Learn about IMS system and application performance

Explore methods and tools for monitoring

Examine recent performance enhancements

Paolo Bruni
Rafael Avigad
James Martin
Maiko Mizuki
Bhups Narsi
John Schlatweiler
Note: Before using this information and the product it supports, read the information in “Notices” on page xxi.

First Edition (January 2013)

This edition applies to Version 12, Release 1 of IBM IMS Transaction and Database Servers (program number 5635-A03) and Version 2, Release 1 of IBM Enterprise Suite (program number 5655-T62).
# Contents

**Figures** .......................... ix

**Examples** .......................... xv

**Tables** .......................... xix

**Notices** .......................... xxi

**Trademarks** .......................... xxii

**Preface** .......................... xxiii

The team who wrote this book ........................................ xxiii

Now you can become a published author, too! .......................... xxv

Comments welcome .................................................. xxv

Stay connected to IBM Redbooks .................................................. xxv

**Chapter 1. Performance monitoring with IMS utilities** .......................... 1

1.1 IMS utilities .................................................. 2

1.2 IMS and DB Monitor .................................................. 2

1.2.1 Activation and control .................................................. 3

1.2.2 Logging data .................................................. 3

1.3 File Select and Formatting Print utility (DFSELA10) .................................................. 9

1.4 Log Transaction Analysis utility (DFSILTA0) .................................................. 11

1.5 Statistical Analysis utility (DFSISTS0) .................................................. 13

1.6 Fast Path Log Analysis utility (DBFULTA0) .................................................. 14

**Chapter 2. Performance concepts and monitoring methodology** .......................... 17

2.1 Performance challenges .................................................. 18

2.2 Events for full-function messages .................................................. 18

2.3 Events for Fast Path messages .................................................. 20

2.4 Events for DBCTL .................................................. 22

2.5 Common log records produced for transaction flows .................................................. 23

2.6 Logging in a single IMS system .................................................. 24

2.7 Open transaction flow .................................................. 26

2.8 Monitoring methodology .................................................. 30

2.8.1 Establishing monitoring strategies .................................................. 30

2.8.2 Monitoring multiple systems in DB/DC and DCCTL environments .................................................. 31

2.8.3 Coordinating performance information in an MSC network .................................................. 32

2.8.4 Monitoring Fast Path systems in DB/DC and DCCTL environments .................................................. 33

2.9 Transaction flow in DB/DC and DCCTL environments .................................................. 34

2.9.1 Principal DB/DC and DCCTL monitoring facilities .................................................. 38

2.9.2 Monitoring procedures in a DBCTL environment .................................................. 39

2.10 Monitoring procedures in an open transaction environment .................................................. 40

**Chapter 3. Performance monitoring tools** .......................... 43

3.1 IMS Performance Analyzer for z/OS, Version 4 Release 3 .................................................. 44

3.1.1 Product overview .................................................. 44

3.1.2 Product features .................................................. 46

3.1.3 Forms-based reports .................................................. 48

3.1.4 Message queue transaction and full-function reports .................................................. 56

3.1.5 Fast Path Reports .................................................. 68

© Copyright IBM Corp. 2013. All rights reserved.
Chapter 4. Platform-related performance functions

4.1 Dynamic full-function buffer pools
   4.1.1 Performance characteristics of the UPDATE POOL command
   4.1.2 Difference between OSAM and VSAM quiesce processing

4.2 Increased VSAM pools

4.3 Logging enhancements
   4.3.1 Striping of OLDS and SLDS
   4.3.2 OLDS buffers specifications
   4.3.3 Migration to buffers above the 2 GB bar
   4.3.4 Migration to striping and buffers in 64-bit storage

4.4 Disk technology on log performance
   4.4.1 Test environment
   4.4.2 Results

4.5 WADS redesign

4.6 System pools in 64-bit real storage

4.7 64-bit ACB processing
   4.7.1 Defining 64-bit ACB
   4.7.2 Creating and sizing the 64-bit storage pool
   4.7.3 QUERY POOL command
   4.7.4 New 4515 log record
   4.7.5 New monitor records are written in certain situations

4.8 zAAP or zIIP times in dependent region accounting log records

4.9 IMS use of zIIP

4.10 Member online change NAMEONLY option

4.11 IMS and Workload Manager (WLM)
   4.11.1 WLM terminology
   4.11.2 WLM concepts of service class and classification
   4.11.3 WLM importance
4.11.4 WLM concepts and goal types .................................................. 175
4.11.5 WLM managed delays ............................................................. 176
4.11.6 WLM concepts and performance level ..................................... 177
4.11.7 WLM service class periods ....................................................... 177
4.11.8 WLM use for IMS ................................................................. 179
4.12 Program load options and IBM Language Environment ............... 180
  4.12.1 Library routine retention ..................................................... 180
  4.12.2 Preload library routines ..................................................... 181
  4.12.3 Preload application programs .............................................. 181
  4.12.4 For OS/VS COBOL programs ............................................... 181
  4.12.5 Checklist for program-related performance problem. ............... 182
4.13 Extended address volume for non-VSAM data sets ........................ 183
  4.13.1 Allocating data sets on EAV volumes .................................... 185
  4.13.2 IMS non-VSAM data sets supported. ..................................... 185
  4.13.3 Example of IMS non-VSAM data sets specification .................. 186
  4.13.4 Special considerations for HALDB OLR output data sets .......... 186
4.14 OTMA support for asynchronous IMS-to-IMS communications .......... 186
  4.14.1 Usage and benefits of Asynchronous IMS-IMS TCP/IP support ...... 188
  4.14.2 Performance evaluation results of IMS connect to IMS connect ... 189
4.15 MSC TCP/IP .......................................................... 189
  4.15.1 MSC TCP/IP uses IMS Connect and the common service layer ...... 189
  4.15.2 IMS to IMS connect functionality ........................................ 190
  4.15.3 IMS considerations .......................................................... 190
  4.15.4 Setup scenario and switching from VTAM to TCP/IP ................... 190
  4.15.5 Fallback scenario ............................................................ 191
  4.15.6 Performance considerations (sample performance test) ............. 191
  4.15.7 Performance expectation ................................................... 192
4.16 Disabling sysplex serial program management (SSPM) ................. 193
4.17 APPC and OTMA shared queue enhancements .................................. 194
  4.17.1 Front-end logging ............................................................ 194
  4.17.2 IMS DISPLAY commands .................................................... 195
  4.17.3 Setup considerations ....................................................... 196
  4.17.4 APPC and OTMA SQ enhancements benefits ............................. 196
  4.17.5 Evaluation results: APPC synchronous shared queues .............. 197
  4.17.6 Evaluation results: OTMA synchronous shared queues .............. 197
4.18 OTMA ACEE reduction for multiple OTMA clients ......................... 197
  4.18.1 Considerations for OTMA setup ........................................... 199
  4.18.2 Benefits ................................................................. 199
  4.18.3 Evaluation results: OTMA ACEE reduction ............................. 199
4.19 OTMA performance enhancement .............................................. 200
4.20 IRLM Version 2.3 and 2.2 .................................................. 200

Chapter 5. Database performance topics ........................................ 201
5.1 Parallel migration to high availability large database (HALDB) ........ 202
  5.1.1 The problem before the enhancement .................................... 202
  5.1.2 Parallel migration to HALDB ................................................. 203
  5.1.3 Parallel migration to HALDB restrictions ............................... 203
5.2 Archive enhancements ............................................................ 203
  5.2.1 Log archiving for non-recoverable databases ........................... 203
  5.2.2 New parameter for Log Archive utility (DFSUARCO) .................. 204
5.3 Online reorganization performance improvements .......................... 204
  5.3.1 OLR VSAM KSDS sequential access ....................................... 204
  5.3.2 Skip GNP call for root-only DB .......................................... 205
5.3.3 Reduce use of the data set busy (ZID) lock during OLR. ................. 205
5.3.4 Eliminate the block (BID) lock for ILDS updates .................. 205
5.3.5 Reduce log records generated during OLR ..................... 205
5.3.6 OLR locking lookside ........................................ 205
5.3.7 Increase in concurrent OLR ................................ 206
5.4 OSAM versus VSAM .............................................. 206
  5.4.1 Features and benefits of OSAM for databases ................. 207
  5.4.2 Considerations for using OSAM ............................ 209
  5.4.3 Summary of benefits of using OSAM ......................... 213
5.5 Understanding IMS locking for performance ....................... 214
  5.5.1 Lock managers ............................................. 214
  5.5.2 Full-function: Database record locks ........................ 215
  5.5.3 Full-function: Segment locks ............................... 215
  5.5.4 Full-function locks: Block locks with IRLM ................. 218
  5.5.5 Full-function locks: Busy locks ............................ 219
  5.5.6 Fast path locks: CI lock ..................................... 219
  5.5.7 Fast path locks: Unit of work (UOW) lock .................. 220
  5.5.8 Lock timeouts .............................................. 220
  5.5.9 Deadlock detection ........................................... 221
  5.5.10 Handling deadlock victims .................................. 222
  5.5.11 Design guidance ............................................ 223
  5.5.12 Space for lock control blocks .............................. 224
  5.5.13 LOCKMAX usage ............................................ 225
  5.5.14 A comparison of PI and IRLM ............................... 225
  5.5.15 Analyze locking with the /TRACE SET program isolation command .... 226
  5.5.16 Sample locking reports ..................................... 227
5.6 CICS threadsafe support ........................................... 232
5.7 Temporary close of VSAM data sets with new extents ............. 233
5.8 Tuning database buffer pools using Buffer Pool Analyzer ....... 233
  5.8.1 Function and effect of I/O, storage, and CPU resources ....... 234
  5.8.2 Buffer pool and subpool organization ....................... 234
  5.8.3 Buffer pool definitions ..................................... 235
  5.8.4 Tuning process overview .................................... 235

Chapter 6. Transaction manager performance: Application considerations ....... 241
  6.1 Full-function transaction ....................................... 242
    6.1.1 Transaction response time ................................ 242
    6.1.2 The analysis of response time ............................ 244
    6.1.3 Input time analysis ....................................... 245
    6.1.4 Input queue time analysis ................................. 248
    6.1.5 Processing time analysis .................................. 253
    6.1.6 Output queue time analysis ............................... 262
    6.1.7 Monitoring ................................................ 262
    6.1.8 Output time analysis ....................................... 264
  6.2 Fast Path transaction ........................................... 266
    6.2.1 Transaction response time ................................ 266
    6.2.2 The analysis of response time ............................ 268
    6.2.3 Input time analysis ....................................... 268
    6.2.4 Input queue time analysis ................................. 269
    6.2.5 Processing time analysis .................................. 271
    6.2.6 Output queue time analysis ............................... 274
    6.2.7 Output time analysis ....................................... 275
  6.3 Application considerations ...................................... 277
10.4.1 Lock structure size .................................................. 356
10.4.2 False contention .................................................... 356
10.4.3 Automatic rebuild .................................................. 356
10.4.4 System-managed duplexing ........................................ 357
10.5 VSAM cache structure ................................................. 357
10.6 OSAM cache structure ................................................. 357
10.7 DEDB considerations .................................................. 358
10.8 Application considerations ............................................ 359
10.9 Shared queues .......................................................... 360
10.9.1 IMS parameters ...................................................... 361
10.9.2 Structure size ........................................................ 361
10.9.3 Structure duplexing ............................................... 361
10.9.4 Overflow .............................................................. 362
10.9.5 Structure checkpoint ............................................... 362
10.9.6 MVS logger ........................................................... 362
10.9.7 FF scheduling differences ........................................ 363
10.9.8 FP Parallel Sysplex processing options ......................... 363
10.9.9 OTMA and APPC synchronous shared queues enhancements .... 363

Chapter 11. Capturing documentation for performance diagnosis .......... 369
11.1 Documentation for defect-related software issues ........................ 370
11.2 Procedures to capture IMS documentation ................................ 371
11.2.1 The IMS Monitor trace ............................................. 371
11.2.2 IMS in storage table traces ....................................... 372
11.2.3 Running and externalizing IMS lock and DL/I traces ............... 372
11.2.4 Storage as a moving target ....................................... 377
11.2.5 The z/OS DUMP command ....................................... 378
11.2.6 GTF trace ........................................................... 381
11.2.7 The /DIAGNOSE command ...................................... 383
11.3 Procedure to request a IMS performance health check .................. 394

Abbreviations and acronyms .................................................. 395

Related publications .......................................................... 399
IBM Redbooks ............................................................... 399
Other publications ........................................................... 399
Online resources ............................................................. 400
Help from IBM ............................................................... 400
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>DFSILTA0 report output</td>
<td>12</td>
</tr>
<tr>
<td>1-2</td>
<td>Sample Transaction Report output</td>
<td>14</td>
</tr>
<tr>
<td>2-1</td>
<td>Events and associated log records for an IMS transaction</td>
<td>24</td>
</tr>
<tr>
<td>2-2</td>
<td>IMS Problem Investigator: Events and log records for an IMS transaction in PI</td>
<td>25</td>
</tr>
<tr>
<td>2-3</td>
<td>Events and associated log records for transaction in shared queues environment</td>
<td>25</td>
</tr>
<tr>
<td>2-4</td>
<td>Connectivity with open environment</td>
<td>26</td>
</tr>
<tr>
<td>2-5</td>
<td>IMS transaction connectivity solutions</td>
<td>27</td>
</tr>
<tr>
<td>2-6</td>
<td>IMS DB connectivity solutions</td>
<td>27</td>
</tr>
<tr>
<td>2-7</td>
<td>IMS Performance Analyzer: IMS Monitor Fast Path Report Set</td>
<td>33</td>
</tr>
<tr>
<td>2-8</td>
<td>IMS Performance Analyzer: IMS Fast Path Transit and Resource Report Set.</td>
<td>34</td>
</tr>
<tr>
<td>2-9</td>
<td>Processing events during transaction flow through IMS</td>
<td>35</td>
</tr>
<tr>
<td>2-10</td>
<td>IMS Problem Investigator: DBCTL Transaction Flow in IMS</td>
<td>39</td>
</tr>
<tr>
<td>2-11</td>
<td>IMS Performance Analyzer: Transaction Resource Usage Summary-DBCTL</td>
<td>40</td>
</tr>
<tr>
<td>2-12</td>
<td>Overview of an IMS configuration that includes ODBM</td>
<td>41</td>
</tr>
<tr>
<td>3-1</td>
<td>IMS Performance Analyzer: Operation overview</td>
<td>45</td>
</tr>
<tr>
<td>3-2</td>
<td>IMS Performance Analyzer: IMS Field Selection</td>
<td>48</td>
</tr>
<tr>
<td>3-3</td>
<td>IMS Performance Analyzer: IMS Connect Field Selection</td>
<td>48</td>
</tr>
<tr>
<td>3-4</td>
<td>IMS Performance Analyzer: OMEGAMON XE TRF Field Selection</td>
<td>49</td>
</tr>
<tr>
<td>3-5</td>
<td>IMS Performance Analyzer: Sorting Summary Report fields</td>
<td>49</td>
</tr>
<tr>
<td>3-6</td>
<td>IMS Performance Analyzer: Summary Report functions.</td>
<td>50</td>
</tr>
<tr>
<td>3-7</td>
<td>IMS Performance Analyzer: IMS Transaction Index creation (Part 1 of 3)</td>
<td>50</td>
</tr>
<tr>
<td>3-8</td>
<td>IMS Performance Analyzer: IMS Transaction Index creation (Part 2 of 3)</td>
<td>51</td>
</tr>
<tr>
<td>3-9</td>
<td>IMS Performance Analyzer: IMS Transaction Index creation (Part 3 of 3)</td>
<td>51</td>
</tr>
<tr>
<td>3-10</td>
<td>IMS Problem Investigator: IMS Transaction Index Record</td>
<td>52</td>
</tr>
<tr>
<td>3-11</td>
<td>IMS Problem Investigator: IMS Transaction Index Merged with IMS Log</td>
<td>53</td>
</tr>
<tr>
<td>3-12</td>
<td>IMS Problem Investigator: Sorting, merging IMS Transaction index-relative view</td>
<td>53</td>
</tr>
<tr>
<td>3-13</td>
<td>IMS Problem Investigator: Sorting, merging IMS transaction index-elapsed view</td>
<td>54</td>
</tr>
<tr>
<td>3-14</td>
<td>IMS Performance Analyzer: IMS Connect Transaction Index creation (Part 1 of 2)</td>
<td>55</td>
</tr>
<tr>
<td>3-15</td>
<td>IMS Performance Analyzer: IMS Connect Transaction Index creation (Part 2 of 2)</td>
<td>55</td>
</tr>
<tr>
<td>3-16</td>
<td>IMS Problem Investigator: IMS Connect Transaction Index</td>
<td>55</td>
</tr>
<tr>
<td>3-17</td>
<td>IMS Performance Analyzer: Transit time by LTERM</td>
<td>56</td>
</tr>
<tr>
<td>3-18</td>
<td>IMS Performance Analyzer: Transit Time by Transaction code</td>
<td>57</td>
</tr>
<tr>
<td>3-19</td>
<td>IMS Performance Analyzer: Transit Time by Transaction code within LTERM</td>
<td>57</td>
</tr>
<tr>
<td>3-20</td>
<td>IMS Performance Analyzer: Transit Time by Line</td>
<td>57</td>
</tr>
<tr>
<td>3-21</td>
<td>IMS Performance Analyzer: Transit Time by Message Class</td>
<td>57</td>
</tr>
<tr>
<td>3-22</td>
<td>IMS Performance Analyzer: Transit Time by Time of Input</td>
<td>58</td>
</tr>
<tr>
<td>3-23</td>
<td>IMS Performance Analyzer: Transaction Resource Usage-Summary</td>
<td>58</td>
</tr>
<tr>
<td>3-24</td>
<td>IMS Performance Analyzer: Resource Availability by Region.</td>
<td>59</td>
</tr>
<tr>
<td>3-25</td>
<td>IMS Performance Analyzer: Resource Availability by Transaction</td>
<td>59</td>
</tr>
<tr>
<td>3-26</td>
<td>IMS Performance Analyzer: Resource Availability by Program</td>
<td>59</td>
</tr>
<tr>
<td>3-27</td>
<td>IMS Performance Analyzer: Resource Availability by Line</td>
<td>59</td>
</tr>
<tr>
<td>3-28</td>
<td>IMS Performance Analyzer: Resource Availability by Database</td>
<td>60</td>
</tr>
<tr>
<td>3-29</td>
<td>IMS Performance Analyzer: CPU Usage, Region</td>
<td>60</td>
</tr>
<tr>
<td>3-30</td>
<td>IMS Performance Analyzer: Internal Resource Usage</td>
<td>61</td>
</tr>
<tr>
<td>3-31</td>
<td>IMS Performance Analyzer: MSC Link Statistics</td>
<td>61</td>
</tr>
<tr>
<td>3-32</td>
<td>IMS Performance Analyzer: Message Queue Utilization</td>
<td>61</td>
</tr>
<tr>
<td>3-33</td>
<td>IMS Performance Analyzer: Database Update Activity</td>
<td>62</td>
</tr>
<tr>
<td>3-34</td>
<td>IMS Performance Analyzer: OSAM Sequential Buffering Report</td>
<td>62</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>3-35 IMS Performance Analyzer: Deadlock Report List Report</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>3-36 IMS Performance Analyzer: Deadlock Summary Report</td>
<td>63</td>
<td></td>
</tr>
<tr>
<td>3-37 IMS Performance Analyzer: System Checkpoint</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>3-38 IMS Performance Analyzer: BMP Checkpoint list report</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>3-39 IMS Performance Analyzer: BMP Checkpoint Summary report</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>3-40 IMS Performance Analyzer: GAP Analysis Report options</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>3-41 IMS Performance Analyzer: Cold Start Analysis Summary Report</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>3-42 IMS Performance Analyzer: Cold Start Analysis detail Report</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>3-43 IMS Performance Analyzer: Transit Log Report</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>3-44 IMS Performance Analyzer: Region Histogram</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>3-45 IMS Performance Analyzer: DC Queue Manager Trace</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>3-46 IMS Performance Analyzer: DC UOW Tracker</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>3-47 IMS Performance Analyzer: Database Trace</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td>3-48 IMS Performance Analyzer: TRF DL/I List</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>3-49 IMS Performance Analyzer: TRF DL/I Summary</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>3-50 IMS Performance Analyzer: TRF DB2 List</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>3-51 IMS Performance Analyzer: TRF DB2 Summary</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>3-52 IMS Performance Analyzer: ATF List</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>3-53 IMS Performance Analyzer: ATF Summary</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>3-54 IMS Performance Analyzer: ATF Trace Level 1</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>3-55 IMS Performance Analyzer: ATF Trace Level 2</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>3-56 IMS Performance Analyzer: ATF Trace Level 3</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>3-57 IMS Performance Analyzer: IMS and DB Monitor Report Set</td>
<td>77</td>
<td></td>
</tr>
<tr>
<td>3-58 IMS Performance Analyzer: IMS Monitor: Region Summary, Schedule/Transaction Calls</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>3-59 IMS Performance Analyzer: IMS Monitor: Region Summary, Message and Database Calls</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>3-60 IMS Performance Analyzer: IMS Monitor: Region Summary, Distributions</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>3-61 IMS Performance Analyzer: IMS Monitor: Region Summary, Program Analysis</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>3-62 IMS Performance Analyzer: IMS Monitor: Region Activity Analysis, Region Detail</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td>3-63 IMS Performance Analyzer: IMS Monitor - Program Analysis Report Set</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>3-64 IMS Problem Investigator: End to End Transaction Flow</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td>3-65 IMS Problem Investigator for z/OS: Process overview</td>
<td>83</td>
<td></td>
</tr>
<tr>
<td>3-66 IMS Problem Investigator: Log file selection</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>3-67 IMS Problem Investigator: Multiple Log File Extract Request</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td>3-68 IMS Problem Investigator: Time slicing selection</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>3-69 IMS Problem Investigator: Log File Merging</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>3-70 IMS Problem Investigator: Navigation in Merged Data</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>3-71 IMS Problem Investigator: Assigning labels</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>3-72 IMS Problem Investigator: Navigation to Labels</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>3-73 IMS Problem Investigator: Log Record View with summary</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>3-74 IMS Problem Investigator: Log Record View without summary</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>3-75 IMS Problem Investigator: Log Record formatted view</td>
<td>91</td>
<td></td>
</tr>
<tr>
<td>3-76 IMS Problem Investigator: Detailed field description</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>3-77 IMS Problem Investigator: Detailed Flag Bit field description</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td>3-78 IMS Problem Investigator: Time specification</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>3-79 IMS Problem Investigator: Log record in HEX1 format</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>3-80 IMS Problem Investigator: Log record in DUMP format</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>3-81 IMS Problem Investigator: Forms creation pane</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>3-82 IMS Problem Investigator: Form field customization</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>3-83 IMS Problem Investigator: Formatted Log Record FORM View</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td>3-84 IMS Problem Investigator: Formatted versus ISPF browse views</td>
<td>97</td>
<td></td>
</tr>
<tr>
<td>3-85 IMS Problem Investigator: Transaction tracking</td>
<td>98</td>
<td></td>
</tr>
<tr>
<td>3-86 IMS Problem Investigator: Unit of recovery tracking</td>
<td>98</td>
<td></td>
</tr>
</tbody>
</table>
3-87  IMS Problem Investigator: OMEGAMON TRF - Resource Trace .......................... 99
3-88  IMS Problem Investigator: OMEGAMON TRF - Database Trace .......................... 100
3-89  IMS Problem Investigator: OMEGAMON ATF tracking ..................................... 101
3-90  IMS Problem Investigator: ATF journal system definition ................................. 101
3-91  IMS Problem Investigator: ODBM open database formatting ............................. 102
3-92  IMS Problem Investigator: DRDA open source code point formatting .................. 103
3-93  IMS Problem Investigator: DRDA IMS specific code point formatting ................... 103
3-94  IMS Problem Investigator: DL/I Call Analysis .................................................. 104
3-95  IMS Problem Investigator: Merged IMS Log and Connect Extension Journal .......... 105
3-96  IMS Problem Investigator: DB2 log formatting ................................................ 105
3-97  IMS Problem Investigator: IRLM-SMF Long Lock record formatting .................... 106
3-98  IMS Problem Investigator: SMF 4F0F record formatted .................................... 107
3-99  IMS Problem Investigator: ITR trace records .................................................. 107
3-100 IMS Problem Investigator: MSC using TCP/IP record formatting ....................... 108
3-101 IMS Problem Investigator: Repository Server record formatting ......................... 109
3-102 IMS Problem Investigator: IMS Transaction Index record formatting ................. 109
3-103 IMS Problem Investigator: Transaction Index record formatted ........................ 110
3-104 IMS Problem Investigator: IMS Connect Transaction Index ............................... 111
3-105 IMS Problem Investigator: IMS Connect Transaction Index formatted (Part 1 of 2) 112
3-106 IMS Problem Investigator: IMS Connect Transaction Index formatted (Part 2 of 2) 112
3-107 IMS Problem Investigator: Requesting BATCH scrub extract processing ............. 113
3-108 IMS Problem Investigator: Building a scrubbed extract .................................. 113
3-109 IMS Problem Investigator: Scrubbed log record ............................................. 114
3-110 IMS Problem Investigator: Batch Reporting BRIEF Request ................................ 115
3-111 IMS Problem Investigator: Batch Reporting BRIEF Output ............................... 115
3-112 IMS Problem Investigator: LOGINFO Report output data set ............................. 116
3-113 IMS Problem Investigator: LOGINFO Report output ....................................... 116
3-114 IMS Connect with IMS Connect Extensions ..................................................... 117
3-115 IMS Connect Extensions: Centralized monitoring and control ............................ 118
3-116 IMS Connect Extensions: Components .............................................................. 120
3-117 IMS Connect Extensions: ISPF dialog client . .................................................... 121
3-118 IMS Connect Extensions: Operations Console ................................................... 122
3-119 IMS Performance Analyzer: IMS Connect Reports Set .................................... 125
3-120 IMS Connect Extensions: Using Problem Investigator with Connect Extensions .. 126
3-121 OMEGAMON XE: Response Time Analysis ....................................................... 133
3-122 OMEGAMON XE: Lock Analysis .......................................................... 134
3-123 OMEGAMON XE: Bottleneck Analysis (Part 1 of 2) ......................................... 135
3-124 OMEGAMON XE: Bottleneck Analysis (Part 2 of 2) ......................................... 136
3-125 OMEGAMON XE: Near-Term History ......................................................... 137
3-126 Transaction Analysis Workbench: Problem registration .................................... 142
3-127 Transaction Analysis Workbench: Session log file management ......................... 143
3-128 Transaction Analysis Workbench: Browsing log files (Part 1 of 2) ...................... 144
3-129 Transactional Analysis Workbench: Browsing log files (Part 2 of 2) .................... 145
3-130 Transaction Analysis Workbench: Tagging and notes ...................................... 146
3-131 Transaction Analysis Workbench: Session reporting ....................................... 147
3-132 Transaction Analysis Workbench: Transaction tracking (Part 1 of 2) .................... 148
3-133 Transaction Analysis Workbench: Transaction tracking (Part 2 of 2) .................... 148
3-134 Transaction Analysis Workbench: Reporting capabilities ................................... 149
3-135 Transaction Analysis Workbench: Building a Filter ......................................... 151
3-136 Transaction Analysis Workbench: ISPF Dialog Map (Part 1 of 2) ...................... 152
3-137 Transaction Analysis Workbench: ISPF Dialog Map (Part 2 of 2) ...................... 153
3-138 Transaction Analysis Workbench: Repository structure ..................................... 154
3-139 Transaction Analysis Workbench: Automated file selection .............................. 156

Figures xi
6-12  Response time concept of Fast Path transaction .......................... 266
6-13  Input time .................................................. 268
6-14  Input queue time ............................................ 269
6-15  Processing time ............................................... 271
6-16  Output queue time .......................................... 274
6-17  Output time .................................................. 275
6-18  Structure of an application program ............................... 278
6-19  Sample hierarchy .............................................. 280
6-20  Command codes by DL/I call type ............................... 281
6-21  Differences between single and multiple positioning .......... 282
6-22  Language environment view .................................. 288
7-1  Parallelism can help improve OTMA performance ................. 294
7-2  Specifying routing rules to improve parallelism ................. 295
7-3  Setting transaction timer options and activating transaction expiration 297
7-4  The effect of ACK delay on overall IMS Connect performance ... 298
7-5  Report demonstrating slow performance from ACK delay ....... 299
7-6  Report demonstrating improved performance with NODELAYACK ... 300
7-7  Open Database distributed architecture .......................... 301
7-8  Wireshark analysis of a DRDA request .......................... 302
7-9  IMS Problem Investigator: ODBM Open Database formatting .... 303
7-10 IMS Problem Investigator: DRDA open source code-point formatting 304
7-11 IMS Problem Investigator: DRDA IMS code-point formatting ... 304
7-12 IMS Problem Investigator: DL/I call analysis ...................... 305
7-13 IMS Problem Investigator: Merged IMS log and connect extension journal 306
7-14 A synchronous callout allows an IMS transaction to reach web services 307
7-15 A synchronous callout consists of multiple interrelated steps ..... 308
7-16 Report showing synchronous callout performance ............... 308
7-17 Synchronous callout event flow ................................ 309
7-18 Distributed workloads change the perspective of performance problems 310
7-19 IMS performance is optimal; most of the time is spent in IMS Connect 311
8-1  DL/I call access patterns to DEDB ................................. 314
9-1  DBRC record types within RECON ............................. 339
9-2  The CLEANUP.RECON command syntax ....................... 346
9-3  Syntax of the CHANGE.CAGRP command ....................... 348
9-4  DBRC access times for serial and parallel RECON access ....... 350
9-5  DBRC BPE/non-BPE processing benefits and performance comparison 351
10-1 Flow using XCF and synlevel=NONE ........................... 366
10-2 Flow using XCF and synlevel confirm ........................... 367
Examples

1-1 IMS Monitor VSAM Buffer Pool report ............................................. 7
1-2 Region Summary Report ................................................................. 8
1-3 DFSERA10 Example JCL ............................................................... 10
1-4 DFSILTA0 sample JCL ................................................................. 11
1-5 Execution of the Statistical Analysis utility ................................. 13
1-6 DBFULTA0 sample JCL ................................................................. 15
1-7 DBFULTA0 report output: Detail Listing of Exception Transactions .... 15
2-1 Sample event flow: Commit mode 1, sync level none ....................... 28
2-2 Sample event flow: Commit mode 1, sync level confirm ................... 28
2-3 Sample event flow: ODBM ............................................................ 29
3-1 IMS Performance Analyzer: Output from IMS Transaction Index build .......................... 51
3-2 Sample JCL to sort IMS Transaction Index in time sequence ........... 54
3-3 Command display output ............................................................. 139
4-1 OLDS and WADS JCL specification .............................................. 186
5-1 Allocate single-volume OSAM database data set: JCL ..................... 210
5-2 Allocate single-volume OSAM database data set: AMS ...................... 210
5-3 Allocate multi-volume OSAM database data set: JCL, non SMS-managed DASD ... 210
5-4 Allocate multi-volume OSAM database data set: AMS, non SMS-managed DASD ... 211
5-5 Allocate multi-volume OSAM database data set: JCL, SMS-managed DASD ... 213
5-6 Allocate multi-volume OSAM database data set: AMS, SMS-managed DASD ... 213
5-7 DFSERA40 formatting ................................................................. 226
6-1 IMS Performance Analyzer for z/OS: Transit Time Analysis .............. 243
6-2 Statistical Analysis utility (DFSISTS0): Transaction Response Report .... 244
6-3 IMS Performance Analyzer for z/OS: Internal Resource Usage report, Miscellaneous Statistics ........................................................ 246
6-4 IMS Performance Analyzer for z/OS: Internal Resource Usage report, Dynamic SAP Statistics .......................................................... 247
6-5 IMS Performance Analyzer for z/OS: Transit Time Analysis .............. 249
6-6 Log Transaction Analysis utility (DFSILTA0) report ......................... 249
6-7 IMS Performance Analyzer for z/OS: IRUR, Scheduling Statistics ........ 251
6-8 IMS Performance Analyzer for z/OS: Performance Exceptions report, Intent Failure Summary ........................................................ 252
6-9 IMS Performance Analyzer for z/OS: Performance Exceptions report, Pool Space Failure Summary ........................................................ 252
6-10 IMS Monitor report: Reports ......................................................... 252
6-11 IMS Performance Analyzer for z/OS: Transit Time Analysis .......... 254
6-12 Log Transaction Analysis utility (DFSILTA0) ................................. 255
6-13 IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis ........................................................ 255
6-14 IMS Monitor report: PROGRAM SUMMARY .................................. 256
6-15 IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis ........................................................ 256
6-16 IMS Monitor report: PROGRAM SUMMARY .................................. 256
6-17 IMS Monitor report: CALL SUMMARY .......................................... 257
6-18 IMS Monitor report: PROGRAM SUMMARY .................................. 258
6-19 IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis ........................................................ 258
6-20 IMS Monitor report: PROGRAM I/O .............................................. 259
6-21 IMS Monitor report: PROGRAM SUMMARY .................................. 259
6-22 IMS Monitor report: PROGRAM I/O .............................................. 259
6-23 IMS Monitor report: PROGRAM SUMMARY .................................. 259
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9-5</td>
<td>LIST.RECON STATUS after redefining the VSAM cluster for RECON1</td>
<td>344</td>
</tr>
<tr>
<td>9-6</td>
<td>The CLEANUP.RECON command</td>
<td>347</td>
</tr>
<tr>
<td>9-7</td>
<td>The DELETE.LOG command</td>
<td>347</td>
</tr>
<tr>
<td>9-8</td>
<td>The CHANGE.CAGRP command with the RECOVPD keyword</td>
<td>349</td>
</tr>
<tr>
<td>11-1</td>
<td>Starting and stopping the IMS Monitor</td>
<td>372</td>
</tr>
<tr>
<td>11-2</td>
<td>The /DIAG SNAP commands</td>
<td>391</td>
</tr>
<tr>
<td>11-3</td>
<td>JCL for printing DIAG records with exit DFSERA30</td>
<td>393</td>
</tr>
</tbody>
</table>
# Tables

1-1 IMS monitoring utilities ................................................................. 2
1-2 IMS Monitor reports ........................................................................ 5
2-1 Event list for full-function messages ............................................. 19
2-2 Event list for Fast Path messages ................................................. 21
2-3 Event list for DBCTL ...................................................................... 22
2-4 Common log records produced by both full-function and Fast Path .................. 23
3-1 IMS monitoring tools ...................................................................... 44
4-1 Results when no striping was used ................................................ 164
4-2 Results when two stripes were used .............................................. 165
4-3 Results when four stripes were used ............................................. 165
4-4 IMS 12 WADS device response time and improvement, ...................... 166
4-5 MSC TCPIP results ......................................................................... 193
4-6 MSC VTAM results .......................................................................... 193
5-1 Online workload measurement results ........................................... 206
5-2 BMP workload measurement results ............................................. 207
6-1 Comparison of standard DL/I call and command code ....................... 280
8-1 Locking level for GU and GN DL/I calls on VSO DEDB .................... 323
8-2 OTHREAD processing for VSO and non-VSO DEDBs ..................... 324
8-3 Scenario 1: HDAM/VSAM with two secondary indexes versus one area DEDB with two secondary indexes ......................................................... 335
8-4 Scenario 2: Access DEDB with secondary index sequence versus no secondary index sequence .............................................................. 335
8-5 Scenario 3: Access DEDB with 0, 2, and 4 secondary indexes .......... 336
11-1 /DIAGNOSE valid environments and keywords ............................ 384
11-2 The /DIAGNOSE SNAP AREA() control blocks ............................ 386
11-3 The /DIAGNOSE SNAP DB() control blocks ................................. 386
11-4 The /DIAGNOSE SNAP LINE() control blocks ............................ 387
11-5 The /DIAGNOSE SNAP LINK() control blocks ............................. 387
11-6 The /DIAGNOSE SNAP PGM() control blocks ............................. 388
11-7 The /DIAGNOSE SNAP REGION() control blocks ........................ 389
Notices

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

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

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

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

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

COPYRIGHT LICENSE:

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

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

CICS®
DB2®
Distributed Relational Database Architecture™
DRDA®
Enterprise Storage Server®
Enterprise Workload Manager™
IBM®
IMS™
IMS/ESA®
InfoSphere®
Language Environment®
MVS™
OMEGAMON®
OS/390®
Parallel Sysplex®
RAA®
RACF®
Redbooks®
Redbooks (logo)®
Resource Measurement Facility™
RMF™
System z®
Tivoli Enterprise Console®
Tivoli®
VTAM®
WebSphere®
z/Architecture®
z/OS®

The following terms are trademarks of other companies:

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

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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

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

IBM® Information Management System (IMS™) provides leadership in performance, reliability, and security to help you implement the most strategic and critical enterprise applications. IMS, IMS utilities, and IMS tools continue to evolve to provide value and meet the needs of enterprise customers.

With IMS 12, integration and open access improvements provide flexibility and support business growth requirements. Scalability improvements have been made to the well-known performance, efficiency, availability, and resilience of IMS by using 64-bit storage.

In this IBM Redbooks® publication we provide IMS performance monitoring and tuning information by describing the key IMS performance functions and by showing how to monitor and tune them with traditional and new strategic applications. This book is for database administrators and system programmers.

We summarize methods and tools for monitoring and tuning IMS systems, describe IMS system-wide performance, database, and transaction considerations.

Based on lab measurements, we provide information about recent performance enhancements that are available with IMS 12, and advice about setting performance-related parameters.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), San Jose Center.

Paolo Bruni is an ITSO Project Leader based in the Silicon Valley Lab in San Jose, CA. Since 1998, Paolo has authored IBM Redbooks publications about IMS, IBM DB2® for z/OS®, and related tools, and he has conducted workshops worldwide. During his many years with IBM in development and in the field, Paulo's work has mostly been related to database systems.

Rafael Avigad is a Product Architect and an Information Developer with Fundi Software, in Perth, Western Australia. For the past six years, he has worked on solutions for managing TCP/IP access to IMS, and has helped to develop current and future IMS tools graphical user interfaces. Before working at Fundi, Rafael developed operational support and billing systems for mobile telephony and independent software vendors (ISVs).

James Martin is a US Representative and a Quality Assurance Specialist for Fundi Software. He has six years of experience working with IMS and IMS Tools. His previous roles at Fundi included two years as a Quality Assurance Specialist for IBM IMS Connect Extensions for IBM z/OS. For the last two years, James has been the Quality Assurance Lead for IBM IMS Problem Investigator for z/OS, and a US Representative, a role that takes him all over the United States, South America, and Canada providing technical support and education for various IMS customers. He is also a Project Manager on the IMS project for the SHARE Enterprise Technology IT Professionals Association.
Maiko Mizuki is working as an IMS SME at IBM in Japan. She belongs to the technical support team of IMS and IMS Tools, providing support on IMS performance monitoring and tuning, IMS Open Database, and IMS Connectivity. Maiko has experience with problem determination, using IMS Monitor, IMS Log Transaction Analysis, IMS Performance Analyzer, and IMS Problem Investigator.

Bhups Narsi is an IMS Level 2 Worldwide Support Specialist with IBM in Australia. He has 30 years of experience with IMS, working as a Technical Support Specialist. His areas of expertise is with IMS full-function databases. He holds a degree in Mathematics and Computer Science from University of the Witwatersrand in South Africa.

John Schlatweiler is an Infrastructure Engineer with Nationwide Mutual Insurance Company, in Columbus, Ohio, US, where he is a member of the Mainframe Engineering Team. He has 28 years of experience in various IT fields. His areas of expertise include IBM CICS®, DB2, IMS, and z/OS. John has worked as a Software Developer, Software Architect, Database Administrator, Systems Programmer, Technical Leader, and Infrastructure Engineer. Prior to joining Nationwide, John was responsible for the strategic direction of database and transaction processing systems in a large telecommunications environment. He recently developed and taught the IMS curriculum for Nationwide's internal IBM System z® University (z/SU) and has participated in a number of workgroups as a Nationwide representative on the IBM zBusiness Leadership Council (zBLC).

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

Glenn Galler
Mark Rader
Nancy Stein
IBM Sales & Distribution

Richard Arellanes
Terry Krein
Jeff Maddix
Bruce Naylor
Geoff Nicholls
Tom Ross
IBM Software Group

Stuart Burnfield
Shirley Collins
Graham Hannington
Jim Martin
Steve Nathan
Fundi Software
Now you can become a published author, too!

Here’s an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at: ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

- Use the online Contact us review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:
  http://www.facebook.com/IBMRedbooks
- Follow us on Twitter:
  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:
  http://www.linkedin.com/groups?home=&gid=2130806
- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
- Stay current on recent Redbooks publications with RSS Feeds:
  http://www.redbooks.ibm.com/rss.html
Performance monitoring with IMS utilities

In this chapter, we introduce the IMS system utilities that can help give you a detailed picture of your IMS system resource usage.

We describe the following IMS system utilities:

- IMS and DB Monitor
- File Select and Formatting Print utility (DFSERA10)
- Log Transaction Analysis utility (DFSILTA0)
- Statistical Analysis utility (DFSISTS0)
- Fast Path Log Analysis utility (DBFULTA0)

Performance tools are described in Chapter 3, “Performance monitoring tools” on page 43.
1.1 IMS utilities

Table 1-1 lists the integrated system utilities that are available for IMS performance evaluations and available as IMS components. To evaluate IMS performance, you do not need all of these utilities. We provide details to help you assess the usefulness of each utility for your IMS installation.

**Table 1-1  IMS monitoring utilities**

<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS Monitor</td>
<td>IMS Monitor traces IMS activities at the ITASK level of detail. An offline program generates reports describing IMS online characteristics. It is the primary tool for IMS tuning.</td>
<td>IMS Version 12 Operations and Automation, SC19-3018</td>
</tr>
<tr>
<td>DB Monitor</td>
<td>DB Monitor is available to IMS batch systems; it can monitor the activity between application programs and databases.</td>
<td>IMS Version 12 Operations and Automation, SC19-3018</td>
</tr>
<tr>
<td>File Select and Formatting utility (DFSERA10)</td>
<td>File Select and Formatting Print utility formats and prints selected records from the IMS log.</td>
<td>IMS Version 12 System Utilities, SC19-3023</td>
</tr>
<tr>
<td>Log Transaction Analysis utility (DFSILTA0)</td>
<td>Log Transaction Analysis utility collects information about individual occurrences of IMS transactions based on records in the IMS log data set.</td>
<td>IMS Version 12 System Utilities, SC19-3023</td>
</tr>
<tr>
<td>Statistical Analysis utility (DFSISTS0)</td>
<td>Statistical Analysis utility analyzes the information in any of the IMS system logs, except those from a batch region.</td>
<td>IMS Version 12 System Utilities, SC19-3023</td>
</tr>
<tr>
<td>Fast Path Log Analysis utility (DBFULTA0)</td>
<td>Fast Path Log Analysis utility prepares statistical reports for Fast Path, based on data that is recorded on the IMS system log.</td>
<td>IMS Version 12 System Utilities, SC19-3023</td>
</tr>
</tbody>
</table>

1.2 IMS and DB Monitor

IMS Monitor and DB Monitor, which are both integrated into the IMS product, collect information about most dispatchable events in the IMS system, including Fast Path events starting with IMS Version 7. They are the primary tools for tuning IMS, analyzing application programs, validating database design, and monitoring system performance. They are also a means of inferring the effect on IMS of the interaction of the hardware and software components of the total system. IMS Monitor collects its data while the online IMS system is in operation while DB Monitor is available for use with your IMS batch jobs.

IMS uses the data produced by each Monitor to generate reports. The report programs run offline and print reports that summarize and categorize IMS activities.

IMS Performance Analyzer for z/OS also uses IMS and DB Monitor log records to generate reports and traces of IMS system activity. See “IMS Monitor reporting” on page 77.
1.2.1 Activation and control

In this section, we describe how to activate and control IMS and DB Monitor Utilities.

IMS Monitor

If you created an IBM MVS™ dynamic allocation (MDA) module named DFSDCMON in the IMS.SDFSRESL data set, you do not need a DD statement in the IMS, DBC, or DCC procedures for IMS Monitor because IMS dynamically allocates and deallocates IMS Monitor data set. This way allows for starting and stopping IMS Monitor with the /TRACE command.

Several activities that you can monitor with the /TRACE command include:

- Telecommunication line and logical link activity
- Scheduling and termination events
- Activity between application programs and message queues
- Activity between application programs and databases (full-function and Fast Path)

You can also use the /TRACE command to limit the monitoring to the following items:

- Particular databases, partitions, or areas
- Particular dependent regions
- A particular interval of time

More information about the /TRACE command can be found in IMS Version 12: System Administration, SC19-3020.

Otherwise, to activate IMS Monitor, you must include a DD statement (using IMSMON as the data set name) in the IMS, DBC, or DCC procedures to specify IMS Monitor log data set. When you include this DD statement, IMS Monitor becomes available. If IMS Monitor is available but inactive, processor usage is unaffected.

DB Monitor

When activating the DB Monitor, you need to specify MON=Y in the PROC statement of the batch job. When you submit the job, IMS uses parameter substitution to update the PARM field of the EXEC statement with a Y in the appropriate position.

Note the following information:

- When stopping the DB Monitor, the system console operator uses the following command:
  MODIFY jobname,STOP
  Message DFS2215A is displayed on the system console when the monitor is inactive.
- The console operator can then reactivate the DB Monitor by using the following command:
  MODIFY jobname,START
  Message DFS2216A is displayed on the console when the Monitor is active again.

1.2.2 Logging data

This section describes how to log data using IMS and DB monitor utilities.

IMS Monitor

Because of how your IMS Monitor log data set is stored, you should process the log after you stop IMS Monitor and before restarting it again. You should start IMS Monitor and allow it to run for a period of time, and then stop it to write a “snapshot” of current activity on IMS Monitor log.
**DB Monitor**

IMS records the data that is produced by the DB Monitor on either the batch log or on a separate DB Monitor log that is defined by the //IMSMON DD statement. When you use the //IMSMON DD statement to control data logging, consider the following information:

- To store the DB Monitor records on the batch log, either include a //IMSMON DD DUMMY statement or omit the //IMSMON DD statement entirely.
- To store the Monitor records on a separate DB Monitor log, include a valid //IMSMON DD statement.

If IMS is unable to open the log data set that is specified with //IMSMON DD, a DFS2217I message is displayed on the IMS console. Note that, when this error condition arises, batch processing continues normally with the DB Monitor inactive.

If the DB Monitor device encounters any I/O errors, the DFS2219I message is displayed on the system console. Batch processing can continue normally and the DB Monitor will be inactive.

To stop the Monitor and force an end-of-volume for the DB Monitor log, use the **MODIFYjobname,STOPEOV** command. When you use the STOPEOV keyword, the batch region does not continue with batch execution until the z/OS mount request for a new data set is satisfied.

**Messages issued:**

- If you enter an incorrect job name with the MODIFY command, an error message is issued by z/OS, not IMS.
- If you make some other error when you enter the MODIFY command, IMS issues a DFS2218I message, followed by either a DFS2215A or DFS2216A message.

Table 1-2 on page 5 lists most of the report titles and the most important fields from the DFSUTR20 and DFSUTR30 reports. In the Comments column, these abbreviations are used:

- Reports marked TM apply to IMS Monitor data.
- Reports marked DB apply to IMS DB Monitor data.
- Reports marked DBCTL apply to DBCTL users.
Table 1-2  IMS Monitor reports

<table>
<thead>
<tr>
<th>Key report fields</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Buffer pool statistics</strong></td>
<td></td>
</tr>
<tr>
<td>Message queue pools report</td>
<td></td>
</tr>
<tr>
<td>Number of immediate fetch (I/F) I/Os</td>
<td>TM</td>
</tr>
<tr>
<td>Number of directory I/O operations</td>
<td></td>
</tr>
<tr>
<td>Number of times blocks washed for FRE</td>
<td></td>
</tr>
<tr>
<td>Database buffer pools report</td>
<td></td>
</tr>
<tr>
<td>Number of read-I/O requests</td>
<td>DB and DBCTL</td>
</tr>
<tr>
<td>Number of blocks written by purge</td>
<td></td>
</tr>
<tr>
<td>Number of locate calls waited because of busy ID</td>
<td></td>
</tr>
<tr>
<td>Number of locate calls waited because of buffer busy write</td>
<td></td>
</tr>
<tr>
<td>Number of locate calls waited because of buffer busy read</td>
<td></td>
</tr>
<tr>
<td>Total number of I/O errors for this subpool</td>
<td></td>
</tr>
<tr>
<td>VSAM buffer pool report</td>
<td></td>
</tr>
<tr>
<td>Number of times control interval requested already in pool</td>
<td>DB and DBCTL</td>
</tr>
<tr>
<td>Number of control intervals read from external storage</td>
<td></td>
</tr>
<tr>
<td>Number of VSAM writes initiated by IBM IMS/ESA®</td>
<td></td>
</tr>
<tr>
<td>Number of VSAM writes to make space in the pool</td>
<td></td>
</tr>
<tr>
<td>Message format buffer pool report</td>
<td></td>
</tr>
<tr>
<td>Number of I/F I/Os</td>
<td>TM</td>
</tr>
<tr>
<td>Number of times blocks washed for FRE</td>
<td></td>
</tr>
<tr>
<td>Quotient</td>
<td></td>
</tr>
<tr>
<td><strong>General reports</strong></td>
<td></td>
</tr>
<tr>
<td>Latch conflict statistics report</td>
<td></td>
</tr>
<tr>
<td>LOGL Contentions</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>General IWAIT time events report</td>
<td></td>
</tr>
<tr>
<td>Mean IWAIT Time</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Region and jobname report</td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td><strong>Region summary</strong></td>
<td></td>
</tr>
<tr>
<td>Scheduling and termination section</td>
<td></td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Not IWAIT Time (Elapsed - IWAIT) Mean</td>
<td></td>
</tr>
<tr>
<td>Schedule to first call section</td>
<td></td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td>TM and DBCTL</td>
</tr>
<tr>
<td>Elapsed execution section</td>
<td></td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>DL/I calls section</td>
<td></td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Not IWAIT Time (Elapsed - IWAIT) Mean</td>
<td></td>
</tr>
<tr>
<td>Idle for intent section</td>
<td></td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Region section</td>
<td>Elapsed Time Mean Not IWAIT Time (Elapsed - IWAIT) Mean</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Region occupancy section</td>
<td>Percentage</td>
</tr>
<tr>
<td><strong>Region IWAIT (report is one per region)</strong></td>
<td></td>
</tr>
<tr>
<td>Scheduling and termination section</td>
<td>IWAIT Time Mean Function</td>
</tr>
<tr>
<td>DL/I calls section</td>
<td>IWAIT Time Mean Function Module</td>
</tr>
<tr>
<td>Checkpoint</td>
<td>IWAIT Time Mean Function</td>
</tr>
<tr>
<td><strong>Programs by region (report is one per region)</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Elapsed execution time mean</td>
</tr>
<tr>
<td></td>
<td>Scheduling end to first call time mean</td>
</tr>
<tr>
<td><strong>Program summary</strong></td>
<td></td>
</tr>
<tr>
<td>Calls/Tran</td>
<td>Calls/Tran</td>
</tr>
<tr>
<td>I/O IWAITs / Call</td>
<td>I/O IWAITs / Call</td>
</tr>
<tr>
<td>CPU Time / Sched</td>
<td>CPU Time / Sched</td>
</tr>
<tr>
<td>Elapsed Time / Sched</td>
<td>Elapsed Time / Sched</td>
</tr>
<tr>
<td>Sched to 1st Call / Sched</td>
<td>Sched to 1st Call / Sched</td>
</tr>
<tr>
<td>Elapsed Time / Trans</td>
<td>Elapsed Time / Trans</td>
</tr>
<tr>
<td><strong>Program I/O (report is one per PSB)</strong></td>
<td></td>
</tr>
<tr>
<td>IWAITS</td>
<td>IWAITS</td>
</tr>
<tr>
<td>IWAIT Time Mean</td>
<td>IWAIT Time Mean</td>
</tr>
<tr>
<td>DDN/Func</td>
<td>DDN/Func</td>
</tr>
<tr>
<td><strong>Communication summary</strong></td>
<td></td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td>Elapsed Time Mean</td>
</tr>
<tr>
<td>Not IWAIT Time Mean</td>
<td>Not IWAIT Time Mean</td>
</tr>
<tr>
<td><strong>Communication IWAIT</strong></td>
<td></td>
</tr>
<tr>
<td>IWAIT time total</td>
<td>IWAIT time total</td>
</tr>
<tr>
<td>IWAIT time mean</td>
<td>IWAIT time mean</td>
</tr>
<tr>
<td>Blksize</td>
<td>Blksize</td>
</tr>
<tr>
<td><strong>Line functions</strong></td>
<td></td>
</tr>
<tr>
<td>Mean Receive Blksize</td>
<td>Mean Receive Blksize</td>
</tr>
<tr>
<td>Max Receive Blksize</td>
<td>Max Receive Blksize</td>
</tr>
<tr>
<td>Mean Trans Blksize</td>
<td>Mean Trans Blksize</td>
</tr>
<tr>
<td>Max Trans Blksize</td>
<td>Max Trans Blksize</td>
</tr>
<tr>
<td>Mean Interval</td>
<td>Mean Interval</td>
</tr>
<tr>
<td>Max Interval</td>
<td>Max Interval</td>
</tr>
</tbody>
</table>
Example 1-1 shows VSAM buffer pool statistics. Each VSAM buffer pool defined in DFSVSMxx member has its own report section.

**Example 1-1  IMS Monitor VSAM Buffer Pool report**

<table>
<thead>
<tr>
<th>Transactions Queuing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Transactions Per Second</td>
</tr>
<tr>
<td>Number of D/I Calls Per Transaction</td>
</tr>
<tr>
<td>Number of Message Queue Pool I/Os</td>
</tr>
<tr>
<td>Number of Format Buffer Pool I/Os</td>
</tr>
<tr>
<td>Ratio of Program Elapsed to D/I Elapsed</td>
</tr>
<tr>
<td>Ratio of D/I Elapsed to D/I IWAIT</td>
</tr>
</tbody>
</table>

### Transaction Queuing

<table>
<thead>
<tr>
<th>Transaction Queuing</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number dequeued</td>
<td>Dequeued mean</td>
</tr>
<tr>
<td>On queue when scheduled mean</td>
<td></td>
</tr>
</tbody>
</table>

### Reports

<table>
<thead>
<tr>
<th>Reports</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intent Failure Summary</td>
<td></td>
</tr>
<tr>
<td>Pool Space Failure Summary</td>
<td></td>
</tr>
<tr>
<td>Deadlock Event Summary</td>
<td></td>
</tr>
<tr>
<td>Monitor Overhead Data</td>
<td></td>
</tr>
</tbody>
</table>

### Run Profile

<table>
<thead>
<tr>
<th>Run Profile</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Transactions Per Second</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Number of D/I Calls Per Transaction</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Number of Message Queue Pool I/Os</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Number of Format Buffer Pool I/Os</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Ratio of Program Elapsed to D/I Elapsed</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Ratio of D/I Elapsed to D/I IWAIT</td>
<td>TM, DB, and DBCTL</td>
</tr>
</tbody>
</table>

### Call Summary (Report is One Per PSB)

<table>
<thead>
<tr>
<th>Call Summary (Report is One Per PSB)</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>IWAIT/Call</td>
<td>TM, DB, and DBCTL</td>
</tr>
<tr>
<td>Elapsed Time Mean</td>
<td></td>
</tr>
<tr>
<td>Not IWAIT Time Mean</td>
<td></td>
</tr>
</tbody>
</table>

Example 1-1 shows VSAM buffer pool statistics. Each VSAM buffer pool defined in DFSVSMxx member has its own report section.
The most important indicator for buffer pool performance is “VSAM writes to make space.” It should always be 0. “Buffer Pool Read I/O Rate” is divided by transaction rate, which helps you to understand which pool is most affected or needs adjusting. The hit ratio is not always an accurate indicator for random access.

For more information, see IMS Version 12: System Administration, SC19-3020. IBM IMS and DB2 Tools portfolio includes a product, IMS Buffer Pool Analyzer (see 3.4, “IMS Buffer Pool Analyzer for z/OS, Version 1 Release 3” on page 126), which might be helpful in this analysis.

Example 1-2 shows important summary information for each dependent region that was active during the time period that the monitor trace was on.

Example 1-2 Region Summary Report

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-------------</td>
<td>------------------------</td>
<td>-------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>SCHEDULING AND TERMINATION</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>REGION 1</strong></td>
<td>1</td>
<td>1</td>
<td>48314669</td>
</tr>
<tr>
<td><strong>REGION 2</strong></td>
<td>1</td>
<td>1</td>
<td>42696686</td>
</tr>
<tr>
<td><strong>REGION 3</strong></td>
<td>1</td>
<td>1</td>
<td>46601395</td>
</tr>
<tr>
<td><strong>REGION 4</strong></td>
<td>1</td>
<td>1</td>
<td>44136700</td>
</tr>
<tr>
<td><strong>REGION 5</strong></td>
<td>1</td>
<td>1</td>
<td>45194561</td>
</tr>
<tr>
<td><strong>TOTALS</strong></td>
<td>5</td>
<td>5</td>
<td>226944193</td>
</tr>
<tr>
<td>ELAPSED EXECUTION</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>REGION 1</strong></td>
<td>1</td>
<td>1</td>
<td>270851515</td>
</tr>
<tr>
<td><strong>REGION 2</strong></td>
<td>1</td>
<td>1</td>
<td>275004759</td>
</tr>
<tr>
<td><strong>REGION 3</strong></td>
<td>1</td>
<td>1</td>
<td>478134880</td>
</tr>
<tr>
<td><strong>REGION 4</strong></td>
<td>1</td>
<td>1</td>
<td>548731226</td>
</tr>
<tr>
<td><strong>REGION 5</strong></td>
<td>1</td>
<td>1</td>
<td>547673367</td>
</tr>
<tr>
<td><strong>TOTALS</strong></td>
<td>5</td>
<td>5</td>
<td>2120395747</td>
</tr>
<tr>
<td>DL/I CALLS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>REGION 1</strong></td>
<td>1</td>
<td>2762</td>
<td>19399174</td>
</tr>
<tr>
<td><strong>REGION 2</strong></td>
<td>2</td>
<td>14848</td>
<td>22430506</td>
</tr>
<tr>
<td><strong>REGION 3</strong></td>
<td>3</td>
<td>1849</td>
<td>12210030</td>
</tr>
<tr>
<td><strong>REGION 4</strong></td>
<td>4</td>
<td>31152</td>
<td>82452077</td>
</tr>
<tr>
<td><strong>REGION 5</strong></td>
<td>5</td>
<td>98838</td>
<td>40940175</td>
</tr>
<tr>
<td><strong>TOTALS</strong></td>
<td>149449</td>
<td>177431962</td>
<td>1187</td>
</tr>
<tr>
<td>IDLE FOR INTENT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHECKPOINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The “Elapsed Time Mean” on each report section is one of the important indicators of system performance. Of course, the percentage of the Region Occupancy is another key indicator.

These examples are only several of the types of performance data that can be recorded and displayed by using IMS and DB Monitors.

For detailed information about using these IMS features, see the following sources:

- *IMS Version 12 Operations and Automation, SC19-3018*
  
  See Chapter 3, Monitoring IMS.

- *IMS Version 12 Database Utilities, SC19-3014*
  
  See Chapter 31, Database Monitor Report Print Utility (DFSUTR30).

- *IMS Version 12 System Utilities, SC19-3023*
  
  See Chapter 14, IMS Monitor Report Print Utility (DFSUTR20).

- *IMS Version 12: System Administration, SC19-3020*
  
  See the following chapters:
  - Chapter 24, Collecting and interpreting IMS monitoring data
  - Chapter 48, DB Monitor Reports
  - Chapter 49, IMS Monitor Reports
  - Chapter 50, IMS Monitor Reports for DBCTL
  - Chapter 51, IMS Monitor reports for DCCTL

### 1.3 File Select and Formatting Print utility (DFSERA10)

You can use File Select and Formatting Print utility (DFSERA10), provided in the base IMS product, to examine and display data from IMS log data sets. Control statements define the input and output options, selection ranges, and various field and record selection criteria:

- Print or copy an entire log data set.
- Print or copy from multiple log data sets based upon control statement input.
- Print Operations Manager (OM) log records.
- Select and print log records on the basis of sequential position in the data set.
- Select and print external trace data sets.
- Select and print log records based upon data contained within the record itself, such as the contents of a time, date, or identification field.
- Allow modules to special process any selected log records.
When using DFSERA10, control flow for the program passes through two major phases:

- Control statement processing
  
  Construction of record test and selection parameters takes place and control statement errors can be diagnosed.

- Record selection and output processing
  
  Input data is read, analyzed, and compared with the selection parameters to determine the applicability of the record for output.

Output data can be formatted and printed on the SYSPRINT data set, copied to a specified data set unchanged, or both. Data to be printed is formatted into 32-byte segments and displayed in both hexadecimal and EBCDIC forms, with the hexadecimal relative offset value preceding each segment.

Example 1-3 shows the required JCL and control statements to print or copy all log records from an IMS log data set.

**Example 1-3  DFSERA10 Example JCL**

```
//EXAMPLE1 JOB
//*
// EXEC PGM=DFSERA10
//STEPLIB DD DISP=SHR,DSN=IMS.SDFSRESL
//*
//SYSUT1 DD DISP=(OLD,KEEP),DSN=IMSLOG,
// UNIT=TAPE,VOL=SER=123456
//SYSUT4 DD DISP=(NEW,PASS),DSN=EXAMPLE1.COPY1,
// UNIT=SYSDA,SPACE=(CYL,(3,1)),
// VOL=SER=IMSPAC
//*
//SYSPRINT DD SYSOUT=A
//*
//SYSIN DD *
*-----------------------------------------------------------------*
* CONTROL STATEMENT : DEFAULTS *
* INPUT = SYSUT1 *
* OUTPUT = SYSPRINT *
* SELECTION QUALIFIERS : *
* 1. DEFAULT = ALL INPUT RECORDS *
*-----------------------------------------------------------------*
OPTION PRINT
END
*-----------------------------------------------------------------*
* CONTROL STATEMENT : DEFAULTS *
* INPUT = SYSUT1 *
* OUTPUT = SYSUT4 *
* SELECTION QUALIFIERS : *
* 1. DEFAULT = ALL INPUT RECORDS *
*-----------------------------------------------------------------*
OPTION COPY
END
/*
```

For detailed information regarding the File Select and Formatting Print utility (DFSERA10), including detailed JCL specifications, utility control statements, and more example JCL, see Chapter 13, File Select and Formatting Print Utility (DFSERA10) of *IMS Version 12 System Utilities*, SC19-3023.
1.4 Log Transaction Analysis utility (DFSLT0A)

You can use another utility that is provided with the base IMS product, the Log Transaction Analysis utility (DFSLT0A), in IMS database/data communication (DB/DC) or Data Communications Control (DCCTL) environments, to collect information about individual transactions based on records on the system log. Many events are tabulated in the Log Analysis report that is produced by this utility, including total response time, time on the input queue, processing time, and time on the output queue.

This utility collects a variety of information regarding your IMS transactions:

- Transaction identification
- Source
- Transaction program name
- Dependent region
- Priority
- Class of the transaction

Canceled messages: Canceled messages are not used.

For the transactions selected, DFSLT0A also accumulates time information:

- The time that each transaction is received
- The time of the message get unique (GU) call
- The time that the transaction processing ends
- The time that the output message is placed on the output queue
- The time that the output message starts to the terminal

From these times, DFSLT0A calculates the following values:

- Total response time
- Time on the input queue
- Processing time
- Time on the output queue

By using this utility, you can create an output data set, in system log format, that is a copy of all or part of the input system logs. By having a copy of the system log, you can monitor system activity without impacting the use of the OLDS for recovery.

You can use this information to help find bottlenecks in transaction processing within the system or to evaluate whether transaction classes have been assigned correctly to ensure proper transaction scheduling.

Example 1-4 shows JCL for that Log Transaction Analysis utility.

Example 1-4  DFSLT0A sample JCL

```plaintext
//DFSILTA4 JOB 'TERRY',CLASS=K,MSGCLASS=A,MSGLEVEL=(1,1), JOB00386
// REGION=0M,TIME=1440,
// USER=USRT001,PASSWORD=
/*ROUTE PRINT THISCPU/IMSTST45
//JOBLIB DD DSN=IMSTESTL.TNUC0,DISP=SHR
// DD DSN=IMSBLD.I1ORTS17.CRESLIB,DISP=SHR
//ILTA EXEC PGM=DFSILTA0
//HEADING DD SYSOUT=A
//PRINTER DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//REPORT DD DUMMY
```
//TITLE DD *
//*
//* LOG INPUT FOLLOWS...
//*
//LOGIN0A DD DSN=IMSTESTL.SLDSP.IMSA.TYTY,DISP=SHR,
// UNIT=SYSDA,VOL=SER=DSHR03
//LOGIN01 DD DSN=IMSTESTL.SLDSP.IMS1.TYTY,DISP=SHR,
// UNIT=SYSDA,VOL=SER=DSHR03
//LOGIN02 DD DSN=IMSTESTL.SLDSP.IMS2.TYTY,DISP=SHR,
// UNIT=SYSDA,VOL=SER=DSHR03
//LOGIN03 DD DSN=IMSTESTL.SLDSP.IMS3.TYTY,DISP=SHR,
// UNIT=SYSDA,VOL=SER=DSHR03

Figure 1-1 shows the report output for a Log transaction Analysis utility run of the JCL in Example 1-4 on page 11.

For detailed information regarding the Log Transaction Analysis utility (DFSILTA0), including procedures, JCL specifications, and detailed utility use description, see the following sources:

- *IMS Version 12 System Utilities*, SC19-3023
  See Chapter 15, Log Transaction Analysis utility (DFSILTA0).
- *IMS Version 12: System Administration*, SC19-3020
  See Chapter 53, Statistical analysis, log-transaction reports.
1.5 Statistical Analysis utility (DFSISTS0)

The IMS Statistical Analysis utility can be used in an IMS DB/DC or DCCTL environment to produce various summary style reports. From these reports, you can gain real insight into your log data to obtain statistics, measuring actual transaction loads and system response times.

The Statistical Analysis utility can be used to analyze data on any of the IMS logs, except for those executing in a batch region. By using DFSISTS0, you can produce reports for the entire log data set, or a portion of the log data set created by the Log Transaction Analysis utility, IMS Performance Analyzer for z/OS, or IMS Problem Investigator for z/OS. These tools are described in Chapter 3, “Performance monitoring tools” on page 43.

By using DFSISTS0, you can produce the following report sets:

- Telecommunication line and terminal; distributed traffic over 24-hour day
- Transaction; distributed activity over 24-hour day
- Transaction response
- Messages queued but not sent; listing by destination and by transaction code
- Program-to-program messages; listing by destination and by transaction code
- Application accounting
- IMS accounting

These reports are dependent on your installation’s current logging process and are also subject to the following restrictions:

- Common Queue Server (CQS) logs cannot be used as input because CQS log records have a different format from IMS log records.
- The utility works only with input log data sets that are created by the same release of IMS as the utility release level.

Example 1-5 shows the JCL for execution of the Statistical Analysis utility.

Example 1-5  Execution of the Statistical Analysis utility

//STATS JOB 1,NAME,MSGCLASS=A,MSGLEVEL=1,PRTY=8
//STEPLIB DD DSNAME=IMS.SDFSRESL,DISP=SHR
//*
//ST1 EXEC PGM=DFSISTS0
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSUDUMP DD SYSOUT=A
//LOGIN DD DSN=IMSLOG,DISP=OLD,
  // UNIT=TAPE,VOL=SER=LOGTAP
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5),,CONTIG)
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5),,CONTIG)
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5),,CONTIG)
//SORTOUT DD DUMMY
//PRINTDCB DD SYSOUT=A
//IMISLOGP DD SYSOUT=A
//SYSIN DD *
TRANS CODE=(ALL,I,O)
NON PRINT=HEX
/*
Figure 1-2 shows sample transaction report output from the Statistical Analysis utility.

<table>
<thead>
<tr>
<th>TRANSACTION REPORT</th>
<th>DATE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE</td>
<td>TOTAL SIZE</td>
<td>DISTRIBUTION</td>
</tr>
<tr>
<td>YGCMSS</td>
<td>1,524</td>
<td>0 0 0 0 1 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGGDIS</td>
<td>1,253</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGDECHC</td>
<td>2,888</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGDECHV</td>
<td>3,040</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGGDIS</td>
<td>2,136</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGGDIS</td>
<td>6,728</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>1,718</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>6,888</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>8,166</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>1,753</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>1,718</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>62,741</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>7,960</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>4,912</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>5,475</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>1,825</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>5,879</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>64,425</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>10,398</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>1,239</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>674</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>412</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>681</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>1,596</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>3,602</td>
<td>1 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>7,154</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>YGCDIS</td>
<td>676</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
</tbody>
</table>

For detailed information regarding Statistical Analysis utility (DFSISTS0), including log record descriptions, report descriptions, JCL specifications, utility control statement descriptions, and several more sample reports, see Chapter 17, Statistical Analysis utility (DFSISTS0) of IMS Version 12 System Utilities, SC19-3023.

1.6 Fast Path Log Analysis utility (DBFULTA0)

The Fast Path Log Analysis utility (DBFULTA0) can be used to prepare statistical reports for Fast Path, based on data that is recorded on the IMS system log. These reports are useful for system installation, tuning, and troubleshooting. This utility is not related to IMS Monitor or the Log Transaction Analysis utility.

This offline utility produces three data sets, one of which contains the following formatted reports:

- Detail Listing of Exception Transactions
- Summary of Exception Detail by Transaction Code for IMS Fast Path (IFP) Regions
- Overall Summary of Transit Times by Transaction Code for IFP Regions
- Overall Summary of Resource Usage and Contentions for All Transaction Codes and PSBs
- Summary of Region Occupancy for IFP Regions by program specification table (PST)
- Summary of Fast Path Virtual Storage Option (VSO) Activity
- Recapitulation of the Analysis
Example 1-6 shows JCL to run Fast Path Log Analysis utility.

Example 1-6  DBFULTA0 sample JCL

```jcl
//DBFULTA0 JOB MSGCLASS=X,NOTIFY=&SYSUID,REGION=64M,MSGLEVEL=(1,1)
//FPLA1 EXEC PGM=DBFULTA0
//STEPLIB DD DISP=SHR,DSN=IMS.SDFSRESL
//SYSPRINT DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//SYSUT1 DD DUMMY,DISP=(,PASS),DSN=&TOTAL,
// UNIT=SYSALLDA,
// DCB=BLKSIZE=7208,
// SPACE=(CYL,(2,2))
//SYSUT2 DD DUMMY,DISP=(,PASS),DSN=&EXCEP,
// UNIT=SYSALLDA,
// DCB=BLKSIZE=7208,
// SPACE=(CYL,(2,2))
//FPLA1.LOGTAPE DD DISP=SHR,DSN=IMSLOG
//*
//FPLA1.SYSIN DD *
NON-MESSAGE
TT(*)=1.0 
MAXDETAIl=1000
CALLS
BUFFER
VSO
FPBP64
//*
```

Example 1-7 shows a report of Fast Path Log Analysis utility.

Example 1-7  DBFULTA0 report output: Detail Listing of Exception Transactions

<table>
<thead>
<tr>
<th>SEQ</th>
<th>TRancode</th>
<th>Sync Point</th>
<th>S Routing</th>
<th>Logical PST</th>
<th>Queue</th>
<th>Transit Times (MSEC)</th>
<th>-Out-</th>
<th>-Deb-</th>
<th>-AdS-</th>
<th>-VSO-</th>
<th>MSDB</th>
<th>BUF Contentsions</th>
<th>R P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IVTFD</td>
<td>14:21:50.66</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 120 192 312 0.0</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>IVTFD</td>
<td>14:21:53.39</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 1 1 0.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>IVTFD</td>
<td>14:21:54.96</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 114 115 0.0</td>
<td>1 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>IVTFD</td>
<td>14:21:56.33</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 1 1 0.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>IVTFD</td>
<td>14:21:58.25</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 0 1 1</td>
<td>0.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>IVTFD</td>
<td>14:22:00.19</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 0 1 1</td>
<td>0.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>IVTFD</td>
<td>14:22:05.03</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 1 9</td>
<td>0.0</td>
<td>2 1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>IVTFD</td>
<td>14:22:10.18</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 1 2</td>
<td>3.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>IVTFD</td>
<td>14:22:13.74</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 0 1 1</td>
<td>0.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>IVTFD</td>
<td>14:22:27.86</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 0 0 1</td>
<td>0.0</td>
<td>1 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>IVTFM</td>
<td>14:23:34.54</td>
<td>IVTFM</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 1 1</td>
<td>0.0</td>
<td>0 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>IVTFM</td>
<td>14:23:36.11</td>
<td>IVTFM</td>
<td>SASAKI1</td>
<td>0</td>
<td>0 0 1</td>
<td>1.0</td>
<td>0 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 0 0 0 0 0 I</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 2. Performance concepts and monitoring methodology

Performance problems are erratic in nature and generally occur when you least expect them. This situation applies to any kind of system; in this book, we discuss the performance of IMS systems. In an environment that has a combination of both internal and external variables, what must we do to be sure we apply good practices and that we are proactive? A number of techniques are available, but the most important is the ability to monitor, profile, track, and trend transactions.

The purpose of this chapter is to highlight areas where performance problems can occur and give you a reference point in identifying and resolving performance problems.

We discuss the lifecycle of an IMS transaction by describing the flow of a message from its inception at a terminal through to its processing by an application, and finally its reply back to the terminal. We provide a brief overview of the performance challenges and then describe IMS full-function, IMS Fast Path, and DBCTL message flows and open transaction interactions with the IMS system. The message flow depicts the life of a transaction through IMS and highlights areas where performance problems can occur. We then describe monitoring methodology and procedures.

This chapter contains the following topics:
- Performance challenges
- Events for full-function messages
- Events for Fast Path messages
- Events for DBCTL
- Common log records produced for transaction flows
- Logging in a single IMS system
- Open transaction flow
- Monitoring methodology
- Transaction flow in DB/DC and DCCTL environments
- Monitoring procedures in an open transaction environment
2.1 Performance challenges

The performance of an IMS system is directly related to a number of internal variables. These variables can be found in the z/OS operating system, in IMS Transaction Manager (TM), in IMS database manager (DM), in the application, or in the hardware. External variables include the network and the physical infrastructure of your network. These external variables are mostly out of our control, although they are integral in ensuring respectable response times. An understanding of your network architecture is paramount in diagnosing network-related response-time issues.

IMS is an event-driven system and events are represented internally by event control blocks (ECB). Each event or a multitude of events can result in performance bottlenecks. We provide a generic method for monitoring and viewing the IMS system to identify problems as events occur.

Several typical IMS performance and availability challenges are as follows:

- Poor IMS response time, transaction queuing, and processing bottlenecks
  - Queued IMS transactions
  - IMS scheduling delays
  - IMS application performance or system performance bottlenecks

- IMS connection bottlenecks
  - CICS/DBCTL connection bottlenecks
  - Network delays
  - Delays related to IMS Connect, Open Transaction Manager Access (OTMA), Open Database Manager (ODBM), Advanced Program-to-Program Communication (APPC), and external driving systems

- IMS database and subsystem delays
  - High I/O
  - Poor buffer pool performance
  - IMS lock and latch conflicts

- External subsystem delays, such as DB2, which can elongate IMS application time
  - DB2 thread connection issues
  - DB2 SQL processing delays
  - DB2 I/O and buffer pool performance issues
  - DB2 lock conflicts

2.2 Events for full-function messages

In this section we describe how to identify possible performance problems for IMS full-function transaction flows, with a view to identifying performance problems, as we step through the various events. Table 2-1 on page 19 shows the list of events for full-function messages. These steps are described in more detail after the table.
The steps are as follows:

1. **Message arrives in IMS**
   
   The message arrival event begins when the message is placed by either IBM VTAM® in an IMS receive any buffer or transported through IMS Connect to IMS Open Transaction Manager Access (OTMA). Either way, the message is moved to a buffer acquired in the high I/O pool (HIOP) where it is edited by Message Format Service (MFS) routines, IMS basic edit, or intersystem communication (ISC) edit.

2. **Message queuing**
   
   Message queuing events require the message in the standard IMS format:
   
   \[11zz|\text{trancode}|\text{data}\]
   
   It is allocated to a position on one of the message queue data sets. The message is moved to the message queue pool and enqueued on the scheduler message block (SMB).

3. **Message scheduling**
   
   Message scheduling events are dependent on a number of variables. The transaction definition is the starting point for any scheduling issues. The following variables affect transaction scheduling:
   
   - SCHDTYP on APPLCTN macro
   - Serial or parallel
   - PROCLIM
   - PARLIM
   - PRIORITY
   - MAXRGN
   - CLASS of transaction
   - SCHD
   
   The next step is to identify any pool intent failures. IMS requires the program specification block (PSB) directory (PDIR) and data management block (DMB) directory (DDIR) for the scheduled PSB to be available. The PSB and DMB are loaded when the transaction first...

---

### Table 2-1 Event list for full-function messages

<table>
<thead>
<tr>
<th>Step</th>
<th>Event</th>
<th>Activity</th>
<th>Problem identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Message arrives in IMS</td>
<td>Message Format Service (MFS) routines, basic edit or intersystem communication (ISC) edit</td>
<td>RECA, high I/O pool (HIOP), CIOP, and MFBP pool shortages evident</td>
</tr>
<tr>
<td>2</td>
<td>Message queuing</td>
<td>Message queued to scheduler message block (SMB)</td>
<td>QBUF shortage evident</td>
</tr>
<tr>
<td>3</td>
<td>Message scheduling</td>
<td>Schedule message in dependent region</td>
<td>Transaction queuing or pool intent failures</td>
</tr>
<tr>
<td>4</td>
<td>Scheduling-end to first DL/I call</td>
<td>Load programs, subroutines, and initialize working storage</td>
<td>Appears as high CPU and elapsed time</td>
</tr>
<tr>
<td>5</td>
<td>Program elapsed time</td>
<td>Application program invoked and DL/I calls performed</td>
<td>High elapsed times, as a result of application, database, WLM, I/O subsystem, or system issues</td>
</tr>
<tr>
<td>6</td>
<td>Sync point</td>
<td>Phase one and two commit</td>
<td>High elapsed times as a result of WADS, OLDS, or I/O subsystem delays</td>
</tr>
<tr>
<td>7</td>
<td>Message output</td>
<td>Send output to destination</td>
<td>HIOP shortage, or network delays before message is dequeued</td>
</tr>
</tbody>
</table>
executes. If not resident, then ACBLIB I/Os are required to load the PSBs and DMBs. The PSB, PSB work pool (PSBW), and DMB pools must be large enough to accommodate all PSBs and DMBs, if possible, and must be monitored regularly.

4. Scheduling-end to first DL/I call

Scheduling-end to first DL/I events include program or subroutine load and any program and working storage initialization being performed by the application. You must review long time periods, when evident in this event, from a program-load or application-design perspective.

5. Program elapsed time

Program elapsed time events are composed of many variables that can influence the elapsed time of a transaction. Elapsed time is measured from the time that the application is scheduled into the IMS regions until synchronization point (sync point) processing.

The following variables can affect elapsed time:

- Program execution time
- Time required to complete the DL/I calls, and consists of two components:
  - IWAIT time, which is the time for DL/I to acquire the database record
  - NOT-IWAIT time, which is the time DL/I spends in code execution
- I/O subsystem delays as a result of DASD response times
- System waits, over which IMS has no influence

6. Sync point processing

Sync point processing events for full-function (FF) require that all I/Os actually take place. IMS follows the standard two-phase commit process. In an IMS only workload, this two-phase commit is actually a call to two separate modules. Generally, a two-phase commit is between IMS and DB2 and follows the standard two-phase commit process. In IMS terms, a get unique (GU) call to the IOPCB to retrieve the next message implies sync point.

7. Message output

Message output events imply termination of the program so that output messages are ready to be sent to their final destination. The HIOP usage is important in identifying possible performance-related issues in this area.

2.3 Events for Fast Path messages

Table 2-2 on page 21 is specific to messages that are being processed through Fast Path expedited message handling (EMH). Certain overlaps exist between Fast Path and full-function, and are also mentioned. These steps are described in more detail after the table.
The steps are as follows:

1. **Message arrives in IMS**
   
   Message arrival event for Fast Path is essentially the same as for full-function.

2. **Fast Path EMH queuing**
   
   The Fast Path EMH event requires IMS to make a decision on whether the message is Fast Path potential (FPP) or Fast Path exclusive (FPE). If FPP, then DBFHAGU0 is called to decide whether the message needs to be processed by full-function IMS or whether a routing code needs to be assigned to make it Fast Path. If FPE, then IMS queues the message from the BALG. The BALG becomes the queue anchor point.

3. **Fast Path EMH scheduling**
   
   The Fast Path EMH scheduling event has no complex-priority scheduling schema when compared to FF. Messages are processed by using a BALG on a first-in-first-out (FIFO) basis and scheduled into an available IMS Fast Path (IFP) region. All IFP regions are always in wait-for-input (WFI) mode so the overhead of program-load is avoided.

4. **Program elapsed time**
   
   The program elapsed time event involves the same principle as mentioned for FF, with the exception that any Fast Path DEDB or MSDB reads are performed under the control of the control region instead of the DL/I separate address space.

5. **Sync point processing**
   
   Fast Path sync point processing differs significantly from full-function sync point processing. A key element of Fast Path sync point processing is that all Fast Path writes happen asynchronously through output threads (OTHR). Two-phase commit process is still used by Fast Path, but it operates differently. IMS Fast Path does not have to wait for the I/O to be physically written. If phase one processing is unsuccessful, the buffers are thrown away, and the transaction is either rescheduled or thrown away in its entirety, depending on the nature of the problem.

---

**Table 2-2  Event list for Fast Path messages**

<table>
<thead>
<tr>
<th>Step</th>
<th>Event</th>
<th>Activity</th>
<th>Problem identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Message arrives into IMS</td>
<td>MFS, basic edit or ISC edit</td>
<td>RECA, MFBP, HIOP, or CIOP shortages evident</td>
</tr>
<tr>
<td>2</td>
<td>Fast Path expedited message handling (EMH) queuing</td>
<td>Determine if Fast Path (FP) potential or exclusive</td>
<td>EMHB pool shortages</td>
</tr>
<tr>
<td>3</td>
<td>Fast Path EMH scheduling</td>
<td>First-in-first-out scheduling by balancing group (BALG)</td>
<td>Queuing on BALG</td>
</tr>
<tr>
<td>4</td>
<td>Program elapsed time</td>
<td>Application program invoked and DL/I calls performed</td>
<td>High elapsed times as a result of application, database, WLM, I/O subsystem, or system issues</td>
</tr>
<tr>
<td>5</td>
<td>Sync point</td>
<td>Phase one and two commit</td>
<td>Output thread shortage (OTHR)</td>
</tr>
<tr>
<td>6</td>
<td>Message output</td>
<td>Send output to destination</td>
<td>EMHB pool shortage, or network delays before message is dequeued</td>
</tr>
</tbody>
</table>
2.4 Events for DBCTL

Table 2-3 is specific to messages being processed through DBCTL. These steps are described in more detail after the table.

Table 2-3 Event list for DBCTL

<table>
<thead>
<tr>
<th>Step</th>
<th>Event</th>
<th>Activity</th>
<th>Problem identification, if relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Message arrives into CICS TOR/AOR</td>
<td>Format message and call/link related application programs</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Application gets control and issues schedule request</td>
<td>EXEC DL/I command</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PSB scheduling</td>
<td>DBT thread TCB is given control. Recovery token establishes</td>
<td>Insufficient PSB, DMB, DB work, PSB work, or EPCB pool space can cause scheduling delays or failures</td>
</tr>
<tr>
<td>4</td>
<td>Program elapsed time</td>
<td>Application program invoked and DL/I calls performed</td>
<td>High elapsed times as a result of application, database, WLM, I/O subsystem, or system issues</td>
</tr>
<tr>
<td>5</td>
<td>Sync point</td>
<td>IMS hardens the log data and writes updated full-function database blocks. Fast Path database updates are asynchronous</td>
<td>WADS and OLDS activity; database DASD I/O</td>
</tr>
<tr>
<td>6</td>
<td>Application issues terminate PSB call</td>
<td>IMS frees scheduled resources</td>
<td></td>
</tr>
</tbody>
</table>

The steps are as follows:

1. Message arrives into CICS TOR/AOR
   Message arrives at CICS TOR first and unless also acting as an AOR, the message can be routed to a CICS AOR.

2. Application gets control and issues schedule request
   Application issues DL/I SCHED call to reserve IMS resources.

3. PSB scheduling
   In most cases, one PSB is scheduled for one CICS transaction. IMS checks whether the PSB and related databases are available. It then allocates space for the necessary control blocks in various scheduling pools and loads the required blocks.

4. Program elapsed time
   The program elapsed time event is composed of many variables that can influence the elapsed time of a transaction. Elapsed time is measured from the time the PSB is scheduled until the PSB is terminated by the application. The following variables can affect elapsed time:
   - Program execution time
   - Time required to complete the DL/I calls, and consists of two components:
     - IWAIT time, which is the time DL/I has to acquire the database record
     - NOT-IWAIT time, which is the time DL/I spends in code execution
   - I/O subsystem delays as a result of DASD response times
   - System waits, which IMS has no influence over
5. Sync point processing
   IMS hardens the log data and writes updated full-function database blocks. Fast Path
   writes are performed asynchronously to sync point processing.

6. Application issues terminate PSB call
   PSB terminates and scheduling resources are freed.

### 2.5 Common log records produced for transaction flows

Table 2-4 lists the most commonly used log records that are produced during the lifecycle of
an IMS transaction. Identifying these log records provides the ability to diagnose
performance-related problems.

<table>
<thead>
<tr>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’01’</td>
<td>Message received from a terminal</td>
</tr>
<tr>
<td>X’03’</td>
<td>Message received from DL/I</td>
</tr>
<tr>
<td>X’07’</td>
<td>An application program was terminated</td>
</tr>
<tr>
<td>X’08’</td>
<td>An application program was scheduled</td>
</tr>
<tr>
<td>X’31’</td>
<td>Message queue GU</td>
</tr>
<tr>
<td>X’32’</td>
<td>Message queue reject</td>
</tr>
<tr>
<td>X’33’</td>
<td>Message queue free</td>
</tr>
<tr>
<td>X’34’</td>
<td>Message cancel</td>
</tr>
<tr>
<td>X’35’</td>
<td>Message queue enqueue</td>
</tr>
<tr>
<td>X’36’</td>
<td>Message queue dequeue</td>
</tr>
<tr>
<td>X’37’</td>
<td>Sync point record</td>
</tr>
<tr>
<td>X’38’</td>
<td>Message after abend</td>
</tr>
<tr>
<td>X’50’</td>
<td>Database undo/redo record</td>
</tr>
<tr>
<td>X’5901’</td>
<td>Fast Path input</td>
</tr>
<tr>
<td>X’5903’</td>
<td>Fast Path output</td>
</tr>
<tr>
<td>X’5936’</td>
<td>Fast Path dequeue</td>
</tr>
<tr>
<td>X’5937’</td>
<td>Fast Path sync point</td>
</tr>
<tr>
<td>X’5938’</td>
<td>Fast Path abend</td>
</tr>
<tr>
<td>X’5950’</td>
<td>Fast Path database update</td>
</tr>
<tr>
<td>X’5953’</td>
<td>Fast Path sequential dependent (SDEP) write</td>
</tr>
</tbody>
</table>
2.6 Logging in a single IMS system

Figure 2-1 shows the transaction processing events, and the associated log records that are related to each event in a single IMS system. This figure shows the associated event processing and the IMS logger (ILOG) functions and associated log records that you can use to review the performance characteristics and flow of an IMS transaction.

![Figure 2-1 Events and associated log records for an IMS transaction]

Figure 2-2 on page 25 shows similar transaction events and associated log records that use IMS Problem Investigator tool. This figure highlights the power and simplicity that the PI tool can provide in viewing the overall flow of an IMS transaction. IMS Problem Investigator is further described in 3.2, “IMS Problem Investigator for z/OS, Version 2 Release 3” on page 81.
Figure 2-2  IMS Problem Investigator: Events and log records for an IMS transaction in PI

Figure 2-3 shows the transaction processing events, and the associated log records that are related to each event in an IMS shared-queues environment. It shows the associated event processing and the ILOG functions and associated log records that you can use to review the performance characteristics and flow of an IMS transaction.

Figure 2-3  Events and associated log records for transaction in shared queues environment
IBM Parallel Sysplex® and data sharing, and shared queues are not specifically discussed in this chapter and are addressed in Chapter 10, “Parallel Sysplex considerations” on page 353. That chapter also has a brief overview of the transaction flow and log records in a shared-queue environment compared to a single system.

2.7 Open transaction flow

IMS offers a variety of solutions for accessing IMS applications and data. There are mainly two patterns of connectivity from the application in an open environment: transaction and database. See Figure 2-4.

The patterns of connectivity are as follows:

- **Transaction connectivity**
  This way is how a client application in an open environment issues an IMS transaction, IMS application processes, and sends a reply to the client application. Conversely, another way referred to as *callout*, in which IMS application can call out the open application. See Figure 2-5 on page 27.

  This connectivity uses Open Transaction Manager Access (OTMA) as an interface for sending and receiving transactions and data from IMS. The commonly used OTMA clients are IMS Connect (ICON) and MQ-IMS bridge.
Database connectivity

This way is how a client application in open environment issues IMS DB processing requests and receives replies. The business logic is processed in the open application. See Figure 2-6.

This connectivity uses Open Database Access (ODBA), database resource adapter (DRA), or both, as an interface to access databases managed by IMS DB manager. Examples of ODBA clients are IBM WebSphere® Application Server and IBM InfoSphere® Classic Federation Server. The Open Database Manager (ODBM) is also used as a client.

When you are using ICON, you can collect the event records with the function of IMS Connect Extensions for z/OS (see 3.3, “IMS Connect Extensions for z/OS, Version 2 Release 3” on page 116).
There are two types of event records:

- **Connect status event**

  A Connect status event identifies a change in the status of your IMS Connect system, for example, a resource (data store, T MEMBER) becoming available or unavailable, or a socket becoming accepted for input by a port task. Connect status events are typically not related to the processing of input messages, but can affect their processing.

  Connect status event records are identified by the EV NT constant event key.

- **Message related event**

  Message related event records identify an event in the processing of an incoming message request. Message related event records have a store clock (STCK) token event key. For non-persistent sockets, each incoming message is assigned a unique event key, and every event that is associated with the processing of the message has the same event key.

  For persistent sockets, all incoming messages are assigned the same event key. All events that are associated with the processing of all messages for the duration of the socket have the same event key.

The following list introduces the records that are involved in an IMS Connect flow for sync level none and sync level confirm transactions:

- **Event flow: Commit mode 1, sync level none**

  Example 2-1 shows the event flow for a single commit mode 1, sync level none transaction. It shows the hexadecimal code and the description of the events records.

  **Example 2-1 Sample event flow: Commit mode 1, sync level none**

  \[
  \begin{align*}
  3C & \text{ Prepare Read Socket} & \text{<= Incoming message from client} \\
  49 & \text{ Read Socket} & \\
  3D & \text{ Message Exit called for READ} & \\
  3E & \text{ Message Exit return for READ} & \\
  41 & \text{ Message sent to OTMA} & \text{<= Sent to OTMA for processing} \\
  42 & \text{ Message received from OTMA} & \\
  3D & \text{ Message Exit called for XMIT} & \\
  3E & \text{ Message Exit return for XMIT} & \\
  4A & \text{ Write Socket} & \text{<= Response sent back to client} \\
  0C & \text{ Begin Close Socket} & \text{<= Non-persistent Socket is closed} \\
  0D & \text{ End Close Socket} & \\
  48 & \text{ Trigger event CLOS} & \text{<= Connect has finished processing message} \\
  \end{align*}
  \]

- **Event flow: Commit mode 1, sync level confirm**

  Example 2-2 shows the event flow for a single commit mode 1, sync level confirm transaction. The difference from the previous example is that the client acknowledgement is displayed in the flow.

  **Example 2-2 Sample event flow: Commit mode 1, sync level confirm**

  \[
  \begin{align*}
  3C & \text{ Prepare Read Socket} & \text{<= Incoming message from client} \\
  49 & \text{ Read Socket} & \\
  3D & \text{ Message Exit called for READ} & \\
  3E & \text{ Message Exit return for READ} & \\
  41 & \text{ Message sent to OTMA} & \text{<= Sent to OTMA for processing} \\
  42 & \text{ Message received from OTMA} & \\
  3D & \text{ Message Exit called for XMIT} & \\
  3E & \text{ Message Exit return for XMIT} & \\
  \end{align*}
  \]
4A Write Socket <= Response sent back to client
49 Read Socket <= ACK received from Client
3D Message Exit called for READ
3E Message Exit return for READ
41 Message sent to OTMA <= ACK sent to OTMA
42 Message received from OTMA
46 De-allocate Session
3D Message Exit called for XMIT
3E Message Exit return for XMIT
4A Write Socket <= De-alloc Response sent back to client
0C Begin Close Socket <= Non-persistent Socket is closed
0D End Close Socket
48 Trigger event CLOS <= Connect has finished processing message

Example 2-3 shows the event flow of processing through ODBM.

Example 2-3 Sample event flow: ODBM

<table>
<thead>
<tr>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>3C Prepare READ Socket</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>5B DRDA 1041 EXCSAT-Exchange Server Attributes</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>5B DRDA 106D ACCSEC-Access Security</td>
</tr>
<tr>
<td>5C DRDA 1443 EXCSATRD-Server Attributes Reply Data</td>
</tr>
<tr>
<td>4A WRITE Socket</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>5B DRDA 106E SECKKH-Security Check</td>
</tr>
<tr>
<td>63 ODBM Security Exit called</td>
</tr>
<tr>
<td>64 ODBM Security Exit returned</td>
</tr>
<tr>
<td>5C DRDA 1219 SECKMKRM-Security Check Reply Message</td>
</tr>
<tr>
<td>4A WRITE Socket</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>5B DRDA 2001 ACCRDB-Access RDB</td>
</tr>
<tr>
<td>5D ODBM begin Allocate PSB (APSB) Program=AUTPSB11</td>
</tr>
<tr>
<td>61 ODBM Routing Exit called</td>
</tr>
<tr>
<td>62 ODBM Routing Exit returned</td>
</tr>
<tr>
<td>AA ODBM Trace: Message sent to ODBM</td>
</tr>
<tr>
<td>69 Message sent to ODBM</td>
</tr>
<tr>
<td>AA ODBM Trace: Message received from ODBM</td>
</tr>
<tr>
<td>6A Message received from ODBM</td>
</tr>
<tr>
<td>5E ODBM end Allocate PSB (APSB) Program=AUTPSB11</td>
</tr>
<tr>
<td>5C DRDA 2201 ACCRDBRM-Access RDB Reply Message</td>
</tr>
<tr>
<td>4A WRITE Socket</td>
</tr>
<tr>
<td>48 Trigger Event for ODBMMSG</td>
</tr>
<tr>
<td>3C Prepare READ Socket</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>5B DRDA 200C OPOQRY-Open Query</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>49 READ Socket</td>
</tr>
<tr>
<td>5B DRDA CC05 DLIFUNC-DL/I function</td>
</tr>
</tbody>
</table>
2.8 Monitoring methodology

Monitoring is the collection and interpretation of IMS data. Monitoring should be an ongoing task for the following reasons:

- Monitoring helps you establish base profiles, workload statistics, and data for capacity planning and prediction.
- Monitoring gives early warning and comparative data to help you prevent performance problems.
- Monitoring validates tuning you have done in response to a performance problem and ascertains the effectiveness of that tuning.

An historical base and conclusions from continuous monitoring provide a good start to answering user complaints and an initial direction for tuning projects.

2.8.1 Establishing monitoring strategies

Several types of monitoring strategies are available:

- Summarize actual workload for the entire online execution. This strategy can include both continuous and periodic tracking. You can track total workload or selected representative transactions.
- Take sample snapshots at peak loads and under normal conditions. Monitoring the peak periods is always useful for two reasons:
  - Bottlenecks and response time problems are more pronounced at peak volumes.
  - The current peak load is a good indicator of what the future average will be.
Monitor critical transactions or programs that have documented performance criteria.

Use the z/OS Workload Manager to help manage workload distribution, balance workloads, and distribute resources.

Plan your monitoring procedures in advance. A strategy should explain the tools to be used, the analysis techniques to be used, the operational extent of those activities, and how often they are to be performed.

Regardless of which strategy you use, you need to develop the following items:

- Performance criteria
- A master plan for monitoring, data gathering, and analysis

### 2.8.2 Monitoring multiple systems in DB/DC and DCCTL environments

Plan to obtain both statistical and performance data for IMS online systems that are part of a multi-system network. You can use the same monitoring utilities and tools that are used for generating performance data for single IMS systems.

An online monitoring facility, such as IBM Tivoli® OMEGAMON® XE for IMS on z/OS (see 3.5, “Tivoli OMEGAMON XE for IMS on z/OS, Version 4 Release 2” on page 129) offers several features that can be used to assist in monitoring multiple IMS systems. It can assist in identifying and resolving immediate issues when monitoring multiple subsystems. IMSplex workspaces can help you monitor shared queues and data sharing groups.

For more information about monitoring multiple systems with OMEGAMON XE for IMS on z/OS, see Chapter 5. Monitoring IMSplex systems of *IBM Tivoli OMEGAMON XE for IMS on z/OS User's Guide V4R2*, SC23-9706.

IMS Monitor can be run concurrently in several systems. You obtain IMS Monitor reports for each IMS system and coordinate your processing analysis:

- IMS Statistical Analysis utility produces summaries of transaction traffic for each system. Again, you combine the statistics for a composite picture.
- IMS Transaction Analysis utility helps you trace transactions across multiple systems and examine the traffic.

With IMS Performance Analyzer for z/OS (3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44) you can define groups. You can group IMS systems for reporting purposes.

IMS Monitor reporting only allows for processing a single IMS system at a time, but with IMS Performance Analyzer for z/OS Groups you can do the following tasks:

- Connect IMS systems participating in a sysplex. Reporting on the group can produce end-to-end response-time statistics for shared queue transactions.
- Connect IMS systems that use data sharing. Reporting on the group can produce consolidated database reporting.
- Connect IMS systems for periodical or ad hoc reporting. Reporting on the group can produce reports for each IMS subsystem in a single run.
- Connect systems (IMS and IMS Connect) to produce combined IMS and Connect forms-based reports.

For more information about using IMS Performance Analyzer for IMS on z/OS Groups to monitor multiple IMS systems in either a shared queue or data sharing environment. See
These options are only some of the options available when monitoring multiple systems in either a shared-queues or data-sharing environment. This process is installation-specific, to be determined by your system administrator.

2.8.3 Coordinating performance information in an MSC network

The IMS system log for each system participating in Multiple Systems Coupling (MSC) contains only the record of events that take place in that system. The logging of traffic received on links is included. You can augment the system log documentation that records the checkpoint intervals with the system identifications of all coupled systems. In this way, you can interpret reports, because you know of transactions that might be present in message queues but are not processed, and you can expect additional transaction loads from remote sources. In your analysis procedures, include ways of isolating the processing that is triggered by transactions that originate from another system.

To satisfy the need for monitoring with typical activity that includes cross-system processing, coordinate your scheduling of IMS Monitor and other traces between master terminal operators. The span of the monitoring does not have to be exactly the same, but if it is widely different, the averaging of report summaries can make it harder to interpret the effect of the processing that is triggered by cross-system messages.

For more information about interpreting MSC reports, see IMS Version 12: System Utilities, SC19-3023 and IMS Version 12: System Administration, SC19-3020.

Additionally, IMS Performance Analyzer for z/OS (see 3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44) provides several reports to assist you in tracking MSC activity in your installation. These IMS Monitor MSC reports include the following information:

- System analysis
- Resource usage
- Monitor data analysis

The reports format the records from the monitor input file into a chronological listing.

IMS Performance Analyzer for z/OS alleviates the need for averaging IMS Monitor report summaries by allowing you to place systems into groups and report against the group rather than individual systems.

IMS log reporting includes the MSC Link Statistics report (Figure 3-31 on page 61) and forms-based transit reporting (3.1.3, “Forms-based reports” on page 48). The MSC Link Statistics report contains information about the use of MSC Links and is generated from IMS log record X’4513’. It provides summary information for each MSC link with a more detailed breakdown of send and receive traffic. Forms-based reports provide summary and detailed transit information about your MSC transactions.

IMS Connect forms-based transit reports, using IMS Connect Extensions for z/OS (see 3.3, “IMS Connect Extensions for z/OS, Version 2 Release 3” on page 116) event journals, allow you to gain detailed information about your MSC over TCP/IP transactions that are initiated through IMS Connect. By using groups to combine IMS and IMS Connect systems, you can obtain end-to-end transit reporting for these types of MSC transactions.
For more information about using IMS Performance Analyzer for z/OS reporting for MSC, see the following sources:

- IBM IMS Performance Analyzer for z/OS User’s Guide V4R3, SC19-3633
  - Part 4, Forms-based transit reporting
  - Part 5, Requesting Log reports
  - Part 6, Monitor reporting
- IBM IMS Connect Extensions for z/OS User’s Guide V2R3, SC19-3632

2.8.4 Monitoring Fast Path systems in DB/DC and DCCTL environments

The major emphasis for monitoring IMS online systems that include message-driven Fast Path application programs is the balance between rapid response and high transaction rates. With Fast Path, performance data is part of the system log information. You can use IMS Fast Path Log Analysis utility (see 1.6, “Fast Path Log Analysis utility (DBFULTA0)” on page 14) to generate statistical reports from the system log records. This utility can provide summaries of the Fast Path transaction loads, reports that highlight exceptional response time, and details of the elapsed time between key events during the time in the system.

Another option, if available to your installation, is to use IMS Performance Analyzer for z/OS (see 3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44) for your Fast Path monitoring needs. It includes IMS Monitor Fast Path Reporting (Figure 2-7), and Fast Path transit and resource usage reporting (Figure 2-8 on page 34). By using it, you can build customizable report sets that can be run daily, weekly, or when you might need them, with one pass of the IMS log or extract.

![Figure 2-7 IMS Performance Analyzer: IMS Monitor Fast Path Report Set](image-url)
### Figure 2-8   IMS Performance Analyzer: IMS Fast Path Transit and Resource Report Set

The system administration tasks of setting up a monitoring strategy, performance profiles, and analysis procedures should be carried into the Fast Path environment.

For more information about using either IMS Monitor or IMS Fast Path Log Analysis utility, see the following sources:
- *IMS Version 12: System Utilities*, SC19-3023
- *IMS Version 12: System Administration*, SC19-3020

For more information about using IMS Performance Analyzer for z/OS, see *IBM IMS Performance Analyzer for z/OS User’s Guide V4R3*, SC19-3633.

#### 2.9 Transaction flow in DB/DC and DCCTL environments

A distinct sequence of events occurs during the processing of a transaction. Message-related processing is asynchronous within IMS, that is, not associated with a dependent region’s processing. Examples of this kind of processing are message traffic, editing, formatting, and recovery-related message enqueuing, any of which can be done concurrently with application program processing for other transactions. Events from application program scheduling to termination are associated with a PST and can be regarded as synchronous.
Figure 2-9 shows you a sequence of events when an online IMS system is processing a mix of transactions concurrently. Each event is explained in the notes that follow.

The unit of work by which most IMS systems are measured is the transaction (or a single conversation iteration, from entering the input message to receipt of one or more output messages in response).

One way of representing the flow of units of work is to compare it to three funnels through which all transactions must pass, as illustrated in Figure 2-9.

![Diagram of processing events during transaction flow through IMS](image)

The events that account for the principal contributions to transaction response time are numbered in the center. The items entered on the left of the diagram are message-related, and those on the right are related to the application program. The arrows trace the flow for an individual transaction. (The diagram does not show the paging element or system checkpoint processing that is distributed through the elapsed times.)
Figure 2-9 on page 35 shows the following events:

1. Wait for poll
   This wait used to be the time between pressing the Enter key and receiving a poll that results in the data being read by the channel program when using BTAM. Now this time does not belong to IMS.

2. Data transfer
   This time includes propagation delay and modem turnarounds for multi-block input messages. You can estimate the data transfer times if the volume of data transmitted is known.

3. Input message processing
   The IMS control of the transaction begins when the input message is available in the HIOP. The time that the message spends in this pool, in MFS processing, and in being moved to the message queue buffers affects response time. Individual transaction I/O to the Format library affects the message queue. A major factor in determining response time is whether the respective pools are large enough for the current volume of transactions flowing into input queuing. In particular, if the message queue pool is too small, overflow to the message queue data sets occurs.

4. Message classification
   This call is to the z/OS WLM to obtain a WLM service classification for the incoming message.

5. Input queuing
   This time is spent on the input queue or in the message queue buffers waiting for a message region to become available. In a busy system, this time can become a major portion of the response time. The pattern of programs scheduled into available regions and the region occupancy percentage are important and should be closely monitored.

6. Scheduling
   Because of class scheduling, regions can be idle while transactions are still in the queue. The effects of scheduling parameters can be as follows:
   - Termination of scheduling as a result of PSB conflict or message class priorities
   - Termination of scheduling as a result of intent conflict
   - Extension of scheduling by I/Os to IMS.ACBLIB for intent lists, PSBs, or DMBs
   - Pool space failures in either the PSB or DMB pools

7. Initialize PB call (activate delay monitoring environment)
   Activate the WLM delay monitoring environment for the message when it is placed into the dependent region. The WLM PB is initialized with the service classification and transaction name, message arrival time, program execution start time (current time), user ID, and so forth.

8. Program load
   See event 9.

9. Program initialization
   After scheduling, several kinds of processing events occur before the application program can start:
   - Contents supervision for the dependent region
   - Location of program libraries and directories to them
   - Program fetch from the program library
   - Program initialization up to the time of the first DL/I call to the message queue
For monitoring, you can obtain the overall time for the these activities. The number of I/Os should be checked periodically.

10. Message queue GU

This GU call is to the message queue. It is chosen as a measuring point because the event is recorded on the system log and is used as a starting point for iterations of processing when more than one message is serviced at a single scheduling of the program.

11. Program execution

The time for program execution, from first message call to the output message-insert, is a basic statistic for each transaction. It is important to account for that time in terms of the work performed:

- Number of transactions processed per schedule
- Number and type of DL/I calls per transaction
- Number of I/Os per transaction

A useful breakdown of elapsed time into processor time and I/O helps determine which transactions use significant resources.

12. Output message insert

This time begins the asynchronous processing for the output response. The output message requests flow into the funnel to be serviced while the application program is either beginning to process another input message or is performing closeout processing and program termination.

13. Wait for sync point

When an output message is issued by a program, it is enqueued on a temporary destination until the program reaches a synchronization point. For programs specified as MODE=MULT, a long delay in output transmission can occur when the program goes on to process several transactions at one scheduling. None of the previous output messages can be released for transmission until the program ends. If the program fails, all current transactions are backed out. Actually, when the messages are dequeued, LIFO sequence is used, from temporary to permanent destination. With MODE=SNGL, the wait for sync point (at the next GU to the message queue) is normally negligible.

14. Program termination

In the case of MODE=MULT, described in event 13, the synchronization point occurs at program termination. Any database updates are purged from the database buffer pools, and the waiting output messages are released.

In the MODE=SNGL case, the synchronization point occurs at the previous message queue GU call (usually a GU with a QC status code), and no database-commit processing occurs at termination, unless the application program has updated a database after the last message queue GU.

15. WLM notify call

This call tells WLM that the application program ended execution. The PB and current time are passed to WLM.

16. Wait for selection

This time is similar to wait for poll on input, with one difference, which is that an output message might not have to wait for the completion of a polling cycle if no polling is in progress on the line at the time the output message is enqueued. However, there might be a wait for the duration of data transmission to other terminals on the line. In a busy system, this wait can account for the majority of time spent on the output queue.
17. Output message processing
   This activity is similar to event 3 on page 36.

18. WLM report call
   This call tells WLM that the response is being sent. IMS passes the input message arrival
   time, the service classification, and the current time (output message send time).

19. Data transfer
   This activity is similar to event 2 on page 36.

20. Output queue processing
   Output messages that were sent are dequeued after their receipt is acknowledged by the
   terminal. In the case of paged output, the acknowledgment is a consequence of another
   input or an IMS MFS 3260/3270 PA2 key entry from the terminal.

2.9.1 Principal DB/DC and DCCTL monitoring facilities

IMS Monitor is the principal monitoring utility that is provided by IMS. It is an integral part of
the control program in the DB/DC environment. The counterpart of IMS Monitor in the batch
environment is the Database Batch Monitor.

IMS Monitor collects data while the DB/DC environment is operating. Information in the form
of monitor records is gathered for all dispatch events and placed in a sequential data set. IMS
Monitor data set is specified on the IMSMON DD statement in the control region JCL; data is
added to the data set when the \texttt{/TRACE} command activates the monitor. The MTO can start
and stop the monitor, guided by awareness of the system's status, to obtain several
snapshots.

For more information about interpreting IMS Monitor reports, see the following sources:

- 	extit{IMS Version 12: System Utilities}, SC19-3023
- 	extit{IMS Version 12: System Administration}, SC19-3020

The principle monitoring tools, provided by IBM IMS Tools, for IMS are as follows:

- Tivoli OMEGAMON XE for IMS on z/OS
  (see 3.5, “Tivoli OMEGAMON XE for IMS on z/OS, Version 4 Release 2” on page 129)
- IMS Performance Analyzer for z/OS
  (see 3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44)

Tivoli OMEGAMON XE for IMS on z/OS monitors and manages the availability, performance,
and resource utilization of your IMS systems, either at a system level or within an IMSplex.
OMEGAMON XE for IMS on z/OS monitors more than 80 groups of attributes, providing a
wealth of IMSplex and system-level data. You can use these attributes to tailor the information
presented in workspaces, or to define situations that target specific thresholds, events, or
performance problems you want to monitor.

For more information about Tivoli OMEGAMON XE for IMS on z/OS monitoring facilities, see

IMS Performance Analyzer for z/OS is a performance analysis tool to help you monitor,
maintain, and tune your Information Management System Database (IMS DB) and
Transaction Manager (IMS TM) systems. IMS Performance Analyzer for z/OS processes IMS
Log, Monitor, IMS Connect event data through IMS Connect Extensions for z/OS event
journals, and OMEGAMON XE for IMS on z/OS Transaction Reporting Facility (TRF) and
Application Trace Facility (ATF) data to provide comprehensive reports for use by IMS
specialists to tune their IMS systems, and managers to verify service levels and predict trends.

For more information about IMS Performance Analyzer for z/OS and its reporting capabilities, see IBM IMS Performance Analyzer for z/OS User’s Guide V4R3, SC19-3633.

2.9.2 Monitoring procedures in a DBCTL environment

This topic explains how to establish monitoring procedures for your DBCTL environment. First, consider that monitoring in a DB/DC environment generally refers to the monitoring of transactions. The transaction is entered by a user on a terminal, is processed by the DB/DC environment, and returns a result to the user. Transaction characteristics that are measured include total response time and the number and duration of resource contentions.

A DBCTL environment has no transactions and no terminal users. It does, however, do work on behalf of coordinator controller (CCTL) transactions that are entered by CCTL terminal users. DBCTL monitoring provides data about the DBCTL processing that occurs when a CCTL transaction accesses databases in a DBCTL environment. This access is provided by the CCTL making the DRA request.

The most typical sequence of database resource adapter (DRA) requests that represents a CCTL transaction are as follows (see Figure 2-10):

- A SCHED request to schedule a PSB in the DBCTL environment
- A DL/I request to make database calls
- A sync point request, COMMTERM, to commit the database updates and release the PSB

The DBCTL process that encompasses this request is called a unit of recovery (UOR).

DBCTL provides UOR monitoring data, such as the following examples (see Figure 2-11 on page 40):

- Total time the UOR exists
- Wait time to schedule a PSB
- I/O activity during database calls

![Figure 2-10 IMS Problem Investigator: DBCTL Transaction Flow in IMS]

The DBCTL process that encompasses this request is called a unit of recovery (UOR).
This information is similar to, and is often the same as, DB/DC monitoring data. However, in a DBCTL environment, the UOR data represents only a part of the total processing of a CCTL transaction. You must also include CCTL monitoring data, using separate tooling, to get an overall view of the CCTL transaction performance.

In this topic, the term transaction refers to a CCTL transaction. When it applies, UOR is specifically named.

The CCTL administrator must decide what strategy to use to monitor transaction performance. Several types of monitoring strategies are available:

- Summarize actual workload for the entire online execution. This can be continuous or at an agreed-to frequency. Total workload or selected representative transactions can be tracked.
- Take sample snapshots at peak loads and under normal conditions. Monitoring the peak periods is always useful for two reasons:
  - Bottlenecks and response time problems are more pronounced at peak volumes.
  - The current peak load is a good indicator of what the future average will be like.
- Monitor critical transactions or programs that have documented performance criteria.

2.10 Monitoring procedures in an open transaction environment

This topic explains how to establish monitoring procedures for your open transaction environment.

In IMS 12, the Open Database Manager provides access to IMS databases that are managed by IMS DB systems in DBCTL and DB/TM environments within an IMSplex. The Open Database Manager supports TCP/IP clients through IMS Connect and application servers running application programs that use the IMS ODBA interface; it is one of the address spaces provided by the IMS Common Service Layer (CSL).

Both independently and with IMS Connect, ODBM supports various interfaces to ease the development of application programs that access IMS databases from many and varied distributed and local environments.
ODBM supports the following interfaces:
- IMS Universal DB resource adapter
- IMS Universal JDBC driver
- IMS Universal DL/I driver
- The ODBA interface
- The ODBM CSLDMI interface for user-written ODBM client application programs

Figure 2-12 is an overview of an IMS configuration that includes ODBM.

With the variety of origination points for these types of transactions, developing monitoring practices can sometimes be difficult. When we talk about transactions coming into the ODBM address space through either user-written ODBM clients, using SCI, or z/OS ODBA applications, enabling BPE tracing for your ODBM address space in combination with the monitoring process described in earlier parts of this chapter can assist you in monitoring transaction performance in this environment. You might also consider using a combination of BPE tracing, IMS Performance Analyzer for z/OS reporting (3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44), and IMS Problem Investigator for z/OS (3.2, “IMS Problem Investigator for z/OS, Version 2 Release 3” on page 81) for interactive analysis of the transactions executing in this environment.
For transactions that are initiated by using the IBM DRDA® protocols coming into the ODBM address space through IMS Connect through TCP/IP, to access IMS databases, we need additional diagnostic information to better determine their performance. With IMS providing little in the way of diagnostic and performance data collection ability for the IMS Connect address space, diagnosing problems with transactions in this environment can prove to be more difficult.

One option when trying to diagnose these problems is to enable BPE tracing for the IMS Connect and ODBM address spaces, and use this trace data along with system utilities, such as the Log Transaction Analysis utility (1.4, “Log Transaction Analysis utility (DFSILTA0)” on page 11) and Statistical Analysis utility (1.4, “Log Transaction Analysis utility (DFSILTA0)” on page 11), to monitor transaction performance for this environment.

Another option is to use additional tooling provided by IBM IMS tools to help you monitor transaction-related performance with the ODBM transactions initiated through IMS Connect. You might use IMS Connect Extensions for z/OS (3.3, “IMS Connect Extensions for z/OS, Version 2 Release 3” on page 116) event journals in combination with IMS Performance Analyzer for z/OS (3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44) to run reports and obtain detailed transaction transit performance information and specific resource usage, for both IMS Connect and IMS, for your ODBM transactions. With this information, you might then use IMS Problem Investigator for z/OS (Chapter 7, “Performance considerations for managing distributed workloads” on page 291) to focus on any of your ODBM transactions that might have poor response times.

The monitoring process in this environment is truly dependent on each individual installation and can be done in a variety of ways. It is highly dependent on individual skill level and knowledge of IMS utilities and tools.

For more information about open transactions, including transaction flows and monitoring tips, see Chapter 7, “Performance considerations for managing distributed workloads” on page 291.
Performance monitoring tools

In this chapter, we introduce IMS monitoring tools capable of providing a detailed picture of your IMS system resource usage. These tools can help to identify potential performance bottlenecks in IMS and help show you how z/OS performance problems affect you overall IMS performance.

We describe the following monitoring tools:

- IMS Performance Analyzer for z/OS, Version 4 Release 3
- IMS Problem Investigator for z/OS, Version 2 Release 3
- IMS Connect Extensions for z/OS, Version 2 Release 3
- IMS Buffer Pool Analyzer for z/OS, Version 1 Release 3
- Tivoli OMEGAMON XE for IMS on z/OS, Version 4 Release 2
- Transaction Analysis Workbench for z/OS, Version 1 Release 1

Today, faster and more efficient usage of IMS system resources is more critical than ever. With the ever-increasing need for reduced CPU cycles and peak performance from storage and I/O devices, good measurement tools that determine how these resources are being used are a necessity for any IMS performance evaluation. Although good tools to measure performance are provided with z/OS, you need additional tools to be able to collect the specific data that is required to evaluate the performance of your IMS systems.

With so many program offerings available today, determining what is needed when doing an IMS performance check can sometimes be difficult. Table 3-1 on page 44 lists the more important integrated program offerings available for IMS performance evaluations. Although not all of the tools listed in the table are necessary to do an IMS performance evaluation, information is provided to assist in evaluating their usefulness to each individual IMS installation. These tools can help improve IMS performance by identifying bottlenecks within IMS and showing how IMS uses system resources. When used in conjunction with z/OS measurements, the tools help identify relationships between z/OS and IMS performance problems.
In this section, we describe the product features, functionality, and benefits you can obtain when using IMS Performance Analyzer for z/OS.

### 3.1.1 Product overview

IMS Performance Analyzer for z/OS processes IMS Log, Monitor, IMS Connect event data through IMS Connect Extensions for z/OS event journals, OMEGAMON XE for IMS Transaction Reporting Facility (TRF) data, OMEGAMON XE for IMS Application Trace Facility (ATF) data, and CQS log data to provide comprehensive reports for use by IMS specialists to tune their IMS systems, verify service levels, and predict trends. IMS Performance Analyzer for z/OS provides an ISPF-based dialog for creating and maintaining your report and extracting requests, and generating the JCL to run them using your specified systems and data files.

From IMS Log data, IMS Performance Analyzer for z/OS provides comprehensive information about transit times (transaction performance times), and IMS resource usage and availability. IMS Performance Analyzer for z/OS can process logs from a single IMS system or from multiple IMS subsystems running in a sysplex and IMS systems using shared queues. You can specify log files explicitly or let IMS Performance Analyzer for z/OS use DBRC Log...
Selection and IMS RECON data sets to automatically locate the files for your required reporting interval. User-defined performance thresholds allow you to set goals and report transactions that deviate from these goals. A history of transaction performance can be maintained in transaction history files.

Transaction transit information can be collected by time or time interval, and reported using standard or custom report formats. Extracts of transaction transit time by time interval data can be created from log files and then graphed or exported for use in spreadsheets or other forms of reporting tools. Extracts of total transaction volume, exception transactions, CPU usage, and database update activity are some of the types of data that can be created for direct import by external programs such as DB2 or PC-based reporting tools.

Figure 3-1 illustrates the ability of IMS Performance Analyzer for z/OS to use or merge multiple sources of IMS log data for reporting, building extracts as comma-separated value (CSV) spreadsheets, or Transaction Index extracts (for IMS and IMS Connect).
The following highlights of IBM IMS Performance Analyzer for z/OS reporting functions show how it can assist you in analyzing types of performance issues:

- Comprehensive suite of reports to help you to manage transaction performance and to monitor resource utilization in IMS TM and DB systems, including the following items:
  - Message queue transaction and full-function database
  - Fast Path EMH transaction and DEDB/MSDB database
- Reports and extracts can be produced from IMS Logs, Monitor files, IMS Connect Extensions journals, and Tivoli OMEGAMON XE for IMS on z/OS TRF files and ATF journals.
- IMS support of shared queues in an IMS sysplex; multi-subsystem log input is merged by time stamp sequence to produce either composite reports or reports by subsystem.

### 3.1.2 Product features

In this section, we describe the main features of IMS Performance Analyzer for z/OS.

**IMS Version 10, 11, and 12 support**
The IMS Performance Analyzer tool exploits all the new features and instrumentation in the IMS log and monitor:

- **IMS Version 10:**
  - Type 56FA transaction accounting record provides accurate transaction CPU time, DL/I call, database I/O, and ESAF usage
  - 4511 general storage statistics
  - 4512 IMODULE statistics
  - 4513 MSC statistics
  - 4514 IBM Enterprise Workload Manager™ (EWLM) statistics

- **IMS Version 11:**
  - Synchronous call-out (ICAL)\(^1\) and Open Database
  - 4515 64-bit cache storage manager
  - 4516 Fast Path 64-bit buffer manager

- **IMS Version 12:**
  - Improved MSC statistics

**OMEGAMON Transaction Recording Facility (TRF) support**
Tivoli OMEGAMON XE for IMS on z/OS for IMS TRF collects detailed transaction accounting, performance and resource utilization information for IMS transactions. IMS Performance Analyzer for z/OS provides a comprehensive set of reports for all of the TRF Extractor record types:

- Transaction transit analysis and resource usage statistics
- DL/I database and DB2 call analysis, including elapsed time per call

---

\(^1\) The IMS Call (ICAL) call enables an application program that runs in the IMS TM environment to send a synchronous request for data or services to a non-IMS application program or service that runs in z/OS or a distributed environment.
OMEGAMON for IMS Application Trace Facility (ATF) support
Tivoli OMEGAMON XE for IMS on z/OS ATF collects detailed information about every application call made by your IMS and DBCTL transactions. DL/I and DB2 call results are recorded in the ATF journal, including detailed information for function code, segment search argument (SSA), key feedback area (KFBA) and I/O areas. This facility provides you the most in-depth analysis of transaction behavior available.

IMS Performance Analyzer for z/OS provides management and reporting of ATF data that includes the following items:

- Archival of the online ATF journals to offline data sets for post-analysis
  Filtering criteria allows you to archive problem transactions only, significantly reducing the volume of data kept and time to resolution.
- Three levels of reporting:
  - Summarization of transaction activity
  - High level transaction detail
  - In-depth traces that report every detail about each application DL/I call

IMS Transaction Index support
Forms-based Log Report Sets provide you with the option to create IMS Transaction Index. The Index contains a record (X'CA01') for each IMS transaction in the log. Each record contains cumulative information from the IMS log about the transaction. It is possible to create a sparse IMS transaction index. The sparse index could contain only transactions that abended or exceeded a specified response time value.

IMS Transaction Index can replace the IMS system log data set (SLDS) files as input for transaction transit reporting, significantly reducing the time and overhead of running IMS Performance Analyzer for z/OS transit reports.

IMS Problem Investigator for z/OS can also use IMS Transaction Index, either on its own or merged with the associated IMS log files, to provide faster problem detection and analysis.

IMS Connect Transaction Index support
IMS Connect forms-based Log Report Set provides you with the capability to create IMS Connect Transaction Index. The index contains a record (X'CA20') that represents each IMS Connect transaction recorded in IMS Connect Extensions for z/OS journal, and includes all the cumulative information from the journal about that transaction. It is possible to create a sparse IMS Connect Transaction Index. The sparse index could contain only transactions that abended or exceeded a specified response time value.

IMS Problem Investigator for z/OS uses IMS Connect Transaction Index, either on its own or merged with the associated IMS Connect Extensions journals and IMS log files, to provide improved levels of problem detection and analysis.

IMS Tools Knowledge Base
IMS Performance Analyzer for z/OS log reporting can now write reports directly to IMS Tools Knowledge Base repository for historical collection and viewing from IMS Tools Knowledge Base dialog. IMS Tools Knowledge Base is a component of IMS Tools Base for z/OS, V1.3 (program number 5655-V93).
3.1.3 Forms-based reports

In addition to traditional transit reporting, IMS Performance Analyzer for z/OS gives you the ability to customize your transit reports and extracts. The ability to customize this type of reporting can be beneficial when you need to drill down on specific areas of interest when working with large amounts of data. The following list describes several advantages of using forms-based reporting when evaluating transaction performance problems:

- Fully customizable reporting or extraction to data set. With field selection (Figure 3-2) you can request only the information you need to see.

![Figure 3-2 IMS Performance Analyzer: IMS Field Selection](image)

Field selection lets you choose what fields are important to you.

IMS, IMS Connect (through IMS Connect Extension for z/OS Event Journals), and OMEGAMON XE Trace Recording Facility reporting are supported. See Figure 3-3 and Figure 3-4 on page 49.

![Figure 3-3 IMS Performance Analyzer: IMS Connect Field Selection](image)
Note the following information:

- More than 160 IMS and 40 IMS Connect transaction information fields are available for selection in reports, providing an improved level of detail for measuring transaction processing and resource usage.

- IMS log and IMS Connect Extensions event data can be merged into a single report, providing end-to-end reporting for IMS Connect transactions.

- Ability to create wide reports eases the restriction of the 132 column page width. When additional fields are requested in the report, IMS Performance Analyzer for z/OS adjusts the page width to accommodate your reports.

- IMS log reporting provides improved accuracy. Additional log record types are used to provide a clearer picture of transaction transit activity.

IMS Version 10 introduced the Type 56FA transaction accounting record that provides more accurate transaction CPU time, DL/I call, database I/O, and ESAF usage statistics. Be sure that the 56FA log record is activated for all transactions, which can improve the accuracy of IMS Performance Analyzer for z/OS transaction transit reporting.

- Precision of elapsed and CPU times is customizable, from millisecond (0.001) to microsecond (0.000001).

- Summary reporting allows up to eight grouping or key fields. See Figure 3-5.
Statistical functions (average, minimum, maximum, standard deviation, peak percentile, total, count, range) can be requested at the field level. See Figure 3-6.

In IMS Forms Based reports set: select (S) Transaction Index
When IMS Performance Analyzer for z/OS batch job completes successfully, the Recap report gives the name of IMS Transaction Index data set and the number of records written. See Example 3-1.

Example 3-1 IMS Performance Analyzer: Output from IMS Transaction Index build

IMS Performance Analyzer
Transaction Index
IPITXOUT Transaction Index completed
Data Set Name . . . JM3.REDBOOK.TRANX
Record count . . . 280
Upon completion, your IMS Transaction Index is ready to use. The Transaction Index can be used effectively in the following ways:

- You can use IMS Transaction Index with IMS Performance Analyzer for z/OS by replacing the IMS log (SLDS) files with the transaction index for subsequent form-based transit report requests. The resulting reports contain the same information as though they were created from the original IMS log files, and have shown in tests to significantly reduce the time necessary to run the same number of transit reports by using the SLDS.

- You can use IMS Transaction Index with IMS Problem Investigator for z/OS to diagnose transaction problems. IMS Transaction Index can be analyzed on its own, or in conjunction with the IMS log files that are used to create it (Figure 3-11 on page 53). The Transaction Index Record (Figure 3-10) is a useful diagnostic tool that provides a summary insight into the dynamics of each transaction, and can provide a shortcut to the root of the problem.

![Figure 3-10  IMS Problem Investigator: IMS Transaction Index Record](image)

When you consider the use of IMS Transaction Index for transaction diagnoses, an important point to remember is that IMS Performance Analyzer for z/OS creates the index records in the order that transactions complete their transit processing in the IMS log file. The index is not sorted in transaction-start-time sequence. This issue occurs only if the index will be merged with the original SLDS or extract that was used to create it for diagnosis. For diagnosis, IMS Problem Investigator for z/OS expects log file data to be in transaction-start-time-sequence to achieve the following goals:

- Merging can display results in correct chronological order (Figure 3-11 on page 53).
- Time formatting can aid views. Relative (Figure 3-12 on page 53) and Elapsed, can display actual (positive) elapsed time deltas.
- When merging IMS Transaction Index and IMS log files, the X’CA01’ type index and the X’01’ type IMS transaction message records, from which the index was derived, can be adjacent in the display. See Figure 3-13 on page 54.
Figure 3-11  IMS Problem Investigator: IMS Transaction Index Merged with IMS Log

Sorting allows for correct time sequence

(R) Initiates relative time view

Figure 3-12  IMS Problem Investigator: Sorting, merging IMS Transaction index-relative view
Figure 3-13  IMS Problem Investigator: Sorting, merging IMS transaction index-elapsed view

Sorting the index is optional; IMS Problem Investigator for z/OS can process an unsorted index. You can decide not to sort the index if you are processing it as stand-alone and looking for individual transaction problems or issues, rather than viewing the file as a time line of system activity or transaction activity.

If you do determine that sorting IMS Transaction Index for your diagnosis purposes is necessary, use the JCL (Example 3-2) to sort the index file in (message arrival) time sequence, by using field ISO, which is the 26-character ISO date and time stamp.

Example 3-2  Sample JCL to sort IMS Transaction Index in time sequence

```bash
//IMSPA JOB ,NOTIFY=&SYSUID
//@
//SORT EXEC PGM=SORT
//SORTIN DD DSN=JCH.WORKSHOP.INDEX,DISP=SHR
//SORTOUT DD DSN=JCH.WORKSHOP.INDEX.SORTED,
// DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(TRK,(1,1),RLSE)
//SORTWK01 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK02 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK03 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK04 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(10,10))
//SYSOUT DD SYSOUT=*  
//SYSJIN DD *
OMIT COND=(349,3,CH,EQ,C'SWI')
SORT FIELDS=(265,26,CH,A)
//@
```
IMS Connect Transaction Index

As with the IMS Transaction Index, IMS Connect Transaction Index is created by IMS Performance Analyzer for z/OS, by using the forms-based reporting sets for IMS Connect. See Figure 3-14 and Figure 3-15.

IMS Connect Transaction Index, denoted by the X'CA20' log code, contains a record for each IMS Connect transaction recorded in the IMS Connect Extensions for z/OS event journal, and includes all the cumulative information from the journal about that transaction (Figure 3-16).

OMIT statement: The OMIT statement is optional. It omits the ‘SWI’ CA01 records, which are of no diagnostic value.

IMS Transaction Index records are variable length and the character positions in the SORT and OMIT statements allow for the first four bytes of each record to contain the record descriptor word (RDW).

Figure 3-14  IMS Performance Analyzer: IMS Connect Transaction Index creation (Part 1 of 2)

Figure 3-15  IMS Performance Analyzer: IMS Connect Transaction Index creation (Part 2 of 2)

IMS Connect Transaction Index, denoted by the X'CA20' log code, contains a record for each IMS Connect transaction recorded in the IMS Connect Extensions for z/OS event journal, and includes all the cumulative information from the journal about that transaction (Figure 3-16).

Figure 3-16  IMS Problem Investigator: IMS Connect Transaction Index
IMS Problem Investigator for z/OS uses IMS Connect Transaction Index, either on its own or merged with the associated IMS Connect Extensions for z/OS event journals and IMS log files, to provide improved levels of problem detection.

When using IMS Connect Transaction Index for problem diagnosis, sorting might be necessary for it to be displayed in correct chronological sequence in IMS Problem Investigator for z/OS. To sort IMS Connect Transaction Index, use the JCL example for IMS Transaction Index (see Example 3-2 on page 54), but remove the OMIT statement because it does not apply to IMS Connect Transaction Index. IMS Problem Investigator for z/OS can process an unsorted IMS Connect Transaction Index when viewed in stand-alone mode so sorting is optional, but when merged with accompanying IMS Connect Extensions for z/OS event journals and IMS Log data, then merging is useful.

### 3.1.4 Message queue transaction and full-function reports

This section describes the various non-forms-based message queue and full-function transit reports and system resource usage and availability reports available for you with IMS Performance Analyzer for z/OS.

#### Analysis reports

Analysis reports are intended for administrators and analysts who want operating information made available to them in an easy-to-read summary format. These types of reports are described next.

**Transaction Transit Analysis**

This report shows the times for the components of transaction transit time:

- Input queue time (local or CQS)
- Processing time
- Output queue time
- Output shared queue (CQS) time
- Output local queue time
- Program switch time

Transaction Transit Analysis reports can be ordered by the following ways:

- LTERM (Figure 3-17)

![Figure 3-17](image-url)

Transaction Transit Analysis reports can be ordered by the following ways:

- Transaction code (Figure 3-18 on page 57)
Chapter 3. Performance monitoring tools

Figure 3-18  IMS Performance Analyzer: Transit Time by Transaction code

- Transaction code within LTERM (Figure 3-19)

Figure 3-19  IMS Performance Analyzer: Transit Time by Transaction code within LTERM

- Line or VTAM node (Figure 3-20)

Figure 3-20  IMS Performance Analyzer: Transit Time by Line

- Message Class (Figure 3-21C)

Figure 3-21  IMS Performance Analyzer: Transit Time by Message Class
Time of Input (Figure 3-22)

Transaction Transit Statistics
These reports consist of statistical distribution graphs of the transit time components that are listed in the preceding reports. See Figure 3-17 on page 56 through Figure 3-22.

Transaction Resource Usage
These reports provide a comprehensive overview of transaction resource usage, including the following items:

- Scheduling statistics, including WFI
- CPU usage
- DL/I call statistics
- Enqueue statistics
- DBCTL DB and I/O usage

This report is available in three formats:

- Long List
- Long Summary (Figure 3-23)
- Short (DL/I Call) Summary

Resource Availability
This report provides a breakdown of active, idle, and unavailable time for regions, lines, programs, transactions, and databases. The time that WFI and pseudo-WFI regions spent waiting for input is also shown.
This type of Report is available for the following items:

- Regions (Figure 3-24)

**Figure 3-24** IMS Performance Analyzer: Resource Availability by Region

- Transactions (Figure 3-25)

**Figure 3-25** IMS Performance Analyzer: Resource Availability by Transaction

- Programs (Figure 3-26)

**Figure 3-26** IMS Performance Analyzer: Resource Availability by Program

- Lines or VTAM nodes (Figure 3-27)

**Figure 3-27** IMS Performance Analyzer: Resource Availability by Line
Databases (Figure 3-28)

CPU Usage

This report and optional extract give statistics of CPU time and elapsed time for the following items:

- Region (Figure 3-29)
- Program
- Transactions

The CPU Usage Report can be useful when determining such things as which region is using too much CPU time or what programs or transactions, in which regions, are in a wait state for too long.

Internal Resource Usage

This report (Figure 3-30 on page 61) provides statistics on the use of various IMS resources and pools. The statistics help you determine how resources and pools are being used and where inefficiencies exist.
Chapter 3. Performance monitoring tools

MSC Link Statistics

This report (Figure 3-31) provides summary information about the overall usage of MSC links and details of send and receive data for each MSC link.

![Figure 3-31 IMS Performance Analyzer: MSC Link Statistics](image)

Message Queue Utilization

This report (Figure 3-32) contains information about the use of message queues.

![Figure 3-32 IMS Performance Analyzer: Message Queue Utilization](image)
It can assist you with the following tasks:

- Balance the I/O between long and short message data sets.
- Adjust record and buffer sizes for the most efficient use of these two data sets.

This report can be helpful to you when trying to reduce storage and minimize I/O.

**Database Update Activity**

The report and optional extract in Figure 3-33 help you determine the cost of database calls.

<table>
<thead>
<tr>
<th>Database Program DDname</th>
<th>Blocks Updated</th>
<th><strong>Generated Update Counts</strong></th>
<th>DB Open Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Inserts</td>
<td>Deletes</td>
</tr>
<tr>
<td>__________</td>
<td>_________</td>
<td>______</td>
<td>______</td>
</tr>
<tr>
<td>DI21PART CEXITPGM</td>
<td>187</td>
<td>22</td>
<td>44</td>
</tr>
<tr>
<td>DI21PART CEXTPGM</td>
<td>347</td>
<td>48</td>
<td>96</td>
</tr>
<tr>
<td>DFSAM04</td>
<td>247</td>
<td>30</td>
<td>18</td>
</tr>
<tr>
<td>MQOITPGM</td>
<td>47</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>DFSAM04</td>
<td>57</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>IVPDR1 DFSIVP1 DFSIVD1</td>
<td>14</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>IVPDR1 DFSIVP2</td>
<td>14</td>
<td>12</td>
<td>6</td>
</tr>
<tr>
<td>DFSIVP3</td>
<td>28</td>
<td>24</td>
<td>12</td>
</tr>
<tr>
<td>DFSIVP35</td>
<td>28</td>
<td>24</td>
<td>12</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>181</strong></td>
<td><strong>192</strong></td>
<td><strong>214</strong></td>
</tr>
</tbody>
</table>

Figure 3-33  IMS Performance Analyzer: Database Update Activity

The report shows the number of blocks that are updated and how many updates of each type were made for each database.

**OSAM Sequential Buffering**

This report (Figure 3-34) provides detailed I/O and buffer-usage statistics that allow you to assess the benefit of OSAM sequential buffering.

Figure 3-34  IMS Performance Analyzer: OSAM Sequential Buffering Report
Deadlock

These reports provide a comprehensive analysis of deadlock events that includes a list (Figure 3-35) of all deadlock events in the log, similar to DFSERA30. In addition, the report summarizes (Figure 3-36) deadlock activity for the following items:

- The frequency of each losing transaction/database combination
- The associated winning transaction/database combinations

The information that is provided can be a useful aid for tuning applications and adjusting your scheduling parameters to avoid this expensive overhead. Note the following information:

- Deadlocks involving DB2 are also reported in this report
- This report is available in List and Summary Views

![Figure 3-35 IMS Performance Analyzer: Deadlock Report List Report](image1)

![Figure 3-36 IMS Performance Analyzer: Deadlock Summary Report](image2)
**System Checkpoint**

The System Checkpoint report (Figure 3-37) provides a detailed analysis of IMS internal checkpoint activity.

![Figure 3-37 IMS Performance Analyzer: System Checkpoint](image)

This report provides details of your IMS resources:

- Databases, with system definition information
- Transactions, with system definition information and basic usage indicators
- Terminals, with system definition information and basic usage indicators

The report also provides a summary of checkpoint activity:

- Breakdown of checkpoint records by type
- Frequency and overhead of internal checkpoint processing

**BMP Checkpoint**

The BMP Checkpoint reporting provides an analysis of batch messaging program (BMP) checkpoint frequency that can affect online performance and system restartability.

- The list report (Figure 3-38) provides details of individual BMP checkpoint activity.

![Figure 3-38 IMS Performance Analyzer: BMP Checkpoint list report](image)
The Summary report (Figure 3-39) provides an overview of each BMP program.

The Gap Analysis (Figure 3-40) report shows periods of time where log records are not being cut, potentially highlighting an external system event that can have caused IMS to slow down.

The Cold Start Analysis (Figure 3-41 and Figure 3-42 on page 66) report provides a snapshot of in-train activity in the event of a cold start to determine what transactions were lost from the IMS message queue, what were the incomplete units of work, what database changes were made and not backed-out, and what external subsystem activity was left in doubt.
The ESAF Trace report lists each connect and disconnect of an external subsystem.

**Detail reports**

Detail reports help analysts and programmers solve problems. These types of reports are described next.

**Transaction Transit Log**

This report (Figure 3-43) shows the transit activity of each message originating from a logical terminal or program switch.

This report can assist you in diagnosing problem areas. It is similar to the report produced by DFSILTA0, but it additionally provides shared queue reporting for multiple IMS subsystems.
Chapter 3. Performance monitoring tools

Region Histogram
This report (Figure 3-44) is a graphic display of region activity.

DC Queue Manager Trace
This report (Figure 3-45) shows a time-sequenced list of each TM event, such as input, message enqueue, get unique, output, and free DRRN (disk relative record number).
You can also request the DC UOW Tracker report (Figure 3-46) so you can trace transaction message flow by using the IMS Tracking UOW.

![Figure 3-46  IMS Performance Analyzer: DC UOW Tracker](image)

### Database Trace

This report (Figure 3-47) shows the versions before and after each changed segment, and, for each change, identifies the application program, transaction, region, and time.

![Figure 3-47  IMS Performance Analyzer: Database Trace](image)

### 3.1.5 Fast Path Reports

The Fast Path EMH transaction and Fast Path database reports are grouped by intended audience into management reports, analysis reports, and detail reports.

#### Management report

A Fast Path EMH management report is available to you with IMS Performance Analyzer for z/OS.

#### Fast Path Transit Extract by Interval and Graphs

This facility allows creation of extract files of Fast Path transit data by time interval.

This data can help you get pictures of system performance by using IMS Performance Analyzer for z/OS graphing facilities, or by exporting for manipulation by external programs or downloading to a PC.
Analysis reports
Analysis reports are intended for administrators and analysts who want operating information in an easy-to-read, summary format.

Note: This report output is similar to the report output shown for the previous examples of full-function output, so no examples will be used for this section of the document.

Fast Path Transit Analysis
This report shows the times for the components of transaction transit time:
- Input queue time
- Processing time
- Output queue time
- Global input and output queue times

This report can be ordered by the following items:
- Transaction Code
- Routing Code
- User ID
- Time of day

The display of transit time by time period can be useful in isolating periods of high response times during the day.

FP Resource Usage and Contention
This report provides statistics on the use of various Fast Path resources:
- DEDB databases and areas
- VSO
- Buffers
- Locks
- Logging
- Transaction throughput

FP Database Call Statistics
This report shows DEDB and MSDB DL/I call statistics, organized down by transaction code.

IFP Region Occupancy
This report provides approximate region occupancy rates for IFP regions.

EMH Message Statistics
This report contains information about the number and length of EMH messages that are processed by balancing groups and shared EMH queues.

DEDB Update Activity
This report provides information about the update activity against your DEDB databases.

VSO Statistics
This report provides detailed statistics on VSO resource usage:
- SHARELVL 0/1 data spaces
- SHARELVL 2/3 coupling facility structures
- SHARELVL 2/3 lookaside buffers
- DEDB area data set I/O
Detail reports
Detail reports help analysts and programmers solve problems.

**Fast Path Transit Log**
This report shows the transit activity of each EMH transaction processed by an IFP region. This report can help you to isolate and diagnose problem transactions in a Fast Path environment.

**Fast Path Transaction Exception**
This series of reports provide details and summary information about IFP transactions, and also message queue transactions that use Fast Path databases. With this report, you can build two extract files: one can be produced containing detail records of all IFP transaction activity and one can be produced containing exception IFP transactions only.

The contents of these reports and extracts are similar to those that are produced by the Fast Path Log Analysis Utility DBFULTA0.

**DEDB Update Trace**
This report provides a record of all DEDB changes, identifying application program, user ID, region, and time.

**DBCTL log reports**
DBCTL has neither transactions nor terminal users in the traditional IMS sense. It does, however, work on behalf of transactions entered by CCTL terminal users. DBCTL generates log data when a CCTL transaction schedules a program to access DBCTL databases.

The following log reports apply to DBCTL:
- Transaction Resource Usage
- Resource Availability
- CPU Usage
- Internal Resource Usage
- Database Update Activity
- Fast Path Resource Usage and Contention
- Fast Path Database Call Statistics
- DEDB Update Activity
- VSO Statistics
- Database Trace (full-function)
- DEDB Update Trace
- Forms-based transit reports that use the following sample report forms:
  - DBCTLIST: List of DBCTL Transactions
  - DBCTSUMM: Summary of DBCTL Transactions

**IMS Connect reports**
IMS Performance Analyzer for z/OS reports that analyze IMS Connect event data are organized by level of detail and area of analysis into categories.
- Transaction Transit
- Transaction Transit (forms-based)
- Resource Usage
- Trace

**Examples:** No examples shown for Connect Reports because the output is similar to that of previous shown full-function reporting.
You can tailor reports for the following options:

- **Time Interval**
  Reports can be summarized by time interval.

- **Selection Criteria**
  You can filter the data based on field values and focus your reports only on the information that you are interested in.

- **Report Interval**
  With time ranges, you can select specific parts of IMS Connect data.

IMS Connect Extensions for z/OS event journals are required for this type of reporting.

**Transaction Transit reports**
The Connect Transaction Transit reports provide performance statistics to measure the performance of your IMS Connect transactions.

Transaction Transit (response) time is categorized into its components: Input, Processing (by OTMA), Acknowledgement from the client, and Output. They can help identify any bottlenecks in transaction flow, and are used for monitoring system performance, gathering diagnostic information, and tuning IMS.

Transaction Transit reports are as follows:

- Analysis
- Log
- Extract

**Resource Usage reports**
These reports contain detailed and summary information about the use and availability of various IMS Connect resources including TCP/IP ports and tpipes.

These reports include the following items:

- Port Usage
- RESUME TPIPE
- ACK/NAK
- Exception Events

**Trace reports**
These reports provide detailed analyses of individual IMS Connect transit event records. These reports are typically used to investigate point in time performance problems because they provide all available information within the report or extract time interval that you specify.
OMEGAMON TRF reports

Transaction performance and resource utilization statistics are collected by Tivoli OMEGAMON XE for IMS on z/OS subsystems with the Transaction Reporting Facility (TRF) Trace running. This data is then written out to an SMF data set or written to the IMS log after collection.

IMS Performance Analyzer for z/OS can then process TRF records extracted from the IMS Log by the TRF post-processor, and provide a set of reports in the TRF Report Set to process the TRF Extractor records. The TRF reports are grouped by functional category so you can request reports individually or by specific category.

The categories are as follows:

- Database Usage reports
- Message Queue reports
- Trace reports

Database Usage reports

These reports provide performance statistics to measure IMS and DB2 database activity:

- DL/I Call List (Figure 3-48)
- DL/I Call Summary (Figure 3-49)
Figure 3-50  IMS PErformance Analyzer: TRF DB2 Call List

Figure 3-51  IMS PErformance Analyzer: TRF DB2 Summary

**Message Queue reports (forms-based)**

These forms-based reports give you flexibility of tailoring the format and content of your reports for message queues. By requesting just the fields of interest you can control the presentation order and format and also create forms-based extracts and load them into DB2 tables.

The Message Queue reports that IMS Performance Analyzer for z/OS provides show performance details about every transaction that is traced by OMEGAMON XE for IMS on z/OS TRF.

These reports include list and summary:

- Message Queue List
- Message Queue Summary

**Trace reports**

The Record Trace reports format the TRF Extractor records for ease of analysis, and provide a list of transactions, each with detailed information, about every event in the life of that transaction. As a result, you can see when a transaction starts, followed by all the events associated with the transaction in the order they occurred.
**OMEGAMON ATF reports**

OMEGAMON XE for IMS on z/OS provides the Application Trace Facility (ATF), which provides application-level accounting statistics of DL/I, DB2, and MQ calls, call times, and CPU utilization. The ATF data that is collected by OMEGAMON XE, can help you with these tasks:

- Identify transaction response-time components
- Fine-tune applications
- Understand how application programs operate

IMS Performance Analyzer for z/OS provides a set of reports for ATF that process your ATF journals. These reports are grouped by functional category, so you can request reports individually or by category. The following categories are available:

- Transaction Transit reports
- Trace reports
- Extracts

**Transaction Transit reports**

These reports provide application-level accounting:

- The List report (Figure 3-52) can be used to list all transaction event data in chronological sequence.

![Figure 3-52  IMS Performance Analyzer: ATF List](image)

The Summary report (Figure 3-53) gives a statistical analysis of your transaction activity.

![Figure 3-53  IMS Performance Analyzer: ATF Summary](image)
**Trace reports**

The ATF Record Trace reports provide three levels of detail for application-level analysis of DL/I and external subsystem (DB2 SQL and MQ adapter) calls:

- **Trace Overview (Figure 3-54)**

- **Trace Detail (Figure 3-55)**

---

### Trace Overview (Figure 3-54)

- **SDSF OUTPUT DISPLAY** JM3F  J0B29061  DSID  105 LINE 2  COLUMNS 02- 133
- **COMMAND INPUT** =>>>  SCROLL ====>  CSR
- **OMEGAMON ATF Trace Level 1 from 25Jul2011 15.50.38.4 Page 1**

#### Transaction Details

- **Transaction**: CEXTCONV
- **Date-Time**: 2011-07-25-15.50.38.450056
- **Jobname**: IADEMP3
- **User ID**: CEX001
- **Elapsed Time**: 2.793486
- **Total CPU Time**: 0.020277
- **DB2 CPU Time**: 0.001753
- **DLI CPU Time**: 0.011827
- **Control Region CPU**: 0.006075
- **Elapsed time DL/I**: 0.560521
- **Event Types and Counts**:
  - DB2 SQL OPEN: 1
  - DLI CALL (TM) ASRT: 1

### Trace Detail (Figure 3-55)

- **SDSF OUTPUT DISPLAY** JM3F  J0B29061  DSID  106 LINE 2  COLUMNS 02- 133
- **COMMAND INPUT** =>>>  SCROLL ====>  CSR
- **OMEGAMON ATF Trace Level 2 from 25Jul2011 15.50.38.4 Page 1**

#### Event Details

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Time Relative</th>
<th>Duration</th>
<th>Function Verb</th>
<th>Ret Code</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2 SQL OPEN</td>
<td>15.50.38.450056</td>
<td>0.000000</td>
<td>SUMMARY COMPLETION</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>DLI CALL (TM) ASRT</td>
<td>15.50.38.4597906</td>
<td>0.147849</td>
<td>OPEN</td>
<td>-818</td>
<td>2</td>
</tr>
</tbody>
</table>

---

**Figure 3-54**  IMS Performance Analyzer: ATF Trace Level 1

**Figure 3-55**  IMS Performance Analyzer: ATF Trace Level 2
Exception Transaction Extract

The Exception Transaction Extract is an ATF data reduction process, which you can use to create a data set of transactions that generated an exception such as an ABEND or a long response time.

You can extract to either of the following types of data sets:

- ATF Journal: A pre-allocated KSDS data set
- REPRO file: A sequential data set, used for archival purposes, that can later load into an empty ATF journal for reporting
IMS Monitor reporting

In addition to IMS, IMS Connect (through IMS Connect Extension for z/OS Event Journals), and OMEGAMON XE TRF and ATF reporting, IMS Performance Analyzer for z/OS also provides extensive reports for both the online and batch DB Monitor. See Figure 3-57.

IMS Performance Analyzer for z/OS produces a comprehensive set of reports, from management summaries to detailed program traces, organized by level of detail and area of analysis, to meet a broad range of IMS system analysis objectives.

Extensive Report Set for Monitor can assist in evaluation of your IMS Systems in the following areas:

- System performance evaluation: Features in IMS Performance Analyzer for z/OS assist you in monitoring and evaluating IMS system performance on a daily basis.

Management level summary reports express key values in terms of rates, ratios, and percentages. Examples of IMS Monitor reports that can help you use IMS Performance Analyzer for z/OS for trend analysis and comparative analysis of systems are shown in Figure 3-58 on page 78, Figure 3-59 on page 78, Figure 3-60 on page 78, and Figure 3-61 on page 78.

- Schedules completed: 90  Transacts= 90  Trans/sched  1.00
- Schedules generated: 0  Trans= 0  Trans/sched .00
- Schedules incomplete: 0  Trans= 0  Trans/sched .00
- WT-made schedules: 19  Transacts= 34  Trans/sched 1.79
- Term Rec Trans count: 90
- Estimated backouts = 0

** Call data **
- ** MESSAGE CALLS **
  - Region Calls: IWTs  IWTs  Elap/Call  Pct of Call Elap
  - Call No. /Tran /Call /IWT /Tran /Call Sc.Mil.Mic CPY DLA IWT
  - 1 11.9 0.9 0.07 4.234 1.1% 98.9% 1.7% 4.9 0.4 0.07 0.986 9.2% 90.8% 5.9%
  - 2 10.1 1.1 0.11 4.317 1.5% 98.5% 0.8% 3.7 0.8 0.21 5.491 2.3% 97.7% 16.8%
  - 3 9.2 0.6 0.07 2.267 1.4% 98.6% 1.9% 5.1 0.5 0.09 3.690 3.0% 97.0% 7.5%
  - Tot: 9.7 0.7 0.08 4.009 1.2% 98.8% 1.6% 4.1 0.4 0.09 1.979 5.0% 95.0% 9.5%

** Call data (FP) **
- ** E M H CALLS **
  - Region Calls: IWTs  IWTs  Elap/Call  Pct of Calls IWTs  Elap/Call  Pct of Calls IWTs  Elap/Call  Pct of Calls
  - Call No. /Tran /Call /IWAIT /Tran /Call Sc.Mil.Mic IWAIT /Tran /Call Sc.Mil.Mic IWAIT /Tran /Call Sc.Mil.Mic IWAIT
  - Totals all Regions: 49.960.431 3.15 549.080.074

** Call data **
- ** D E D B CALLS **
  - Region Calls: IWTs  IWTs  Elap/Call  Pct of Calls IWTs  Elap/Call  Pct of Calls IWTs  Elap/Call  Pct of Calls
  - Call No. /Tran /Call /IWAIT /Tran /Call Sc.Mil.Mic IWAIT /Tran /Call Sc.Mil.Mic IWAIT /Tran /Call Sc.Mil.Mic IWAIT
  - Totals all Regions: 49.960.431 3.15 549.080.074

** Call data (FP) **
- ** M S D B CALLS **
  - Region Calls: IWTs  IWTs  Elap/Call  Pct of Calls IWTs  Elap/Call  Pct of Calls IWTs  Elap/Call  Pct of Calls
  - Call No. /Tran /Call /IWAIT /Tran /Call Sc.Mil.Mic IWAIT /Tran /Call Sc.Mil.Mic IWAIT /Tran /Call Sc.Mil.Mic IWAIT
  - Totals all Regions: 49.960.431 3.15 549.080.074

** Program Analysis **
- ** Per Schedule **
  - Trans Scd-DLI DLI-Term Schd Elapsed CPUtime Pct of Tran
  - PSBname TranCode Scheds Trans /Schd Mil.Mic Sc.Mil.Mic on Q Calls IWTs Sc.Mil.Mic Mil.Mic CPU Call IWT
  - CEXSPGM CEXSCONV 4 4 1.00 5.592 16.569 .0 10.0 .0 16.569 18.460 111% 35.7% .0%
  - CEXSPGM CEXSNONC 3 3 1.00 4.809 13.795 .0 8.0 .0 13.795 12.879 93.4% 56.4% .0%
  - CEXTPGM CEXTCONV 5 5 1.00 12.020 85.255 .0 10.0 1.8 85.255 30.420 35.7% 84.6% 6.6%
  - CEXTPGM CEXTNONC 4 4 1.00 3.988 60.841 .0 8.0 .3 60.841 27.794 45.7% 25.7% .5%
  - DFSIVP1 IVTNO 1 1 1.00 14.642 270.121 .0 4.0 10.0 270.121 42.406 15.7% 101% 17.5%
  - DFSIVP2 IVTNV 1 1 1.00 4.344 1.780 .0 4.0 1.0 1.780 0.000 .0% 129% 38.5%
  - DFSIVP3 IVTCV 24 24 1.00 5.479 41.357 .0 10.3 .9 41.357 2.236 5.4% 94.8% 2.0%
  - DFSIPV1 PVN1 1 1 1.00 3.462 270.121 .0 4.0 10.0 270.121 42.406 15.7% 101% 17.5%
  - DFSIPV2 PVN2 1 1 1.00 4.344 1.780 .0 4.0 1.0 1.780 0.000 .0% 129% 35.7%
  - DFSIPV3 PVNC 24 24 1.00 5.479 41.357 .0 10.3 .9 41.357 2.236 5.4% 94.8% 2.0%
  - DFSIPV4 PVNT 11 10 .91 44.884 76.844 .0 3.6 1.1 84.548 0.000 0% 99.7% 5.8%
  - DFSIPV5 PVDF 7 7 .88 72.319 0.509 .0 3.6 .0 0.582 0.000 0% 78.1% .0%
  - DFSIPW11 DW11 9 9 .90 5.063 26.800 .0 12.2 .1 26.800 5.538 19.2% 38.7% .9%
  - DFSIPW12 DSW12 8 8 1.00 3.381 75.824 .0 18.9 .8 75.824 7.820 10.3% 65.9% .8%
  - DFSIPW13 DSW13 5 5 1.00 3.352 63.596 .0 8.8 1.2 63.596 5.996 9.4% 71.3% 1.6%
  - DFSIPW14 ADD3 4 4 1.00 3.605 51.639 .0 9.5 1.3 51.639 6.087 11.8% 76.5% 2.3%

Figure 3-58  IMS Performance Analyzer: IMS Monitor: Region Summary, Schedule/Transaction

Figure 3-59  IMS Performance Analyzer: IMS Monitor: Region Summary, Message and Database Calls

Figure 3-60  IMS Performance Analyzer: IMS Monitor: Region Summary, Distributions

Figure 3-61  IMS Performance Analyzer: IMS Monitor: Region Summary, Program Analysis
Chapter 3. Performance monitoring tools

System tuning reports can help you enhance IMS system performance through system tuning and are a key feature of IMS Performance Analyzer for z/OS:

- Summarized IMS Monitor output helps you rapidly identify problem areas.
- Detail analysis reports help you investigate and evaluate these problem areas and also see the effect of changes to the system. See Figure 3-62.

![Figure 3-62 IMS Performance Analyzer: IMS Monitor: Region Activity Analysis, Region Detail](image)

Application and program evaluation through IMS Performance Analyzer for z/OS reports on program activity in message processing or batch regions.

IMS Performance Analyzer for z/OS can be a valuable tool for evaluating existing applications and programs, and for validating whether new applications and programs conform to installation standards.
Program activity reports and program traces add to system documentation (Figure 3-63).

Also reported with IMS Performance Analyzer for z/OS IMS Monitor data are these items:

- External subsystem statistics can be incorporated into Region and Program reports.
- Schedules in progress including wait-for-input (WFI), IMS Fast Path (IFP), and BMPs, are also reported using IMS Performance Analyzer for z/OS and IMS Monitor data.
- Reports on IMS batch programs are also provided with IMS Performance Analyzer for z/OS.

IMS Performance Analyzer for z/OS is a tool for management, system programmers, and technical support personnel. The following list indicates only some of the performance-type areas that IMS Performance Analyzer for z/OS can help you with, when used on a regular basis:

- Improve system performance.
- Improve transaction transit time.
- Manage your IMS Connect Internet communications more efficiently.
- Use IMS regions and message queues more efficiently.
- Reduce virtual and real storage requirements in buffer pools.
- Increase the availability of IMS resources.
- Evaluate applications and programs against system standards before installation.
- Do ongoing system measurement and management reporting.
- Debug IMS applications.
- Increase the productivity of analysts and programmers.
- Determine future system requirements.
- Enhance system and program documentation.
Reduce the need to run IMS utilities.
Reduce the requirement to run the monitor reports supplied with IMS.
Provide auditors with valuable data for a number of potential audit tasks.

For more information about IMS Performance Analyzer for z/OS, see *IBM IMS Performance Analyzer for z/OS Users Guide V4R3*, SC19-3633.

### 3.2 IMS Problem Investigator for z/OS, Version 2 Release 3

In this section, we describe the product features, functionality, and benefits you can obtain when using IBM IMS Problem Investigator for z/OS.

#### 3.2.1 Product overview

IMS Problem Investigator for z/OS is a problem analysis tool for Information Management System Database (IMS DB) and Transaction Manager (IMS TM) version 10, 11, and 12 systems.

IMS Problem Investigator for z/OS is a log analysis tool built both to help IMS systems and application programmers determine the cause of problems and trace the flow of events end to end, and to help identify the cause of IMS database system problems by providing crucial information such as who or what incorrectly updated a database, when the database was updated, and how to reverse the changes. IMS Problem Investigator for z/OS is also useful when diagnosing IMS transaction management performance issues by tracking your IMS transactions end to end (Figure 3-64 on page 82) through IMS and related systems to help in determining transaction times and event latencies for identifying bottlenecks.
IMS Problem Investigator allows interrogation of log files from a variety of sources to assist in the analysis of performance problems associated with your IMS system environment.

IMS Problem Investigator enables IMS administrators and programmers to interactively explore formatted, interpreted, and easily customizable views of log records. With this information, they have the ability to identify and analyze problems quickly, removing the need for expert understanding of log data structures and relationships between log records.

IMS Problem Investigator supports the following types of log record data:

- IMS log
- IMS Transaction Index created by IMS Performance Analyzer
- IMS TM and IMS DB monitor data sets
- Common Queue Server (CQS) log stream and extracts
- IMS Connect event data collected by IMS Connect Extensions
- IMS Connect Transaction Index created by IMS Performance Analyzer
- OMEGAMON XE for IMS Transaction Reporting Facility (TRF) log and extract
- OMEGAMON XE for IMS OMEGAMON XE for IMS on z/OS Application Trace Facility (ATF) journal
- DB2 log

Figure 3-64  IMS Problem Investigator: End to End Transaction Flow
- WebSphere MQ log extract
- SMF - IRLM Long Lock records
- IMS trace table records (67FA, 67FF) in the IMS log
- IMS Repository Audit log stream and extracts (introduced in IMS 12)

You can analyze these records through an ISPF dialog, batch reports, and REXX programming services; and, you can create filtered extracts and CSV files to aid problem investigation.

You can submit batch requests to format Common Queue Server (CQS) and IMS Repository (FRP) log streams or create extracts and CSV files.

IMS Problem Investigator for z/OS can format only CQS and FRP extract files; it cannot directly format CQS and FRP log streams.

IMS logs are a rich source of information about your IMS environment and they provide essential data for many business functions. Without IMS Problem Investigator for z/OS, much of the valuable information in the logs is difficult to access and in some cases even more difficult to understand. Although utility programs such as DFSERA10 can assist in the examination and display of log records, you should have an intimate knowledge of the IMS log data and the relationship of the records in them to be of much use.

With an emphasis on interactive analysis and customization, IMS Problem Investigator for z/OS helps simplify the process of log analysis; you can more easily use IMS and related logs for debugging, performance tuning, tracing, and creating audit trails.

Figure 3-65 shows IMS Problem Investigator for z/OS process overview.
3.2.2 Product features

IMS Problem Investigator offers several unique and significant features to assist problem investigation by making the log data easier to understand and analyze. We examine each of the following log areas:

- Log file selection and extraction
- Log file merge and navigation
- Log record presentation

Log file selection and extraction
You can explicitly select the log files you want to browse. See Figure 3-66.

---

**Figure 3-66** IMS Problem Investigator: Log file selection
You have the following options:

- Select and extract data from multiple IMS and IMS Connect systems (Figure 3-67):
  - Request log data for a time period.
  - Request log data matching filtering criteria.

![Process Log Files](image)

**Figure 3-67  IMS Problem Investigator: Multiple Log File Extract Request**
Use time slicing to analyze large log files more expediently (Figure 3-68).

With time slicing, you can process large log files with the speed and convenience of a small extract by viewing them through a small window of time.

IMS Problem Investigator for z/OS indicates the extent to which the data in each file covers the time slice you specified, so that you can narrow your analysis to only the records that are relevant to the period of time of interest to you.

Use time slicing to merge log files and the overlapping time range is made obvious so you can quickly and easily analyze the data beginning from the point where merged records start intersecting.

IMS Problem Investigator for z/OS processes only data in the time range you specify, treating the log data as though you had previously extracted the data into a smaller file, allowing your session to perform with the responsiveness and convenience of a smaller extract file.
Log file merge and navigation

You can merge together and navigate within various types of data sources by using IMS Problem Investigator for z/OS:

- Merge multiple files to process as a single file (Figure 3-69):
  - Process records from multiple files, merged in time sequence.
  - Merge multiple files to process as a single file.

![Figure 3-69: IMS Problem Investigator: Log File Merging](image)

- Go to an exact point of time in a log file with microsecond precision.
- Navigate forwards and backwards in the record using time increments (Figure 3-70 on page 88).
Move forward and back easily through merged data sources
Navigate directly to a specified time within merged data

Figure 3-70 IMS Problem Investigator: Navigation in Merged Data

- Quickly set labels to remember positions within a file (Figure 3-71).

Figure 3-71 IMS Problem Investigator: Assigning labels

Use L line command to assign labels to log records of interest
Navigate quickly back to labels with the LOCATE primary command (Figure 3-72).

![Figure 3-72  IMS Problem Investigator: Navigation to Labels]

Note that upon exit (PF3) of an investigative session, all assigned labels are lost.
**Log record presentation**

Presentation accesses and interprets the information in the log files so you can do the following tasks:

- List records with (Figure 3-73) or without (Figure 3-74) a brief summary of their contents.

```plaintext
<table>
<thead>
<tr>
<th>Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>05:47:05</td>
<td>IMS 2000 Problem Investigator: Log Record View with summary</td>
</tr>
<tr>
<td>05:47:05</td>
<td>IMS 2000 Problem Investigator: Log Record View without summary</td>
</tr>
</tbody>
</table>
```

![Figure 3-73](image1)  
IMS Problem Investigator: Log Record View with summary

![Figure 3-74](image2)  
IMS Problem Investigator: Log Record View without summary
Select Log records to view extensive details (Figure 3-75).

![Database Update](image)

Select log record to format details

Figure 3-75  IMS Problem Investigator: Log Record formatted view

Highlight fields to see detailed descriptions (Figure 3-76 on page 92).
Highlight flag fields to view the description and state of individual bits in those fields and which ones are on or off (Figure 3-77).

Place cursor in Flag bit fields to see detailed information regarding Flag bits.
Chapter 3. Performance monitoring tools

- Show time in GMT, local, or fixed offsets (Figure 3-78).
- Apply a time zone offset to synchronize time with the MVS system log (Figure 3-78).

**Specify time zone in GMT or local or fixed offset**

**Apply time zone offset if needed to match record times on PMR or joblog**

Figure 3-78   IMS Problem Investigator: Time specification

In certain situations, a log record can be changed by an APAR, and PI might fail to format it. When that happens, the log record can be viewed in DUMP, HEX0, or HEX1 format. See Figure 3-79 and Figure 3-80 on page 94.

![IMS Problem Investigator: Log record in HEX1 format](image)

Figure 3-79   IMS Problem Investigator: Log record in HEX1 format
Build customized forms to customize which fields are shown to you in ISPF formatted log record display (Figure 3-81).
In this pane, you can make the following selections:

- Select only the fields you are interested in, or that are pertinent to the current issue (Figure 3-82).

![Figure 3-82  IMS Problem Investigator: Form field customization](image-url)
– Apply form to log record, and now see only data of interest (Figure 3-83).

**Figure 3-83**  IMS Problem Investigator: Formatted Log Record FORM View
If you want to view the log records by using ISPF browse, toggle between formatted presentation and ISPF-browse presentation (Figure 3-84).

**Figure 3-84  IMS Problem Investigator: Formatted versus ISPF browse views**
Log analysis

Use IMS Problem Investigator for z/OS to streamline your log analysis; you can use a variety of options to do the following tasks:

- Use the TX line command to track log records that belong to the same transaction but might be logged across multiple log files of various types (Figure 3-85).

- When a transaction does program-to-program switches, each transaction uses the same original unit of work ID. Viewing individual transactions within the entire set of transactions this way can be difficult.

You can isolate individual transactions within the set by using the TU line command (Figure 3-86). This command isolates a transaction by its unit of recovery (UOR) token and shows you only the events that are related to the selected transaction.
New log record type support

IMS PI supports all the new instrumentation in the IMS log and monitor:

- The 56FA transaction-level statistics provide detailed performance and resource usage metrics, including CPU time, DL/I call, database I/O and ESAF usage, and is fully exploited by IMS Problem Investigator in terms of reporting, tracking, and filtering.

  If you are not currently collecting the 56FA log record, consider enabling them for all transactions to greatly improve your ability to diagnose transaction performance issues.

- Synchronous call-out (ICAL).

- The 4517 User Exit statistics, 4511 general storage statistics, 4512 IMODULE statistics, 4513 MSC statistics, 4515 64-bit cache storage manager, 4516 Fast Path 64-bit buffer manager, and 4514 EWLM statistics

IBM Tivoli OMEGAMON XE for IMS on z/OS Transaction Reporting Facility (TRF) support

Format and analyze IBM Tivoli OMEGAMON XE for IMS on z/OS TRF records, providing useful monitoring statistics such as DL/I and DB2 call response times. See Figure 3-87 and Figure 3-88 on page 100.
IBM Tivoli OMEGAMON XE for IMS on z/OS Application Trace Facility (ATF) support

The online ATF journals can be merged with the IMS log data allowing you to investigate IMS and DBCTL applications and their DL/I calls. See Figure 3-89 on page 101. You may do any of these tasks:

- Use tracking to see all the application calls that are associated with a single transaction and identify bottlenecks.
- Merge the IMS and DB2 logs, which can provide the complete end-to-end picture of the transaction lifecycle. When you include the ATF journal you can also see how the sequence of application calls affects the lifecycle.
A new IMS system view (6. OMEGAMON ATF Journals) is added to system definitions to define the ATF journal data sets (Figure 3-90).
Open Database (DRDA) analysis using IMS Connect Extensions for z/OS

IMS Connect Extensions for z/OS event journaling provides a comprehensive trace of every IBM Distributed Relational Database Architecture™ (DRDA) request issued by the DRDA application. See Figure 3-91. You can do the following tasks:

- Track application calls associated with a single thread to identify bottlenecks.

Figure 3-91   IMS Problem Investigator: ODBM open database formatting

- Provide easy-to-read formatting of all DRDA code points, for both the open-standard (Figure 3-92 on page 103) and IMS-specific (Figure 3-93 on page 103).
Figure 3-92  MS Problem Investigator: DRDA open source code point formatting

Figure 3-93  IMS Problem Investigator: DRDA IMS specific code point formatting
Analyze DL/I call results including I/O and feedback areas (Figure 3-94).

Figure 3-94  IMS Problem Investigator: DL/I Call Analysis
- Merge the CEX journal with the IMS log to see the complete end-to-end picture of the session thread of a distributed transaction (Figure 3-95).

```
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Date 2010-03-31 Wednesday</th>
<th>Time (GMT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>08</td>
<td>Application Start Program=AUTPSB11 Region=0003</td>
<td>05.46.45.066230</td>
<td></td>
</tr>
<tr>
<td>5607</td>
<td>Start of UOR Program=AUTPSB11 Region=0003</td>
<td>05.46.45.066232</td>
<td></td>
</tr>
<tr>
<td>5616</td>
<td>Start of protected UOW Region=0003</td>
<td>05.46.45.066530</td>
<td></td>
</tr>
<tr>
<td>A0AA</td>
<td>ODBM Trace: Message received from ODBM</td>
<td>05.46.45.067979</td>
<td></td>
</tr>
<tr>
<td>A06A</td>
<td>Message received from ODBM</td>
<td>05.46.45.067977</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 3-95  IMS Problem Investigator: Merged IMS Log and Connect Extension Journal

**DB2 log support**

Format and analyze DB2 Versions 8, 9, and 10 log records, providing useful information about DB2 database accesses from your IMS TM system (Figure 3-96).

```
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Date 2008-01-25 Friday</th>
<th>Time (GMT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>002</td>
<td>DB2 Pageset Control - Open Pageset</td>
<td>01.49.26.685872</td>
<td></td>
</tr>
<tr>
<td>0002</td>
<td>DB2 Pageset Control - Open Pageset</td>
<td>01.49.26.693336</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - Begin UR</td>
<td>01.49.26.694784</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Insert into a Data Page</td>
<td>01.49.26.694800</td>
<td></td>
</tr>
<tr>
<td>0002</td>
<td>DB2 Pageset Control - Open Pageset</td>
<td>01.49.26.697600</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Type 2 Index Update</td>
<td>01.49.26.697712</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Type 2 Index Update</td>
<td>01.49.26.700784</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - Begin Commit Phase 1</td>
<td>01.49.26.700896</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - Phase 1 to 2 Transition</td>
<td>01.49.26.700928</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - End Commit Phase 2</td>
<td>01.49.26.702592</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - Begin UR</td>
<td>01.49.26.702896</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Update In-Place in a Data Page</td>
<td>01.49.26.702912</td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td>DB2 Savepoint</td>
<td>01.49.26.703200</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Insert into a Data Page</td>
<td>01.49.26.703512</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - Enable Commit Phase 1</td>
<td>01.49.27.372992</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - Begin Commit Phase 2</td>
<td>01.49.27.402800</td>
<td></td>
</tr>
<tr>
<td>0020</td>
<td>DB2 Unit of Recovery Control - End Commit Phase 2</td>
<td>01.49.27.404512</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 3-96  IMS Problem Investigator: DB2 log formatting
**SMF record support for IRLM Long Locks**

Format and analyze IRLM Long Lock records (SMF 4F0F - decimal 79.15) to assist in analysis of transaction delays. See Figure 3-97.

These SMF records are created by the IMS Long Lock exit when IRLM detects that an IMS transaction is waiting for a lock exceeding the IMS long lock value.

![Figure 3-97  IMS Problem Investigator: IRLM-SMF Long Lock record formatting](image)

The information in SMF 79.15 (X'4F0F') records allows you to measure transaction event latencies and to investigate performance problems when IMS transactions experience IRLM Long Locks.
This information is provided for an IRLM Long Lock when a database lock conflict occurs during an update (Figure 3-98).

Figure 3-98  IMS Problem Investigator: SMF 4F0F record formatted

Merging IMS and SMF log files, IMS Problem Investigator for z/OS enables end-to-end analysis of an IMS transaction to identify the transaction that is holding and the transaction that is waiting for the lock.

Enhanced trace record formatting

Enhanced formatting of the IMS log trace records (log codes 67FA and 67FF) generated by the /TRACE command. See Figure 3-99.

Figure 3-99  IMS Problem Investigator: ITR trace records

**IMS 12 MSC enhancement using IMS Connect**

Starting with IMS Version 12, IMS Connect can be used to provide Multiple Systems Coupling (MSC) connectivity.

IMS Connect Extensions for z/OS, through its event journaling, provides comprehensive detail about the connect-to-connect events that are associated with MSC transaction processing.
IMS Problem Investigator for z/OS provides the usual interactive event formatting and tracking across the front-end and back-end systems. See Figure 3-100.

**IMS Version 12 Repository Audit log stream**

IMS 12 introduced its repository server for storing your Dynamic Resource Definitions, and can optionally create an audit log stream.

IMS Problem Investigator for z/OS allows you to analyze the repository events, through log stream extract, which include server control and client unit-of-work requests so you can monitor how changes made might affect your IMS Dynamic Resource Definitions. See Figure 3-101 on page 109.
Special log record types

Types include IMS Transaction Index and IMS Connect Transaction Index:

- **IMS Transaction Index:**
  
  This type is a specialized extract file created by IMS Performance Analyzer for z/OS.
  
  - Each record in the index represents an IMS transaction and contains cumulative information from the IMS log about that transaction.
  
  - The index can be processed on its own or merged with the associated IMS log files in an interactive diagnosis session to provide improved levels of problem determination.
  
  - The index is displayed in IMS user log record format, and is denoted by a record code of 'X'CA01' (Figure 3-102).
The index record (Figure 3-103) contains all of the transit data information about the transaction.

This information includes the following items:

- **Transaction identification**
  - Transaction code, terminal, and other identifying names
  - Time stamp of when the transaction started or entered the system

- **Tracking tokens**
  - Unit of work, recovery, OTMA, Connect, and other identifying tokens

- **Event time stamps**

- **Performance metrics**
  - Transit and response time details
  - Processing delays, including Commit Mode 0 and 1 delays that are incurred while waiting for client acknowledgement
  - Additional response time components, including RESUME TPIPE elapsed time delays
  - An abend and other processing failure indicators
– Resource usage

  • CPU time
  • Full-function and Fast Path database calls
  • Database buffer, I/O, and locking
  • Fast Path buffer; VSO and contention
  • External subsystem
  • Input and output message count and lengths

All transaction types are comprehensively supported, including: MPP, BMP, Fast Path, MSC (end-to-end), APPC (including CPI-CI), OTMA (including Connect and MQ).

However, you should be aware of some restrictions when you consider the use of Transaction Index records:

  • If the type 56FA transaction-level accounting records are not being captured, the type X’07’ application termination accounting record is required to provide resource usage estimates, and in a WFI environment the X’07’ log records might not be available in the log.
  • Type 56FA transaction-level accounting records are required for accurate and complete resource usage statistics.

▶ IMS Connect Transaction Index

IMS Performance Analyzer for z/OS can be used to create X’CA20’ IMS Connect Transaction Index records for transactions that are sent to OTMA (Figure 3-104).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Date</th>
<th>LSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA20</td>
<td>Connect Transaction</td>
<td>2011-10-28</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TranCode=PRT2 Userid=CEX2 IMSID=ICDH ClientID=CLNAC02 Port=3801 LogToken=c8962e0e2e33103 SSN=0689 Response=0.041073 CM=1 SYNCELEVEL=1 TOV=38_MIN Socket=Tran</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3-104  IMS Problem Investigator: IMS Connect Transaction Index

– Each record in the index represents an IMS transaction and contains cumulative information from IMS Connect Extensions journal about that transaction (Figure 3-105 on page 112 and Figure 3-106 on page 112).
IMS Connect Transaction Index can be analyzed on its own, or merged with IMS Connect Extensions journals used to create it.

When merged with IMS Connect Extensions event journal and IMS Log, IMS Connect Transaction Index can help you see the end-to-end transaction lifecycle of your IMS Connect transactions.

The Connect Transaction Index, in a similar way as IMS Transaction Index, can be a useful diagnostic mechanism that provides a summary insight to the dynamics of the transaction and provide a shortcut to the cause of the transaction problem.
**Specialized log processing**

Several specialized log processing options are available to you with IMS Problem Investigator for z/OS. The SCRUB utility removes sensitive or confidential user data from IMS log records, preparing them for sending off-site or making them available in-house for problem analysis.

IMS Problem Investigator for z/OS provides two methods for implementing SCRUB:

- A batch utility creates a scrubbed extract data set for sending off-site:
  
  a. Select log data for which you want to build a scrubbed extract (Figure 3-107).

  ![Figure 3-107 IMS Problem Investigator: Requesting BATCH scrub extract processing](image)

  b. Request extract, specify interval, and request scrubbing of sensitive log data (Figure 3-108).

  ![Figure 3-108 IMS Problem Investigator: Building a scrubbed extract](image)
c. Batch processing is carried out and new scrubbed extract is created (Figure 3-109).

Original log data remains unchanged and only the newly created log extract contains scrubbed data.

- An assembler exit can enforce scrubbing of IMS log records prior to display in the ISPF dialog or batch reporting.
  - This exit is a list of authorized users who have the authority to view sensitive data. Any users who are not on this list are allowed only to view the scrubbed data.
  - This exit must be built and linked into your IMS Problem Investigator for z/OS load library.
Batch reporting option: BRIEF

The BRIEF batch reporting option provides a quick and concise way of reporting log records through the batch reporting process. See Figure 3-110.

Only general and high-level information is reported by using this option, similar to the way records are browsed in the dialog. See Figure 3-111.

Examples are in the following list:

- Log code and description.
- High-level identification information, through global fields.
- Time stamps and delta elapsed time between log record events.
Log Information report: LOGINFO

The Log Information (LOGINFO) report (Figure 3-112), which is created each time a batch process is requested (Figure 3-110 on page 115), provides a synopsis of the log record types in the IMS log, including record count, lengths, rate per second, and volume. See Figure 3-113.

---

**Figure 3-112** IMS Problem Investigator: LOGINFO Report output data set

---

**Figure 3-113** IMS Problem Investigator: LOGINFO Report output

---

Selected record types are organized further to provide additional information about transaction arrival and processing rates.

### 3.3 IMS Connect Extensions for z/OS, Version 2 Release 3

In this section, we describe the product features, functionality, components, and benefits you can obtain when using IBM IMS Connect Extensions for z/OS.

IBM IMS Connect Extensions is covered in this book because it provides the instrumentation data for IMS Connect, necessary to do performance reporting and tuning.
3.3.1 Product overview

IBM IMS Connect Extensions for z/OS is a tool that enhances the operation of IMS Connect. IMS Connect, a function of IMS, is the premier pathway for accessing IMS applications and databases by using TCP/IP protocols. See Figure 3-114.

IMS Connect Extensions for z/OS can help you maintain and manage your IMS Connect resources by providing the following product features:

- Monitoring and recording of IMS Connect activity
  IMS Connect Extensions for z/OS provides detailed event records for your IMS Connect activity, and gives the information you need to analyze important metrics:
  - Performance
  - Throughput
  - Resource availability
  - Security

You can also use the information collected by IMS Connect Extensions for z/OS to debug existing clients and new applications.

- Single point of control for multiple IMS Connect systems
  IMS Connect Extensions for z/OS gives you centralized management and control of all your IMS Connect systems. This feature includes the ability to monitor and manipulate MSC physical links and remote IMS Connect systems and their associated resources, from either an ISPF operations dialog or Operations Console GUI client. See Figure 3-115 on page 118.
Enhanced transaction management
Dynamically manage your TCP/IP transactions so you can define rules to automatically distribute your workloads between multiple IMS systems in a sysplex, or reroute messages in the case of an unexpected IMS system outage.

Enhanced Open Database management
With IMS Connect Extensions for z/OS, you can dynamically manage your ODBM (TCP/IP DRDA) requests because you can manage your ODBM resources and define rules to distribute your ODBM workloads.

Improved client services; additional features are provided for IMS Connect clients:
- Enhanced information in error messages
- Password change facility
- Extended message translation.

Using these and other features of IMS Connect Extensions for z/OS gives you the ability to improve the availability, reliability, and performance of IMS Connect. IMS Connect Extensions for z/OS, through its event journals, can also help you speed and simplify problem determination when used in synergy with other tools, such as IMS Performance Analyzer for z/OS and IMS Problem Investigator for z/OS. By using IMS Connect Extensions for z/OS, your IMS Connect systems become more transparent so they are easier to manage and audit.
3.3.2 Product features

The following product features are available to you with the new IMS Connect Extensions for z/OS product:

- **MSC over TCP/IP support**
  IMS Connect Extensions for z/OS introduces MSC and remote ICON (RICON) object types, which are both are supported in the Status Monitor. The MSC session type is also introduced and is supported in the Active Sessions display.

- **EVENTLOGGING**
  This control optimizes performance by buffering events before they are written to the active journal using new CEXCTLIN DD values. Options are as follows:
  - **EVENTLOGGING WRITE=IMMEDIATE**
    This option provides best protection from lost events if IMS Connect region is cancelled purposely or inadvertently.
  - **EVENTLOGGING WRITE=BUFFER**
    This option reduces CPU usage for CEX, but some events can be lost if IMS Connect region is cancelled purposely or inadvertently.

- **Active Sessions FORM pre-fill option**
  This feature automatically selects fields corresponding to the session type being used (OTMA, ODBM, or MSC).

- **Open Database support**
  - **Open Database Manager**
    This support enables routing and monitoring of DRDA requests. The Status Monitor and Active Sessions dialogs are enhanced to display information about ODBMs and aliases.
  - **New ODBM security exit**
    Exit CEXAUTH0 provides the same access control options that are provided for your OTMA workloads. An installation verification program (IVP) is introduced; it produces a DRDA workload.
  - **Event logging**
    The ODBMDEFAULTS keyword is introduced. It is supported in the CEXCTLIN DD statement; it specifies the default behavior for ODBM routing:
    - **DEFAULTACTION=INPUTALIAS**
      When specified, the input alias is copied as the output alias and the ODBM name is set to blank.
    - **DEFAULTACTION=BLANKALIAS**
      When specified, the output alias and the ODBM name are both set to blank.

- **IRM_Timer value display**
  The IRM_Timer value is now displayed on the Active Sessions dialog for OTMA or MSC sessions.
- Command Shell improvements
  The command shell now supports IMS 12 IMS Connect WTOR commands and IMS Connect z/OS commands.
  Support for IMS Type-1 commands is added to the ISPF command shell and the Operations Console.
  The command shell now supports new IMS Connect commands in IMS 12.
- Security enhancements
  Security offers the ability to validate IMS Connect access based on the client IP address and IMS Connect port.

These are only several of the enhancements available in IMS Connect Extensions for z/OS. For a complete product listing, see *IMS Connect Extensions for z/OS User's Guide V2R3, SC19-3632.*

### 3.3.3 Product components

IMS Connect Extensions for z/OS runs in IMS Connect address space and is initiated by IMS Connect at startup. Figure 3-116 shows the main product components.

![Figure 3-116  IMS Connect Extensions: Components](image-url)
Components are described in the following list:

- **ISPF dialog client**
  
  The ISPF dialog client (Figure 3-117) connects through TCP/IP to one or more IMS Connect Extensions for z/OS console listeners, and helps provide centralized monitoring and control of IMS Connect systems across your enterprise. You can also use the ISPF dialog client to configure IMS Connect Extensions for z/OS server. These configuration settings are stored in a VSAM repository.

```
File Menu Help
-----------------------------------------------
Option ====> IMS Connect Extensions 2.3 - Primary Menu
0 Profile Customize your dialog profile
1 Definitions Display or maintain IMS Connect Extensions definitions
2 Operations System Monitor and Control Facility
X Exit Exit IMS Connect Extensions
Definitions repository . . 'REDBOOK.JM3.REPOSITORY'
```

*Figure 3-117  IMS Connect Extensions: ISPF dialog client*

- **Operations Console GUI client**
  
  An Eclipse-based PC application provides a graphical interface to do IMS Connect Extensions for z/OS operations (Figure 3-118 on page 122). With it, you can monitor and control your IMS Connect systems and their components from a distributed workstation. IMS Connect Extensions Operations Console runs under IBM Tools Base Connection Server.
Console listener
This agent runs in IMS Connect address space that listens on a TCP/IP port for connections from IMS Connect Extensions for z/OS clients. The console listener allows clients to access information about an IMS Connect system and issue commands to that system. A console listener provides information about a single IMS Connect system; a single ISPF or GUI client can connect to multiple console listeners.

Repository
This VSAM key-sequenced data set (KSDS) contains configuration data for IMS Connect Extensions for z/OS. With IMS Connect Extensions for z/OS you can manage this repository from the ISPF dialog interface.

Exit manager
When IMS Connect is started, IMS Connect Extensions for z/OS initializes the user exit manager and uses it to load all of the user exits on behalf of IMS Connect and dynamically configure event collection, real-time monitoring, message translation, and workload management for these exits. The exit manager handles both user exits that are supplied by IBM and custom user exits.

Event collector
The event collector collects IMS Connect events and data about the input and output to every function call to every exit, and including data sent to and returned by custom user
exits. Event records are collected in active and archive journals. The event collector is initialized by the exit manager.

- **Status monitor agent**
  This agent provides real-time statistics on message processing activity for a system, IMS Connect message exits, data stores, ODBMs, aliases, and ports. Status monitor statistics are available through both the ISPF Operations dialog or Operations Console (GUI).

- **Active session agent**
  This agent provides real-time tracking of active TCP/IP, DRDA, and LOCAL port sessions. Active session information is available through the ISPF Operations dialog, Operations console, or publisher API. Using the ISPF Operations dialog and Operations Console (GUI), you can view details of active sessions.

- **Publisher interface**
  Provides to monitoring applications real-time data about event records and active sessions running in an IMS Connect region, and contains configuration information about the IMS Connect system. The publisher interface supports IMS Connect monitoring function of OMEGAMON XE for IMS on z/OS.

- **Active and archive journals**
  These files contain IMS Connect event data recorded by IMS Connect Extensions for z/OS. The *active* journals are where the event collector writes the event records. The *archive* journals are sequential data sets, on disk or tape, containing IMS Connect event records archived from the active journals. IMS Connect Extensions for z/OS journals can be analyzed by other IBM IMS tools, such as IMS Problem Investigator for z/OS and IMS Performance Analyzer for z/OS to provide debugging, performance analysis, and auditing information about your IMS Connect systems. IMS Connect Extensions for z/OS also includes a print utility that can format journal data.

- **Archive utility**
  This utility is used to archive your IMS Connect Extensions for z/OS journals.

- **Batch command, definition maintenance, and reporting utilities**
  IMS Connect Extensions for z/OS includes batch utilities that provide additional features such as submission of IMS Connect Extension command from a batch job and creation of resource definitions from a batch job.

### 3.3.4 Product benefits

IMS Connect Extensions provides a solid set of enhancements to IMS Connect functionality of your IMS systems. These enhancements can help you reduce the cost of providing TCP/IP access to your IMS applications, while improving the overall quality of service and increasing the security and audit-ability of IMS.

Manage all your IMS Connect systems from a single user interface. IMS Connect Extensions for z/OS provides two management options, the ISPF Dialog and the Operations Console (GUI). Use either to accomplish the following tasks:

- Centralize monitoring and control of all of your IMS Connect Extensions for z/OS operations.

- Provide an easy-to-use sysplex view of your IMS Connect instances, including remote IMS Connect systems over an MSC link.
Both consoles summarize activity for your groups of IMS Connect instances, and secure your operation functions with flexible access control rules so you can tailor which commands users can issue based on their roles in your organization and access authority.

IMS Connect Extensions for z/OS assists in maximizing IMS and IMS Connect availability, helping you meet service-level objectives with the following features:

- Add, delete, or reload IMS Connect message exits without interrupting IMS Connect.
- Control access to IMS Connect instances through a system authorization facility (SAF) security class.
- Refresh IMS Connect SAF security profiles dynamically.
- Record and report IMS Connect activity.

IMS Connect Extensions for z/OS records information about the transactions processed in IMS Connect. IMS Connect Extensions for z/OS Event Recorder collects records throughout the transaction, and then writes these records out to the Active journals. When the Active journals are full, these records are then written by the Archive Manager to the Archive journals. After being archived, the records can be used for reporting by Performance Analyzer for z/OS, and be used for interactive analysis by Problem Investigator for z/OS. These records provide extensive information about performance, response time, and throughput for your IMS Connect based transactions. Examples are as follows:

- Performance and response time information for IMS, IMS Connect, and user message exits
- Availability information for data stores and ports
- Throughput information for various programming models, such as the following items:
  - Conversational
  - Non-conversational
  - Send Only
- Resource availability

- Optimize TCP/IP performance.

IMS Connect Extensions helps you better manage performance, accurately plan capacity, and dynamically manage workloads.

IMS Connect Extensions for z/OS collects internal event data from IMS Connect and writes that data to journals so that it can be used with IMS Performance Analyzer for z/OS for extensive reporting. See Figure 3-119 on page 125.
By using these products together, you can produce detailed performance reports with useful information including the following examples:

- **Flow control data**, for example ACK, NAK, and OTMA
- **Internal and external transit times and latencies** for IMS Connect transactions
- **Message activities**: OTMA, READ, and XMIT exits, read and write socket utilization, and security and commit confirm elapsed times
- **Network activity**: Port and socket utilization, amount of data processed and accepted, and read and write socket request counts
- **IMS information**: RESUME TPIPE requests, including information about each command type

IMS Connect Extensions for z/OS assists you to manage IMS Connect workloads by providing flexible control of input message traffic. It can balance workloads between eligible data stores and reroute all messages to an alternate data store when a given data store is unavailable. IMS Connect Extensions for z/OS can also reject input messages or generate warnings when user-defined message rates are exceeded.

IMS Connect Extensions for z/OS also caches IMS Connect user ID profiles, reducing the overhead of user ID password validation. This cache can be cleared automatically or manually.

IMS Connect Extensions for z/OS helps you to administer and operate IMS Connect by providing the information you need to diagnose problems in real-time, and then, to fix problems by allowing you to change more IMS Connect Extensions options dynamically.

You can use the ISPF Operations dialog or Operations Console (GUI) to view status, information, and details:

- **Status of all active IMS Connect instances**
- **Information about active sessions**, highlighting delays and inactivity and TCP/IP statistics
- **Details of port status and utilization** for systems or sysplexes
When combined with IMS Problem Investigator for z/OS (Figure 3-120), IMS Connect Extensions for z/OS enables rapid problem determination. Determination includes the ability to view important state event information such as data store and OTMA transaction member (tmember) availability, OTMA failures and time-outs, exit failures, and session errors.

<table>
<thead>
<tr>
<th>BROWSE</th>
<th>CEX220.QAUNIT.EVTLOG.PUT0904(RG4410E)</th>
<th>Tracking active</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>Scroll</td>
<td>Time (GMT)</td>
</tr>
<tr>
<td>A03C Prepare READ Socket</td>
<td>02.33.15.672538</td>
<td></td>
</tr>
<tr>
<td>A049 READ Socket</td>
<td>02.33.15.672614</td>
<td></td>
</tr>
<tr>
<td>A0A4 Event Collection IRM Trace</td>
<td>02.33.15.672642</td>
<td></td>
</tr>
<tr>
<td>A03D Message Exit called for READ</td>
<td>02.33.15.672647</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.15.672704</td>
<td></td>
</tr>
<tr>
<td>A03E Message Exit returned from READ TranCode=PART</td>
<td>02.33.15.672712</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.15.672819</td>
<td></td>
</tr>
<tr>
<td>A041 Message sent to OTMA</td>
<td>02.33.15.672829</td>
<td></td>
</tr>
<tr>
<td>01 Input Message TranCode=PART Source=Connect</td>
<td>02.33.15.673328</td>
<td></td>
</tr>
<tr>
<td>35 Input Message Enqueue TranCode=PART</td>
<td>02.33.15.673354</td>
<td></td>
</tr>
<tr>
<td>08 Application Start TranCode=PART Region=0003</td>
<td>02.33.15.674926</td>
<td></td>
</tr>
<tr>
<td>31 DLT GU TranCode=PART Region=0003</td>
<td>02.33.15.674927</td>
<td></td>
</tr>
<tr>
<td>20 Database Open Database=DI21PART Region=0003</td>
<td>02.33.16.114169</td>
<td></td>
</tr>
<tr>
<td>20 Database Open Database=DI21PART Region=0003</td>
<td>02.33.16.181941</td>
<td></td>
</tr>
<tr>
<td>5E SB Handler requests Image Capture Region=0003</td>
<td>02.33.16.182205</td>
<td></td>
</tr>
<tr>
<td>03 Output Message Response LTerm=7901 Source=Connect</td>
<td>02.33.16.195511</td>
<td></td>
</tr>
<tr>
<td>31 Message GU for APPC LTerm=7901</td>
<td>02.33.16.195533</td>
<td></td>
</tr>
<tr>
<td>33 Free Message</td>
<td>02.33.16.195717</td>
<td></td>
</tr>
<tr>
<td>5610 Syncpoint End of Phase 1 Region=0003</td>
<td>02.33.16.195743</td>
<td></td>
</tr>
<tr>
<td>3730 Syncpoint End of Phase 1 Region=0003</td>
<td>02.33.16.195788</td>
<td></td>
</tr>
<tr>
<td>4893 Free Message</td>
<td>02.33.16.195803</td>
<td></td>
</tr>
<tr>
<td>5612 Syncpoint End of Phase 2 Program=DFSSAM02 Region=0003</td>
<td>02.33.16.195924</td>
<td></td>
</tr>
<tr>
<td>07 Application Terminate TranCode=PART Region=0003</td>
<td>02.33.16.196252</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.16.196857</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.16.197003</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.16.197009</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.16.197232</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.16.197237</td>
<td></td>
</tr>
<tr>
<td>A0A3 Event Collection OTMA Trace</td>
<td>02.33.16.197242</td>
<td></td>
</tr>
<tr>
<td>A0A6 Event Recording EXIT Output Message Trace</td>
<td>02.33.16.197287</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3-120  IMS Connect Extensions: Using Problem Investigator with Connect Extensions

3.4 IMS Buffer Pool Analyzer for z/OS, Version 1 Release 3

In this section, we describe the product features, functionality, and benefits you can obtain when using IBM IMS Buffer Pool Analyzer for z/OS.

3.4.1 Product overview

IBM IMS Buffer Pool Analyzer for z/OS provides a way to analyze database buffer pool performance for both IMS batch jobs (DL/I and DBB) and IMS subsystems (DBCTL and DB/DC).

IMS Buffer Pool Analyzer provides more information than only IMS database buffer pool hit ratios and I/O rates. It provides a way to determine the impact of buffer pool changes before they are made.
3.4.2 Product features

The following product features are available with IMS Buffer Pool Analyzer for z/OS:

- Reviews your buffer pool environment and recommends changes to the number of buffers in each subpool to improve performance.
  
  Specific recommendations are made for the number of buffers for each subpool, and the resulting change in storage usage and reduction in number of database reads.

- Provides information about the efficiency of OSAM cached buffers in a coupling facility (CF), and shows the effect of changing the amount of storage allocated to OSAM buffers in the CF cache structure.

- Reviews the databases that are allocated to each subpool and document subpools that include databases with I/O access ratios that are inconsistent, providing the information that you can use to change your subpool configuration.

- Identifies storage that is wasted, because the most efficient buffer sizes are not available for database data sets.

- Determines whether adding or subtracting buffers will improve the performance of a selected buffer pool.

- Models buffer pool usage to determine I/O rates for various numbers of buffers in each pool.

- Identifies the databases that most heavily use each of the database subpools.

- Performs scenario analysis, including analyzing the impact of creating new buffer pools or changing the block size of a database.

- Reports on IMS OSAM, VSAM, and DEDB look-aside buffer pools, and IMS OSAM buffers that are cached in a coupling facility structure.
  
  The reports provide information that is required to make informed decisions about implementing changes to the structure and sizes of your database buffer pools.

- Requires no changes to any IMS control region or batch job during IMS Buffer Pool Analyzer installation.
  
  IMS can continue running while you are implementing the product. IMS Buffer Pool Analyzer dynamically accesses IMS data sets and control blocks from the trace address space to gather the information that is required to produce buffer pool reports.

3.4.3 Reports overview

IMS Buffer Pool Analyzer offers several reports to help you analyze IMS database buffer pools to provide statistical analysis of the impact of changes that affect buffer pools.

IMS Buffer Pool Analyzer analyzes I/O rates and buffering information for a specific database to facilitate buffer pool changes. These I/O rates and buffering information might be required for changes to a database structure and to develop models to assist you with making informed decisions on the addition of buffers to an existing pool, or sizing requirements for a new buffer pool.
The summary reports that are included in the IMS buffer pool report provide recommendations for changes to your database buffer pool environment. The IMS buffer pool reports include the following items:

- A list of databases that are not using the most efficient buffer size because an eligible subpool is not defined with the most efficient buffer size
  
  The report shows the amount of storage that can be saved if a buffer pool with the most efficient size is defined, based on the average number of buffers that are present in the subpool.

- Recommendations for the number of buffers to be allocated to each buffer subpool
  
  These recommendations result from examining each subpool to identify where added storage will provide the greatest reduction in I/O, and where reducing storage results in the least increase in I/O. This technique takes storage from the subpools that use it least efficiently, and gives additional storage to the subpools where the storage provides the most benefit.

- Detailed information regarding each buffer subpool used in your environment
  
  You can use these reports to help you understand the reasons for and the details of the recommendation sets in the summary reports. The following information is included in these reports:
  
  - Performance information about the current subpool configuration
  - Projections for subpool performance with different numbers of buffers allocated for the subpool
  - A list of each DBDS that used buffers in the subpool during the data-gathering period
  - Performance information for each DBDS

You can create reports for both the buffer pool configuration that was in use when the data-gathering process was performed, and for an updated buffer pool configuration that you define. Based on reports that you create, you can predict the impact of adding a new buffer subpool and allocating several database data sets to that subpool. IMS Buffer Pool Analyzer can then examine your new environment and suggest optimal storage allocation for that configuration.

Based on the IMS buffer pool report, you can model a change to your DBDS environment by specifying that one or more data sets has a different block size, CI size, or access method from that which was used at the time that the data was gathered. This information is helpful when you design a new buffer pool environment where a major DBDS configuration change will be made.

Include a statement in your report job that specifies the access method, either OSAM or VSAM, and the new block size or CI size to be used for each data set. IMS Buffer Pool Analyzer creates a model environment with changes to your data sets. When the block size or CI size is changed, IMS Buffer Pool Analyzer adjusts the individual buffer requests to model the new block number that is required to fulfill the buffer request.

### 3.4.4 Product benefits

IMS Buffer Pool Analyzer for z/OS helps you eliminate guesswork when making changes to your buffer pool configuration, so that you can understand the impact of changes to your system before you implement them. It can also help reduce IMS outages that are needed to make changes to your buffer pool configuration. This product assists you in buffer pool tuning without having to try one change, and then another, and so on. IMS Buffer Pool Analyzer for z/OS also reduces IMS real storage and virtual storage usage without the fear of severe
impact on IMS performance and user response time. You can use IMS Buffer Pool Analyzer for z/OS to model changes and see what the impact of subpool storage reductions will be before you implement any changes.

IMS Buffer Pool Analyzer for z/OS can determine which buffer subpools have the most space for additional storage, and, therefore, will result in the highest reduction in I/O rate when the subpool is increased in size. It can also help you make changes to your buffer pool configuration by adding and removing subpools or changing databases that are using subpools. By understanding what the impact of the change will be on your system, IMS Buffer Pool Analyzer for z/OS allows you to model separate buffer pool configurations and see what the effect is on overall system performance.

IMS Buffer Pool Analyzer for z/OS helps identify database data sets with much higher or lower hit ratios than other database data sets in each subpool. You can use this information to create new subpools that include database data sets that are more similar in access patterns and improve overall buffer pool performance. The product is also useful when making changes to your database access method or block size or CI size. You can model the changes and understand the impact of database changes on your buffer pool configuration. With this information, you can have IMS Buffer Pool Analyzer for z/OS make recommendations about changes to your buffer pool configurations to accommodate the changes that are being made to your database data sets.

Other benefits of IMS Buffer Pool Analyzer for z/OS include OSAM coupling facility caching information. With this feature, you can analyze the benefit of caching OSAM buffers in a coupling facility; and with support for IMS Tools Knowledge Base, you are able to store and retrieve your Buffer Pool Analyzer reports for later viewing.

3.5 Tivoli OMEGAMON XE for IMS on z/OS, Version 4 Release 2

In this section, we describe the product features, functionality, components, and benefits you can obtain when using the Tivoli OMEGAMON XE for IMS on z/OS product.

3.5.1 Product overview

OMEGAMON XE for IMS is a member of the latest generation of IBM Tivoli Monitoring family of mainframe monitoring products. This product can be used to monitor and manage the availability, performance, and resource utilization of your IMS systems at either a system level or within an IMSplex.

OMEGAMON XE for IMS on z/OS can assist you in performing the following tasks:

- Manage performance and availability of IMS systems from a single, integrated interface.
- Track and optimize both resource usage and transaction processing.
- View coupling facility statistics to identify factors that affect the performance of IBM IMSplex environments.
- Use shared queue support to monitor workload balancing and use data sharing support to minimize the impact of locks on shared databases.
- Integrate information from OMEGAMON XE for IMS monitoring agents on z/OS across multiple platforms and third-party software into a single view.
When used in conjunction with other OMEGAMON XE monitoring products, the data, analyses, and alerts presented by OMEGAMON XE for IMS help you develop a holistic view of your entire computing enterprise from a single console.

In the following sections, we explain how OMEGAMON XE for IMS works, describe the resources it provides, and show examples of how it can be used to monitor, analyze, and manage IMS systems, workloads, and shared resources in an IMSplex environment.

### 3.5.2 Provided resources

This section has an overview of the resources that are provided by OMEGAMON XE for IMS.

**Attributes**

Attributes are characteristics or properties of the logical and physical objects monitored by OMEGAMON XE for IMS on z/OS. Attributes define the queries that collect the information presented in your workspace and define situations that trigger alerts and automated actions.

OMEGAMON XE for IMS monitors more than 80 groups of attributes, providing a vast amount of IMSplex and system-level data. You can use these attributes to tailor the information that is presented in workspaces, or to define situations that target specific thresholds, events, or performance problems that you want to monitor.

You might use attributes to create situations that monitor for specific conditions that can require an alert or type of alerts. An example of this is to create a situation that monitors for conditions with a specific severity level. When the values for attributes of alerts that are relayed to a Tivoli Monitoring Server match the values you specify in situations, the managed objects that are associated with the situations change in appearance, alerting you to problems.

**Predefined workspaces**

OMEGAMON XE for IMS on z/OS also offers a variety of default workspaces. The product provides both IMSplex-level workspaces and system-level workspaces, with each of these workspaces or groups of workspaces displaying a specific set of data.

The IMSplex-level views in these workspaces provide data on IMSplex features, such as the following items:

- Coupling facility structures
- Global locks
- Shared queues

The system-level workspaces provide data on IMS system features, such as the following items:

- IMS address space CPU usage and storage usage
- IMS Connect information including response time summaries and detail, TCP/IP usage summary, RESUME TPIPE summaries, ACK and NAK summaries, and Exception Events
- DBCTL Thread summaries and detail
- Device activity
- Extended Recovery and External Subsystem information
- Fast Path Balancing Groups, data entry databases (DEDBs), main storage databases (MSDBs), and Virtual Storage Option (VSO) Data Spaces information
- High availability large database (HALDB) summaries and detail
Chapter 3. Performance monitoring tools

- IMS, IRLM, and Fast Path environment status
- IMS Connect regions
- IMS rate information to help determine the health of an IMS system
- Multiple systems coupling physical and logical links
- Online log data set (OLDS) and write-ahead data set (WADS) statistics
- OTMA statistics by group, Tmember, and tpipe
- Overflow sequential access method (OSAM) statistics
- Pool storage and buffer pool metrics
- Response Time Analysis data
- Startup parameter details
- Virtual Storage Access Method (VSAM) activity measures

You can use the information from these workspaces to manage the performance and availability of systems and their resources, identify potential problems, trace the causes of alerts or exceptions, make tuning and resource distribution decisions, and identify particular conditions you might want to monitor with OMEGAMON for IMS on z/OS.

Predefined monitoring situations
Situations are descriptions of conditions to which you want to be alerted. Situations periodically verify the values of attributes used in the situation description. When they are distributed to systems monitored by OMEGAMON XE for IMS on z/OS agents, situations can, for example, issue a message when a queue count is greater than or equal to 100 messages. Situations can also trigger simple (reflex) actions, or complex automation policies.

If situations are associated with Navigator items, they can generate auditory or visual event indicators, which provide access to special event workspaces that contain more information about the event and guidance for how it should be handled.

OMEGAMON XE for IMS on z/OS provides an extensive set of predefined situations. These situations check for conditions that are typically considered to be problematic or noteworthy. They can also serve as templates for creating customized situations of your own. All these situations include expert advice for handling these conditions should they arise.

Take Action commands
Using OMEGAMON XE for IMS on z/OS, you can define and use Take Action commands, and stop and start processes from within Tivoli Enterprise Portal. You can add a Take Action view to a workspace. You can send an action you have defined or you can enter a command manually through the Command field.

Historical data collection
In addition to providing real-time performance and availability data, you can use OMEGAMON XE for IMS on z/OS to collect data over extended periods of time. By studying the information that is gathered from a historical perspective, you can, for example, identify trends and determine whether current performance is typical or exceptional, or evaluate the effect of tuning decisions. Use the historical data collection function to specify the following information:
- Interval at which data is to be collected
- Interval at which data is to be warehoused
- Location at which the collected data is to be stored
You can view the historical data that is collected by OMEGAMON XE for IMS on z/OS in Tivoli Enterprise Portal workspaces or in reports that are generated by reporting tools like IMS Performance Analyzer for z/OS (see 3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44).

### 3.5.3 Product features

OMEGAMON XE for IMS provides full support for IBM Tivoli Monitoring V6.2.1 or later. In addition, V4.2.0 of this product incorporates new features, enhances existing functionality, and continues migration of functionality from OMEGAMON classic and OMEGAMON II for IMS.

#### Version numbering of components and queries

All components are identified by the version number in the Managed System Status table view of the Tivoli Enterprise Portal. The new version numbering feature helps you to quickly identify the version of each monitoring agent in an environment that comprises multiple versions.

Also, the version number appears in the description of queries in the Situation Editor. Multiple queries with the same name and description are required to support upgrade scenarios. The addition of a version number enables you to identify which query is appropriate for a particular agent at a particular version.

#### Event forwarding

If you are using IBM Tivoli Enterprise Console® or IBM Tivoli Netcool/Omnibus, in addition to IBM Tivoli Monitoring, to manage events in your enterprise, you can now forward events reported by OMEGAMON XE for IMS monitoring agents to these event management products.

Before events can be forwarded, event forwarding must be enabled on the hub monitoring server, and a default destination server must be defined. In addition, the Tivoli Enterprise Console or Tivoli Netcool/OMNibus server (the event server) must be configured to receive the events, a situation update forwarding process must be installed on the event server, and, for events forwarded to IBM Tivoli Enterprise Console the BAROC file for the agent must be installed and imported on the event server.

After situation forwarding is enabled, by default all situation events are forwarded to the specified event server. However, you can customize which situation events are forwarded and to which event server, using the Situation Editor in the Tivoli Enterprise Portal. You might also need to assign an event status compatible with the target event server.

#### Dynamic linking from OMEGAMON XE to OMEGAMON for IMS

Dynamic linking from OMEGAMON XE to its OMEGAMON for IMS interface is made possible by the support for dynamic terminal integration available with IBM Tivoli Monitoring V6.2.1. Dynamic terminal integration is an extension to the Tivoli Enterprise Portal that provides seamless access to TN3270-based applications through context-sensitive links.

A Tivoli Enterprise Portal terminal view enables you to connect to any TN3270, TN5250, or VT100 host system with TCP/IP from inside a Tivoli Enterprise Portal workspace. For 3270 or 5250 terminal views, you also have scripting capability with record, playback, and authoring of entire scripts. By associating a terminal view with a connection script and a query that returns appropriate values, you can configure a view that opens to a specific panel of a 3270 application. This feature is useful for creating contextual workspace links for investigating issues.
Chapter 3. Performance monitoring tools

OMEGAMON XE for IMS for z/OS takes advantage of this capability to create predefined links from several workspaces to target workspaces that contain a related OMEGAMON for IMS screen in a Terminal Emulator view. The data used to connect to the target screens is retrieved from environmental variables specified during configuration of OMEGAMON XE on z/OS monitoring agents using the Configuration Tool.

Response Time Analysis
OMEGAMON XE for IMS on z/OS analyzes the response time of IMS transactions and categorizes the response time into input queue time (I), program input queue time (PI), processing time (P), output queue time (O), internal systems application response time (R0), and total IMS internal response time (R1). See Figure 3-121.

In addition to end-to-end (ETE) response time, host response time, and network response time are displayed. The OMEGAMON XE TEP interface offers comprehensive reporting of response time analysis information based on the data collected in the OMEGAMON classic address space. The classic 3270 Response Time Analyzer (RTA) now features response times that are calculated in millionths of a second.

Monitor the flow of the workload
Use Response Time Analysis to identify problems and outliers

Figure 3-121 OMEGAMON XE: Response Time Analysis
Program isolation and IRLM locking
OMEGAMON XE for IMS on z/OS conveys program isolation (PI) and Internal Resource Lock Manager (IRLM) locking information with workspaces to display local locks held by each IMS region and to display lock conflicts. These new workspaces display data for PI and IRLM systems. See Figure 3-122.

IMS Lock Analysis information in the Tivoli portal

More detailed analysis of lock holders/waiters, and full support for both IRLM and PI locking in the TEP

Lock owner/waiters

Drill into application detail

Figure 3-122  OMEGAMON XE: Lock Analysis
**Bottleneck Analysis**

The Bottleneck Analysis feature gathers and displays the percentage of time that is spent in each of the separate IMS execution states. The OMEGAMON XE for IMS on z/OS portal interface offers comprehensive reporting of bottleneck analysis information based on the data that is collected in the OMEGAMON classic address space (PDEX and MDEX classic major commands). See Figure 3-123 and Figure 3-124 on page 136.

Where is the bottleneck? Use Bottleneck Analysis to identify Waits By Category

![Figure 3-123 OMEGAMON XE: Bottleneck Analysis (Part 1 of 2)](image)

Bottleneck analysis will help identify workload bottlenecks

Bottleneck analysis does a detailed analysis of IMS workload and determines where the workload is spending its time. Delay percentages are broken out for short term and long term intervals.

% delay by category
Application Trace Facility
Application Trace Facility (ATF) is fully integrated into the OMEGAMON Real-time interface. In addition, enhancements include full support of all DL/I calls, WebSphere MQ application support, DB2 SQL statement support, and filtering and grouping in the display function so users can display meaningful subsets of ATF activity. A new component, Journal Logging Facility (JLF), handles archiving of application trace data to VSAM data sets for later reporting. Configuration has been simplified by removing the need for link-edits and updates to IMS control region JCL.

Not supported: Application Trace support is removed from the CUA interface.

Near-Term History
The online Trace Recording Facility was moved to the OMEGAMON real-time interface and renamed “Near-Term History.” Near-Term History, a separate component from Trace Recording Facility for Batch, is specified during OMEGAMON classic address space startup. See Figure 3-125 on page 137.
Chapter 3. Performance monitoring tools

Figure 3-125 OMEGAMON XE: Near-Term History

Changes to this feature include the following enhancements:

- The Journal Logging Facility (JLF) archives Near-Term History data to VSAM data sets for later reporting.
- Classic 3270 commands display summarized and grouped transaction history, including meaningful subsets of transaction information.
- Ability to control collection and status of Near-term History data set in the classic 3270 interface.
- Dynamic Workspace Links (DWL) from Tivoli Composite Application Manager (ITCAM) for Transactions to Near-Term History.
- Ability to specify and activate trace definitions at OMEGAMON startup.

**WebSphere MQ//DB2 support in Dependent Region workspace**

For OMEGAMON XE for IMS on z/OS and the Real-time Monitor (Classic), an external subsystem status indicates when an application running in a dependent region issues an external subsystem request to WebSphere MQ, and also DB2. The external subsystem status reflects the processing that is occurring within the WebSphere MQ subsystem or DB2 subsystem. So if an application is waiting on an external subsystem such as WebSphere MQ or DB2, the region status indicates an external subsystem request has been issued. And the external subsystem status identifies the process the subsystem is performing. In addition, the external subsystem ID is displayed when the application is processing an external subsystem request.
The dependent region displays include OTMA information:

- TMEMBER name
- TPIPE name
- Commit mode
- Synchronization level
- TPIPE status

The OMEGAMON XE for IMS on z/OS Dependent Regions workspace provides OTMA information for an application. In addition, this workspace provides two new links:

- OTMA TMEMBER Status workspace
- OTMA tpipe workspace

The Real-time Monitor (Classic) displays OTMA information for a single region by including a new minor command for the region major commands.

**Long-Wait Lock Owner**

OMEGAMON XE for IMS on z/OS makes it easier to identify lock owners that affect overall system performance and to isolate specific waiters and owners for a given lock. The collected data allows identification of system bottlenecks and determination of the reasons for possible system performance degradations. The Global Lock Conflicts workspace shows the recovery token, the waiter counts, and the status of the top blocker. This workspace also contains a new table view featuring only owners that are considered top blockers. In addition, you can navigate to the Topology workspace to display relationships between owners and waiters for a given resource.

**Log Trace Analyzer**

The product integrates with the Log Trace Analyzer, a tool that links messages that are displayed in the RKLVLOG to information regarding workarounds and solutions that are available as authorized program analysis reports (APARs).

**Reporting capability through Tivoli Common Reporting**

OMEGAMON XE for IMS on z/OS includes reports that run under Tivoli Common Reporting, a reporting tool and strategy common across Tivoli products. Tivoli Common Reporting provides a consistent approach to viewing and administering reports. This reporting environment runs on Windows, Linux, and UNIX.

**Transaction substitution for IMS umbrella transactions**

If you use umbrella transactions in your IMS environment, you can identify those IMS umbrella transactions for which you want a user code to be substituted in place of the IMS external transaction code for data collection and reporting. This way allows IMS umbrella transactions to be monitored based on the individual user codes, rather than the IMS external transaction code, thereby providing improved performance metrics. Response Time Analysis (RTA), Near-Term History (NTH), the Transaction Reporting Facility (TRF batch), and the Application Trace Facility (ATF) support the substitution of a user code in place of the IMS external transaction code for data collection and reporting.

**IMS Health workspace**

The IMS health workspace displays rate information that can be used to determine the health of the IMS system. In addition, a sysplex-level workspace displays a high-level overview of each IMS system being monitored by the agents.
Standardized display of response time and CPU time
Real-time commands for Response Time Analyzer (GRSP, SRSP, and IRSP), Near-Term History (NTVG, NTVS, NTVW, and NTVD), and Application Trace Facility (ATVG, ATVS, ATVW, and ATVD) display response and CPU times as actual values, instead of a fixed format display (Example 3-3).

Example 3-3 Command display output

00:00:00.000321
is displayed as
321 ms

New functions in V4.2.0 Interim Feature 2 (IF2)
This section describes new functions in V4R2.0 Interim Feature 2 (IF2).

Using 64-bit Tivoli Monitoring architecture to display 32-bit unsigned integers
When you have systems that remain active for six or more months, many of the unsigned 32-bit counters in OMEGAMON XE reports exceed the X’7FFFFFFF’ value limit of the Tivoli Enterprise Portal table and graph views. Currently, Tivoli Monitoring supports only signed 32-bit counters, which poses a problem for the display of 32-bit signed counters in OMEGAMON XE agents that run on z/OS systems. OMEGAMON XE for IMS on z/OS V4.2 Interim Feature 2 (IF2) converts z/OS and IMS 32-bit signed counters to 64-bit (signed) fields. This conversion allows the Tivoli Enterprise Portal table and graph views to display these counters correctly.

DBCTL thread enhancements
The information collected and displayed for DBCTL threads is enhanced primarily to include the following unit of work (UOW) metrics:

- The IMS DBCTL Thread Summaries workspace displays summary information for each connected CICS region. Data was added to the IMS DBCTL Thread Summaries workspace in the Tivoli Enterprise Portal and the underlying table for the UOW input rate, UOW processing rate, and input and processed thread counts. From this summary workspace, you can select a given CICS system to display additional information about the DBCTL threads for that system. Thread-elapsed time in microseconds and thread-occupancy percentage were added for each thread.

- A plex-level table and workspace provide DBCTL summary information for each monitored IMS system by a data sharing group. This summary data includes the UOW input rate, UOW processing rate, and input and processed thread counts for each IMS system in the data sharing group.

- The 3270 interface displays DBCTL summary information, including the UOW input rate, UOW processing rate, and input and processed thread counts for each connected CICS. Elapsed time in microseconds and thread occupancy percentage has been added when the individual threads are displayed.

ATF and NTH detail record navigation
Within the Application Trace Facility (ATF) and Near-Term History (NTH) detail records, you can press the PF11 key to display further detail of the same item. By using the PF11 key, you can navigate from summary to overview to detail. Use the PF5 and PF6 keys to navigate at the same display level from one item to the next (summary to summary or detail to detail).

Collection of CPU times consumed in each type of IMS regions per transaction
OMEGAMON XE for IMS on z/OS V4.2 Interim Feature 2 (IF2) displays the total application CPU and CPU used for application calls (DL/I, DB2, and WebSphere MQ) within the ATF.
**TRF Extractor enhancements**

In batch mode, the Trace Recording Facility (TRF) Extractor reads the IMS SLDS data sets and extracts both IMS and TRF log records. If the SLDS does not contain TRF log records, the TRF Extractor ends the job step with a condition code of 0 (zero). Most customers use an automated job scheduling process that checks return codes for each job. A return code of zero implies that all is well; missing TRF records might indicate a problem. The NOTRF parameter can be used to set the condition code (0 - 99) of the TRF Extract job step when no TRF records are found on the IMS logs.

**New functions in V4.2.0 Interim Feature 3 (IF3)**

This section describes new functions in V4R2.0 Interim Feature 3 (IF3).

**Dependent Region DL/I Calls**

The Dependent Region DL/I Calls Details attribute table and the Fast Path Region DL/I Calls workspace (linked to from the Dependent Region workspace) are now removed. This table and workspace contained data only for Fast Path regions. The data from this attribute table and workspace are in the Dependent Regions Statistics attribute table and Dependent Region DL/I calls workspace (linked to from the Dependent Regions workspace). The Dependent Regions Statistics table and the Dependent Region DL/I calls workspace present DL/I call information for all dependent regions including fast path.

**TRF Class and TRF DL/I Summary**

The TRF tables and workspaces, including the Local TRF Class attribute table, IMS TRF Class workspace, Local TRF DL/I Summary attribute table, and IMS Trace Recording Facility (TRF) DL/I Summary workspace, have been removed and are no longer supported.

**Enhanced application metrics**

The Dependent Region and Fast Path Region workspaces and their underlying attribute tables, Dependent Regions and Fast Path Regions, are enhanced to include the following additional application metrics for IMS V10 and later:

- Elapsed time for the executing transaction (UOR) in microseconds
- Number of DL/I database calls, system service calls, and message calls for the executing transaction (UOR)
- Number of external subsystem calls for the executing transaction (UOR)
- Database I/O counts
- Elapsed wait times for intent conflict, pool space, scheduling process, database I/O, and database locking

The Dependent Region and Fast Path workspaces can obtain additional metrics for an executing application by selecting the link to the Dependent Region DL/I Calls workspace which displays data from the Dependent Regions Statistics attribute table. For IMS V10 and higher, existing call counts now represent the calls issued for a given application instance or unit of recovery (UOR); however, prior to IMS V10, the counts reflected the total calls for the application schedule. The following new statistics are provided for IMS V10 and later:

- Counts for additional DL/I message call types
- Counts for additional DL/I database call types
- Counts for DL/I system service type calls
- Number of external subsystem (ESAF) calls
- Counts for the number of reads
- Database I/O statistics
- Number of waits on enqueues
3.6 Transaction Analysis Workbench for z/OS, Version 1
Release 1

This section describes product features, functionality, components, and benefits you can obtain when using the Transaction Analysis Workbench for z/OS product.

Transaction Analysis Workbench is being included in this chapter because understanding today’s complex applications requires the ability to view all components of the transaction in terms of how each component affects performance.

3.6.1 Product overview

Transaction Analysis Workbench for z/OS is a tool for analyzing problems with the performance or behavior of z/OS-based transactions. It provides a platform for investigating logs and other historical data that is collected during transaction processing and system operations.

Because transaction processing today is often a complex process, modern applications frequently interact with external subsystems and use system services; as a result, pinpointing the cause of a problem can be extremely difficult. Transaction Analysis Workbench for z/OS extends the scope of traditional transaction analysis techniques and enables you to more easily identify problems that are caused by internal and external events by allowing you to interactively explore formatted, interpreted, and easily customizable views of log records from various types of log files.

With Transaction Analysis Workbench for z/OS you can identify and analyze transaction problems quickly, without requiring an expert understanding of log data structures and the relationships between log records. It provides a framework for analyzing transaction problems when working in synergy with the following IBM tools for z/OS:

- IBM IMS Connect Extensions for z/OS (see 3.3, “IMS Connect Extensions for z/OS, Version 2 Release 3” on page 116)
- IBM IMS Performance Analyzer for z/OS (see 3.1, “IMS Performance Analyzer for z/OS, Version 4 Release 3” on page 44)
- IBM CICS Performance Analyzer for z/OS (program number 5655-U87), documented in CICS Performance Analyzer for z/OS User’s Guide, SC34-7153

Transaction Analysis Workbench for z/OS can help you analyze the following types of log records:

- IMS
  - IMS log
  - IMS transaction index, created by IMS Performance Analyzer for z/OS
  - IMS Monitor and DB monitor
  - Common Queue Server (CQS) log stream
  - IMS Connect event data, collected by IMS Connect Extensions for z/OS
  - Internal resource lock manager (IRLM) long lock detection (SMF type 79, subtype 15)
  - OMEGAMON XE for IMS Application Trace Facility (ATF) journal and Transaction Reporting Facility (TRF)
- For CICS
  - CICS monitoring facility (CMF) performance class (SMF type 110, subtype 1, class 3)
- For DB2
  - DB2 log
  - DB2 accounting (SMF type 101)
  - DB2 performance (SMF type 102)
- For WebSphere
  - WebSphere MQ log extract
  - WebSphere MQ statistics (SMF type 115, subtypes 1 and 2)
  - WebSphere MQ accounting (SMF type 116)
- For z/OS
  - OPERLOG: the z/OS operations log (log stream)
  - Selected SMF record types applicable to problem analysis (including IBM RMFTM, APPC, job-level accounting, and VSAM activity), in either log streams or dumped SMF data sets

### 3.6.2 Product features

The Transaction Analysis Workbench for z/OS contains features that can help simplify problem analysis in your z/OS environment. These features are designed to be effective whether you are a first responder who is trying to log a problem and collect the necessary data for resolution, or a subject matter expert who is doing a deeper analysis for problem determination.

#### Session manager

The session manager is an ISPF dialog that you use to register problem details and perform analysis. By using it, you can save a history of each problem session, which is useful when resuming an existing analysis or reassigning the analysis to a different subject matter expert.

The session manager can help you to control the lifecycle of a problem:

- Register the problem (Figure 3-126).

---

**Figure 3-126  Transaction Analysis Workbench: Problem registration**
Select the log files needed to analyze the problem (Figure 3-127).

![Figure 3-127](https://example.com/figure3127.png)

**Figure 3-127** Transaction Analysis Workbench: Session log file management
Browse log files individually, or merged in any combination (Figure 3-128 and Figure 3-129 on page 145).

**Figure 3-128  Transaction Analysis Workbench: Browsing log files (Part 1 of 2)**
While browsing log files, tag any log records that are of special interest (Figure 3-130 on page 146).
- Write notes about your analysis (Figure 3-130).

---

**Figure 3-130 Transaction Analysis Workbench: Tagging and notes**

- Resume browsing at the tagged position.
- Reassign the problem to the appropriate subject matter expert.
Run SMF reports, IMS Performance Analyzer reports, or CICS Performance Analyzer reports that are specific to the problem (Figure 3-131).

| Option ====> | 3 |
| Summary . . : Redbook 1 |
| 1 Register Update the problem registration details |
| 2 Files Locate and manage the log files required for diagnosis |
| 3 Reporting Run batch reports |
| 4 Investigate Perform interactive log file analysis |
| 5 History Review the problem history |

Automated file selection

Automated file selection eliminates the usual tedious process of manually locating the data required for problem analysis. All you need to do is specify a date and time period for analysis and the z/OS systems and subsystems that you are interested in; Transaction Analysis Workbench for z/OS then locates the corresponding log files related to the issue.

Currently, Transaction Analysis Workbench supports automated file selection for the following types of log files:

- DB2 log
- IMS log
- IMS Connect Extensions for z/OS journal

Interactive analysis

As with Problem Investigator, Transaction Analysis Workbench presents the associated records in an integrated interactive investigative session on an individual basis or merged together. With enhanced and expanded log record formatting, you can use features such as end-to-end transaction tracking, and TX and TU line commands to identify significant events throughout the transaction’s lifecycle.
Transaction Analysis Workbench for z/OS expands on the capabilities of IMS Problem Investigator for z/OS so you can view end-to-end transaction lifecycle events for your z/OS based transactions, including the following items:

▶ CICS with DBCTL

From a CICS initiated transaction, view the associated events in the IMS log and OMEGAMON journals (Figure 3-132 and Figure 3-133).

▶ CICS or IMS with DB2

From a CICS or IMS initiated transaction, view the associated DB2 accounting records and performance trace events (SMF), and also the DB2 log events.

▶ CICS or IMS with WebSphere MQ

From a CICS or IMS initiated transaction, view the associated MQ accounting records (SMF), and also the MQ log (extract) events.
DB2
View application activity using the DB2 log data and performance trace events.

IMS Connect:
View IMS transaction and DRDA ODBM database activity.

Reporting
The Transaction Analysis Workbench for z/OS currently combines its own reporting with other products, IBM IMS Performance Analyzer for z/OS and IBM CICS Performance Analyzer for z/OS, to help investigate and report on multiple facets of a problem (Figure 3-134).

The following reporting capabilities are currently available by using the Transaction Analysis Workbench for z/OS:

- Use workbench’s own report and extract utility.
  - z/OS system-level analysis, including OPERLOG, address space activity, system resource utilization for CPU processors, virtual storage and page data sets, MVS system logger, and VSAM data set performance
  - DB2 thread accounting, including SQL call elapsed time breakdown
  - WebSphere MQ thread accounting, including GET and PUT call counts and CPU usage
  - APPC/MVS conversational transaction performance analysis.

- Use CICS Performance Analyzer for z/OS.
  CICS transaction performance analysis

- Use IMS Performance Analyzer for z/OS.
  IMS transaction performance and system analysis

Figure 3-134  Transaction Analysis Workbench: Reporting capabilities
3.6.3 Product components

Transaction Analysis Workbench for z/OS consists of the following components:

- ISPF dialog
- Repositories
- Report and extract utility
- Automated file selection utility
- Knowledge modules

ISPF dialog
The Transaction Analysis Workbench ISPF dialog user interface is divided into the following parts:

- Profile: Dialog Settings
  Each user of Transaction Analysis Workbench for z/OS has a profile that consists of personal settings that affect the behavior of the ISPF dialog.

- Sessions: Session Manager
  A session is a collection of information about a particular problem, such as a list of associated log files, reports JCL and output, and history. Transaction Analysis Workbench for z/OS stores sessions in a repository that can be shared between users. These sessions offer a structured, team-oriented framework for problem analysis, enabling you to manage each problem separately, while sharing problem analysis between users.

- System definitions
  A system definition is a collection of information that associates the name of a system (such as an MVS image, an IMS region, a CICS region, or a DB2 system) with details about that system, including the location of the log files to which it writes log records.

  After you define these systems to the Transaction Analysis Workbench for z/OS, you can use the automated file selection utility to locate the corresponding log files, based on the system names and a time period, rather than having to manually locate and specify the log files.

- Controls
  With controls you can customize how the Transaction Analysis Workbench for z/OS presents your log data. Several types of controls are available to you:

  - Filters
    Select only the log records that are relevant to the problem at hand (Figure 3-135 on page 151).
Figure 3-135  Transaction Analysis Workbench: Building a Filter

- Object lists
  These lists allow you to define a set of related field values, such as TRANCODE or IMSID, once, and then refer to that set of values in many filter conditions.

- Forms
  Use forms to select only the fields from a particular log record type that are relevant to the problem.

- IMS user log records
  You may format IMS log records written by IMS applications.

  Process: ad hoc list of log files.

  With this option, you can work with a list of log files without them being associated with any of your existing sessions.
Figure 3-136 and Figure 3-137 on page 153 show a map of the ISPF dialog and its options.
Repositories

Repositories are the data sets where the Transaction Analysis Workbench for z/OS stores data that is related to transaction analysis. These repositories can be shared by multiple users; some of them allow sharing with other products. See Figure 3-138 on page 154.
All four repositories are optional; they may be used, depending on which features of Transaction Analysis Workbench for z/OS you require for your installation.

The repositories are divided into the following two categories:

- **Problem analysis repositories**
  
  These repositories contain information that Transaction Analysis Workbench will use to help you analyze problems and includes both the Session and Control Repositories. The Session repository contains information pertinent to your sessions; the Control repository contains any filters, forms, and object lists you defined and can be shared with IMS Problem Investigator for z/OS.

- **System definition repositories**
  
  These repositories contain system definitions that Transaction Analysis Workbench for z/OS uses for automated log file selection. You need these repositories only if you want to use the Transaction Analysis Workbench for z/OS automated log selection utility.

  There are two types of system definition repositories:

  - **IMS system definition repository**
    
    This repository can be shared with IMS Problem Investigator for z/OS and Performance Analyzer for z/OS and is required only if you intend to use Transaction Analysis Workbench for IMS automated log selection.

  - **CICS, DB2, WebSphere MQ, and MVS Image system definition repository:**
    
    This repository can be shared with the CICS Performance Analyzer for z/OS and is required only for DB2 automated log selection.

Transaction Analysis Workbench can create, write to, and read from all of these repositories.
Report and extract utility
The report and extract utility is a batch program that creates reports, extracts, and comma-separated value (CSV) files.

Reports
This utility generates the following types of reports:

- Formatted record reports
  These reports reproduce the formatting that the ISPF dialog displays when you browse an individual log record.

- OPERLOG reports
  These reports display formatted listings of OPERLOG records.

- SMF reports
  These reports display information that is specific to a particular SMF record type or subtype. The report layout is specific to the record type or subtype, and is available only for some SMF record types/subtypes.

Extracts
An extract is a file that contains log records that are copied from one or more log files. You can use extracts in Transaction Analysis Workbench as substitutes for the original log files. An extract can contain any combination of log record types, including combinations that do not occur in the original log files, such as SMF records and IMS records together.

Extracts are extremely useful for copying selected log records from log streams or log files that are stored on tape, or conserving storage and processing when you need to keep only a subset of the original log files.
Automated file selection utility
The automated file selection utility (Figure 3-139) is a batch program that locates the log files for a session, based on a time period and a system definition.

Figure 3-139  Transaction Analysis Workbench: Automated file selection

When you register a session in the Transaction Analysis Workbench for z/OS ISPF dialog, you can choose to manually specify the data set names of the associated log files.
Automated Log selection is available for the following types of data:

- **DB2 log**
  
  To select DB2 log files, the utility uses output from the DB2-supplied print log map utility (DSNJU004). This output lists the locations of DB2 log files, based on information in the DB2 bootstrap data set (BSDS).

- **IMS log**
  
  To select IMS log files, the utility uses the IMS database recovery control (DBRC) API to get the locations of IMS system log data sets (SLDS) or online data sets (OLDS) from the RECON data sets.

- **IMS Connect Extensions journals**
  
  To select IMS Connect Extensions journals, the utility reads IMS Connect Extensions definition repository.

The utility saves the details of the located log files to the session, which is stored in the session repository.

**Knowledge modules**

A knowledge module is a file, more specifically a type of executable load module, containing information that the Transaction Analysis Workbench for z/OS uses to format a particular type of log record. Each knowledge module contains the following information about a particular type of log record:

- The structure and data types of its fields
- Descriptions of each field that are used to display these descriptions when you zoom on a field while browsing logs in the ISPF dialog
- Which fields in the record type map to Transaction Analysis Workbench global fields

With the Transaction Analysis Workbench for z/OS you can also create your own knowledge modules for IMS log records, referred to as user log records, and analyze these records the same way you would any other IMS log record.

### 3.6.4 Product benefits

Transaction Analysis Workbench contains many of the features of IMS Problem Investigator, with additional features exclusive to Transaction Analysis Workbench that make it beneficial in certain environments, as in the following examples:

- If your installation is an IMS TM shop with DB2, but you have no need for DB2 performance or accounting data, IMS Problem Investigator can be the correct choice for you.
- If your installation is CICS DB2 or DBCTL, Transaction Analysis Workbench provides expanded formatting of SMF data for CICS and DB2 accounting data, making it a better choice for you.
It depends on your installation. Although these two products are similar in the way they look and handle, several significant functional differences exist:

- **Log browser**
  - Transaction Analysis Workbench for z/OS provides additional knowledge modules for SMF records and OPERLOG records.
  - Transaction Analysis Workbench for z/OS enables you to browse log streams without first offloading log stream records to an extract.
  - In addition to working with a personal ad hoc list of log files, Transaction Analysis Workbench for z/OS enables you to define sessions, each with its own list of log files. Users who share the same session repository can select a session and use its list of log files, rather than having to insert that set of log files into their own ad hoc list or session.

- **Automated file selection utility**
  - Transaction Analysis Workbench for z/OS provides an improved process for the batch automated log selection process.
  - Automated log selection now provided for DB2 log in addition to IMS and IMS Connect Extensions Journals.

- **Report and extract utility**
  In Transaction Analysis Workbench, the REPORT command has two additional parameters, SMF and OPERLOG, which create reports that are available only in the Transaction Analysis Workbench for z/OS.

### 3.6.5 Features introduced by Transaction Analysis Workbench

The following features are offered by Transaction Analysis Workbench:
- Knowledge modules for SMF record types
- Sessions
- Requesting IMS Performance Analyzer and CICS Performance Analyzer batch reports
- SMF reports
- Automated selection of DB2 logs
- Support for log streams

### 3.7 Summary

This chapter and Chapter 1, “Performance monitoring with IMS utilities” on page 1 described the most commonly used tools and utilities that technical professionals use when doing system and transaction performance evaluations or system and application tuning. The tools and utilities we discussed include both online monitoring and offline analysis capabilities. When used in synergy with each another, these tools and utilities, and the reports and log information that they provide can be powerful allies when you do any type of performance evaluation and tuning.
Platform-related performance functions

In this chapter, we provide information about IMS performance-related items. We explore IMS 12 enhancements to several IMS system parameters, which can directly affect performance.

This chapter contains the following topics:
- Dynamic full-function buffer pools
- Increased VSAM pools
- Logging enhancements
- Disk technology on log performance
- WADS redesign
- System pools in 64-bit real storage
- 64-bit ACB processing
- zAAP or zIIP times in dependent region accounting log records
- IMS use of zIIP
- Member online change NAMEONLY option
- IMS and Workload Manager (WLM)
- Program load options and IBM Language Environment
- Extended address volume for non-VSAM data sets
- OTMA support for asynchronous IMS-to-IMS communications
- MSC TCP/IP
- Disabling sysplex serial program management (SSPM)
- APPC and OTMA shared queue enhancements
- OTMA ACEE reduction for multiple OTMA clients
- OTMA performance enhancement
- IRLM Version 2.3 and 2.2
4.1 Dynamic full-function buffer pools

With IMS 12, users can dynamically add, change, and delete VSAM and OSAM buffer pool definitions for full-function databases. This support is provided by using new specifications in the DFSDFxxx proclib member in conjunction with the UPDATE POOL command. With this support, full-function buffer pools can be managed without restarting IMS. IMS is able to internally quiesce application read and update activity to allow the UPDATE POOL command to complete with little disruption to transaction workloads. With the ability to dynamically update full-function buffer pool specifications, system availability and application performance are better when the buffer pools are sufficient to avoid unnecessary I/O.

4.1.1 Performance characteristics of the UPDATE POOL command

OSAM waits for the target subpool to be “not owned” and goes through the following buffer pool quiesce processing:

1. Applications can own one OSAM (or VSAM) buffer at a time.

2. UPDATE POOL command quiesces buffers only after application ownership of buffer goes to zero.

3. After buffer is quiesced, applications must wait for the buffer to be reconfigured.

4. Buffer reconfiguration causes the following occurrences:
   a. Subpool is purged and altered buffers are committed.
   b. Subpool and buffer prefixes are released.
   c. Fixed pages and CF resources are released.
   d. Local cache is released.
   e. Sequential buffers are invalidated.

5. After reconfiguration of buffers, applications that are requesting a buffer resume processing.

VSAM waits for DL/I activity to reach commit points and goes through the following buffer pool quiesce processing:

1. IMS looks at program specification tables (PSTs) to find program specification blocks (PSBs) with intent (read or update) on databases.
   
   If intent is found, PST is quiesced at the commit point.

2. When all PSTs are quiesced, the following events occur:
   - Buffer pools are purged.
   - Open database data sets are closed and reopened (applies to both READ and UPDATE access).
   - New applications with sensitivity to database are held until UPDATE POOL command completes.
   - Old subpool is deleted and new subpool is created.

4.1.2 Difference between OSAM and VSAM quiesce processing

VSAM and OSAM quiesce processing differs:

- For VSAM, IMS waits for the application to reach sync point in the sync point processor.
- For OSAM, IMS waits in the buffer handler for the application ownership of the buffer going to zero.
Therefore, the impact of issuing the UPDATE POOL command against an online IMS environment differs, depending on the target of the command: VSAM or OSAM buffer pools.

In controlled performance studies that were conducted at IBM Silicon Valley Laboratory, we observed a certain benefit of using the UPDATE POOL command.

**Benefit:** Using the UPDATE POOL command against OSAM databases performs more quickly and with less online activity impact than issuing the command against VSAM databases.

### 4.2 Increased VSAM pools

IMS 12 allows up to 255 VSAM database buffer pools; previous versions were limited to 16 pools. The limit for online systems is 255 and 254 for batch jobs and utilities. Each buffer pool can have separate subpools for different buffer sizes and for data and index components.

More VSAM pools and subpools can be specified, which gives the capability to tune VSAM pools for increased database performance.

VSAM buffer pools are defined with POOLID statements in the DFSVSMxx member or DFSVSAMP data set. IMS 12 allows users to specify up to 255 POOLID statements.

z/OS 1.11 APAR OA32318 is a prerequisite for defining more than 16 VSAM pool IDs.

**Benefit:** Having more VSAM pools and subpools gives the capability to tune VSAM pools for increased database performance.

### 4.3 Logging enhancements

The IMS logger is enhanced. The OLDS and SLDS can be extended format data sets in IMS 12. Extended format enables the use of striping, which can increase the maximum data rates for logging. Log buffers are optionally moved above the 2 GB boundary in virtual storage. Previous versions of IMS allowed users to place the log buffers above the 2 GB boundary in real storage, but not in virtual storage. These enhancements can contribute towards increased logging rates while using less storage in extended common storage area (ECSA).

#### 4.3.1 Striping of OLDS and SLDS

Striping allows multiple concurrent I/Os for sequential processing:

- Data set is spread across multiple volumes.
- Logging rates are increased.
- Extended format is specified with “EXT” data set type.
- JCL allocation requires DATAACLAS and STORACLAS parameters on DD statement.
- Striping is invoked for extended format data sets when the storage class has sustained data rate (SDR) value of 5 or higher.
Table 4-1 on page 164 shows the logging rate with no striping, Table 4-2 on page 165 shows the logging rate with 2 stripes and Table 4-3 on page 165 shows the logging rate with 4 stripes, reaching a maximum of 351 MB per second on a DS8800 storage device.

**Benefit:** Higher logging rates can be achieved with the enhanced IMS 12 logger and sequential access method (SAM) striping.

### 4.3.2 OLDS buffers specifications

OLDS buffers are allocated above the 2 GB boundary, note the following specifications:

- Is specified with BUFSTOR=64 on OLDSDEF statement in DFSVSMxx member of the IMS PROCLIB data set.
- Puts buffers in 64-bit virtual storage.
- Requires that block size is 4 KB multiple.
- OLDS must be an extended format.

Note the following information about the OLDS block size:

- Optionally, it is specified with BLKSZ parameter on the OLDSDEF statement in the DFSVSMxx member.
- Previously, it obtained from the data set characteristics
- The BLKSZ= parameter should be used.

If BLKSZ and BUFSTOR=64 are both specified, note what happens:

- IMS rounds up the number of buffers (BUFNO=) to a megabyte boundary.
- The 64-bit storage is acquired in megabyte increments.
- The number of buffers created in the acquired storage is maximized.

**Benefit:** Obtain ECSA virtual storage constraint relief (VSCR) by moving the log buffers above the 2 GB bar.

### 4.3.3 Migration to buffers above the 2 GB bar

The following considerations are for buffers above 2 GB bar:

- Be sure that all OLDS are in extended format for the buffers to be above the 2 GB bar.
- Ensure that the Logger Exit (DFSFLGX0) or RSR Log Filter Exit (DFSFTFX0) is capable of handling buffers above 2 GB bar.
- Note that the IMS supplied DFSFTFX0 exit was modified to handle buffers above the 2 GB bar.
- The BUFSTOR=64 must be specified on the OLDSDEF statement.
- IMS must be restarted.
- Restart operations can be cold or warm.
4.3.4 Migration to striping and buffers in 64-bit storage

We list two possible sets of migration steps to implement striping with OLDS and buffers in 64-bit storage:

One possible set of migration steps is as follows:

1. Define new OLDS data sets with extended formatting.
2. Use data class in which data set type is EXT (extended format).
3. Set the storage class with sustained data rate (SDR) to a value greater than 5 MBps (one stripe for every 4 MBps).
4. Define MDAXxx members for dynamic allocation.
5. Start new OLDS data sets with /START OLDS commands. Note the following information:
   - Striping will be used for these data sets
   - A mixture of extended format, basic format and large format OLDS is acceptable when buffers are below the 2 GB bar
6. Stop old OLDS data sets with the /STOP OLDS command.
7. Specify a BUFSTOR=64 on OLDSDEF statement.
8. Specify BLKSZ= on OLDSDEF statement.
9. Terminate and restart IMS to get buffers above the bar.

Another set of possible migration steps is as follows:

1. Terminate IMS.
2. Rename the OLDS data sets.
3. Define new OLDS data sets using the old data set names.
4. Use data class in which EXT is the data set type.
5. Use storage class with SDR greater than or equal to 5 MBps.
6. Copy the old OLDS data sets to the new OLDS data sets.
7. Specify BUFSTOR=64 and BLKSZ= on OLDSDEF statement.
8. Restart IMS.
   - Striping will be used on all OLDS.
   - Buffers will be above the bar.
4.4 Disk technology on log performance

A comparison test was run between DS8700 and DS8800 disk subsystems at IBM Silicon Valley Laboratory. The results are described in this section.

4.4.1 Test environment

The following test environment configuration is what we used:

- z196
- 5 CPUs for 1-IMS on LPAR1
- 4 CPUs for TPNS on LPAR2
- 72 IFP regions
- 4000 logical terminals (LTerms)
- z/OS 1.13 for IMS 12
- z/OS 1.12 for IBM Teleprocessing Network Simulator (TPNS) V3.5

The testing was done as follows:

- Full CI logging; size is 74 KB per transaction.
- Log buffer is 2516 KB.

4.4.2 Results

Three test runs were done with the following configurations:

- No striping
- Two stripes
- Four stripes

In the following tables, measurements are listed for IMS Performance Analyzer for z/OS (IMS PA) and IBM z/OS Resource Measurement Facility™ (RMF).

Table 4-1 shows results when no striping was used.

<table>
<thead>
<tr>
<th>Table 4-1 Results when no striping was used</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS8700</td>
</tr>
</tbody>
</table>

**IMS PA**

<table>
<thead>
<tr>
<th></th>
<th>DS8700</th>
<th>DS8800</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLDS writes per second</td>
<td>155 MB</td>
<td>207 MB</td>
<td>+52 (+34%)</td>
</tr>
<tr>
<td>Transactions per second</td>
<td>2143</td>
<td>2871</td>
<td>+728 (+34%)</td>
</tr>
<tr>
<td>Waits for writes per second</td>
<td>2,143</td>
<td>2,859</td>
<td>+716 (+33%)</td>
</tr>
<tr>
<td>Logbuf waits per second</td>
<td>869</td>
<td>715</td>
<td>-154 (-18%)</td>
</tr>
<tr>
<td>WADS EXCPVRs per second</td>
<td>16.52</td>
<td>10.44</td>
<td>-6.08 (-37%)</td>
</tr>
</tbody>
</table>

**RMF**

<table>
<thead>
<tr>
<th></th>
<th>DS8700</th>
<th>DS8800</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLDS I/O activity per second</td>
<td>662</td>
<td>882</td>
<td>+220 (+33%)</td>
</tr>
<tr>
<td>OLDS I/O response time in ms</td>
<td>1.48</td>
<td>1.10</td>
<td>-0.38 (-26%)</td>
</tr>
</tbody>
</table>
Table 4-2 shows results when two stripes were used.

<table>
<thead>
<tr>
<th></th>
<th>DS8700</th>
<th>DS8800</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IMS PA</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLDS writes per second</td>
<td>274 MB</td>
<td>310 MB</td>
<td>+36 (+13%)</td>
</tr>
<tr>
<td>Transactions per second</td>
<td>3797</td>
<td>4289</td>
<td>+492 (+13%)</td>
</tr>
<tr>
<td>Waits for writes per second</td>
<td>3.785</td>
<td>4.291</td>
<td>+ 506 (+13%)</td>
</tr>
<tr>
<td>Logbuf waits per second</td>
<td>3,190</td>
<td>3,424</td>
<td>+234 (+07%)</td>
</tr>
<tr>
<td>WADS EXCPVRs per second</td>
<td>4.15</td>
<td>1.86</td>
<td>-2.29 (-55%)</td>
</tr>
<tr>
<td><strong>RMF</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLDS I/O activity per second</td>
<td>1166</td>
<td>1468</td>
<td>+302 (+26%)</td>
</tr>
<tr>
<td>OLDS I/O response time in ms</td>
<td>1.62</td>
<td>1.28</td>
<td>-0.34 (-21%)</td>
</tr>
</tbody>
</table>

Table 4-3 shows results when four stripes were used.

<table>
<thead>
<tr>
<th></th>
<th>DS8700</th>
<th>DS8800</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IMS PA</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLDS writes per second</td>
<td>308 MB</td>
<td>351 MB</td>
<td>+43 (+14%)</td>
</tr>
<tr>
<td>Transactions per second</td>
<td>4250</td>
<td>4850</td>
<td>+600 (+14%)</td>
</tr>
<tr>
<td>Waits for writes per second</td>
<td>4,263</td>
<td>4,852</td>
<td>+589 (+14%)</td>
</tr>
<tr>
<td>Logbuf waits per second</td>
<td>3,197</td>
<td>3,860</td>
<td>+663 (+21%)</td>
</tr>
<tr>
<td>WADS EXCPVRs per second</td>
<td>2.01</td>
<td>0.98</td>
<td>+1.03 (+51%)</td>
</tr>
<tr>
<td><strong>RMF</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLDS I/O activity per second</td>
<td>2518</td>
<td>3750</td>
<td>+1232 (+49%)</td>
</tr>
<tr>
<td>OLDS I/O response time in ms</td>
<td>1.128</td>
<td>.667</td>
<td>-0.461 (-41%)</td>
</tr>
</tbody>
</table>

**Observation:** Upgrading disk technology for OLDS data sets can improve IMS logging rate. Logging rates over 350 MBps were achieved by using the IMS 12 Logger with 64-bit virtual buffering and sequential access method (SAM) striping on DS8800.

## 4.5 WADS redesign

The IMS write-ahead data set (WADS) contains a copy of committed log records that are in the OLDS buffers, but have not yet been written to the OLDS because the OLDS buffer is not yet full. IMS 12 changes the way that WADS are written. The concept of track groups is no longer used with IMS 12. This concept changes the calculation for the space required for the WADS and changes the data written by log-write-ahead requests.
In IMS 12, WADS should be sized to provide enough space for the data in the OLDS buffers, which are not yet written to disk, at any time plus one track. This way can dramatically reduce the space requirement for the WADS. In previous versions, the WADS was sized by using the WADS track-group concept. The number of tracks in a track group was calculated as follows:

\[
\frac{\text{OLDS block size}}{\text{WADS segment size}} + 1
\]

A WADS segment size was 2 KB for OLDS buffers below the bar in real storage, and 4 KB for OLDS buffers above the bar in real storage.

For example, an installation with 200 24 KB OLDS buffers above the bar can use a WADS with 1400 tracks.

With IMS 12, a system with 200 24 KB buffers would require no more than enough tracks to hold 200 x 24 KB plus one track, which would be approximately 101 tracks.

In previous versions of IMS, the WADS was written in segments from the OLDS buffers. Successive writes were to different tracks. The scheme is much simpler in IMS 12. Each WADS write is to the next block in the data set. This way generally provides improved performance with modern storage systems. Their algorithms for cache usage generally favor sequential writes. OLDS buffers are conceptually divided into 4 KB segments. When a write-ahead request is made, the WADS writes include the current 4 KB segment of the OLDS buffer and any preceding segments that were not previously written. This request can include segments in the same OLDS buffer or segments in previous OLDS buffers, which are not yet written to an OLDS data set.

To provide the best response times for WADS writes, keep the WADS data in the cache of the storage subsystem. Because the WADS is written in a wrap-around fashion, the entire WADS should be in the cache for optimum performance.

**Benefits of WADS redesign**

In controlled studies at IBM Silicon Valley Laboratory, IMS 12 with the enhanced WADS demonstrated an improvement in WADS device response time compared to IMS 11 WADS response time.

Table 4-4 shows IMS version 12 WADS device average response times, in milliseconds, and the percentage of improvement.

<table>
<thead>
<tr>
<th>IMS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS 11</td>
<td>.384 ms</td>
</tr>
<tr>
<td>IMS 12</td>
<td>.344 ms</td>
</tr>
<tr>
<td>Improvement</td>
<td>10.41%</td>
</tr>
</tbody>
</table>

**Observation:** IMS 12 demonstrated improvements of approximately 10% in WADS device response time in comparison to IMS 11 with the same full-function HALDB workload.
4.6 System pools in 64-bit real storage

The following pools are moved to 64-bit real storage but remain in 31-bit virtual storage:

- DBWP: Database work pool
- DLDP: DMB pool
- DLMP: CSA PSB pool
- DPSB: DL/I PSB pool
- PSBW: PSB work pool

Benefits are as follows:

- Reduction in 31-bit fixed real frames for fixed pools
- Some users are now able to fix these pools; previously, they were constrained by 31-bit real storage.

The most likely pools to be affected by this are the CSA PSB pool and the DL/I PSB pool because they tend to be the largest pools at most installations.

Benefit: It is now possible to have long-term page-fix CSA PSB pool and the DL/I PSB pool.

4.7 64-bit ACB processing

With IMS 12, non-resident application control block (ACB) members can be cached in a 64-bit storage pool to potentially improve the performance of program scheduling and reduce the usage of 31-bit storage.

Prior to 64-bit ACB processing, the following load occurs:

- Resident DMB and PSB are loaded at control region initialization.
- All DEDBs are loaded at control region initialization.
- Non-resident DMB and PSB are loaded on demand.

With 64-bit ACB processing, the following loading occurs:

- Non-resident DMB and PSB are loaded into 64-bit storage.
- Data entry databases (DEDBs) are not loaded into 64-bit storage.
- Users can change the resident option if they want more loaded into 64-bit storage.

By changing the resident option for a PSB, the user will have a smaller resident pool and also some MVS common service area (CSA) reduction. No performance degradation is expected at scheduling because the PSB is obtained from the 64-bit pool.
4.7.1 Defining 64-bit ACB

Caching ACBs in the 64-bit storage is enabled by specifying the ACBIN64 parameter on the DFSDFxxx PROCLIB member:

```
ACBIN64= (NNN)
```

The specification is in gigabytes, where NNN can be a value in the range of 1 - 999. You can display information about the ACBs that are cached in 64-bit storage by using the type-2 command `QUERY POOL TYPE(ACBIN64)`.

At scheduling, the following events occur:

- At first scheduling, a PSB and any DMBs are loaded into the non-resident pool. These members are also loaded in 64-bit storage.
- At the next scheduling, the member is obtained from 64-bit storage instead of reading it from ACBLIB.
- OLC will delete the ACB members in 64-bit storage. You can add or change individual members of the ACB library and bring these members online while IMS is running by using the ACB library member online change function.

**Benefit:** Load ACB members (DMB and PSB) into 64-bit storage to reduce scheduling delays.

4.7.2 Creating and sizing the 64-bit storage pool

You can make your calculations based either on DASD allocation or on the sizes of the 31-bit non-resident pool sizes.

To calculate how much 64-bit storage must be allocated for PSB and DMB ACB members, calculate how much storage is being used for all the non-resident PSBs and non-resident DMBs in the ACB library. DEDB DMBs always reside in 31-bit storage. Because DMBs and PSBs both reside in the same 64-bit pool, sufficient storage must be allocated to accommodate all these members.

Note the following examples:

- The current DASD space allocation for all the non-resident PSBs and non-resident DMBs in the ACB library is 2150 cylinders on a 3390-9 device type, which is approximately 1.9 GB of DASD space. To have sufficient space to include as many ACB members as possible in the 64-bit pool, specify ACBIN64=2.
- There are 57000 ACB members in the ACBLIB data set. The cumulative size of these ACB members is approximately 850 MB. The size of the non-resident DMB pool is 250 MB. The size of the non-resident PSB pool is 300 MB. Therefore, specify ACBIN64=1.

4.7.3 QUERY POOL command

The QUERY (or QRY) POOL command displays information about the current usage of the buffers that are managed by the Fast Path 64-bit buffer manager, processor storage utilization statistics for the pool type specified, and full-function database (OSAM or VSAM) buffer pools:

```
QRY POOL NAME(ACBIN64) SHOW(ALL)
```
The most important output from the command is as follows:

- **PoolNm**: Pool name
- **PoolSz**: Pool size
- **InUse**: Percentage of pool in use
- **Free**: Allocated but unused buffers in pool
- **TotBuf**: Total buffers in pool
- **Finds**: FIND calls
- **Hits**: Successful FIND calls
- **Misses**: Unsuccessful FIND calls
- **Deletes**: DELETE calls
- **LgBfNm**: Name of largest buffer
- **LgBfSz**: Length of largest buffer
- **SmBfNm**: Name of smallest buffer
- **SmBfSz**: Length of smallest buffer

### 4.7.4 New 4515 log record

The statistics displayed on the QRY POOL command are also externalized on the 4515 log record.

### 4.7.5 New monitor records are written in certain situations

The following new monitor records are written in the following instances:
- Type 74 is issued when a get request for a PSB is made from the 64-bit pool
- Type 75 is issued when a put request for a PSB is made to the 64-bit pool
- Type 76 is issued when a get request for a DMB is made from the 64-bit pool
- Type 77 is issued when a put request for a DMB is made to the 64-bit pool.

### 4.8 zAAP or zIIP times in dependent region accounting log records

The time fields for zAAP or zIIP are added to the X’07’, X’0A07’ and X’56FA’ log records:
- X’07’ program termination
- X’0A07’ CPIC program termination
- X’56FA’ optional transaction level statistics record

The CPU time field is changed to include only the standard central processor (CP) time, not zAAP/zIIP time.

The sum of CP and zAAP/zIIP times is the total CPU time.

**Benefits:**
- Users can distinguish between CP and zAAP/zIIP times.
- Can be used for accounting or charge-out purposes.
- Is significant for software licensing.
- Is most significant for JMP and JBP regions.
4.9 IMS use of zIIP

Request response processing for authorized Common Queue Server (CQS) clients in IMS Version 12 is executed under enclave service request blocks (SRBs). In IMS 12 and subsequent releases, IMS requests z/OS to process such work on an available zIIP. *Request response processing* is the processing of the return of data from the CQS address space to an authorized CQS client address space in response to a request that the client directed to the CQS. Authorized CQS clients are those clients that register to IMS 12 CQS while executing in supervisor state and with a “system” program status word (PSW) key (keys 0 - 7).

The following examples are of IMS 12 operations that involve such authorized CQS clients:

- When the IMS control region is running with IMS Shared Message Queues or Shared IMS Fast Path Message Queues enabled
- When IMS Resource Manager (RM) address space is using a resource structure

4.10 Member online change NAMEONLY option

In an IMSplex environment, you can use the ACB member online change (OLC) function to add or change individual members of the ACB library, or the entire ACB library, and bring these new or changed members online without quiescing the IMSplex or refreshing the active ACB library.

IMS 12 APAR PM28802 added the new option for member online change.

With OPTION(NAMEONLY) specified on the INIT.OLC TYPE(ACBMBR) command, IMS processes only the DBDs and PSBs that are specified in the NAME keyword:

```
INIT OLC TYPE(ACBMBR) PHASE(PREPARE) NAME(name1,name2,...) OPTION(NAMEONLY)
```

The command processes only the named member (or members) of the staging ACBLIB. Without this option, the command processes the related members that were changed, for example, the DBDs in the intent list of the PSB.

OPTION(NAMEONLY) can be used for the following instances:

- New DBDs: These DBDs cannot reference existing DBDs which have been modified
- New or changed PSBs which do not reference changed DBDs in their intent lists

**Benefit:** This option provides a performance benefit when there are many members in the staging and active ACB libraries.

4.11 IMS and Workload Manager (WLM)

Workload Manager (WLM) is the z/OS component responsible for ensuring that multiple workloads in a z/OS system use resources based on goals and the importance of each workload. Each workload is also given a service class to distinguish it from other workloads.

For a discussion about setup and use of WLM, see *IMS Performance and Tuning Guide*, SG24-7324.
Figure 4-1 shows how WLM manages tasks in z/OS.

### 4.11.1 WLM terminology

The following terminology is used in WLM:

- A **service definition** consists of one or more service policies.
- A **service policy** contains several workloads; one service policy is active at a time in a LPAR or Parallel Sysplex.
- Each **workload** consists of one or more service classes.
- Each **service class** has at least one period and each period has one goal.
- A **goal** can be one of five types: system, average response time, percent response time, execution velocity, and discretionary. A goal can have a duration.
- **Address spaces** and **transactions** are assigned to service classes by classification rules.
4.11.2 WLM concepts of service class and classification

The following descriptions explain WLM classification and service class:

- **Classification**
  - Assignment of incoming work to a service class, and optionally to a report class
  - Based on a wide variety of filters, or qualifiers

- **Service class**
  - Is a set or group of related work:
    - Production CICS, IMS, and DB2 address spaces might be in same service class: STCHI or PRODHI or ONLNPRD, and so on.
    - Separate Report Classes can report on CICS or IMS or DB2.
  - Can combine goals of various types in multiple periods:
    - A period is the combination of importance (IMP), goal, and duration.
    - A service class period is the target of WLM measurement and management actions.
Figure 4-3 shows WLM classification rules.

- WLM assigns work to a service class based on qualifiers that apply to the subsystem from which the work arrived.
Figure 4-4 shows the subsystem types used for classification by WLM.

<table>
<thead>
<tr>
<th>Transaction Type</th>
<th>Allowable Goal Types</th>
<th>Allowable Number of Periods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address space</td>
<td>Response Time Discretionary</td>
<td>Multiple</td>
</tr>
<tr>
<td>Enclave</td>
<td>Execution Velocity Discretionary</td>
<td>DDF (and DB2)</td>
</tr>
<tr>
<td>CICS/IMS</td>
<td>Response Time IMS Txn (class)</td>
<td>1</td>
</tr>
</tbody>
</table>

- Subsystems follow one of three transaction type models.
- Need to understand how this affects the value of figures shown in the RMF Workload Activity Report.
- SYSH is used for LPAR load balancing.

Figure 4-4 Subsystems types used for classification
4.11.3 WLM importance

Figure 4-5 shows how WLM applies importance to an address space.

![WLM Concepts – Importance](image)

4.11.4 WLM concepts and goal types

The various goals that WLM manages:

- System goals
  - SYSTEM and SYSSTC service classes have fixed dispatching priorities above importance 1 (IMP 1).

- Response time goals
  - Average response time, including queue time and execution time.
  - Percentile response time, reduces impact of outliers, for example, 90% of transactions complete within 0.7 seconds.

- Execution Velocity goals or velocity goals
  - Velocity goals are intended for work for which response time goals are not appropriate, such as address spaces or long running jobs.
  - Indicates how fast work should run relative to other work requests when ready, without being delayed for CPU, storage, or I/O.
  - Is expressed as a number, for example, 60 or 40.

  A value of 60 means “ready” work runs 60% of the time.
Differentiates velocity goals within an importance level by 10.

Appropriate velocity goal depends on number of engines (CPs).

Discretionary

Appropriate for low priority, long-running work.

Figure 4-6 shows an example of WLM importance levels and IMS.

### WLM Importance Levels and IMS, an example

- Importance 1 is highest priority after SYSSTC.
- IMS address spaces should have velocity goals and a single period defined.
- Non-production IMSs could be IMP 2 or IMP 3 or IMP 4 if in the same LPAR (or Parallel Sysplex) with production IMS.
- Discretionary work gets service after all other importance levels.
  - Not appropriate for IMS address spaces.
  - Not recommended for IMS work.
  - Very little service if CPU is 100% busy.

<table>
<thead>
<tr>
<th>Importance</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMP 1</td>
<td>Highest priority after SYSSTC.</td>
</tr>
<tr>
<td>IMP 2</td>
<td>High</td>
</tr>
<tr>
<td>IMP 3</td>
<td>Medium</td>
</tr>
<tr>
<td>IMP 4</td>
<td>Low</td>
</tr>
<tr>
<td>IMP 5</td>
<td>Lowest</td>
</tr>
<tr>
<td>DISCRETIONARY</td>
<td>Default service class</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>z/OS</td>
</tr>
<tr>
<td>SYSSTC</td>
<td>IRLM</td>
</tr>
<tr>
<td>IMP 1</td>
<td>IMS CTL, OM, RM, SCI, DLISAS, CQS, DBRC</td>
</tr>
<tr>
<td>IMP 2</td>
<td>MPPs, IFPs, (most important IMS TRXs if Server Mgmt. is used)</td>
</tr>
<tr>
<td>IMP 3</td>
<td>BMPs</td>
</tr>
<tr>
<td>IMP 4</td>
<td>IMS/DLI Batch</td>
</tr>
<tr>
<td>IMP 5</td>
<td>Lowest priority work</td>
</tr>
<tr>
<td>DISCRETIONARY</td>
<td>DISCRETIONARY Default service class</td>
</tr>
</tbody>
</table>

![Figure 4-6  WLM importance levels and IMS, an example](WLM_importance_levels_and_IMS.png)

### 4.11.5 WLM managed delays

WLM can only affect work by adjusting the following resources:

- Processor (dispatching priority)
- Non-paging DASD I/O (IOSQ, subchannel pending, control unit queue)
- Storage (paging, swapping)
- Tasks (multi-programming level, server address space creation, batch initiation)

For example, WLM manages stored procedure address spaces.

WLM cannot manage the following items, for example:

- User delays (coffee breaks)
- Network delays
4.11.6 WLM concepts and performance level

The system administrator classifies work to service classes. The classification mechanism uses work attributes like transaction names, user identifications or program names defines goals and importance levels for the service classes representing the application work. Goals can be expressed as response times or velocity.

The value of the importance parameter indicates how vital it is to the installation that this performance goal be met relative to the goals of other periods.

- Defined as part of the Service Class: 1 (high) to 5 (low)
- Assigned to a Service Class Period
- A way to prioritize critical goals
- For work at the same importance level, WLM attempts to equalize the PIs

Service class periods are compared by calculating the performance index (PI) for each service class. PI gives WLM a common way to track how well the work is doing regardless of goal type.

**Important:** The performance index for a service class is a single number, which tells whether the goal definition can be met, was overachieved, or was missed. WLM modifies the access of the service classes based on the achieved performance index and importance.

- If PI equals 1, then the work in the period is exactly meeting its goal.
- If PI is less than 1, then the work is doing better than its goal.
- If PI is more than 1, then the work is missing its goal.

4.11.7 WLM service class periods

WLM manages service class periods. This section explains how.

WLM heuristic behavior is applied to service class periods. Figure 4-7 on page 178 shows WLM performance heuristic behavior.
WLM can effectively manage 25-30 active service class periods.

- If you have more than 30 active service class periods, WLM might not be able to adjust resources for all of them when the system is busy.
- When the system is busy is when you want WLM to adjust resources to meet your business goals.

*Loose goals* are performance goals that are too easily achieved.

- Service class periods with loose goals are likely to have a PI of less than 1, so WLM will always perceive that the class periods are meeting their goals.
- Service class periods with loose goals can have a PI of less than 0.7, in which case they can become a donor.

Service units are the basis for period-switching within a service class that has multiple periods. The duration of a service class period is specified in terms of service units. When an address space or enclave running in the service class period consumes the amount of service that is specified by the duration, workload management moves it to the next period. The work is managed to the goal and importance of the new period. Figure 4-8 on page 179 shows an example of Service Class Period Switch. *Service units* describes a hardware independent measure of CPU consumption.
Figure 4-8  Example of Service Class Period Switch

- All transactions assigned to this service class start in Period 1.
  - WLM manages the transactions in period 1 to the percentile response time goal of 90% completing in half a second, with an importance of 3.
- Transactions that accumulate 300 service units (DUR = 300) before completing migrate to Period 2 (a new service class period).
  - WLM manages the transactions in period 2 to the goal of 90% completing in 4 seconds, with an importance of 4. [That is, 90% of those that did not complete in period 1.]
- Transactions that accumulate 900 service units (DUR 300 + DUR 600) before completing migrate to Period 3 (a new service class period).
  - WLM manages the transactions in period 3 to a velocity goal of 40, with an importance of 5.
- "Service units" is a hardware independent measure of CPU consumption. If your transaction consumes 1000 service units on a z9, it should consume 1000 service units on a z196.

4.11.8 WLM use for IMS

The following list is a brief summary of setting up WLM for IMS:

- IMS schedules transactions into regions and address spaces, based on their (IMS) class, except for FP and BMP.
- WLM adjusts region priority in an attempt to meet goals.
- If multiple classes can be scheduled into a given IMS region, then all transactions in those classes should have the same goal.
- If many transactions with different goals have the same class and do not manage at the transaction level, WLM will constantly adjust the address space after a given transaction runs, which can have a different goal than the next.
- Consider carefully before trying to manage below the region level.
- Keep things as simple as possible.
4.12 Program load options and IBM Language Environment

Various techniques of loading programs in a region are described in the following sources:

- IMS Performance and Tuning Guide, SG24-7324
- IMS Version 7 Performance Monitoring and Tuning Update, SG24-6404

From an IMS performance point of view, the considerations that are described in this section apply.

4.12.1 Library routine retention

Using the library routine retention (LRR) function can significantly improve the performance of COBOL transactions running under IMS TM. LRR provides function similar to that of the VS COBOL LIBKEEP runtime option. It keeps the IBM Language Environment® initialized and retains in memory any loaded Language Environment library routines, storage associated with these library routines, and storage for Language Environment startup control blocks. To use LRR in an IMS dependent region, you must do the following tasks:

- In your startup JCL or procedure to open the IMS dependent region, specify the PREINIT=xx parameter (xx is the two-character suffix of the DFSINTxx member in your IMS PROCLIB data set).
- Include the name CEELRRIN in the DFSINTxx member of your IMS PROCLIB data set.
- Open your IMS dependent region.

You can also create your own load module to initialize the LRR function by modifying the CEELRRIN sample source in the SCEESAMP data set. If you do this step, use your module name in place of CEELRRIN. See Language Environment for z/OS, SA22-7564 for details.

Important: If the RTEREUS runtime option is used, the top level COBOL programs of all applications must be preloaded.

Note that using RTEREUS will keep the Language Environment running until the region goes down or until a STOP RUN is issued by a COBOL program. This means that every program and its working storage (from the time the first COBOL program was initialized) is kept in the region. Although this option improves performance, you might find that the region can soon fill to overflowing, especially if many separate COBOL programs are invoked.

When you do not use RTEREUS or LRR, you should preload the library routines (described in 4.12.2, “Preload library routines” on page 181).
### 4.12.2 Preload library routines

Preload the following items:

- For all COBOL applications:
  - CEEBINIT
  - IGZCPAC
  - IGZCPCO
  - CEEEV005
  - CEEPLPKA
  - IGZETRM
  - IGZEINI
  - IGZCLNK

- If the application contains VS COBOL II programs:
  - IGZCTCO
  - IGZEPLF
  - IGZEPCL

- If the application contains OS/VS COBOL programs:
  - IGZCTCO
  - IGZEPLF
  - IGZCLNC
  - IGZEPCL

When running under the Language Environment, continue to preload the same ILBO library routines that you previously preloaded. If ILBOSTT0 was in the preload list with OS/VS COBOL, replace it with ILBOSTT2 and make sure that it is in there twice. You should, at a minimum, include all heavily used ILBO library routines in your preload list.

**COBP ACK** refers to a collection of individual modules that are packaged into a single load module to reduce the time that would otherwise be needed to load the individual load modules. Because the two COBPACKs, IGZCPAC and IGZCPCO, used by IMS, contain only those COBOL library routines that run above the line (AMODE 31, RMODE ANY), you should also preload any of the below-the-line routines that you need. A list of below-the-line routines is in *Language Environment for z/OS, SA22-7564*.

### 4.12.3 Preload application programs

Heavily used application programs can be compiled with the RENT option and preloaded to reduce the LOAD overhead for them. An example of a heavily used application program is a shared, dynamically called subprogram.

### 4.12.4 For OS/VS COBOL programs

OS/VS is a version of COBOL that ran before Language Environment. STACK controls the allocation of the thread's stack storage for COBOL LOCAL-STORAGE data items and work areas for COBOL library routines. For the runtime option, use STACK(32K,32K,BELOW,KEEP). The default is STACK(128K,128K,BELOW,KEEP).

The CEEWUOPT or CEEWQUOP sample job can create a new CEEUOPT options module in a user-specified library. The application programmer can then include one of these CEEUOPT modules when link-editing an application. The options in CEEUOPT override the default options in CEEDOPT or CEECOPT, unless NONOVR was specified for the option when CEEDOPT or CEECOPT was created.
4.12.5 Checklist for program-related performance problem

The following questions can help you to isolate the problem area if a performance-related problem arises with COBOL. Most performance-related problems that were reported in the past fall under one or more of the following categories of questions. Each category does not always apply to all operating environments.

This information basically serves as a checklist of considerations when you investigate a performance problem.

- **Has something recently changed that caused the performance problem?**
  Look for changes (including upgrades or maintenance) to the hardware, operating system, Language Environment, COBOL compiler, application code, workload, and so on. Determining changes in these areas can help to identify the problem.

- **What are all of the compiler options that were used? Is OPTIMIZE being used? Is TRUNC(OPT) being used?**
  Make sure you understand the performance implications of the options you are using.

- **What runtime options were used?**
  Make sure you understand the performance implications of the options you are using.

- **Is the program compiled with OS/VS COBOL or VS COBOL II and run with the Language Environment library?**

- **Does the application have a mixture of OS/VS COBOL or VS COBOL II with COBOL/370, COBOL for MVS and VM, COBOL for IBM OS/390® and VM, or Enterprise COBOL programs?**

- **Is the application called by a program that does not conform to Language Environment?**
  For an assembler driver, repeatedly calling COBOL, is RTEREUS or IGZERRE or ILBOSTP0 being used? Is LRR or CEEPIPI being used? Are the CEEENTRY and CEETERM macros being used? Make sure to check for vendor-provided driver programs too.

- **Does the application have any calls to any other programs? If so, what languages are involved? What is the approximate number of calls and the depth of the calls? Are the calls static or dynamic? How many unique programs are called?**

- **What Space Management Tuning is being used?**
  The RPTSTG(ON) runtime option can help you to determine the correct values to use.

- **What are the JOBLIB and STEPLIB data sets and where is the Language Environment library in the search order? Is the Language Environment library in LNKLST or LPA/ELPA?**

- **For IMS, is the application, library, or both, preloaded? Is LRR being used?**

- **Do you have an execution profiler? If so, have you used it to try to identify the “hot spots” of where time is spent in the application?**
  This information can be useful in identifying and solving performance problems.

- **What are the release levels of all COBOL products being used? Has all current maintenance been applied?**
  If the most current release of IBM Enterprise COBOL is not being used, you should try it before reporting the problem to IBM because the problem might have already been addressed.

- **What are the release levels of the operating and subsystems being used (IMS, CICS, OS/390, z/OS)? Has all current maintenance been applied?**
If using SORT, what release of DFSORT is being used? Has all current maintenance been applied? Is the FASTSRT compiler option being used?

In case you need to seek assistance from IBM in solving the performance problem, what other information can you tell us?

This information can help us understand the overall program structure. Examples include heavy use of a particular COBOL verb, the application program alters the save area in a nonstandard way, subscripts that are not binary, what data types are used (USAGE DISPLAY, INDEX, COMP-n, and so on.

4.13 Extended address volume for non-VSAM data sets

An extended address volume (EAV) is a volume with more than 65,520 cylinders. An extended address volume increases the amount of addressable DASD storage per volume beyond 65,520 cylinders by changing how tracks are addressed. IMS 12 allows some non-VSAM data sets to use EAV volumes:

- z/OS 1.12 is required.
  - Data sets can reside in extended address space (EAS) on EAV volumes.
- Architecture can support hundreds of terabytes on single volume.
- Storage is addressed by using 28-bit cylinder/track address.
- IMS OSAM data sets can reside in EAS.
  - IMS 11 provided support for VSAM data sets to reside in EAS.

**Benefits:**

- Support exists for the placement of more data sets on a single volume.
- Users can manage fewer numbers of larger volumes.
- Less need exists for multi-volume OSAM.
- Fewer extents are being taken by data sets that use secondary allocation.

Figure 4-9 on page 184 shows an EAV, and Figure 4-10 on page 184 shows several EAV key design points.
Extended address volume (EAV)

- EAV is a volume with more than 65,520 cylinders
  - 3390 Model A
  - 1 to 268,434,453 cylinders
    > Architectural EAV maximum

Maximum sizes

- 3390-3
  - 3 GB
  - Max cyls: 3,339

- 3390-9
  - 9 GB
  - Max cyls: 10,017

- 3390-27
  - 27 GB
  - Max cyls: 32,760

- 3390-54
  - 54 GB
  - Max cyls: 65,520

- 3390-A
  - "EAV"
  - 100s of TBs

EAV key design points

- EAV maintains 3390 track format
  - Track-managed space:
    > Area on EAV within the first 65,520 cyls
    > Space allocated in track or cyl increments
    > Storage for "small" data sets
  - Cylinder-managed space:
    > Area on EAV located above first 65,520 cyls
    > Space is allocated in multicylinder units
    > Storage for "large" data sets
  - New DSCB format types identify EAS data sets
    > New formats (Format 8 and 9) in VTOC
      - Data set resides in cylinder-managed space
4.13.1 Allocating data sets on EAV volumes

The following list has several JCL considerations for allocating data sets on EAV volumes:

- **EATTR** is a data set attribute. Note the following information:
  - Specifies whether extended attributes can be created for a data set.
  - Overrides system-determined extended addressing space (EAS) eligibility.
  - Data set cannot have extended attributes or optionally reside in EAS (no extended attributes):
    
    ```
    EATTR=NO
    ```
  - Data set can have extended attributes and can optionally reside in EAS (extended attributes are optional):
    
    ```
    EATTR=OPT
    ```

- You can use the EATTR parameter to indicate whether the data set can support extended attributes in the following definitions:
  - AMS DEFINE CLUSTER (VOL (xxxxxx))
  - ALLOCATE
  - JCL (VOL=SER=xxxxxx)
  - Dynamic Allocation
  - Data Class

4.13.2 IMS non-VSAM data sets supported

The following data sets are supported:

- Overflow sequential access method (OSAM) data sets
  - OSAM database data sets
  - Restart data set (RDS)
  - Message queue blocks data set
  - Long and short message data set
- IMS online log data sets (OLDS)
- IMS log write ahead data sets (WADS)
- IMS SPOOL data sets
- BPE External Trace Data Sets
  Specified on EXTTRACE in BPE configuration parameter member in proclib.
4.13.3 Example of IMS non-VSAM data sets specification

Example 4-1 shows IMS OLDS and WADS JCL specification of non-VSAM data sets with the EATTR attribute.

Example 4-1   OLDS and WADS JCL specification

```plaintext
//ALLOC   EXEC PGM=IEFBR14
//SYSPRINT DD  SYSOUT=A
//DFSOLP00 DD  DSN=IMSTESTL.IMS01.OLDSP0,UNIT=SYSDA,VOL=SER=EAV002,
//             DISP=(,CATLG),EATTR=OPT,
//             DCB=(RECFM=VB,BLKSIZE=22528,LRECL=22524,BUFNO=5),
//             SPACE=(CYL,(1,1))
//DFSWADS0 DD  DSN=IMSTESTL.IMS01.WADS0,UNIT=SYSDA,VOL=SER=EAV002,
//             DISP=(,CATLG),EATTR=OPT,
//             DCB=(DSORG=PS,RECFM=F,BLKSIZE=4096,KEYLEN=1),
//             SPACE=(CYL,5)
```

4.13.4 Special considerations for HALDB OLR output data sets

Considerations for high availability large database (HALDB) online reorganization (OLR) output data sets are as follows:

- Is created automatically during OLR. If OLR input data set is in EAS of EAV, IMS creates OLR output data set in EAS of EAV.
- Can be manually pre-allocated to an EAV volume. The following example shows pre-allocated OLR output data sets:

```plaintext
DEFINE  CLUSTER (NAME (IMSTESTS.DBVHDJ05.M00001) -
  REUSE -
  EATTR(OPT) -
  CYL(1,1)  RECSZ (1017,1017) -
  VOL (EAV001) SHAREOPTIONS (3,3)  CISZ (1024)  NIXD)
```

4.14 OTMA support for asynchronous IMS-to-IMS communications

IMS Connect to IMS Connect messaging is a function that sends messages from IMS to a remote IMS through a local IMS Connect and a remote IMS Connect across TCP/IP connections.

Existing methods typically require the creation of customer gateway applications that implement RESUME TPIPE code to retrieve the message from one IMS, along with code to establish a connection, and send the message to another IMS system. With this capability, IMS and IMS Connect provide the necessary communication path and code that eliminate the need for a gateway application.
Figure 4-11 shows asynchronous IMS-IMS TCP/IP support.

### Asynchronous IMS-IMS TCP/IP support

- TCP/IP connections between the local and remote IMS systems
  - Are managed by IMS Connect to IMS Connect communications
    - Without having to write client code or invoke additional gateways
  - The goal is simplification and ease of use

**Example:**

The IMS application in IMS1 makes an ISRT ALTPCB call. If an OTMA descriptor is used, then the application specifies a descriptor name as the destination on the change call. The OTMA of IMS1 looks up the descriptor name to retrieve the values that describe the remote connection and inserts them into the OTMA headers of the message. IMS1/OTMA then sends the message to the local IMS Connect that is specified in the TMEMBER parameter of the descriptor. If the OTMA descriptor is not specified, the OTMA DFSYDRU0 user exit can be coded to provide the appropriate values for the remote connection.
Figure 4-12 shows local IMS to remote IMS connection.

**Asynchronous IMS-IMS TCP/IP support …**

- **OTMA**
  - Sends OTMA remote ALTPCB messages to IMS Connect using new destination information
    > OTMA destination descriptors or DFSYDRU0 exit Routine
- **IMS Connect**
  - Receives OTMA ALTPCB messages from a local IMS and sends them to the remote IMS Connect for processing in the remote IMS
    > Enhanced IMS Connect configuration specifications

---

**4.14.1 Usage and benefits of Asynchronous IMS-IMS TCP/IP support**

Usage includes the following items:

- IMS applications: ISRT ALTPCB
- IMS environment: destination descriptor or a DFSYDRU0 exit routine
- IMS Connect: configuration specifications

Benefits are as follows:

- Supports TCP/IP communications to invoke transactions between IMS systems without having to create or maintain a separate gateway solution (IMS-provided and supported solution).
- IMS to IMS communication does not rely on an external gateway program for communications.
- Reduces maintenance costs by eliminating the need to maintain a user-developed IMS Connect gateway application to receive the output message and send it to another IMS system
4.14.2 Performance evaluation results of IMS connect to IMS connect

At IBM Silicon Valley Laboratory, we conducted controlled performance studies for IMS Connect to IMS Connect communication across TCP/IP connections.

**Observation:** In our testing environment, we achieved a message sending rate of over 9,500 messages a second between our local IMS Connect and our remote IMS Connect over a TCP/IP network. All measurements were conducted within a stable and isolated environment by using the enhanced SMQL workload.

4.15 MSC TCP/IP

Environments that primarily use TCP/IP networks for communications, but require SNA/VTAM to sustain their Multiple Systems Coupling (MSC) links, now have the option of also migrating their MSC connections entirely to TCP/IP. A TCP/IP link type was added to the MSPLINK macro. These links can be used to replace, complement or back up existing MSC VTAM connections. Operationally, the new support is compatible with other link types, for example, CTC, MTM, VTAM, using the same command structure.

From an MSC perspective, the operation of TCP/IP physical links and VTAM physical links is similar. Apart from some differences in system definition and buffer size requirements, IMS and z/OS components that support each connection type present the most significant operational differences. Both TCP/IP and VTAM physical link types can be used either as your primary physical link type or as a backup link type in case the other link type fails for any reason.

Depending on various factors, such as network traffic and the distance between the two connected IMS systems, a TCP/IP physical link is likely to provide better performance than a VTAM physical link.

MSC TCP/IP requires no additional requirements over the basic version 12 requirements but is supported only between two version 12 IMS systems.

4.15.1 MSC TCP/IP uses IMS Connect and the common service layer

IMS Connect is a key component of the MSC support. Note the following information:

- IMS Connect sends and receives messages through the TCP/IP network.
  - IMS Connect manages the TCP/IP communications
  - IMS MSC manages the message processing
- CSL provides the Structured Call Interface (SCI) for communications between IMS components including IMS Connect.
  - Each IMS and its local MSC-routing IMS Connect system must be part of the same IMSPlex. The IMSPLEX= plexname parameter is in the Common Layer Section of the DFSDFxxx of IMS proclib.
  - The Operations Manager (OM) is not required but is preferred for type-2 command support.
4.15.2 IMS to IMS connect functionality

Figure 4-13 shows how IMS to IMS communication is implemented.

- IMS to IMS connect functionality
  - Isolates TCP/IP from the IMS Control Region
    >Uses the existing IMS Connect TCP/IP support
  - Provides a new MSC driver as well as TCP/IP driver for MSC
  - Supports communication with IMS via the Structured Call Interface (SCI)

IMS Connect Extensions for z/OS and IMS Problem Investigator for z/OS can be used to assist in data collection for MSC over TCP/IP performance issues.

4.15.3 IMS considerations

Note the following information about the use of TCP/IP MSC links:

- Requires IMSplex architecture.
- Provides SCI interface to IMS Connect
- Relies on Type2 command support for optimally managing the environment Operations Manager.
- Uses only bandwidth (BW) mode. BW mode was introduced in IMS 10 as optional for CTC, MTM, and VTAM links.

Installing at least one MSC link of any type also installs MSC. If MSC is defined in one IMS within a shared queues IMSplex, it must also be installed in the other IMS systems. There is no change from previous releases.

4.15.4 Setup scenario and switching from VTAM to TCP/IP

A suggested setup scenario is for all IMS systems to be at version 12 to use MSC TCP/IP. Consider a VTAM to TCP/IP migration and TCP/IP to VTAM fallback scenario.

The existing VTAM link in stage 1 sysgen is as follows:

```
PLNK1  MSPLINK  TYPE=VTAM,NAME=L6APPL1,SESSION=1,_BUFSIZE=8192
LNK12  MSLINK   PARTNER=AK,MSPLINK=PLNK1,OPTIONS=FORCSESS
MSN12  MSNAME   SYSID=(20,10)
```
The following steps are an example for switching an MSC link from VTAM to TCP/IP:

1. Add a TCP/IP physical link to stage 1:
   
   PLNK2 MSPLINK TYPE=TCP/IP, NAME=IMS2, LCLICON=ICON1,
   LCLPLKID=MSC12, SESSION=1, BUFSIZE=8192

2. Add the MSC and RMTIMSCON configuration statements to the local IMS Connect configuration proclib member:
   
   MSC=(LCLPLKID=MSC12, RMTPLKID=MSC21, LCLIMS=IMS1, RMTIMS=IMS2,
   IMSPLEX=(MEMBER=HWS1, TMEMBER=PLEX1), RMTIMSCON=CONNECT2)
   RMTIMSCON=(ID=CONNECT2, HOSTNAME=ICON2.IBM.COM, PORT=9999, RSVSOC=2)

3. Add an SCI, and optionally add OM if it is not already defined.

4. Schedule and activate new IMS generations as required, and restart IMS.

5. Stop the physical and logical links and move the logical link from VTAM to TCP/IP.

6. Repeat these steps for the remote IMS.

7. Restart the links.

### 4.15.5 Fallback scenario

Use the following steps to fall back to MSC VTAM:

1. Stop the physical TCP/IP links and associated logical links.

2. Move the logical links from the TCP/IP physical links back to the MSC VTAM physical links.

3. Restart the MSC VTAM links and logical links.

### 4.15.6 Performance considerations (sample performance test)

The following steps run a benchmark test of MSC links bandwidth performance. This test can be done for a VTAM and a TCP/IP link to compare the performance between the two links.

1. Queue up 10,000 messages to the link.

2. UPDATE MSLINK NAME(name) START(STATISTICS) OPTION (NORESET, CHKPT) IMS will not reset the query statistics at each IMS checkpoint.

3. UPDATE MSLINK NAME(name) START(STATISTICS) OPTION (RESET).
   IMS resets the link statistics to zero and sets the start time to the current time.

4. (Only for VTAM): UPDATE MSLINK NAME(name) SET(BANDWIDTH(ON)).
   IMS sets bandwidth on for VTAM (TCP/IP always uses BANDWIDTH).

5. UPDATE MSLINK NAME(name) START(COMM).
   IMS starts the link and lets all the messages process (messages sent one way).

6. QUERY MSLINK NAME(name) SHOW(STATISTICS).

7. Look at Send_Msg_Time to see the interval of time between the first and last message sent. Used for benchmark testing of a block of messages. Can also use Rec_Msg_Time.

8. Repeat these steps with larger buffer sizes.
4.15.7 Performance expectation

Expect TCP/IP bandwidth to improve over VTAM. Verify the results by using the QUERY MSLINK SHOW (STATISTICS) command. The following statistics are supported:

- **General statistics**
  - Total ITASK dispatch count; total processing time; Hi, Lo, Avg processing times per dispatch; and check write counts and rates

- **Send statistics**
  - Total messages sent; total byte count sent; Hi, Lo, Avg send message sizes; queue manager get (QGET) counts (GU, GN, DEQ calls); Hi, Lo, Avg QGET times; and Hi, Lo, Send I/O times

- **Receive statistics**
  - Total messages received; total byte count received; Hi, Lo, Avg receive message sizes, QMGR ISRT (QPUT) counts (ISRT, and ENQ calls); Hi, Lo, Avg QPUT and times; and Hi, Lo, Receive I/O times

Note that the TCP/IP elapsed time value requires use of an external time reference to be accurate. IMS cannot detect if external time reference is being used and therefore the value will be zero if the calculation result is a negative number.

IMS Performance Analyzer for z/OS provides performance reports for MSC Link.

**Evaluation results: MSC TCP/IP**

Our results are from the controlled performance studies conducted at IBM Silicon Valley Laboratory. The objective of this evaluation is to determine the performance capability of the new TCP/IP MSC enhancement and existing links by conducting high stress and bandwidth evaluations.

**Results:** IMS 12 TCP/IP MSC type link achieves 10,587 transactions per second.

**Observations:** TCP/IP MSC links have a higher bandwidth than MSC links.

Figure 4-14 shows the overall configuration of the test environment.
The ITR values used with this evaluation were calculated by using an average CPU busy percentage of the FE and BE LPARs.

Table 4-5 shows MSC TCP/IP results and Table 4-6 shows MSC VTAM results.

**Table 4-5  MSC TCPIP results**

<table>
<thead>
<tr>
<th>IMS 12 MSC TCP/IP 16K buffers (1 KB message size)</th>
</tr>
</thead>
<tbody>
<tr>
<td>External throughput rate (ETR)</td>
</tr>
<tr>
<td>Internal throughput rate (ITR)</td>
</tr>
<tr>
<td>FE CPU</td>
</tr>
<tr>
<td>BE CPU</td>
</tr>
</tbody>
</table>

**Table 4-6  MSC VTAM results**

<table>
<thead>
<tr>
<th>IMS 12 MSC VTAM 16K buffers (1 KB message size)</th>
</tr>
</thead>
<tbody>
<tr>
<td>External throughput rate (ETR)</td>
</tr>
<tr>
<td>Internal throughput rate (ITR)</td>
</tr>
<tr>
<td>FE CPU</td>
</tr>
<tr>
<td>BE CPU</td>
</tr>
</tbody>
</table>

### 4.16 Disabling sysplex serial program management (SSPM)

An optional capability to disable sysplex serial program management is added in IMS 12 APAR PM31422.

Before IMS 12, serial programs (SCHDTYPE=SERIAL on APPLCTN macro or DEFINE PGM command) enforce serialization of PSB scheduling in the following circumstances:

- When RM is used
- When an RM structure is defined
- When serialization is enforced across the shared queues group (which does not apply to CICS or ODBA application schedules)

IMS 12 adds the option to *not* enforce serialization. The option is specified with the following control statement:

```plaintext
GBL_SERIAL_PGM=N | Y
```

For example, `GBL_SERIAL_PGM=N` turns off serialization across the shared queues group.

The statement can be in either of the following locations:

- Specified in DFSDFxxx member COMMON_SERVICE_LAYER section
- DFSCGxxx member

**Benefit:** Disabling SSPM enables greater parallelism.
4.17 APPC and OTMA shared queue enhancements

A capability removes the dependency on Resource Recovery Services (RRS) in a shared queues environment for the following items:

- APPC synchronous conversations and OTMA CM1 (send-then-commit) interactions; applies only to the following statement:
  
  synclevel=NONE|CONFIRM

  The statement Synclevel=Syncpoint still requires RRS.

- Communications use XCF services

The following options are available for the existing AOS= parameter, in DFSDCxxx, to request the use of XCF:

- AOS=B

  Synchronous transactions synclevel=NONE|CONFIRM can be processed in a back-end system by using XCF communications. Processing synchronization level of SYNCPT depends on the RRS option:
  
  - RRS=Y: Transactions can be processed at either FE or BE, using RRS.
  - RRS=N: Transactions are processed only at the FE.

- AOS=S

  Allows synchronous transactions with synclevel=NONE|CONFIRM to be processed in a back-end system using XCF communications. Processing synchronization level SYNCPT is equivalent to AOS=F.

- AOS=X

  Allows synchronous transactions with synclevel=NONE|CONFIRM to be processed in a back-end system using XCF communications. Processing synchronization level of SYNCPT is equivalent to AOS=N

Choice of B, S, or X depends on how syncpoint messages are to be processed.

4.17.1 Front-end logging

This version introduces keywords and commands to set front-end (FE) logging:

- AOSLOG=Y or N keyword in IMS DFSDCxxx proclib specifies whether the FE system is to write a 6701 log record for the following items:
  
  - Response messages returned from the BE system through XCF
    
    This item is applicable to all synchronization levels (NONE, CONFIRM, and SYNCPT).
  
  - Error messages returned from the BE system through XCF
    
    This item is applicable to all synchronization levels of (NONE, CONFIRM, and SYNCPT).
  
  - ID=TIB3

- For diagnostics, the /DIAGNOSE command is enhanced to control AOSLOG capture for events related to APPC and OTMA synchronous transactions in a shared queues environment:
  
  /DIAGNOSE SET AOSLOG(ON|OFF)
4.17.2 IMS DISPLAY commands

Figure 4-15 shows enhancements made to the display command.

**IMS commands**

- Enhancements to the output of /DISPLAY commands
  - /DIS A DC, /DIS A REG, /DIS OTMA
    - New statuses in APPC/OTMA SHARED QUEUE STATUS – LOCAL=\texttt{status1} GLOBAL=\texttt{status2}

  - \texttt{status1} can be one of the following:
    - ACTIVE-RRS
    - ACTIVE-XCF
    - ACTIVE-RRS/XCF
    - FORCE-RRS
    - FORCE-RRS/XCF
    - INACTIVE
    - UNSUPPORTED

  - \texttt{status2} can be one of the following:
    - ACTIVE-RRS
    - ACTIVE-XCF
    - ACTIVE-RRS/XCF
    - CHECK
    - INACTIVE

  > New text that shows the AOSLOG setting

  Figure 4-15 Enhancement to IMS display command

The display command is shown in the following examples:

- Figure 4-16 shows an example of a display active DC command.

**IMS commands ...**

- Example

  XX, /DIS A DC
  DFS0001 VTAM STATUS AND ACTIVE DC COUNTS
  DFS0001 VTAM ACB OPEN -LOGONS DISABLED
  DFS0001 IMSLU=N/A,N/A APPC STATUS=DISABLED TIMEOUT=0
  DFS0001 OTMA GROUP=N/A STATUS=NOTACTIVE IMS1
  DFS0001 APPC/OTMA SHARED QUEUE STATUS - LOCAL=ACTIVE-XCF GLOBAL=ACTIVE-XCF
  DFS0001 APPC/OTMA SHARED QUEUES LOGGING=Y
  DFS0001 APPC/OTMA RRS MAX TCBS - 40 ATTACHED TCBS - 1 QUEUED RRSWKs=0
  DFS0001 APPLID=APPL8 GRNAME= STATUS=DISABLED
  DFS0001 LINE ACTIVE-IN - 1 ACTIV-OUT 0
  DFS0001 NODE ACTIVE-IN - 0 ACTIV-OUT 0
  DFS0001 LINK ACTIVE-IN - 0 ACTIV-OUT 0

  Figure 4-16 An example of a display active DC command
Figure 4-17 shows an example of a display active region command.

### IMS commands . . .

- Example of /DIS ACTIVE REG output
  - New status of WAIT-XCF and TERM-WAIT XCF
  - New text: TMEM: tmember-name TPIPE: tpipe-name for OTMA and LUNAME: networkid.luname for APPC

<table>
<thead>
<tr>
<th>Command</th>
<th>REGID</th>
<th>JOBNAME</th>
<th>TYPE</th>
<th>TRAN/STEP</th>
<th>PROGRAM</th>
<th>STATUS</th>
<th>CLASS</th>
<th>ORIGIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS0001</td>
<td></td>
<td></td>
<td>JMPRGN</td>
<td>JMP</td>
<td>IMS1</td>
<td></td>
<td></td>
<td>IMS1</td>
</tr>
<tr>
<td>DFS0001</td>
<td>1</td>
<td>IMSPPA</td>
<td>TPI</td>
<td>APOL11 IMS1 APOL1</td>
<td>WAIT-RRS/PC</td>
<td>1,2,3,4</td>
<td></td>
<td>IMS1</td>
</tr>
<tr>
<td>DFS0001</td>
<td>URID:</td>
<td>C2D6B6917DE8200000000000001010000</td>
<td>ORIGIN: IMS2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>2</td>
<td>IMSPPA</td>
<td>TPI</td>
<td>APOL12 IMS1 APOL1</td>
<td>TERM-WAIT RRS</td>
<td>1,2,3,4</td>
<td></td>
<td>IMS2</td>
</tr>
<tr>
<td>DFS0001</td>
<td>URID:</td>
<td>C2D6B6917DE8300000000000001010000</td>
<td>ORIGIN: IMS2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>3</td>
<td>IMSPPC</td>
<td>TPI</td>
<td>APOL13 IMS1 APOL1</td>
<td>WAIT-XCF</td>
<td>1,2,3,4</td>
<td></td>
<td>IMS2</td>
</tr>
<tr>
<td>DFS0001</td>
<td>TMEM:</td>
<td>IMSNETWK.LU621MS1</td>
<td>TPIPE: CLIENT01 ORIGIN: IMS2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>4</td>
<td>IMSPPD</td>
<td>TPI</td>
<td>APOL14 IMS1 APOL1</td>
<td>TERM-WAIT XCF</td>
<td>1,2,3,4</td>
<td></td>
<td>IMS2</td>
</tr>
<tr>
<td>DFS0001</td>
<td>LUNAME:</td>
<td>IMSNETWK.LU621MS1</td>
<td>ORIGIN: IMS2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>JBP</td>
<td>JBP</td>
<td>BMP</td>
<td>NONE IMS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>FP</td>
<td>FP</td>
<td>BMP</td>
<td>NONE IMS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>DBTTGN</td>
<td>DBT</td>
<td>NONE IMS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>DBRCTAB</td>
<td>DBRC</td>
<td>IMS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS0001</td>
<td>DLISDEP</td>
<td>DLS</td>
<td>IMS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4-17 An example of a display active region command

### 4.17.3 Setup considerations

Several considerations for setting up shared queue enhancements are described in the following list:

- Choice of AOS = B, S, or X depends on how syncpoint messages are to be processed.
- An IMS restart (NRE/ERE) is required to change the AOS= and AOSLOG= settings.
- DBRC MINVERS of 12.1 is required. Note the following information:
  - All IMS systems in a shared queues group must be at this level to enable the enhancement.
  - IMS systems at lower versions using the same RECON data sets are not able to join the SQ group.
  - IMS systems at lower versions using a different RECON data set will still be able to join the shared queue group, but can experience ABENDU0711.
- Different AOS parameter settings in different IMS systems can complicate the environment. Understand the impact.

### 4.17.4 APPC and OTMA SQ enhancements benefits

Using XCF rather than RRS offers certain benefits. Allowing IMS to be the syncpoint manager enhances the performance of the commit processing by eliminating the following items:

- RRS logging overhead
- Potential RRS commit processing bottleneck
- Overhead associated with communicating with an external sync point manager
4.17.5 Evaluation results: APPC synchronous shared queues

These findings for APPC synchronous shared queues are from our controlled performance studies, conducted at IBM Silicon Valley Laboratory. The results of the IMS 12 APPC synchronous shared queues enhancement evaluation confirm significant performance gains are achieved by using XCF in comparison to RRS.

**Observations:**
- 78% reduction (average) in CPU utilization for back-end and front-end IMS
- 35% reduction IFP region occupancy, with ETR gains of 22%, demonstrating quicker transaction transit times
- Eight IFPs used with 57.13% region occupancy for AOS=Y and 42.12% for AOS=B

4.17.6 Evaluation results: OTMA synchronous shared queues

These findings for OTMA synchronous shared queues are from our controlled performance studies, conducted at IBM Silicon Valley Laboratory.

IMS 12 OTMA synchronous shared queues enhancement with XCF for commit mode 1 (CM1, which is send-then-commit) provides significant CPU efficiency improvements leading to increased throughput compared to RRS for CM1 transactions with a synchronization level of NONE or CONFIRM. Based on our performance evaluation, we saw a transaction rate (ETR) improvement of 6 - 18% and path-length (ITR) improvement of 27 - 81%. Note that the ITR values used with this evaluation were calculated by using an average CPU busy percent of the FE and BE LPARs.

**Observations:**
- 63% ITR path-length improvement for SYNCLVL=NONE, using XCFIMS
- 81% ITR path-length improvement for SYNCLVL=CONFIRM, using XCFIMS
- 27% improvement for SYNCLVL=NONE, by splitting the workload between a FE and BE situation pathlength (ITR)

4.18 OTMA ACEE reduction for multiple OTMA clients

The following enhancements are included:

- OTMA ACEE reduction for multiple OTMA clients capability to create, share, and cache a single ACEE associated with a RACF user ID. Sharing is across multiple OTMA member clients (TMEMBER).
- A reduced maximum ACEE aging value during client-bid is introduced: $999999$ seconds (11.5 days). Previously, the value was 68 years. The range is $300$ seconds to $999999$ seconds. If OTMA receives a value less than $300$, the value is reset to 0 and OTMA will not refresh ACEEs.
  
  A cached ACEE has an aging value based on the OTMA member client with the lowest value.
**Challenge addressed: multiple ACEEs for the same user**

- More storage
- More RACF calls to create an instance of an ACEE
- Possible security exposure if a change has to be made to a user profile
  - Different versions of the ACEE based on which OTMA client is used

---

### Command enhancement

- /DISPLAY OTMA command response shows the ACEE aging value for each OTMA client

<table>
<thead>
<tr>
<th>XCFGRP1</th>
<th>GROUP/MEMBER</th>
<th>XCF-STATUS</th>
<th>USER-STATUS</th>
<th>SECURITY</th>
<th>TIB</th>
<th>INPT</th>
<th>SMEM</th>
<th>DRUEXIT</th>
<th>T/O</th>
<th>ACEEAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IMS1</td>
<td>ACTIVE</td>
<td>SERVER</td>
<td>FULL</td>
<td>8000</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>-HWS1</td>
<td>ACTIVE</td>
<td>ACCEPT</td>
<td>TRAFFIC</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>HWSYDRU0</td>
<td>239</td>
<td>3600</td>
<td></td>
</tr>
<tr>
<td>-HWS2</td>
<td>ACTIVE</td>
<td>ACCEPT</td>
<td>TRAFFIC</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>HWSYDRU0</td>
<td>239</td>
<td>7200</td>
<td></td>
</tr>
<tr>
<td>-HWS3</td>
<td>ACTIVE</td>
<td>ACCEPT</td>
<td>TRAFFIC</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>HWSYDRU0</td>
<td>239</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td><em>09121/172200</em></td>
<td>IMS1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Cached ACEEs that are uniquely used by HWS1 and HWS2 have aging values of 3600 and 7200 respectively
- Cached ACEEs that are used by both HWS1 and HWS2 have an aging value of 3600
- HWS3 has no aging value and a non-cached ACEE is created for users that use this TMEMBER client
4.18.1 Considerations for OTMA setup

Consider performance, operational, security, and environment factors:

- **Performance**
  - For multiple OTMA clients, for example, multiple copies of IMS Connects and WebSphere MQs there is potential reduction of total number of RACROUTE REQUEST=VERIFY (RACF I/Os) to create ACEEs for user IDs.
  - When an expired RACF ACEE is detected for an input transaction, an asynchronous request updates the cached ACEE. The input transaction gets an ACEE for transaction authorization, which is deleted when authorization is complete.

- **Operational**
  - The following refresh commands have the same effect; both commands work on the one and only one OTMA ACEE table for all users:
    - `/SECURE OTMA REFRESH TMEMBER membername`
    - `/SECURE OTMA REFRESH`

- **Security**
  - OTMA detects obsolete RACF ACEEs through an internal IMS timer.
  - Every two minutes, OTMA does an ACEE cleanup operation for 10 expired user IDs
  - Reduction of maximum aging value to 11.5 days can increase RACF I/Os if more refreshes are done.

- **Environments**, for example IMSplex with mixed IMS versions
  - Can have wide differences in the maximum aging values for ACEE refreshes:
    - IMS 12: 999999 seconds
    - IMS 10 and IMS 11: 2147483647 seconds

4.18.2 Benefits

Cached ACEEs reduce the system storage requirements while providing better security and performance. Only one copy of the ACEE is used instead of multiple per OTMA client offers the following benefits:

- Reduced storage usage
- Reduced security exposure
- Improved performance

Cached ACEEs provide consistency. The same security results regardless of which OTMA client is used.

A lower maximum ACEE aging value triggers faster ACEE cache refresh, which reduces security exposure, for example when user ID is revoked or access permissions are changed.

4.18.3 Evaluation results: OTMA ACEE reduction

These finding for OTMA ACEE reduction are from our controlled performance studies, conducted at IBM Silicon Valley Laboratory.

Executing with IMS 11 OTMA, 3000 TCP/IP clients, and three IMS Connect clients, we observe that 9000 ACEEs are created, one per TCP/IP client per TMEMBER. With IMS 12, the number is reduced to only 3000 ACEEs, demonstrating a reduction of 67% of the subpool
As the number of TMEMBERs and clients increase, the amount of storage savings increases.

Observations:
- We expected significantly reduced EPVT usage for the IMS control region when many TMEMBERs have the same user IDs. We observed a 67% EPVT subpool reduction for IMS 12 over IMS 11.
- Greater savings are expected as the number of OTMA clients increases.

4.19 OTMA performance enhancement

With IMS 10 APARs PM20292, IMS 11 PM20293, and IMS 12 in the base code, the following enhancements are realized:
- Reduced path length for OTMA transaction processing
- Simplification in logic when validating a TPIPE name, only when a new TPIPE name is received on a message instead of when each message is received.
- Includes the ICAL enhancements.

Benefit: OTMA performance is improved.

4.20 IRLM Version 2.3 and 2.2

IRLM 2.3 and IRLM 2.2 are both included with IMS 12. IRLM 2.3 and IRLM 2.2 can be used with any supported version of IMS. Note the following information.
- IRLM 2.3 is required by DB2 Version 10. IRLM 2.3 has a 64-bit caller interface; IMS continues to use the 31-bit caller interface.
- IRLM 2.3 requires z/OS 1.10 or later.

IRLM 2.3 provides improved performance for some requests. We do not expect a substantial performance improvement with IRLM 2.3 with IMS.
Database performance topics

In this chapter, we provide additional details about database-related performance improvements in IMS.

This chapter contains the following topics:
- Parallel migration to high availability large database (HALDB)
- Archive enhancements
- Online reorganization performance improvements
- OSAM versus VSAM
- Understanding IMS locking for performance
- CICS threadsafe support
- Temporary close of VSAM data sets with new extents
- Tuning database buffer pools using Buffer Pool Analyzer
5.1 Parallel migration to high availability large database (HALDB)

IMS HD Reorganization Unload utility (DFSURGU0) was enhanced in IMS 10 (APAR PM06635), IMS 11 (APAR PM06639), and IMS 12 to allow unloads of key ranges of an HDAM, HIDAM or HISAM database when migrating to HALDB. Multiple unloads for the same database can be run in parallel. This can significantly reduce the elapsed time for a migration to HALDB. This is especially important for databases with logical relationships because their unloads for migration could require a long time.

5.1.1 The problem before the enhancement

Before this HALDB enhancement, the migration unloading of a logically related database often had a long time duration. When a logical child segment is unloaded, its logical parent must be read, in most cases by using a random read operation. These random reads account for almost all of the elapsed time of the unload.

The logically related database must always be read unless the unload is being executed on the physical logical child of a bidirectional logical relationship with virtual pairing where the PHYSICAL option was specified in the DBD to include the concatenated key in the physical logical child.

The logically related database is always read with the following logical relationships:
- Unidirectional logical relationships
- Physically paired bidirectional relationships
- Virtually paired bidirectional logical relationships when reading the virtual logical child
- Virtually paired bidirectional logical relationships when reading the physical logical child and the VIRTUAL option is specified in the DBD (the concatenated key of the logical parent is not stored in the physical logical child)

The solution to this problem allows multiple unload jobs to be run in parallel. The enhancement allows an unload job to unload part of a database. Without the enhancement, an unload always reads an entire database. Figure 5-1 shows parallel migration to HALDB.

![Figure 5-1 Parallel migration to HALDB](image-url)
5.1.2 Parallel migration to HALDB

Each execution of HD Reorganization Unload can be limited to a key range:

- Unloads of HIDAM and HISAM read only the database records in the key range.
- Unloads of HDAM read all root segments in the database and select only the records for the specified key range.

An execution of HD Reorganization Unload can be restricted to a range of keys when the MIGRATE=YES control statement is included for migration to HALDB. The low (FROMKEY) and high (TOKEY) values must be specified in hexadecimal. These values can be obtained from the output of a LIST.DB DBRC command for the partition.

For SYSIN DD control statements, paste key values from RECON listing):

\[ \text{MIGRATE=YES FROMKEY(key value) TOKEY(key value) KEYLEN(key length)} \]

The statement is shown in the following example:

\[ \text{MIGRATE=YES FROMKEY(FOFOFO) TOKEY(D2F2F0) KEYLEN(3)} \]

5.1.3 Parallel migration to HALDB restrictions

The following restrictions apply for parallel migration to HALDB:

- Can be used only with the MIGRATE keyword.
- Cannot be used with MIGRATX.
- Unload range must match HALDB partition boundaries.
- HD Reload accepts only one input data set.
- Input data set must contain all the records for the partition (or partitions) being loaded.
- HD Reload accepts only one input data set.
- HD Reload can load one or multiple partitions.
- Concurrent unloads require DISP=SHR for database data sets.
- Concurrent unloads require VSAM share options that allow multiple readers.
- SHAREOPTION(3 3) can be used.

5.2 Archive enhancements

Enhancements to IMS archive are related to non-recoverable databases and compression.

5.2.1 Log archiving for non-recoverable databases

\[ \textbf{SPEs:} \text{This enhancement is introduced by small programming enhancements (SPEs): IMS 10 PM18093; IMS 11 PM19363, and IMS 12 PM54945.} \]

Archive (DFSUARCO) is enhanced to write “undo” records for non-recoverable full-function databases. Previously, “undo” records were not archived.

A control statement option is available to not archive “undo” records or write these records as was done previously.
The following tasks are for non-recoverable database logging:

1. The “after-image” log records are not written for non-recoverable databases.
2. The “undo” log records are written to OLDS for non-recoverable full-function databases. In this way, uncommitted updates can be backed out.
3. The “undo” log records are not archived from OLDS before this enhancement. They are archived with this enhancement unless overridden by a control statement.

The problems addressed by this enhancement are as follows:

- Online backout by using archived log stops when it encounters a missing log record. The backout is incomplete; other database updates cannot be backed out and will be stopped.
- Batch backout by using archived log does not back out updates to non-recoverable database but sets no flags in DBRC to prevent its use.

### 5.2.2 New parameter for Log Archive utility (DFSUARC0)

The new record compression parameter CMPRSNR for the Log Archive utility SLDS control statement has the following options (APAR PM71759):

- **CMPRSNR(ALL)**
  - ALL indicates that the log records will be compressed. Compressing the records causes them to be replaced by minimal size placeholder records.
- **CMPRSNR(dddddddd, dddddddd,...)**
  - An eight character string is represented by dddddddd. Log Archive utility compresses log records only if the DB name matches this string.

Log Archive utility does not check whether a DB name is defined to the IMS being archived.

### 5.3 Online reorganization performance improvements

The following enhancements were made for online reorganization (OLR) for HALDBs and can reduce CPU and elapsed times, and log volumes:

- OLR VSAM KSDS sequential access
- Skip GNP call for root-only DB
- Reduce use of the data set busy (ZID) lock during OLR
- Eliminate the block (BID) lock for ILDS updates
- Reduce log records generated during OLR
- OLR locking lookaside
- Increase in concurrent OLR

#### 5.3.1 OLR VSAM KSDS sequential access

OLR is a sequential process, reading the input data set from beginning and going forward. Prior to IMS 11, OLR issues VSAM GET requests to retrieve sequentially from the input data sets, it sets a direct access option in the request parameter list (RPL). That results in VSAM going through one or more index levels to retrieve the data control interval (CI).

OLR is enhanced to take advantage of the VSAM sequential access option when issuing KSDS GET requests to retrieve sequentially from the input data set (or sets). The result is a reduction in CPU and elapsed time.
5.3.2 Skip GNP call for root-only DB

For a root-only database, there is no reason to issue the GNP call because there is no dependent segment to be read. Skipping the GNP call saves CPU and elapsed time.

5.3.3 Reduce use of the data set busy (ZID) lock during OLR

Prior to IMS 11, PHIDAM root insert was executed as follows:
- Insert each root segment:
  - Acquire ZID lock on primary index.
  - Insert primary index segment.
  - Release ZID lock.
- Records in UOR are locked and not accessible by other applications.

With IMS 11, PHIDAM root insert is executed as follows:
- Insert all root segments.
- Acquire ZID lock on primary index once at end of UOR.
- Mass primary index inserts.
- Release ZID lock.

This change reduces locking for PHIDAM.

5.3.4 Eliminate the block (BID) lock for ILDS updates

For a HALDB partition with logical relationships, the updates for the indirect list data set (ILDS) by OLR do not need to obtain the block (BID) lock. The BID lock is used for serialization of the updates to a block across IMS data sharing systems. However the design of HALDB does not allow the ILDS to be updated by more than one IMS at a time. Therefore, the BID lock does not need to be obtained.

However, the ZID lock needs to be obtained to notify other IMS data sharing systems in the case of CI/CA splits. To reduce the use of the ZID lock during OLR, the updates for the ILDS can be saved for insertion at the end of the UOR. The ZID lock will be obtained once before starting to insert all the saved ILDS updates, and then released once. When the UOR covers many entries to the ILDS, many ZID lock requests will be eliminated, which results in a reduction of CPU usage and elapsed time.

5.3.5 Reduce log records generated during OLR

For a HALDB partition undergoing OLR, the database update log records (type x'50') will be consolidated when possible into full block updates. By combining all the updates for a full block into a single type x'50' log record, many of the smaller type x'50' log records will be eliminated. The result is a reduction in the log volume generated by OLR.

5.3.6 OLR locking lookaside

When a lock request is made by the OLR owning region, a lookaside operation is done. The lookaside function determines whether this lock is new or already owned. If the lock is new, normal processing to call IRLM to obtain the lock is done. If this lock is already owned, the lock manager returns to the caller. The savings in path length by optimizing the lock manager out of the call flow results in reduction in CPU and elapsed time.
5.3.7 Increase in concurrent OLR

With IMS 11 OLR enhancements, the optimal number of concurrent OLR tasks is increased. In controlled studies, IBM Silicon Valley Laboratory observed the optimal concurrent OLR increased from eight in IMS 10 to sixteen in IMS 11.

5.4 OSAM versus VSAM

A comparison of IMS full-function overflow sequential access method (OSAM) hierarchical direct access method (HDAM) and OSAM hierarchical indexed direct access method (HIDAM) databases, versus equivalent virtual storage access method (VSAM) databases when using specific IMS workload scenarios, is documented in IMS Performance and Tuning Guide, SG24-7324.

However, in this book, we demonstrate various performance gains inherent within OSAM databases in IMS workload environments. Specialists at IBM Silicon Valley Laboratory studied performance of IMS OSAM versus VSAM databases for partitioned HIDAM (PHIDAM) high availability large database (HALDB) databases. What follows are the main findings using their particular workload.

The database set up was as follows:
- A 32-partition 64 GB HALDB database with 2 GB per partition
- Access method of PHIDAM/OSAM and PHIDAM/VSAM
- Multi-volume databases spanning three volumes each
- Specific customer-like DBD and database data
- Equivalent buffer settings for both OSAM and VSAM environments

Figure 5-1 on page 202 shows an 11% reduction in CPU usage that is required to process roughly the same number of transactions per second, resulting in a 12% improvement in CPU efficiency for the OSAM database workload environment.

<table>
<thead>
<tr>
<th>Table 5-1</th>
<th>Online workload measurement results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PHIDAM/VSAM</td>
</tr>
<tr>
<td>CPU usage (%)</td>
<td>29.71%</td>
</tr>
<tr>
<td>Transactions completed (per second)</td>
<td>1,209</td>
</tr>
<tr>
<td>CPU efficiency (transactions per second)</td>
<td>4,096</td>
</tr>
</tbody>
</table>
A BMP OSAM database workload had a 36% reduction in total elapsed time and a 22% reduction in CPU time over the equivalent VSAM database workload resulting in cost savings in time and processing power. Table 5-2 shows BMP workload measurement results.

<table>
<thead>
<tr>
<th>PHIDAM/VSAM</th>
<th>PHIDAM/OSAM</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Usage (%)</td>
<td>6.18%</td>
<td>6.96%</td>
</tr>
<tr>
<td>Elapsed time</td>
<td>107 minutes 14 seconds</td>
<td>68 minutes 33 seconds</td>
</tr>
<tr>
<td>Total CPU time</td>
<td>10 minutes 28 seconds</td>
<td>8 minutes 7 seconds</td>
</tr>
</tbody>
</table>

Benefits:
- IMS OSAM databases demonstrated a 10%+ improvement in CPU efficiency over the same workload using VSAM databases with our online IMS workload. This means the workload environments that were using the OSAM databases were able to complete 10% more transactions per processor cost over the VSAM database workload environments.
- IMS OSAM databases demonstrated 30%+ and 20%+ savings in total elapsed time and CPU time respectively than VSAM databases with our IMS BMP workload.

5.4.1 Features and benefits of OSAM for databases

The features and benefits of OSAM for databases are described next.

Specific purpose software
OSAM was written for specific IMS usage and is optimized for this function. OSAM is implemented within about seven modules; VSAM is a general purpose access method.

We can conclude that the more specific the function, the more efficient the process. The benefit is reduced CPU cost, less maintenance, and the maintenance is done within IMS.

OSAM blocks and buffers
OSAM block size can be any value up to 32,752 bytes; VSAM CISIZE is a multiple of 512 up to 8 KB, and multiple of 2 KB up to 30 KB.

There is one OSAM buffer-pool containing any number of subpools, a subpool is a set (4 to 32,767) of buffers of the same size. Buffer sizes are 512, 1 KB, 2 KB or multiple of 2 KB up to 32 KB (VSAM buffer sizes more than 4 KB must be multiples of 4 KB)

Any number of subpools can have the same buffer size (VSAM allows a maximum of 16 subpools prior to IMS 12, and 255 in IMS 12). Database data set can be assigned to a specific subpool, which is specified with the IOBF= in DFSVSMxx member.

Multiple OSAM subpools with same size buffers are encouraged, especially in online systems, to enable more parallelism in IMS buffer management and reduce buffer-search cost for hierarchic direct (HD) space search.

OSAM sync point processing
For transactions, BMPs and checkpointing batch programs, all IMS DB writes should take place at sync point time.
At sync point, VSAM writes each buffer individually (one at a time). OSAM chains together blocks for the same data set by using a single I/O operation, for non-page-fixed subpool, which is limited to 49 blocks per start I/O (SIO).

OSAM does parallel writes, for multiple subpools to multiple volumes and in parallel with VSAM. This way leads to reduced elapsed time and region occupancy.

**OSAM sequential buffering (SB)**
Sequential buffering is implemented with chained reads (10 consecutive blocks) instead of single-block reads. Sequential buffering assumes that if you need the first block, you will also need the immediately following ones.

With look-ahead reading (asynchronous read-ahead) while processing the current sequential set of blocks, OSAM will read the next set of data required by the application so that the required data is always in buffers.

OSAM SB is “enabled” by the user but is dynamically switched on or off by IMS, according to the estimated or measured benefit calculated by various algorithms.

OSAM reads data into special OSAM sequential buffering (OSB) buffer sets and does not interfere with normal OSAM buffers. It copies a buffer from the OSB buffer set to a normal OSAM buffer when needed instead of waiting to retrieve it from DASD.

OSAM SB is exploited by BMPs (and theoretically, MPPs), stand-alone batch, utilities, online image copy (IC), unload, scan, prefix update, surveyor, HALDB online reorganization, and more.

Totally sequential processes can run in less time, but jobs with some element of sequential processing can see a benefit.

**OSAM background write**
This OSAM facility is only for stand-alone batch and is not a true “background” write; it occurs when a forced write for space is needed by a batch program. OSAM issues a synchronous chained write for all the buffers in the subpool, similar to an application CHKP for just the subpool.

**OSAM 8 GB data sets**
Unlike VSAM, which has a 4 GB data set limit, OSAM data sets (non HALDB) can be up to 8 GB in size (for even block size).

**Caching OSAM data in the coupling facility**
Caching small, highly volatile, shared OSAM DBs reduces the impact of OSAM reread activity due to buffer invalidation and replaces DASD I/O with cache coupling facility (CF) access.

Caching uses store-through cache; when an application reads data from DASD, it is copied (as a user option) into the CF. At application commit time, changed data is written first to DASD, and then to CF before locks are released.

Caching is a user-specified option at the OSAM subpool level, with the IOBF statement in the DFSVSMxx member. It has an additional option to cache updated data only or cache all referenced data.
5.4.2 Considerations for using OSAM

With OSAM, there are certain aspects to consider regarding hiperspace buffers and allocating multi-volume data sets.

**Hiperspace buffers**

Only VSAM allows a subpool to be defined to include buffers in hiperspace.

**Allocating multi-volume data sets**

Consider the following information:

- The number of OSAM secondary extents is limited to a range of 52 - 60 (depending on block size).
- OSAM can use job control language (JCL) or access method services (AMS) to allocate database data sets.
- The rules differ for allocating multi-volume SMS managed and non-SMS managed direct access storage device (DASD) data sets.
- In reusing an OSAM multi-volume data set, you might potentially leave an EOF mark on a volume that is not used on a reload.
Example 5-1 shows allocation of a single-volume OSAM database data set by using JCL.

Example 5-1  Allocate single-volume OSAM database data set: JCL

```
//ALLOC1 EXEC PGM=IEFBR14
//DD1 DD DSN=HIGHNODE.MIDNODE.LOWNODE,
// DISP=(NEW,CATLG),
// SPACE=(CYLS,(200,100)),
///* add UNIT and VOLSER if needed
///* add SMS parameters if needed
///* do not code DCB parameters
/*
```

Example 5-2 shows allocation of a single-volume OSAM database data set by using AMS.

Example 5-2  Allocate single-volume OSAM database data set: AMS

```
//ALLOC1 EXEC PGM=IDCAMS,REGION=4096K
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
/* */
/* DELETE OLD DATA SET IF IT EXISTS */
/* */
DELETE 'HIGHNODE.MIDNODE.ENDNODE'
IF LASTCC <= 8 THEN SET MAXCC = 0
/* */
/* ALLOCATE - WITH SMS PARAMETERS */
/* */
ALLOCATE DSN('HIGHNODE.MIDNODE.ENDNODE') -
NEW CATALOG -
DATACLAS(DCLAS) -
MGMTCLAS(MCLAS) -
STORCLAS(SCLAS) -
SPACE(200 100) CYLINDERS
/*
```

Example 5-3 shows allocation of a multi-volume OSAM database data set by using JCL for non-SMS managed DASD.

Example 5-3  Allocate multi-volume OSAM database data set: JCL, non SMS-managed DASD

You must allocate and catalog the data set exactly as shown
If you try VOL=SER=(VOL111,VOL222,VOL333),DISP=(NEW,CATLG)
or
VOL=(,,,3),DISP=(NEW,CATLG)
you will get a primary extent only on the first volume, the other volumes will get only secondary extents
// EXEC PGM=IEFBR14
//DD1 DD DSN=HIGHNODE.MIDNODE.ENDNODE,
// DISP=(NEW,KEEP),
// SPACE=(CYLS,(200,100)),
// UNIT=3390,
// VOL=SER=VOL111
/*
*/
// EXEC PGM=IEFBR14
//DD2 DD DSN=HIGHNODE.MIDNODE.ENDNODE,
// DISP=(NEW,KEEP),
Example 5-4 shows allocation of a multi-volume OSAM database data set by using AMS for non-SMS managed DASD.

Example 5-4  Allocate multi-volume OSAM database data set: AMS, non SMS-managed DASD

Restriction: Multi-volume OSAM database data sets on non-SMS DASD cannot be backed up with IMS Image Copy 2. Each data set has a sequence number of 1. DF/DSS cannot process this information.
Figure 5-3 shows an IEHLIST listing, where each data set has a sequence number of 1.

![Partial IEHLIST output]

Figure 5-3  IEHLIST listing where each data set has a sequence number of 1

Multi-volume OSAM database data sets on SMS DASD can be backed up with IMS Image Copy 2. Each data set has the correct sequence number.

Figure 5-4 shows an EHLIST listing where each data set has the correct sequence number.

![Partial IEHLIST output]

Figure 5-4  IEHLIST listing where each data set has the proper sequence number

You must do the following steps when you allocate a multi-volume OSAM database data set by using JCL for SMS-managed DASD:

- Use an SMS storage class with the guaranteed space attribute.
- Specify the volume serial numbers.

If you do not do these steps, only the first volume will get a primary extent; the other volumes will get only secondary extents.
Example 5-5 shows allocation of a multi-volume OSAM database data set by using JCL for SMS-managed DASD.

**Example 5-5  Allocate multi-volume OSAM database data set: JCL, SMS-managed DASD**

```bash
// EXEC PGM=IEFBR14
//DD123 DD DSN=HIGHNODE.MIDNODE.ENDNODE,
// DISP=(NEW,CATLG),
// SPACE=(CYL,(200,100)),
// UNIT=3390,
// VOL=SER=(VOL111,VOL222,VOL222),
// STORCLAS=GTDSPACE
/*

Example 5-6 shows allocation of a multi-volume OSAM database data set by using AMS for SMS-managed DASD.

**Example 5-6  Allocate multi-volume OSAM database data set: AMS, SMS-managed DASD**

```bash
// EXEC PGM=IDCAMS,REGION=4096K
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *
DELETE 'HIGHNODE.MIDNODE.ENDNODE'
IF LASTCC <= 8 THEN SET MAXCC = 0
ALLOC DSN('HIGHNODE.MIDNODE.ENDNODE') -
  NEW CATALOG -
  SPACE(200,100) CYLINDERS -
  UNIT(3390) -
  VOL(VOL111 VOL222 VOL333) -
  STORCLAS(GTDSpace)
/*

### 5.4.3 Summary of benefits of using OSAM

Benefits of using OSAM are summarized in the following list:

- OSAM is shown to be more efficient than VSAM in the following ways:
  - Less CPU
  - Chained writes
  - Parallel writes
  - Chained and look-ahead reading with OSAM SB
- OSAM supports up to 8 GB data sets (for non-HALDB)
- OSAM allows CF caching for volatile shared DBs
- Online region occupancy is reduced.
- Batch elapsed times are reduced.

**Tip:** If performance or cost are key factors in your system, you might want to consider the use of OSAM.
5.5 Understanding IMS locking for performance

The information in this chapter is based on the white paper IMSS Locking with Program Isolation or the IRLM, by Rich Lewis of IBM:

http://www.whitepapersdb.com/white-paper/3539/ims-locking-with-program-isolation-or-the-irlm

5.5.1 Lock managers

IMS has the following lock managers:

- Program isolation (PI)
  - Does not support data sharing.
  - Locks are managed by IMS online systems.

- Internal resource lock manager (IRLM)
  - Can be used with or without data sharing.
  - IRLM is a separate address space.
  - Multiple IRLMs are used with data sharing across logical partitions (LPARs).

- Fast Path lock manager
  - Used without data sharing.
  - Fast Path (FP) also uses PI or IRLM, and is required for deadlock detection.

Figure 5-5 shows lock compatibility.

<table>
<thead>
<tr>
<th>Lock Level</th>
<th>1 – read</th>
<th>2 – share</th>
<th>3 - update</th>
<th>4 - exclusive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 – read</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>2 – share</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>3 - update</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>4 - exclusive</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lock Level</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 – read</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>3 – erase</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>4 – share</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>6 – update</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>8 - exclusive</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

Figure 5-5  Lock compatibility matrix
5.5.2 Full-function: Database record locks

Consider the following information about database record locks:

- They are requested when a database record is accessed. HDAM and PHDAM lock the root anchor point (RAP).
- They are used to serialize access to database records.
- The level depends on the PROCOPT value:
  - PROCOPT=G
    - PI level 2; IRLM level 4.
    - Readers can be positioned in a database record concurrently.
  - PROCOPT=update
    - PI level 3 (or 1); IRLM level 6.
    - Updaters have exclusive access to database record.
- They are released as follows:
  - If no update is issued, they are released when program communication block (PCB) position is moved to another database record.
  - If an update is issued, the locks are held until sync point; PI demotes level 3 to level 1 when position is moved off the record and root is not updated.
- For HDAM and PHDAM DB, the lock is held on the RAP.
  - Lock is on the root anchor point (RAP), which is another reason to have more RAPs than roots. Typical guidance is for the number of RAPs to be greater than twice the number of roots.
- Often, this lock is the most important. “Control” records often produce lock conflicts.

5.5.3 Full-function: Segment locks

If the PI lock manager is used, individual segments are locked when they are updated. Other segments in the database record can be accessed when the updating program moves its position to another database record. If the IRLM is used, individual segments are not locked, instead, the lock on the database record is held until the updates are committed.

Note the following information about full-function segment locks:

- The locks are used only with PI.
- A segment lock is always requested for updates to dependent segments:
  - It is used to serialize access to updated dependent segments.
  - Updates include updates to pointers in the segment (to other segments).
  - Hierarchical indexed sequential access method (HISAM) lock is for overflow logical records.
- The lock is always at PI level 3 (only read). See Figure 5-5 on page 214.
- Segment locks are released at sync point.
- If a database record lock is held at level 1 (by another program), some dependent segment (or segments) are locked at level 3. Segment lock is tested when the dependent segment is accessed; test waits if lock is held but does not get the lock.
The following figures illustrate how PI can provide greater concurrency by taking locks at the segment level:

- Figure 5-6 shows that PI can provide more concurrency.

![Figure 5-6: PI can provide more concurrency](image)

- Figure 5-7 shows a lock at root level of a DB record.

![Figure 5-7: Lock at root level of a DB record](image)
Figure 5-8 shows a lock at dependent 1 level of a DB record.

- PI may provide more concurrency
  - Non-shared lock of dependent makes all of its children inaccessible

Figure 5-9 and Figure 5-10 on page 218 show a lock at dependent 2 level of a DB record.

- PI may provide more concurrency
  - Non-shared lock of twin does not make preceding twins inaccessible
Figure 5-10 shows a lock at dependent 2 level of a DB record.

![Figure 5-10](image)

Figure 5-10  Lock at dependent 2 level of a DB record (Part 2 of 2)

Figure 5-11 shows how separate branches of a DB record can be accessed at the same time.

![Figure 5-11](image)

Figure 5-11  Different branches of a DB record can be accessed at the same time

---

### 5.5.4 Full-function locks: Block locks with IRLM

Full-function block locks are used only for updates (insert, delete and replace calls). Note the following information about block locks with IRLM:

- Used only with block level data sharing (SHARELVL=2 or 3).
- Are requested when a block is updated.
- Are used to serialize updates from different IMS systems and are requested with a private attribute. Cannot be shared across different IMS systems (no matter what level).
- Level for OSAM and ESDS is always IRLM level 4.
Level for KSDS (primary and secondary indexes):
- Inserts and replaces IRLM level 4.
- Deletes IRLM level 3.
- CI/CA splits IRLM level 6.

These locks are released at sync point.
They are shared within an IMS system unless there is a delete with insert or replace of a KSDS record or a CI/CA split.

Block lock conflicts typically occur for updates in a small database or small part of a database. They are common for the following items:
- Secondary index with high insert/delete activity to small range of records
  - Records in the same CI
  - Often because of keys based on current time
- Small database with "control" records
  - Statistics maintenance
  - Other similar functions

5.5.5 Full-function locks: Busy locks

Note the following information about busy locks:
- These locks are requested to serialize the following activity to a data set:
  - Update to KSDS with block level data sharing
    - Insert IRLM level 8
    - Non-insert IRLM level 2
  - Open and close of data set PI level 4 IRLM level 8
  - Creation of new block in data set PI level 4 IRLM level 8
- These locks are released at the end of an operation (open, close, update, and so on).
- Lock waits are rarely a problem with busy locks.
- The number of lock requests can be important for data sharing.
  They depend on the coupling facility (CF) accesses for the lock structure for index updates.

5.5.6 Fast path locks: CI lock

Note the following information about CI lock:
- Lock is similar to the database record lock for full-function.
- Lock is requested when a CI is read into a buffer and is used to serialize access to segments in a CI.
- The level depends on the PROCOPT value:
  - If PROCOPT=6, then FP is level 1 and IRLM is level 2
  - If PROCOPT=update, then FP is level 4 and IRLM is level 8
- The lock is released with update by the output thread (sync point with VSO). Without an update, the lock is released by sync point or when buffer is stolen.
5.5.7 Fast path locks: Unit of work (UOW) lock

Note the following information about UOW lock:

- UOW lock is used only when high speed sequential processing (HSSP) or high speed reorganization (HSR) is active.
  - It is requested instead of a CI lock by HSSP and HSR.
  - It can be requested in addition to CI lock by other IMS components.
- The lock level depends on the PROCOPT value:
  - For non-HSSP or HSR request: FP level 1 and IRLM level 2
  - For HSSP or HSR request: FP level 4 and IRLM level 8
- The lock is released as follows:
  - For non-HSSP, non-HSR request: When all locks on CIs in the UOW are released.
  - For HSSP request: If update, then by output thread; if no update, then by sync point.
  - For HSR request: At the end of reorganization of the UOW.

5.5.8 Lock timeouts

Lock timeouts are managed by IMS as follows:

- PI and Fast Path lock managers never time out (end) a lock request.
- The `Modify (F) irmlproc,SET` command dynamically alters the TIMEOUT value known to IRLM, for example:
  ```
  F irmlproc,SET,TIMEOUT=seconds,imssubsystemname
  ```
  - Controls the reporting of "long locks" for an IMS system that uses the IRLM.
  - Does not end a lock request.
  - Drives an IMS LOCKTIME process to check on timeouts.
- The IRLM lock timeout function is controlled by the LOCKTIME control statement in the DFSVSMxx or DFSVSAMP (VSPEC) PROCLIB member. The LOCKTIME statement allows the user to specify the maximum time that a dependent region waits for an IRLM lock before it is abended.
  ```
  LOCKTIME=(mtime,maction,btime,baction)
  ```
- The LOCKTIME parameter can be changed with the following command:
  ```
  UPDATE IMS SET(LCLPARM((LOCKTIME(MSG(mtime),MSGOPT(maction),BMP(btime),BMPOPT(baction))))
  ```

Transaction is retried after a U3310 lock timeout abend

Lock timeouts occur when the IRLM is used as the lock manager and IMS LOCKTIME value is exceeded. Installations can choose whether lock timeouts produce a BD status code for the DL/I call or a U3310 abend. Note the following information:

- Exceptions for IFP, CPI-C driven programs, and protected conversations; error messages are discarded.
- A retry can be overridden by a DFSNDMX0 exit routine.
- Previous IMS versions always discarded input message after a timeout, unless DFSNDMX0 indicated that a retry should be done.

**Benefit:** The default is to attempt to process the input message.
Lock timeout message and logging

IMS 12 adds optional DFS2291I diagnostic messages for lock timeouts:

- Timeouts occur only with IRLM and IMS LOCKTIME specified.
- Previous IMS releases provide information only through RMF reports.
- IMS 12 writes log record X’67D0’ subtype X’1B’ for lock timeouts.
- The trace record contains same information as the DFS2291I message.

**Benefit:** The benefit of this enhancement is that information about lock conflicts is more readily available and helps identify, and reduce or eliminate, long locks.

Installation chooses whether the DFS2291I messages are issued:

- The parameter that controls the messages is in the DIAGNOSTIC section of DFSDFxxx:
  ```
  <SECTION=DIAGNOSTIC>
  MSG2291I=ISSUE | SHORT | SUPPRESS
  
  The values are as follows:
  - ISSUE creates multiline messages
  - SHORT creates one line messages
  - SUPPRESS is the default
  
  The number of messages is limited for a U3310 situation. A message is issued only for the first five U3310s for a transaction.

Figure 5-12 shows a sample of the lock timeout message.

### Lock Timeout Message

- New DFS2291I message issued with U3310 abend or "BD" status code
  - U3310 or "BD" indicates that waiter has exceeded the specified wait time
  - DFS2291I is either a multiple line message
    ```
    DFS2291I LOCKNAME=0900004288800201D7
    DFS2291I DBNAME=DLVNT202 LOCKFUNC=GET LCL AND GBL ROOT LOCKS
    DFS2291I BLOCKER PST=0001 TRAN=NQF1 PSB=PMVAP212 TYPE=MPP
    DFS2291I BLOCKER TRANELAPSEDTIME=00:01:11 IMSID=IMS1
    DFS2291I BLOCKER RECOVERY TOKEN=IMS1 0000000200000000
    DFS2291I VICTIM PST=0002 TRAN=SHF1 PSB=PMVAP213 TYPE=MPP
    DFS2291I VICTIM TRANELAPSEDTIME=00:00:49 IMSID=IMS1
    DFS2291I VICTIM RECOVERY TOKEN=IMS1 0000000300000000
    
    - Or a "short" one line message
      ```
      DFS2291I BLOCKER PST=0001 TRAN=NQF1 PSB=PMVAP212 TYPE=MPP
      
    Figure 5-12  Lock timeout message
  ```

5.5.9 Deadlock detection

IRLM provides deadlock management. By using the DEADLOK parameter in the DXRJPROC procedure, you can specify the local deadlock-detection interval and the number of local
cycles that are to occur before IRLM initiates a global deadlock detection. You can also
dynamically modify the DEADLOK values by using the z/OS MODIFY IRLMPROC command.
The following list describes how IMS detects deadlocks:

- Fast Path lock manager does not detect deadlocks.
  When a lock request waits, Fast Path passes information to another lock manager (PI or
  IRLM) to perform deadlock detection.
- PI checks for deadlocks when a lock request waits.
- IRLM checks for deadlocks on a timer basis.
  - The DEADLOK parameters in the DXRJPROC procedure to start up IRLM defines
    local and global values, for example:
    DEADLOK='5,1'
  - A wait must exist through two cycles before IRLM checks for a deadlock. With local
    value of 1 second, deadlock could last 2 seconds before detection.

  **Tip:** Reasonable values for local are 1 second or less.

Deadlocks can be created with IMS and non-IMS resources as follows:
- CICS applications with IMS and VSAM
- IMS TM applications with IMS DB and DB2
- DB2 stored procedures with IMS DB and DB2

The following example shows deadlock:

Tran A holds IMS lock X
Tran B holds DB2 lock Y
Tran A requests DB2 lock Y and waits
Tran B requests IMS lock X and waits – DEADLOCK!

These deadlocks are resolved only by timeouts, usually resolved by the “other” resource
manager, not IMS. IMS only times out lock requests when the LOCKTIME value for IMS is
specified with IRLM.

### 5.5.10 Handling deadlock victims

The following actions are for deadlock victims:

- Abend U0777
  Message processing program (MPP), Java message processing region (JMP), IMS Fast
  Path (IFP), batch message processing (BMP), or Java batch processing region (JBP)
- Rescheduled
  MPP, JMP, IFP and BMP messages are rescheduled.
- Abend U0123
  Advanced Program-to-Program Communication (APPC) Common Programming
  Interface - Communications (CPIC) driven or modified standard application
- CICS task: CICS ADCD abend
- Terminated thread
  Open Database Access (ODBA) thread: AIB “system failure” return code X’00000108’,
  reason code X’00000244’ and error extension code X’10000309’ and thread is terminated.
Exceptions of abend for deadlock victims are as follows:

- **INIT STATUS GROUPB**
  Back out occurs and program receives a BC status code.
- **Non-message-driven BMP or JBP with Fast Path program communication block (PCB)**
  Back out occurs and program receives an ‘FD’ status code.
- **Deadlock during sync point processing with FLD calls**
  Back out and reprocessing occur.

### 5.5.11 Design guidance

The following design options reduce lock contention and deadlocks:

- **Minimize PROCOPT values.**
  The value of PROCOPT=A produces “non-shared” level locks.
- **Take frequent checkpoints.**
  However, do not create a logging problem by checkpointing too much user data, such as all of working storage.
- **Be wary of communications during a synchronization interval:**
  - **OTMA commit mode 1** with either of the following values:
    - `synclevel=syncpoint`
    - `synclevel=confirm`
  - **APPC with either of the following values:**
    - `synclevel=syncpoint`
    - `synclevel=confirm`
- **Synchronous callout (ICAL).**
  The default timeout for ICAL is 10 seconds. The application can set any value
- **Communications delays are likely to cause locking problems.**
- **Try to limit high frequency updates to any record.**
  - “Control” records can be a problem, for example, obtaining the “next invoice number” from a control record. Possible solutions are as follows:
    - Delay calls to the record until the end of the transaction.
    - Use multiple records, one for each series of numbers.
    - Use non-sequential numbers, such as choosing numbers at random.
  - Databases with only a few database records are often problems.
- **Provide free space in PHIDAM or HIDAM with block level data sharing.**
  Without free space all inserts go to end of data set causing block lock conflicts
Be wary of the PROCOPT=E value:
- PROCOPT=E on root
  - Schedules program exclusively for the database in a designated IMS subsystem so that it does not affect scheduling or locking in other IMS subsystems.
  - If no data sharing, then no locks are used for the database
  - If data sharing, then all locks for database are held until sync point.
PROCOPT=E on root is sometimes used to allow BMPs with infrequent checkpoints to run.
- PROCOPT=E on a dependent segment
  - Schedules program exclusively for the segment in an IMS subsystem.
  - Locks are used for the database records.
  - No PI locks are used for the segment.

Ensure that applications access the databases and segments in the same order.
- For example, all databases are accessed in alphabetical order by name and the segments are accessed in key sequence.
- As long as all programs follow the same conventions, the chances of a deadlock are reduced.

Tune the system and applications.
- Use lots of database buffers.
- The faster an application runs, the shorter the time it holds locks.

### 5.5.12 Space for lock control blocks

When using PI, consider the following information:
- Each locked resource uses 24 byte control block
- Each holder of a resource lock uses 24 byte control block
- A good rule is that each lock requires 48 bytes
- PI lock control block storage locations are as follows:
  - With Fast Path: extended common storage area (ECSA)
  - Without Fast Path: extended private of DL/I SAS address space
  - Without Fast Path or DL/I SAS: extended private of control region
- PI storage is limited by the PIMAX execution parameter.
  - If PIMAX is not specified, storage is limited by second subparameter of CORE= on IMSCTF macro.
  - PIINCR specifies the increments in which storage is acquired.

When using IRLM, consider the following information:
- Each lock requires about 540 bytes in 64-bit storage of IRLM address space.
- Space can be limited by the z/OS MEMLIMIT parameter on the job or job step.
When using coupling facility lock structure, consider the following information:

- Each lock protecting an update uses an entry in the lock record list.
  - All block locks
  - Level 6 database record locks
  - Level 8 Fast Path CI and UOW locks
- Record list entries are approximately 250 bytes.
- The goal for the lock table is 1000 entries per held lock.
  - Provides false contention rate of 0.1%
  - Entries are typically 2 bytes, therefore, about 2000 bytes per held lock

Excessive space for locks is usually caused by BMPs or a small subset of BMPs.

The following messages can be received when lock space is exhausted:

- PI: U0775 abend of requestor
- IRLM: U3300 abend of requestor
- Lock structure record list: U3307 of requestor

### 5.5.13 LOCKMAX usage

The LOCKMAX parameter limits the number of locks held by a dependent region or batch job at any time. The number that is specified indicates units of 1000. The parameter can be specified on the PSBGEN statement of PSB. If specified as region parameter, it overrides PSB specification.

When an application exceeds the value specified for the LOCKMAX parameter, a pseudoabend of type U3301 results.

Log records contain the maximum number actually used:

- For online systems: X’37’ and X’5937’
- For batch data sharing: X’41’

Specify LOCKMAX in all dependent regions in test systems.

**Tip:** Analyze log record type X’37’, X’5937’, and X’41’ to trend locks that are used by application programs to detect those applications that obtain a large number of locks.

### 5.5.14 A comparison of PI and IRLM

The following list compares PI and IRLM:

- IRLM is required for block level data sharing
- PI has a shorter path length, but might not be significant in total application path length
- PI has a maximum of 63 waiters, the sixty-fourth waiter receives U2478 abend and MPP or JMP is rescheduled.
- IRLM has no limit on the number of waiters.
- IRLM has a “long locks” capability; it reports locks that wait for a long time.
- Lock timeout capability requires IRLM
5.5.15 Analyze locking with the /TRACE SET program isolation command

In an online environment, PI tracing is executed by the /TRACE command. For batch or DB/DC environments, specify the LOCK=OUT value on the OPTIONS statement at system initialization time.

PI trace entries are written in the same trace table as DL/I and lock activity trace entries. A PI trace entry contains information about program isolation ENQ/DEQ calls, and DL/I calls. The trace entry that is created by /TRACE TABLE DL/I contains different information about DL/I calls and is written as a separate entry in the same trace table. Starting the LOCK trace also causes PI tracing to occur.

If PI is entered without the OPTION keyword, the program isolation trace is kept in storage without being logged. If you are using the program isolation trace to provide statistics and performance data, enter the OPTION(ALL) keyword.

To turn on program isolation tracing, include the additional time field in the trace record and have the trace information logged by issuing the following command:

/TRACE SET PI OPTION ALL

To stop program isolation tracing, issue the following command.

/TRACE SET OFF PI

This command is not selective and will trace locking activity for all workload in the IMS system. The PI trace supports both lock managers, IRLM or PI, even though it is called the PI trace. The command produces X'67FA' records when directed by the command for the trace records to be logged. Additional log records being written out to the IMS logs might mean more frequent OLDS switches and more frequent log archiving.

Use the file select and formatting print utility (DFSERA10) with program isolation trace record format and print module DFSERA40 to print the X'67FA' trace records.

Example 5-7 shows the control statements that are required for DFSERA40.

Example 5-7  DFSERA40 formatting

<table>
<thead>
<tr>
<th>CONTROL</th>
<th>CNTL</th>
<th>OPTION</th>
<th>PRINT</th>
<th>OFFSET=5, VALUE=67FA, FLDLEN=2,</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>COND=E, EXITR=DFSERA40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Additional OPTION control statements can be added to filter the output to your requirements.

The program isolation trace report utility (DFSPIRP0) can also be used to print a report from the X'67FA' log records that are produced by a PI trace. The report created by the DFSPIRP0 utility shows only those enqueues that were required to wait (the resource was not immediately available). You can restrict the report produced by the DFSPIRP0 utility to a time period by specifying a PRINT control statement.
5.5.16 Sample locking reports

Figure 5-13 shows sample output from a program isolation trace record format and print module DFSERA40.

![Sample Output from DFSERA40](image1)

Figure 5-13  Sample output from DFSERA40 (the spacing of fields is altered)

Figure 5-14 shows a sample report from program isolation trace report utility DFSPIRP0.

![Sample Output from DFSPIRP0](image2)

Figure 5-14  Sample output from program isolation trace report utility DFSPIRP0
Figure 5-15 shows a IMS monitor report.

**IMS monitor**

- PROGRAM I/O Report

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>PCB NAME</th>
<th>IWAIT</th>
<th>TOTAL</th>
<th>MEAN</th>
<th>MAXIMUM</th>
<th>DDN/FUNC</th>
<th>MODULE</th>
</tr>
</thead>
<tbody>
<tr>
<td>AZLACL</td>
<td>RZCMA001</td>
<td>2</td>
<td>3419</td>
<td>1709</td>
<td>1991</td>
<td>PI</td>
<td>RZCMA001</td>
</tr>
</tbody>
</table>

- REGION IWAIT Report

<table>
<thead>
<tr>
<th><strong>REGION</strong></th>
<th>OCCURRENCES</th>
<th>TOTAL</th>
<th>MEAN</th>
<th>MAXIMUM</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DL/I CALLS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>20959</td>
<td>1309</td>
<td>4696</td>
<td>PI=SMWLJ001</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>48901</td>
<td>2573</td>
<td>26494</td>
<td>PI=RZCMA001</td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
- "PI" appears for both PI and IRLM
- Segment code is "1" except for PI segment locks
- You can examine these reports to see if you have a lot of locks and to determine their average wait times

Figure 5-15  IMS Monitor report

Figure 5-16 shows an RMF II long lock detection report.

**RMF II - IRLM long lock detection report**

- Shows lock waits greater than IRLM LOCKTIME value
  - Also shows holders of lock and other waiters for lock

<table>
<thead>
<tr>
<th>Command ====&gt;</th>
<th>RMF - ILOCK IRLM Long Lock Detection</th>
<th>Line 1 of 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU= 37/35 UIC=2540 PR= 0</td>
<td>Scroll ===&gt; HALF</td>
<td>System= RMF5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State</th>
<th>Type</th>
<th>Lock_Name</th>
<th>PSB_Name</th>
<th>Elap_Time</th>
<th>CICS_ID</th>
<th>IMS_ID</th>
<th>Recovery-Token</th>
<th>PST#</th>
<th>Trx/Job</th>
<th>Wait_Time</th>
<th>DB/Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF Structure ACOXLOCK</td>
<td>at 07/28/2006 13:02:10 Deadlock Cycle 00002EC7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOP BMP</td>
<td>09C943CFA7800101D7000000000000000</td>
<td>DFSSAMB1</td>
<td>00:06:04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOCKER ACO3</td>
<td>AC03</td>
<td>0000003000000000</td>
<td>0006</td>
<td>IRLMTOPZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOP BMP</td>
<td>09C3614505800101D7000000000000000</td>
<td>DFSSAMB1</td>
<td>00:06:09</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOCKER ACO1</td>
<td>AC01</td>
<td>0000000600000000</td>
<td>0006</td>
<td>IRLMTOPA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAITER BMP</td>
<td>09C3614505800101D7000000000000000</td>
<td>DFSSAMB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO2</td>
<td>AC02</td>
<td>0000000800000000</td>
<td>0007</td>
<td>IRLMWTA1</td>
<td>00:05:52</td>
<td>DI21PART</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAITER BMP</td>
<td>09C943CFA7800101D7000000000000000</td>
<td>DFSSAMB7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO2</td>
<td>AC02</td>
<td>0000000900000000</td>
<td>0008</td>
<td>IRLMWTA2</td>
<td>00:05:42</td>
<td>DI21PART</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5-16  RMF II long lock detection report
DFSERA30 deadlock report

- Provides detailed information about each deadlock

```
DEADLOCK ANALYSIS REPORT - LOCK MANAGER IS IRLM

RESOURCE DMB-NAME LOCK-LEN LOCK-NAME - WAITER FOR THIS RESOURCE IS VICTIM
01 OF 02 CMLDDCDB 08 7EB22000843A01D7
KEY FOR RESOURCE IS FROM DELETE WORK AREA
KEY=(200414913326180)

IMS-NAME TRAN/JOB PSB-NAME PCB--DBD PST# RGN CALL LOCK LOCKFUNC STATE
WAITER IMS2 TRLDDC1 CMLDDCDB CMLDDCDB 00003 MPP DLET GBIDP 22400318 04-P
HOLDER IMS1 USMEED2 CMLDDCDB ------- 00007 MPP ----- ----------- 04-P

RESOURCE DMB-NAME LOCK-LEN LOCK-NAME
02 OF 02 CMLDDCDB 08 7EB22B3E843A01D7
KEY IS ROOT KEY OF DATA BASE RECORD ASSOCIATED WITH LOCK
KEY=(200414913326180)

IMS-NAME TRAN/JOB PSB-NAME PCB--DBD PST# RGN CALL LOCK LOCKFUNC STATE
WAITER IMS1 USMEED2 CMLDDCDB CMLDDCDB 00007 MPP GET GRIDX 30400358 06-P
HOLDER IMS2 TRLDDC1 CMLDDCDB ------- 00003 MPP ----- ----------- 06-P

DEADLOCK ANALYSIS REPORT - END OF REPORT
```

RMF coupling facility reports

- Coupling Facility Usage Summary – Structure Summary

```
COUPLING FACILITY NAME = CF01
TOTAL SAMPLES(AVG) = 240 (MAX) = 240 (MIN) = 240

STRUCTURE SUMMARY

<table>
<thead>
<tr>
<th>STRUCTURE TYPE</th>
<th>NAME</th>
<th>STATUS</th>
<th>% OF ALLOC</th>
<th>% OF CF</th>
<th>% OF ALL</th>
<th>% OF CF REQ/SEC</th>
<th>% OF AVG</th>
<th>% OF UTIL</th>
<th>% OF REQ</th>
<th>% OF UTIL SEC</th>
<th>% OF TOT/CUR</th>
<th>% OF TOT/CUR</th>
<th>% OF TOT/CUR XI'S</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCK</td>
<td>MMLL_IRLML</td>
<td>ACTIVE</td>
<td>34M</td>
<td>0.2</td>
<td>71551</td>
<td>0.1</td>
<td>0.1</td>
<td>59.63</td>
<td>62K</td>
<td>0</td>
<td>8388K</td>
<td>N/A</td>
<td>28</td>
</tr>
</tbody>
</table>

COUPLING FACILITY USAGE SUMMARY
```

Figure 5-17 shows a DFSERA10 deadlock report.

Figure 5-18 shows an RMF coupling facility structure summary report.
Figure 5-19 shows an RMF coupling facility structure summary report; important fields are highlighted.

**RMF coupling facility reports**

- Coupling facility usage summary – structure summary

```
<table>
<thead>
<tr>
<th>STRUCTURE SUMMARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
```

- **Structure Size**

```
<table>
<thead>
<tr>
<th>Structure Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>34M</td>
</tr>
</tbody>
</table>
```

- **Record List**

```
<table>
<thead>
<tr>
<th>AVG</th>
<th>LST/DIR</th>
<th>DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>59.63</td>
<td>62K</td>
<td>0</td>
</tr>
</tbody>
</table>
```

- **Lock Table**

```
<table>
<thead>
<tr>
<th>LOCK</th>
<th>DIR REC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8389K</td>
</tr>
</tbody>
</table>
```

```
Figure 5-19  RMF coupling facility structure summary report showing important fields
```
Figure 5-20 shows an RMF coupling facility structure activity report.

### RMF coupling facility reports

- **Coupling facility structure activity**

<table>
<thead>
<tr>
<th>STRUCTURE NAME</th>
<th>TYPE</th>
<th>STATUS</th>
<th>SYSTEM NAME</th>
<th>AVG/SEC</th>
<th>% OF</th>
<th>SERV TIME (MIC)</th>
<th>REQ</th>
<th>STD_DEV</th>
<th>REQ</th>
<th>STD_DEV</th>
<th>AVG</th>
<th>CONTENTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMSL_1MSIRM</td>
<td>LOCK</td>
<td>ACTIVE</td>
<td>SYSL</td>
<td>584</td>
<td>0.8</td>
<td>18.3</td>
<td>0</td>
<td>8.2</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>MMSL_2MSIRM</td>
<td>LOCK</td>
<td>ACTIVE</td>
<td>SYSL</td>
<td>584</td>
<td>0.8</td>
<td>18.3</td>
<td>0</td>
<td>8.2</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>MMSL_3MSIRM</td>
<td>LOCK</td>
<td>ACTIVE</td>
<td>SYSL</td>
<td>584</td>
<td>0.8</td>
<td>18.3</td>
<td>0</td>
<td>8.2</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>MMSL_4MSIRM</td>
<td>LOCK</td>
<td>ACTIVE</td>
<td>SYSL</td>
<td>584</td>
<td>0.8</td>
<td>18.3</td>
<td>0</td>
<td>8.2</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

---

- Also shows holders of lock and other waiters for
Figure 5-21 shows an RMF coupling facility structure activity report; important fields are highlighted.

RMF coupling facility reports

- Coupling facility structure activity

<table>
<thead>
<tr>
<th>STRUCTURE NAME = MMHL_IMSRLM</th>
<th>TYPE = LOCK</th>
<th>STATUS = ACTIVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>AVG/SEC</td>
<td># REQ</td>
</tr>
<tr>
<td>SYSL</td>
<td>584</td>
<td>584</td>
</tr>
<tr>
<td></td>
<td>0.49</td>
<td>0.0</td>
</tr>
<tr>
<td>CHNGD</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSM</td>
<td>69547</td>
<td>69K</td>
</tr>
<tr>
<td></td>
<td>57.96</td>
<td>103</td>
</tr>
<tr>
<td>CHNGD</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSN</td>
<td>406</td>
<td>394</td>
</tr>
<tr>
<td></td>
<td>0.34</td>
<td>12</td>
</tr>
<tr>
<td>CHNGD</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSO</td>
<td>1014</td>
<td>1014</td>
</tr>
<tr>
<td></td>
<td>0.84</td>
<td>0</td>
</tr>
<tr>
<td>CHNGD</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>71551</td>
<td>71K</td>
</tr>
<tr>
<td></td>
<td>59.63</td>
<td>115</td>
</tr>
<tr>
<td>CHNGD</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- SERV TIME (MIC) - AVG | STD_DEV

Figure 5-21  RMF coupling facility structure activity report showing important fields

5.6 CICS threadsafe support

CICS 4.2 adds support for threadsafe IMS database calls with IMS 12:

- Eliminates TCB switches for IMS database calls.
  - Without threadsafe support, IMS call must be done under an IMS TCB:
    - Requires a switch from CICS QR TCB to IMS TCB and back to CICS QR TCB.
    - If an application is running under an OPEN TCB, requires a switch from OPEN TCB to QR TCB and back from QR TCB to OPEN TCB.
  - With threadsafe support, an IMS call can be done under a CICS OPEN TCB:
    - Requires no TCB switch
    - CICS has multiple OPEN TCBs. Multiple DL/I calls can be done in parallel under CICS OPEN TCBs.

- Enhancement applies to both EXEC DL/I and CALL DL/I.

Benefits:

- CPU use is lower.
- Throughput is increased.
Figure 5-22 illustrates the difference between accessing IMS from a CICS application without the use of threadsafe support and with the use of threadsafe support.

### CICS threadsafe support

#### Without IMS threadsafe support
- **OPEN TCB**
  - EXEC CICS
  - process CICS cmd
  - EXEC SQL
  - process DB2 call
  - EXEC DLI
  - switch
- **QR TCB**
  - switch
- **IMS TCB**
  - process IMS call
  - switch

#### With IMS threadsafe support
- **OPEN TCB**
- **EXEC CICS**
- **process CICS cmd**
- **EXEC SQL**
- **process DB2 call**
- **EXEC DLI**
- **process IMS call**

---

**Figure 5-22  IMS threadsafe support**

### 5.7 Temporary close of VSAM data sets with new extents

IMS 12 issues a temporary close (`CLOSE TYPE=T`) when a new extent is taken on a VSAM database data set. The `CLOSE TYPE=T` updates the catalog but leaves the data set open. Jobs that open the database data set but do not handle verifying, are aware of the new extent.

**Benefits:**
- Tools that read a database while it is open in online systems or batch jobs are aware of the new extents.
- VSAM VERIFY processing can take less time to execute because the VSAM catalog is now more current and aware of any new extents.

### 5.8 Tuning database buffer pools using Buffer Pool Analyzer

Tuning is the process of balancing the distribution of resources to maximize throughput.

We look at I/O, storage, and CPU resources. We also look at how the buffer pool and subpool are organized, how to define buffer pools, and how to tune them.
5.8.1 Function and effect of I/O, storage, and CPU resources

Functions of various resources are described in the following list:

- **I/O**
  - **DASD I/O**
    DASD I/O is usually the target of reduction, because it is slow. Consider that I/O is also a CPU-intensive operation. In CPU-constrained environments, reducing I/O can relieve the CPU resource.
  - **Paging I/O**
    Paging occurs when z/OS determines that there is not enough real storage to meet the needs of all tasks on an LPAR. If increasing IMS buffer storage to reduce DASD I/O causes paging I/O, the net result can be worse performance than would result with increased DASD I/O.

- **Storage**
  - **Central storage (before IBM z/Architecture®)**
    This storage is the traditional storage that is present in the mainframe environment. Increasing the number of buffers in a pool increases Information Management System’s use of central storage space.
  - **Expanded storage (before z/Architecture)**
    Expanded storage can be used for VSAM hiperspace buffers or for MVS paging. Excessive use of expanded storage can increase MVS paging I/O, and worsen IMS and system performance.
  - **64-bit storage (z/Architecture)**
    In the z/Architecture, expanded storage is combined with central storage. Because the combined total of real storage can exceed 2 GB, storage above this line is referred to as 64-bit real storage.

- **CPU**
  The CPU resource can increase if there are not enough buffers, causing excessive DASD I/O, or if buffer pools are so large that IMS has to search excessively on each buffer request.

Be aware that database I/O avoidance should not be the only consideration when tuning IMS database buffer pools. Before making such changes, review RMF data (such as paging rate) and virtual storage availability in your IMS region for the planned additional buffers. Also discuss the situation with your system performance team.

5.8.2 Buffer pool and subpool organization

IMS allows for separating buffer pools by data set, by group of data sets or, for HALDB, by partition. You can then size OSAM and VSAM buffer pools to reduce the number of database I/Os required to complete a transaction or batch workload.

- **OSAM buffer pools**
  OSAM database buffer pools can direct specific database data sets to specific buffer pools. There is a default buffer pool, typically with several subpools containing buffers of various sizes. For example, the default buffer pool can have one subpool containing 1024-byte buffers, another subpool containing 8,192-byte buffers, and a third subpool containing buffers with 32,768 bytes each.
Chapter 5. Database performance topics

5.8.3 Buffer pool definitions

IMS buffer pool specifications are defined in statements that are read from the following sources:

- DFSVSAMP DD for a batch DL/I or DBB job.
- DFSVSMMxx member of IMS PROCLIB for a DB/DC or DBCTL subsystem, where xx is the VSPEC= value in the DFSPBxxx member of PROCLIB.
- Buffer pools can also be dynamically changed in IMS 12 by specifying changes in DFSDFxxx member and issuing the UPDATE POOL command to activate the changes.

The following example shows how an OSAM buffer pool with three subpools could be defined. The example defines a 512-byte subpool with 4 buffers, a 2 KB subpool with 5 buffers, and a 4 KB subpool with 12 buffers:

\[
\text{IOBF} = (512,4,Y,Y) \\
\text{IOBF} = (2048,5,Y,Y) \\
\text{IOBF} = (4096,12,N,N)
\]

VSAM pool and subpool definitions are similar. If multiple VSAM pools are to be defined, a POOLID statement defines the name of a VSAM pool, and VSRBF statements that follow the POOLID define each subpool within that pool. The following example defines a default subpool named DFLT with the same subpool configuration as in the preceding OSAM example:

\[
\text{VSRBF}=512,4 \\
\text{VSRBF}=2048,5 \\
\text{VSRBF}=4096,12
\]

For further information, see *IMS V12 System Definition*, GC19-3021.

5.8.4 Tuning process overview

During the tuning process, you can review resources, analyze I/O information, balance subpools, and validate the results.
The topic describes the process of tuning buffer pools, which involves the following activities:

- Reviewing storage resources and paging activity.
- Gathering BPLTRACE database I/O information.
- Reviewing buffer pool sizes.
- Balancing subpools.
- Reviewing subpool buffer allocations.
- Validating the results.
- Implementing the updated buffer specifications.

**Reviewing storage resources and paging activity**
Before increasing any IMS buffer pool, understand the implications of real storage and paging activity, and also the implications of virtual storage. Review real storage and paging rates to ensure that the addition of buffers will not adversely affect either IMS or the other tasks on that z/OS logical partition (LPAR). Also consider virtual storage when increasing database buffer pools. All database buffers are located in 31-bit virtual storage (above the 16M line) except for hiperspace buffers, which are located in a separate data space. When increasing database buffer pool sizes, make sure that sufficient virtual storage is available in the batch job’s address space or in the IMS DL/I(LSO=S) or CTL(LSO=Y) address space.

**Gathering BPLTRACE database I/O information**
The data-gathering component of buffer pool analyzer uses the generalized trace facility (GTF) to gather data for modeling reports. You can use multiple GTF trace data sets to generate a single report. The BPLTRACE started task is used to initiate the data gathering.

Analysis of trace data is critical to the steps in the tuning process. For an online subsystem, gathering multiple traces at separate times helps provide a better sample and ensures that performance is reasonable at all times. Sample several workload types if appropriate, for example: prime time workload, night time batch workload, and so on. The length of time to perform trace activity depends on the amount of space in the GTF trace data set, and on the number of buffers in the buffer pools. To provide reasonable projections, the number of OSAM and VSAM records that are captured in the traces should be at least five times the total number of buffers in your buffer pools. The more trace data gathered, the more accurate the projections will be.

If the batch job first reads large amounts of data and then does updates, then either one long trace interval or several shorter trace intervals should provide reasonable projections.

**Reviewing buffer pool sizes**
Review buffer subpool sizes to ensure that subpools with appropriate buffer sizes exist. For example, if there is a VSAM data set with a 2K CI size but the VSAM buffer pool assigned to that data set only has 1K, 4K, and 8K buffers, then half the storage that is used by buffers of the 2K data set will be wasted because the data set will use the 4K buffers.

IMS Buffer Pool Analyzer will identify any database data sets that do not have the most efficient buffer size available. If the inefficient buffer size report is not present in the subpool report SPLRPT DD or model report MDLRPT DD, then no databases were identified with inappropriate buffer sizes. If the report is present, you can use the report to identify potential savings from adding new buffer subpools with more efficient buffer sizes.

You can then use IMS Buffer Pool Analyzer’s modeling feature to define new subpools, and rerun the report to see the impact of the change. After reviewing the inefficient buffer size report, if you choose to add new subpools, you can use the DFSVSAMP input to allow IMS Buffer Pool Analyzer to determine how many buffers to assign to the old and new subpools.
The MDLRPT DD reports will show recommendations for how many buffers to assign to each subpool in your buffer pool configuration.

**Balancing subpools**

When reviewing the existing buffer pool configuration, also review the busiest data sets in each subpool. A database subpool can be overwhelmed by a large DBDS. A single BMP could easily scan a data set and replace every other database's buffers. Therefore, if the hit ratio for any of the busy data sets is lower than the hit ratio for other data sets in the subpool, consider moving the data set with the lower hit ratio to its own subpool or a subpool that is shared with other data sets with similar activity.

Among the most obvious examples of balancing subpools is separating VSAM data and index buffers into separate subpools. Presumably, VSAM index components will have a high hit ratio, which means that they should not be in a subpool that is shared with a data set that could replace all the index buffers on a regular basis.

IMS buffer pool analyzer will document any subpools in your configuration that have data sets with varied hit ratios. Review this report to determine whether you could make changes in your subpool configuration that might provide more efficient storage utilization.

You can use IMS buffer pool analyzer’s modeling feature to split a subpool and reassign high or low hit ratio databases into different subpools. If you confirm that there are problems with unbalanced subpools, use the DFSVSAMP input to the buffer analysis job to add new subpools and reassign database data sets to different subpools. The MDLRPT DD will include recommendations for the number of buffers to allocate to all impacted subpools.

**Reviewing subpool buffer allocations**

Because I/O rates and storage are the primary drivers of a general performance tuning strategy, hit ratios and buffer life should not drive buffer pool allocations, but should be used as an ongoing monitor of performance.

To determine the appropriate number of buffers for each subpool, consider the following information. Note, however, there can be application-specific requirements for performance that overwhelm this general tuning strategy.

- The number of buffers for each subpool should be large enough to hold the buffers that are required by any concurrently running applications. This size should help you avoid writing buffers to make room for additional reads. Be careful to not make any buffer pool too small.
- Avoid making buffer subpools too large. Wasting storage causes performance degradation of, not only IMS, but of the entire z/OS LPAR.
- OSAM sequential buffering can significantly improve performance of some applications.
- OSAM buffer caching in a coupling facility can improve performance, but, unless storage in your environment is severely constrained, is not a substitute for a well-tuned buffer pool.

A number of strategies have been used to tune a buffer pool. Buffer pools can be tuned to try to achieve a specific hit ratio in each subpool (for example, 80% for data subpools and 90% for index subpools) or to try to achieve a specific buffer life in each subpool (for example, 30 seconds to 5 minutes in each subpool). These strategies fail to consider that different databases have different access patterns.

While tuning buffer pools to achieve these objectives is one option, another might be to change buffer pool allocations to achieve significantly lower I/O rates, causing the usage of significantly less storage.
The strategy presented here for tuning buffer pool allocations strives to achieve the lowest I/O rate with the least use of storage.

IMS Buffer Pool Analyzer now provides a report that uses the following methodology to produce buffer pool recommendations.

You can review IMS Buffer Pool Analyzer's recommendations to determine whether they make a sensible alternative configuration in your environment.

However, note that IMS Buffer Pool Analyzer does not make recommendations for the addition or removal of subpools. Therefore, be sure you still understand the process that follows, and review whether to add new subpools or consolidate existing subpools.

The marginal reduction statistics section in the database subpool statistics report (see Figure 5-23) provides a way to compare subpool efficiency as the number of buffers is increased. Each line of the report shows a different number of buffers and information about subpool performance using that number of buffers. The marginal reduction number shows how many I/Os are reduced from the prior line's buffer number per kilobyte of storage that was added to the pool. This approach provides a way to identify which subpool would benefit the most from adding more buffers.

<table>
<thead>
<tr>
<th>NUMBER</th>
<th>POOL SIZE (K)</th>
<th>PROJECTIONS</th>
<th>BUFFER LIFE</th>
<th>MARGINAL REDUCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFERS</td>
<td></td>
<td>HITS PER SECOND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>81.3%</td>
<td>188.6</td>
<td>0.02</td>
</tr>
<tr>
<td>8</td>
<td>32</td>
<td>85.3%</td>
<td>148.5</td>
<td>0.05</td>
</tr>
<tr>
<td>16</td>
<td>64</td>
<td>88.4%</td>
<td>117.2</td>
<td>0.13</td>
</tr>
<tr>
<td>32</td>
<td>128</td>
<td>91.2%</td>
<td>86.9</td>
<td>0.35</td>
</tr>
<tr>
<td>64</td>
<td>256</td>
<td>93.1%</td>
<td>69.0</td>
<td>0.92</td>
</tr>
<tr>
<td>128</td>
<td>512</td>
<td>94.9%</td>
<td>51.1</td>
<td>2.50</td>
</tr>
<tr>
<td>192</td>
<td>768</td>
<td>95.9%</td>
<td>40.5</td>
<td>4.73</td>
</tr>
<tr>
<td>256</td>
<td>1,024</td>
<td>96.7%</td>
<td>33.0</td>
<td>7.74</td>
</tr>
<tr>
<td>384</td>
<td>1,536</td>
<td>97.4%</td>
<td>26.1</td>
<td>14.69</td>
</tr>
<tr>
<td>512</td>
<td>2,048</td>
<td>97.6%</td>
<td>24.0</td>
<td>21.24</td>
</tr>
<tr>
<td>768</td>
<td>3,072</td>
<td>97.6%</td>
<td>23.4</td>
<td>32.73</td>
</tr>
<tr>
<td>1,024</td>
<td>4,096</td>
<td>97.6%</td>
<td>23.4</td>
<td>43.64</td>
</tr>
<tr>
<td>1,536</td>
<td>6,144</td>
<td>97.6%</td>
<td>23.4</td>
<td>65.46</td>
</tr>
<tr>
<td>2,048</td>
<td>8,192</td>
<td>97.6%</td>
<td>23.4</td>
<td>87.28</td>
</tr>
</tbody>
</table>

Figure 5-23  Sample report: Marginal reduction statistics

The process of determining how many buffers should be allocated requires that you use the following tasks:

1. Determine, for each subpool, the marginal reduction number that applies to the subpool's current number of buffers. If the current number of buffers in the pool is not listed, estimate the number by using the marginal reduction number from the two closest lines. In the marginal reduction statistics example, if the current number of buffers in the subpool is 64, the marginal reduction number would be 557. If the number of buffers defined is 100, you could estimate the marginal reduction number as 400 (approximately the average of 557 and 252, the marginal reduction numbers for 64 and 128 buffers, respectively).

2. For any subpools with a marginal reduction of 0 (zero), find the lowest number of buffers in that subpool that has a marginal reduction of 0. For example, if the number of buffers currently defined in the subpool is 2048, then 1024 buffers is the least number of buffers with a marginal reduction number of 0. Because these are 4096-byte buffers, note that 4 MB of storage is freed for use in increasing other subpools.
Suggested changes to the number of buffers are as follows:

- If there are fewer buffers remaining in the pool than the number of dependent regions or the PST= parameter in DFSPBxxx, use the number of dependent regions for the number of buffers.

- If the number of buffers in the pool is less than 10, consider merging this subpool into the next larger subpool. If either this pool or the next larger pool is busy (either in I/O rate or request rate), leave the subpool as it is. For subpools with a small number of buffers, consider adding buffers to allow for growth. For example, if an index has a CI split and is the only data set in a buffer pool that is tuned exactly to the number of index CIs, the CI split will cause a very large number of I/Os to occur.

3. Compare the marginal reduction numbers for each subpool. Those with a large marginal reduction number benefit the most from the addition of buffers. Those subpools with low marginal reduction numbers can have the number of buffers reduced with the least impact. Adjust the number of buffers in each subpool to make the marginal reduction for each subpool closer. Depending on whether additional real storage is available, it might be necessary to reduce the size of some subpools to increase the size of other subpools.

Validating the tuning results

To ensure that you have achieved the tuning results that you were seeking, you can validate the results using the process that is described in this section. To validate results of your tuning efforts, complete the following steps:

1. Run IMS Buffer Pool Analyzer reports using the number of buffers you determined with the preceding reviewing subpool buffer allocations process.

2. Compare the total buffer pool size in the original configuration with the total buffer pool size in the new configuration.

   Because this value is the amount of storage that will be added to the virtual storage that is used by the address space, ensure that the job or started task has enough virtual storage to accommodate the addition. Also, ensure that the additional use of real storage will not adversely affect MVS performance and paging.

3. Verify that the total I/O rates for the new configuration are lower than those for the original configuration.

4. Run additional trace sessions to validate the new configuration.

   If you take several traces at different times, run all the traces into a single report, along with running the traces individually. Ensure that the changes to improve the average workload do not adversely affect any particular workload, such as nighttime BMP work or other work.

Implementing the updated buffer specifications

To implement buffer pool specifications:

- For an IMS online subsystem, make changes to the buffer specifications in DFSVSMxx member, and then stop and start the IMS subsystem again.

- If you use IMS 12 you have the option to make changes to the buffer specifications dynamically while the IMS subsystem is executing by changing DFSDFxxx member and then issuing the UPDATE POOL command. Later do not forget to make the same updates to DFSVSMxx, prior to the next IMS WARM or COLD start, otherwise the changes will be lost.

- For an IMS batch subsystem, change buffer specifications prior to the next scheduled run.

Ensure that any changes that you make to affect one IMS or batch job are not unintentionally implemented in other jobs or IMS subsystems.
Transaction manager performance: Application considerations

In this chapter, we closely examine the process of transaction and describe the ways to monitor, analyze, and tune it. We explore following two types of transaction flows:

- Full-function transaction flow
- Fast Path transaction flow

We also discuss IMS application considerations and talk about designing applications in an IMS environment. We cover aspects that are related to performance and the design of applications.

This chapter contains the following information:

- IMS application interface, the language interface, and what you must do to invoke the interface
- Structure of a typical program using the language interface, and application considerations for performance when looking at writing applications that operate in a IMS environment
- IMS and language environment
6.1 Full-function transaction

Figure 6-1 shows the whole flow of a full-function transaction (non-response mode) that we discuss in this chapter.

Figure 6-1  Full-function transaction flow image

6.1.1 Transaction response time

Transaction response time is an important index when you consider performance tuning of IMS. Two response time indexes are user and host:

- **User response time**
  
  IMS log: X’35’(Input MSG ENQ) - X’36’(MSG DEQ)

  This response time is for a user from the viewpoint of IMS. It is the time from when IMS enqueues the message from the LOGICAL TERMINAL (X’35’) terminal to when IMS deletes the message from the message queue pool after sending the output message and receiving the acknowledgment (X’36’). It does not include the network processing time.

- **Host response time**
  
  IMS log: X’35’(Input MSG ENQ) - X’31’(CNT GU)

  This internal processing time includes the time from when IMS enqueues the message (X’35’) to when IMS starts sending output message (X’31’). In most cases, the host response time is regarded as a target of tuning.
The image of these two response times is shown in Figure 6-2.

![Diagram showing response time indexes: User response time and host response time](image)

The following examples are the average response time per transaction, as displayed by some IMS tools:

- IMS Performance Analyzer for z/OS: Transit Time Analysis report (Example 6-1)
  
  The Total column under the Average Transit Time category shows the average host response time. IMS Performance Analyzer for z/OS calculates the average host response time per transaction code.

**Example 6-1  IMS Performance Analyzer for z/OS: Transit Time Analysis**

<table>
<thead>
<tr>
<th>Transact Code</th>
<th>PSB</th>
<th>Min Resp Count</th>
<th>PSB Tran Time</th>
<th>Input Swit Queue</th>
<th>Pgm</th>
<th>-----</th>
<th>Output Queue</th>
<th>Avg Swit Queue</th>
<th>CQS Local</th>
<th>Avg</th>
<th>Total</th>
<th>Input Swit Queue</th>
<th>Pgm</th>
<th>-----</th>
<th>Output Queue</th>
<th>CQS Local</th>
<th>Avg</th>
<th>Total</th>
<th>Total</th>
<th>Total</th>
<th>Max Avg</th>
<th>Max Avg</th>
<th>Max Avg</th>
<th>Max Avg</th>
<th>Max Avg</th>
<th>Max Avg</th>
<th>Max Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSTR1</td>
<td>ZMAPSAP1</td>
<td>349</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>33</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>33</td>
<td>47</td>
<td>76</td>
<td>858</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZMAPSTR2</td>
<td>ZMAPSAP2</td>
<td>223</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>23</td>
<td>24</td>
<td>121</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZMAPSTR3</td>
<td>ZMAPSAP3</td>
<td>76</td>
<td>7</td>
<td>5</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>55</td>
<td>0</td>
<td>68</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>68</td>
<td>0</td>
<td>68</td>
<td>121</td>
<td>695</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZMAPSTR4</td>
<td>ZMAPSAP4</td>
<td>133</td>
<td>11</td>
<td>2</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>22</td>
<td>24</td>
<td>0</td>
<td>67</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>67</td>
<td>0</td>
<td>67</td>
<td>89</td>
<td>597</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZMAPSTR5</td>
<td>ZMAPSAP5</td>
<td>187</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>1</td>
<td>0</td>
<td>40</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>1</td>
<td>40</td>
<td>41</td>
<td>188</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZMAPSTR6</td>
<td>ZMAPSAP6</td>
<td>41</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td>46</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>46</td>
<td>1</td>
<td>46</td>
<td>47</td>
<td>135</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1,009</strong></td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
<td>25</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>47</td>
<td>69</td>
<td>858</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MSG SEND**

**MSG RECEIVE**

**Response time which end user actually experiences**

**Infographic showing response time indexes: User response time and host response time**

**End user response time**

**Host response time**
Example 6-2 Statistical Analysis utility (DFSISTS0): Transaction Response Report

<table>
<thead>
<tr>
<th>TRANSACTION CODE</th>
<th>TOTAL RESPONSES</th>
<th>LONGEST RESPONSE</th>
<th>95% RESPONSE</th>
<th>75% RESPONSE</th>
<th>50% RESPONSE</th>
<th>25% RESPONSE</th>
<th>SHORTEST RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSTR1</td>
<td>349</td>
<td>.85S</td>
<td>.16S</td>
<td>.01S</td>
<td>.01S</td>
<td>.00S</td>
<td>.00S</td>
</tr>
<tr>
<td>ZMAPSTR2</td>
<td>223</td>
<td>.32S</td>
<td>.14S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
</tr>
<tr>
<td>ZMAPSTR3</td>
<td>76</td>
<td>.94S</td>
<td>.08S</td>
<td>.01S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
</tr>
<tr>
<td>ZMAPSTR4</td>
<td>133</td>
<td>.59S</td>
<td>.13S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
</tr>
<tr>
<td>ZMAPSTR5</td>
<td>187</td>
<td>.30S</td>
<td>.16S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
</tr>
<tr>
<td>ZMAPSTR6</td>
<td>41</td>
<td>.26S</td>
<td>.14S</td>
<td>.02S</td>
<td>.02S</td>
<td>.01S</td>
<td>.01S</td>
</tr>
</tbody>
</table>

The average transaction response time can be calculated from the report by the following formula:

\[ \text{Average response time} = \frac{(\text{value of LONGEST RESPONSE} + \text{value of 95\% RESPONSE})}{2} \times 0.05 + \frac{(\text{value of 95\% RESPONSE} + \text{value of 75\% RESPONSE})}{2} \times 0.20 + \frac{(\text{value of 75\% RESPONSE} + \text{value of 50\% RESPONSE})}{2} \times 0.25 + \frac{(\text{value of 50\% RESPONSE} + \text{value of 25\% RESPONSE})}{2} \times 0.25 + \frac{(\text{value of 25\% RESPONSE} + \text{value of SHORTEST RESPONSE})}{2} \times 0.25 \]

In addition, with IMS Performance Analyzer for z/OS you can see response time averages and percentiles through the use of forms-based transit reporting.

6.1.2 The analysis of response time

In this chapter, we show the method to analyze the transaction processing time and discuss tuning. We divide the processing time into five parts (see Figure 6-3) and explain each.

![Figure 6-3 Five parts of transaction flow]
Figure 6-4 shows the IMS log types that are written during transaction processing, and the scope of each reporting utility or tool. There are similarly named columns in multiple utilities, but the scopes differ slightly between them.

6.1.3 Input time analysis

In this section, we analyze the period of time from the receipt of the input message at the VTAM buffer to the time it is written to a message queue. See Figure 6-5.

Processing involves the following steps:
1. Receiving the transaction message and putting it into IOBUF by VTAM
2. Placing the message into a receive-any (RECANY) buffer by VTAM
3. Fetching of MID/DIF

   If the required MID or DIF is not in the format block pool, IMS reads it from the format library.
4. Editing of MFS
   - IMS acquires an MFS workspace in the High I/O Pool (HIOP).
   - IMS builds an input message segment using the field information included in the MID.

5. Acquiring an MSGQ record
   - IMS checks message length and distinguishes which is suitable: SHMSG or LGMSG.
   - If the message queue buffer (QBUF) is too small, a buffer steal is invoked and the I/O
to the message queue data set occurs.

6. Processing ENQ
   A message is moved into the message queue buffer and enqueued on the scheduler
message block (SMB)

7. Writing log records
   - IMS creates a type X'01' log record and puts it into the OLDS buffer to record the data
put in a message queue buffer.
   - IMS creates a type X'35' log record and puts it into the OLDS buffer to record the
enqueueing of the message, and issues a wait write request to write the log into WADS
or OLDS.
   - IMS restores a keyboard of the terminal from where the message came in (except the
case of response-mode).

Monitoring and consideration
Although there is no utility to report this time, you can determine the length with one of the
following methods:

► Use of RECANY buffer
   It can be reported by using IMS Performance Analyzer for z/OS: Internal Resource Usage,
Miscellaneous Statistics report (Example 6-3) from the type X'450B' log record.

Example 6-3  IMS Performance Analyzer for z/OS: Internal Resource Usage report, Miscellaneous Statistics

<table>
<thead>
<tr>
<th>Miscellaneous Statistics</th>
<th>Count /Transact</th>
<th>/Second</th>
<th>Interval :</th>
<th>12.27 (HHHH-MM-SS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest PST used</td>
<td>47</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Highest CCB ID used</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum # of CCBs in use</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Number of Conversations started</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Number of Conversational</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Transactions</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Number of Wait-For-Input</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Transactions</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Number of NonRecoverable</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Transactions</td>
<td>0</td>
<td>.00</td>
<td>0.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Number of Response-Mode</td>
<td>1,009</td>
<td>1.35</td>
<td>100.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Transactions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total number of Transactions</td>
<td>1,009</td>
<td>1.35</td>
<td>100.00% of all Transactions</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

► The invocation of selective dispatching caused by the shortage of dynamic save area
prefix (SAP)
   Dynamic save area sets (maximum 999) and SAPs are the area that is used by
communications ITASKs. This area is allocated for each ITASK during the following times:

Receive ITASK             RECEIVED POST - MSG ENQ
Send ITASK                CNT GU - SEND
When needed, IMS expands the number of SAPs up to 10 times the SAV\(^1\) parameter value. If more SAPs are needed, IMS goes into a selective dispatching mode and the new communication ITASKs will wait until SAPs become available.

This way causes IMS to issue the DFS0769I message at the time of system checkpoint.

The selective dispatching can result in communication delays; the delays can be resolved by increasing the SAPs. The usage of SAPs can be monitored with IMS Performance Analyzer for z/OS: Internal Resource Usage, Dynamic SAP Statistics (Example 6-4) from the type X'450F' log record.

**Example 6-4  IMS Performance Analyzer for z/OS: Internal Resource Usage report, Dynamic SAP Statistics**

<table>
<thead>
<tr>
<th>Dynamic SAP statistics</th>
<th>TCB Name</th>
<th>ALM</th>
<th>Interval :</th>
<th>12.27 (HHHH.MM.SS)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Start 16Aug2012 16:05:01:84</td>
<td>IMS Performance Analyzer for Internal Resource Usage - IMS</td>
</tr>
</tbody>
</table>

- **Total # avail Non-Priv SAPs**: 18, 18, 18
- **Total # avail Privileg SAPs**: 0, 0, 0
- **# SAPs on Stage Queue**: 0, 0, 0
- **Hi # SAPs asgn cur Contraction Interval**: 0, 0, 0
- **Current SAP Generation**: 0, 0, 0
- **Minimum # SAPs**: 18, 18, 18
- **Generation Size**: 36, 36, 36
- **# Expands done**: 0
- **# Contractions done**: 0
- **# moves Stage to Free**: 0
- **# times Upper Limit on SAPs**: 0
- **# times in SD for SAPs**: 0
- **# times Expansion failed**: 0
- **Total Non-Priv SD Waits**: 0, 0, 0
- **High Non-Priv SD Waiters**: 0, 0, 0
- **Total Priv SD Waits**: 0, 0, 0
- **High Priv SD Waiters**: 0, 0, 0
- **# Priv Disps during SD**: 0, 0, 0
- **HI # SAPs asgn cur Checkpoint Interval**: 0, 0, 0
- **HI # Priv asgn cur Checkpoint Interval**: 0, 0, 0
- **Number of Statistics Intervals**: 1

- **I/O to the format library**
  This can be monitored by using IMS Performance Analyzer for z/OS: Internal Resource Usage - Message Format Buffer Pool Statistics or the IMS Monitor, Buffer Pool Statistics - Message Format Buffer Pool.

- **I/O to the message queue data set**
  This can be monitored by using IMS Performance Analyzer for z/OS: Internal Resource Usage - Message Queue Pool Statistics or the IMS Monitor, Buffer Pool Statistics - Message Queue Pool.

---

\(^1\) SAV is the Data Communication EXEC parameter that specifies the allowed maximum number of concurrently active device I/Os.
6.1.4 Input queue time analysis

In this section, we analyze the period of time spent on the input queue or in the message queue buffers waiting for a message region to become available. See Figure 6-6.

![Figure 6-6 Input queue time](image)

In a busy system, this time can be a major part of the response time. The pattern of programs that are scheduled into available regions and the region occupancy percentage should be monitored.

Processing occurs as follows:

1. Acquiring of the scheduling resources:
   a. The PST IPOST is issued or triggered by the enqueueing of transaction SMB, and a message processing program (MPP) region chooses the SMB to be processed.
   b. If some required DMBs do not exist in the DMB pool, IMS acquires the pool space and loads them from the ACBLIB.
   c. If the required PSB does not exist in the PSB pool, IMS must load it from the ACBLIB or copy it from the resident PSB copy.
   d. The PSBW space is acquired.

2. Scheduling completion:
   - IMS creates a type X’08’ log record, which records the scheduling of an application program.
   - IMS creates a type X’31’ log record, which records the SMB GU.
   - IMS moves the first message from QBUF to PSBW.

Monitoring

The input queue time can be monitored from the following reports:

- IMS Performance Analyzer for z/OS: Transit Time Analysis or a customized forms-based Transit Report keyed by transaction code.

The Input Queue column under the Average Transit Time category of this report shows you the average input queue time for each transaction code. See Example 6-5 on page 249.
### Example 6-5  IMS Performance Analyzer for z/OS: Transit Time Analysis

<table>
<thead>
<tr>
<th>Transaction Code</th>
<th>PSB</th>
<th>Min Resp Count</th>
<th>Average Transit Time (msec)</th>
<th>90% Peak Transit Time (msec)</th>
<th>Max</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSTR1 ZMAPSAP1</td>
<td>127</td>
<td>13</td>
<td>0</td>
<td>14</td>
<td>33</td>
<td>0</td>
</tr>
<tr>
<td>ZMAPSTR2 ZMAPSAP2</td>
<td>123</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>ZMAPSTR3 ZMAPSAP3</td>
<td>76</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>55</td>
</tr>
<tr>
<td>ZMAPSTR4 ZMAPSAP4</td>
<td>133</td>
<td>11</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ZMAPSTR5 ZMAPSAP5</td>
<td>187</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>ZMAPSTR6 ZMAPSAP6</td>
<td>41</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>0</td>
</tr>
</tbody>
</table>

**System Totals**

<table>
<thead>
<tr>
<th>PSB</th>
<th>Min Resp Count</th>
<th>Average Transit Time (msec)</th>
<th>90% Peak Transit Time (msec)</th>
<th>Max</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,009</td>
<td>4</td>
<td>1</td>
<td>14</td>
<td>0</td>
<td>15</td>
</tr>
</tbody>
</table>

---

Considerations

Tuning considerations are as follows:

- **The time waiting for scheduling**
- **The processing time of scheduling**

**Consider the time waiting for scheduling**

The primary questions to ask are as follows:

- Are there enough dependent regions to process the arriving transactions?
  - How high is the transaction arriving rate?
  - How long is the average transaction processing time?
  - How many MPPs are available for scheduling?
- Are the parameters that concern transaction scheduling properly tuned?
  - TRANSACT macro parameters:
    - CLASS=
    - PRTY=
    - PROCLIM=
    - PARLIM=
  - EXEC parameter of MPP regions:
    - CLn=
  - How frequently do the scheduling failures occur?
**Consider the processing time of scheduling**

The following tuning aspects can affect this time:

- **I/O to the message queue data set**
  This item can be monitored by using IMS Performance Analyzer for z/OS: Internal Resource Usage -Message Queue Pool Statistics, or the IMS Monitor: Buffer Pool Statistics -Message Queue Pool.

- **I/O to the ACBLIB data set**
  This item can be monitored by using IMS Performance Analyzer for z/OS Total System IWAIT Detail ACBLIB block loading IWAITs and ACBLIB miscellaneous IWAITs. It can also be calculated from the IMS Monitor report: Region IWAIT and Region Summary.

- **The scheduling failure**

- **Multiple loads of same DMB caused by the pool shortage**

When you investigate the problem concerning scheduling, consider the following methods:

- **/DISPLAY PROGRAM command**
  Determine whether the resources have the following status:
  - NOTINIT
  - STOPPED
  - LOCK

- **/DISPLAY TRANSACTION command**
  Determine whether the resources have the following status:
  - STOPPED
  - PSTOP
  - USTOP
  - LOCK

- **/DISPLAY POOL (PSBW/DMBP/PSBP/EPCB) command**
  Determine if there are enough pool spaces. If a PSB pool space shortage happens, a scheduling failure occurs. If a DMB must be loaded into the pool and the necessary free space is unavailable, one or more DMBs not currently being referenced must be deleted to make pool space, and the databases defined by those DMBs are closed. This OPEN/CLOSE process of databases can highly affect performance.

- **/DISPLAY ACTIVE REGION command**
  Determine whether the resources have the following status, because these states indicate the occurrence of scheduling failure:
  - WAIT FOR INTENT
  - POOL SPACE FAILURE

- **/DISPLAY DATABASE command**
  Determine whether the Fast Path database resources have the following status:
  - STOPPED

**Database status:** In case of full-function database, the transaction that is accessing it can be scheduled even though the database status is stopped. Also in case of Fast Path databases, the transaction can be scheduled if the database state is STOPPED caused by a /DBR command (DL/I calls get an FH status.)
Consider the system definitions:

- **APPLCTN macro parameter: SCHDTYP=SERIAL/PARALLEL**

Check the following parameters if the transaction is not scheduled in parallel:

- Is the PARALLEL keyword explicitly defined? The default is SERIAL.
- Is the MAXRGN=1 defined on the TRANSACT macro? This parameter limits the number of MPP regions that can be concurrently scheduled to process a transaction. The default is 0 (zero), which means no limit.
- Are there enough transactions enqueued to fulfill the condition defined by PARLIM= and MAXRGN= on the TRANSACT macro to start scheduling to another MPP?

**Scheduling:** The scheduling to another MPP starts when the current number of the enqueued transaction is greater than the number defined on PARLIM multiplied by the number of dependent regions currently scheduling the transaction:

\[ A > B \times C \]

A  Current number of enqueued transaction  
B  Number defined on PARLIM=  
C  Number of dependent regions currently scheduling the transaction

**DBPCB parameter: PROCOPT=**

PSBs including DBPCB defined as PROCOPT=xE cannot be scheduled concurrently.

You can check the occurrence of scheduling failure from following reports:

**IMS Performance Analyzer for z/OS: Internal Resource Usage report**

If scheduling failures exist, the number is reported for each of these reasons: program conflicts, database intent conflicts, and others. See Example 6-7.

**Example 6-7  IMS Performance Analyzer for z/OS: IRUR, Scheduling Statistics**

<table>
<thead>
<tr>
<th></th>
<th>Count</th>
<th>/Transact</th>
<th>/Second</th>
<th>Interval : 12.27 (HHHH.MM.SS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Conflicts</td>
<td>0</td>
<td>.00</td>
<td>.00</td>
<td>0.00% of all Conflicts</td>
</tr>
<tr>
<td>Database Intent Conflicts</td>
<td>0</td>
<td>.00</td>
<td>.00</td>
<td>0.00% of all Conflicts</td>
</tr>
<tr>
<td>Conflicts for Miscellaneous Reasons</td>
<td>1</td>
<td>.00</td>
<td>.00</td>
<td>100.00% of all Conflicts</td>
</tr>
<tr>
<td>SMBS tried for Scheduling</td>
<td>1,008</td>
<td>1.00</td>
<td>1.35</td>
<td></td>
</tr>
<tr>
<td>Programs Scheduled</td>
<td>1,007</td>
<td>1.00</td>
<td>1.35</td>
<td></td>
</tr>
<tr>
<td>Total Conflicts</td>
<td>1</td>
<td>.00</td>
<td>.00</td>
<td>0.10% of SMBS tried</td>
</tr>
<tr>
<td>Number of active BMPs</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of active MPPs</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^2\) PROCOPT=E forces exclusive use of this database or segment by the online transaction. Other application programs that are scheduling a PSB that are referring to this database/segment wait during their scheduling process. PROCOPT E can be specified with x, assuming the G, I, D, R, and A values.
Taking an IMS Monitor log, you can acquire more detailed information about scheduling failure. IMS Performance Analyzer for z/OS and IMS Monitor report can give you the information.

- IMS Performance Analyzer for z/OS: Performance Exceptions report, Intent Failure Summary (Example 6-8)

  This report can show you the number of failures and name of related PSB and DBD.

**Example 6-8  IMS Performance Analyzer for z/OS: Performance Exceptions report, Intent Failure Summary**

<table>
<thead>
<tr>
<th>Report from 22Jun2012 14.03.21.55</th>
<th>IMS 12.1.0</th>
<th>IMS Performance Analyzer 4.3</th>
<th>Report to 22Jun2012 14.04.57.18</th>
</tr>
</thead>
</table>

**Intent Failure Summary**

| From 22Jun2012 14.03.34.15 To 22Jun2012 14.04.36.34 | Elapsed= 0 Hrs 1 Mins 2.185.734 Secs |

<table>
<thead>
<tr>
<th>PSB Name</th>
<th>DMB Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMP255</td>
<td>DH41SK01</td>
<td>2</td>
</tr>
<tr>
<td>DDLTRN15</td>
<td>DH41ST01</td>
<td>5</td>
</tr>
<tr>
<td>DDLTRN16</td>
<td>DH41ST01</td>
<td>7</td>
</tr>
<tr>
<td>MPP01</td>
<td>DMB52FIN</td>
<td>3</td>
</tr>
</tbody>
</table>

Total for DH41SK01: 2
Total for DH41ST01: 12
Total for DMB52FIN: 3
*** Total ***: 17

- IMS Performance Analyzer for z/OS: Performance Exceptions report, Pool Space Failure Summary (Example 6-9)

  This report can show the Pool ID, the required byte size, and number of times.

**Example 6-9  IMS Performance Analyzer for z/OS: Performance Exceptions report, Pool Space Failure Summary**

|-----------------------------------|-----------|------------------------------|----------------------------------|

**Pool Space Failure Summary**

| From 22Jun2012 14.03.34.15 To 22Jun2012 14.14.36.34 | Elapsed= 0 Hrs 10 Mins 2.185.734 Secs |

<table>
<thead>
<tr>
<th>Pool ID</th>
<th>Bytes req.</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLDP</td>
<td>12561</td>
<td>1</td>
</tr>
<tr>
<td>DLHP</td>
<td>4325</td>
<td>2</td>
</tr>
<tr>
<td>DLMP</td>
<td>95697</td>
<td>1</td>
</tr>
</tbody>
</table>

*** Total ***: 4

- IMS Monitor report: Pool Space Failure Summary (Example 6-10)

  This report can show you the number of failures and name of related PSB and DBD when several intent conflicts occur in the Intent Failure Summary. It can also report the Pool ID, the required byte size, and number of times in the Pool Space Failure Summary.

**Example 6-10  IMS Monitor report: Reports**

|----------------------------------|---------------------------------|-----------|

**Intent Failure Summary**

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>DMBNAME</th>
<th>OCCURRENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRANDB</td>
<td>MFDB02</td>
<td>4</td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

**Pool Space Failure Summary**

<table>
<thead>
<tr>
<th>POOL ID</th>
<th>BYTES REQ.</th>
<th>OCCURRENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPSB</td>
<td>2592</td>
<td>2</td>
</tr>
<tr>
<td>DPSB</td>
<td>2848</td>
<td>4</td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td>6</td>
</tr>
</tbody>
</table>
6.1.5 Processing time analysis

In this section, we analyze the time of application processing after the completion of scheduling. See Figure 6-7.

The following events occur during this time frame:

1. Preparation of PCB
   IMS copies the required PCB to DIRCA (PCB work area in MPP) from PSB pool.

2. Creation of a link to required application program:
   a. IMS checks whether the application is defined to be preloaded, or not.
   b. If the application is not defined to be preloaded, IMS checks the BLDL list and tries to find the TTR of the program.
   c. If the TTR is not on the BLDL, IMS issues the BLDL macro to create an entry.
   d. IMS link to the program.

3. Initialization of the application program
   The user application program initializes the I/O area.

4. Issuing the message prime GU call
   The message segment that stayed in the PSBW is now moved to the I/O area of GU call.

5. Updating the database
   a. The application program issues GHU call.
   b. The CI containing the requested segment is read and put into the DB buffer pool.
   c. IMS passes the acquired segment to the I/O area of the application.
   d. The segment is updated by the REPL call.
   e. IMS reads the application I/O area and returns it into DB buffer pool.
   f. IMS creates a type X’50’ log and put it into OLDS buffer to record the before and after image of the segment.
   This step is valid only if the application program actually updates an IMS database.

6. Issuing the ISRT IOPCB
   a. The output message segment is moved to the PSBW from the I/O area of ISRT call.
   b. The output message segment is moved to the QBUF pool from the PSBW pool.
7. Sync point processing
   a. The program updates the database physically.
   b. IMS creates a type X’03’ log record and puts it into OLDS buffer to record the output message data.
   c. IMS creates a type X’35’ log record and puts it into OLDS buffer to record the enqueueing of the message, and enqueue the output message into QBLKS.
   d. IMS creates a type X’3730’ log record and puts it into OLDS buffer to record the sync point.
   e. IMS releases the database lock.
   f. IMS creates a type X’3701’ log record and puts it into OLDS buffer to record the sending of the message and invoke the sending process.
   g. IMS dequeues the processed message and creates a type X’33’ log record, which records the release of the message record.

8. Releasing the scheduled resources
   a. IMS creates a type X’07’ log record which records the program termination.
   b. IMS releases the PSBW pool space.
   c. IMS changes the status of PSB in the PSB pool to unused.

Monitoring
The processing time can be monitored from the following reports or a customized Forms Based Transit report:

- IMS Performance Analyzer for z/OS: Transit Time Analysis report (Example 6-11)
  The Program Exec column of this report shows the average processing time for each transaction code.

Example 6-11  IMS Performance Analyzer for z/OS: Transit Time Analysis

<table>
<thead>
<tr>
<th>Transact Code</th>
<th>PSB Code</th>
<th>Resp Count</th>
<th>Tran Count</th>
<th>Input Swit</th>
<th>Queue Count</th>
<th>Output Queue Count</th>
<th>Exec Count</th>
<th>Pgm CQS Local</th>
<th>Queue Count</th>
<th>Output Queue Count</th>
<th>Exec Count</th>
<th>Pgm CQS Local</th>
<th>Total Time</th>
<th>Input Swit</th>
<th>Queue Count</th>
<th>Output Queue Count</th>
<th>Exec Count</th>
<th>Pgm CQS Local</th>
<th>Total Time</th>
<th>Input Swit</th>
<th>Queue Count</th>
<th>Output Queue Count</th>
<th>Exec Count</th>
<th>Pgm CQS Local</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSTR1</td>
<td>ZMAPSAP1</td>
<td>349</td>
<td>7</td>
<td>1</td>
<td>0</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>80</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>ZMAPSTR2</td>
<td>ZMAPSAP2</td>
<td>223</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>1</td>
<td>76</td>
<td>0</td>
<td>76</td>
<td>0</td>
<td>858</td>
</tr>
<tr>
<td>ZMAPSTR3</td>
<td>ZMAPSAP3</td>
<td>76</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>68</td>
<td>0</td>
<td>68</td>
<td>0</td>
<td>121</td>
</tr>
<tr>
<td>ZMAPSTR4</td>
<td>ZMAPSAP4</td>
<td>133</td>
<td>11</td>
<td>2</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>67</td>
<td>0</td>
<td>89</td>
<td>0</td>
<td>89</td>
<td>0</td>
<td>597</td>
</tr>
<tr>
<td>ZMAPSTR5</td>
<td>ZMAPSAP5</td>
<td>187</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>0</td>
<td>41</td>
<td>0</td>
<td>41</td>
<td>0</td>
<td>188</td>
</tr>
<tr>
<td>ZMAPSTR6</td>
<td>ZMAPSAP6</td>
<td>41</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>1</td>
<td>0</td>
<td>46</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>47</td>
<td>0</td>
<td>135</td>
</tr>
</tbody>
</table>

**System Totals** | 1,009 | 4 | 1 | 0 | 14 | 0 | 0 | 0 | 15 | 25 | 0 | 47 | 0 | 0 | 0 | 0 | 69 | 858 | 2
Log Transaction Analysis utility, DFSILTA0 (Example 6-12)

The PROC column of this report shows the processing time for each one transaction. Therefore, if you want to know the average time for each transaction, some calculation is needed by using this output.

Example 6-12  Log Transaction Analysis utility (DFSILTA0)

The tuning areas where you can try to tune processing time are as follows:

- Processing time of scheduling to first DL/I call
- Application processing time

Monitoring the processing time of scheduling to first DL/I call

This time is from the completion of scheduling to the issuing of the first GU IOPCB call. Almost all of this time is spent loading a program module.

By taking an IMS Monitor log, you can determine the precise time of scheduling to the first DL/I call. The IMS Performance Analyzer for z/OS, and the IMS Monitor report can give you the information:

- IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis (Example 6-13)

The Scd-DLI column of this report shows you the average scheduling to first DL/I call time of each transaction code.

Example 6-13  IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis

<table>
<thead>
<tr>
<th>PSBname</th>
<th>TranCode</th>
<th>Scheds</th>
<th>Trans /Sched</th>
<th>Scd-DLI Mil.Mic</th>
<th>DLI-Term Schd</th>
<th>Mil.Mic on Q</th>
<th>Calls</th>
<th>IWTs</th>
<th>Scd-DLI Mil.Mic</th>
<th>Elapsed CPUs</th>
<th>Pct of Tran Elap</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSAP1</td>
<td>ZMAPSTR1</td>
<td>294</td>
<td>296 1.01</td>
<td>0.084</td>
<td>9.280</td>
<td>0.0</td>
<td>21.9</td>
<td>12.3</td>
<td>9.217</td>
<td>14.3%</td>
<td>93.5%</td>
</tr>
<tr>
<td>ZMAPSAP5</td>
<td>ZMAPSTR5</td>
<td>172</td>
<td>172 1.00</td>
<td>0.066</td>
<td>14.387</td>
<td>0.0</td>
<td>33.4</td>
<td>17.6</td>
<td>14.387</td>
<td>2.493</td>
<td>69.4%</td>
</tr>
<tr>
<td>ZMAPSAP4</td>
<td>ZMAPSTR4</td>
<td>109</td>
<td>110 1.01</td>
<td>0.074</td>
<td>13.410</td>
<td>0.0</td>
<td>33.5</td>
<td>20.9</td>
<td>13.288</td>
<td>1.743</td>
<td>75.4%</td>
</tr>
<tr>
<td>ZMAPSAP2</td>
<td>ZMAPSTR2</td>
<td>198</td>
<td>198 1.00</td>
<td>0.075</td>
<td>5.808</td>
<td>0.0</td>
<td>23.0</td>
<td>6.0</td>
<td>5.808</td>
<td>1.030</td>
<td>55.8%</td>
</tr>
<tr>
<td>ZMAPSAP6</td>
<td>ZMAPSTR6</td>
<td>64</td>
<td>64 1.00</td>
<td>0.064</td>
<td>8.916</td>
<td>0.0</td>
<td>26.6</td>
<td>11.4</td>
<td>8.916</td>
<td>1.849</td>
<td>64.0%</td>
</tr>
</tbody>
</table>

**Grand Total**

874 877 1.00 0.081 10.461 0.0 27.2 13.5 10.425 1.631 15.6% 92.1% 68.5%

Chapter 6. Transaction manager performance: Application considerations 255
IMS Monitor report: PROGRAM SUMMARY (Example 6-14)

The SCHED TO 1ST CALL/SCHED column of this report shows you the average scheduling to first DL/I call time of each PSB. The time per transaction cannot be shown by the IMS Monitor report.

Example 6-14  IMS Monitor report: PROGRAM SUMMARY

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>SCHEDS.</th>
<th>DEQ.</th>
<th>CALLS /TRAN</th>
<th>I/O IMAITS /CALL</th>
<th>I/O IMAITS /SCH.</th>
<th>CPU TIME DISTR.</th>
<th>ELAPSED SCHED.TO</th>
<th>ELAPSED TIME /TRANS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSBDUMMY</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANDB</td>
<td>3</td>
<td>3</td>
<td>36 12.0</td>
<td>0</td>
<td>1.0</td>
<td>1256 42A,B</td>
<td>2569</td>
<td>7871 44A,B</td>
</tr>
<tr>
<td>DFSIVP2</td>
<td>5</td>
<td>5</td>
<td>43 8.6</td>
<td>6 1</td>
<td>1.0</td>
<td>743 46A,B</td>
<td>37447</td>
<td>2209 48A,B</td>
</tr>
<tr>
<td>DFSIVP1</td>
<td>5</td>
<td>5</td>
<td>45 9.0</td>
<td>14 3</td>
<td>1.0</td>
<td>816 52A,B</td>
<td>68665</td>
<td>2173 54A,B</td>
</tr>
<tr>
<td><strong>TOTALS</strong></td>
<td>14</td>
<td>13</td>
<td>124 9.5</td>
<td>20 1</td>
<td>0.9</td>
<td>828</td>
<td>38450</td>
<td>3252 41408</td>
</tr>
</tbody>
</table>

Monitoring the application processing time

This time is from the first GU IOPCB call that is issued by the application to the completion of program processing after the sync point. IMS Monitor log is needed to know the time spent for this process. The IMS Performance Analyzer for z/OS and IMS Monitor report can give you the information:

IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis or a customized forms-based report keyed by application name or PSB (Example 6-15)

The Elapsed column of this report shows you the average processing time of each PSB and transaction code.

Example 6-15  IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>TranCode</th>
<th>Scheds</th>
<th>Trans /Sched</th>
<th>Sched-DLI Trans</th>
<th>DEQ-DLI Trans</th>
<th>Elapsed IWAIT Elapsed CPUtime</th>
<th>Elapsed Pct of Tran Elapsed</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSAP1 ZMAPSTR1</td>
<td>294</td>
<td>296</td>
<td>1.01</td>
<td>0.064</td>
<td>9.280</td>
<td>0.084</td>
<td>9.217</td>
</tr>
<tr>
<td>ZMAPSAP5 ZMAPSTR5</td>
<td>172</td>
<td>172</td>
<td>1.00</td>
<td>0.086</td>
<td>14.387</td>
<td>0.0</td>
<td>14.387</td>
</tr>
<tr>
<td>ZMAPSAP4 ZMAPSTR4</td>
<td>109</td>
<td>110</td>
<td>1.01</td>
<td>0.074</td>
<td>13.410</td>
<td>0.335</td>
<td>13.288</td>
</tr>
<tr>
<td>ZMAPSAP6 ZMAPSTR6</td>
<td>37</td>
<td>37</td>
<td>1.00</td>
<td>0.075</td>
<td>5.808</td>
<td>0.230</td>
<td>5.808</td>
</tr>
<tr>
<td>ZMAPSAP3 ZMAPSTR3</td>
<td>64</td>
<td>64</td>
<td>1.00</td>
<td>0.084</td>
<td>8.916</td>
<td>0.266</td>
<td>8.916</td>
</tr>
<tr>
<td><strong>Grand</strong> <strong>TOTALS</strong></td>
<td>874</td>
<td>877</td>
<td>1.00</td>
<td>0.081</td>
<td>10.461</td>
<td>0.272</td>
<td>10.425</td>
</tr>
</tbody>
</table>

IMS Monitor report: PROGRAM SUMMARY (Example 6-16)

The ELAPSED TIME/TRANS column of this report shows you the average processing time of each PSB. The time per transaction cannot be shown by the IMS Monitor report.

Example 6-16  IMS Monitor report: PROGRAM SUMMARY
Analysis
You can determine the most effective tuning point by analyzing this application processing time. This processing time can be divided into components (Figure 6-8) by using the IMS Performance Analyzer for z/OS or monitor report capabilities.

Figure 6-8   Components of application processing time

The other parts include the system overhead (sync point, middleware, z/OS CPU wait), application logic, paging, and a variety of other variables.

DL/I call processing time
This time can be calculated by using the IMS Monitor reports or IMS Performance Analyzer for z/OS:

- You can calculate by using the ELAPSED TIME MEAN (PSB TOTAL) column of the CALL SUMMARY report (Example 6-17), which shows an average elapsed time per DL/I call, and TRANS.DEQ column of the PROGRAM SUMMARY report (Example 6-18 on page 258), which shows the number of transactions:
  
  \[
  \text{DL/I call processing time} = \frac{\text{ELAPSED TIME MEAN (PSB TOTAL)} \times \text{CALLS (PSB TOTAL)}}{\text{TRANS.DEQ}}
  \]

Example 6-17   IMS Monitor report: CALL SUMMARY

<table>
<thead>
<tr>
<th>IMS MONITOR</th>
<th>CALL SUMMARY</th>
<th>TRACE START 2008 256, 10:57:47</th>
<th>TRACE STOP 2008 256, 10:59:37</th>
<th>PAGE 0029</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSB NAME</td>
<td>PCB NAME</td>
<td>CALL FUNC</td>
<td>STAT NO.SEGMENT</td>
<td>CALLS</td>
</tr>
<tr>
<td>TRANDB</td>
<td>I/O PCB</td>
<td>ASRT ( )</td>
<td>QC</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GU ( )</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ISRT ( )</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(GU) ( )</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I/O PCB SUBTOTAL</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td>MFD002</td>
<td>I/O PCB</td>
<td>STAT ( )</td>
<td>GE</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT ( )</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GN (02)CLSEG2</td>
<td>Gk</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GN (02)CLSEG1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GU (01)RTSEG</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DL/I PCB SUBTOTAL</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSB TOTAL</td>
<td>36</td>
<td>0</td>
</tr>
</tbody>
</table>
Example 6-18  IMS Monitor report: PROGRAM SUMMARY

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>TRANSCS.</th>
<th>DEQS.</th>
<th>CALLS /TRAN</th>
<th>I/O</th>
<th>IWaits /CALL</th>
<th>CPU TIME /SCHED</th>
<th>ELAPSED TIME /SCHED</th>
<th>1ST CALL DISTR.</th>
<th>1ST CALL TIME /SCHED</th>
<th>ELAPSED TIME /TRAN.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSBDUMMY</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0.0</td>
<td></td>
<td>26</td>
<td>38A,B</td>
<td>36</td>
<td>0</td>
<td>40A,B</td>
</tr>
<tr>
<td>TRANDB</td>
<td>3</td>
<td>3</td>
<td>120</td>
<td>1.0</td>
<td></td>
<td>1256</td>
<td>42A,B</td>
<td>2569</td>
<td>7871</td>
<td>44A,B</td>
</tr>
<tr>
<td>DFSIVP2</td>
<td>5</td>
<td>5</td>
<td>43</td>
<td>8.6</td>
<td>6</td>
<td>743</td>
<td>46A,B</td>
<td>37447</td>
<td>2209</td>
<td>48A,B</td>
</tr>
<tr>
<td>DFSIVP1</td>
<td>5</td>
<td>5</td>
<td>45</td>
<td>9.0</td>
<td>14</td>
<td>816</td>
<td>52A,B</td>
<td>68665</td>
<td>2173</td>
<td>54A,B</td>
</tr>
<tr>
<td>**TOTALS</td>
<td>14</td>
<td>13</td>
<td>124</td>
<td>9.5</td>
<td></td>
<td>828</td>
<td></td>
<td></td>
<td></td>
<td>41408</td>
</tr>
</tbody>
</table>

IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis

You can calculate by using the Elapsed column and Call (Pct of Tran Elap) column of the PROGRAM SUMMARY report (Example 6-19):

DL/I call processing time = Elapsed x Call (Pct of Tran Elap) / 100

Example 6-19  IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis


**Program Analysis**

<table>
<thead>
<tr>
<th>PSBname</th>
<th>TranCode</th>
<th>Scheds</th>
<th>Trans /Sched</th>
<th>Scd-DLI</th>
<th>DLI-Term Schd</th>
<th>Calls</th>
<th>IWaits Sc.Mil.Mic on Q</th>
<th>CPUtime</th>
<th>Pct of Tran Elap</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSAP1</td>
<td>ZMAPSTR1</td>
<td>294</td>
<td>296</td>
<td>1.01</td>
<td>0.084</td>
<td>9.280</td>
<td>.0</td>
<td>21.9</td>
<td>12.3</td>
</tr>
<tr>
<td>ZMAPSAP5</td>
<td>ZMAPSTR5</td>
<td>172</td>
<td>172</td>
<td>1.00</td>
<td>0.086</td>
<td>14.387</td>
<td>.0</td>
<td>33.4</td>
<td>17.6</td>
</tr>
<tr>
<td>ZMAPSAP4</td>
<td>ZMAPSTR4</td>
<td>109</td>
<td>110</td>
<td>1.01</td>
<td>0.074</td>
<td>13.410</td>
<td>.0</td>
<td>33.5</td>
<td>20.9</td>
</tr>
<tr>
<td>ZMAPSAP2</td>
<td>ZMAPSTR2</td>
<td>198</td>
<td>198</td>
<td>1.00</td>
<td>0.075</td>
<td>5.808</td>
<td>.0</td>
<td>23.0</td>
<td>6.0</td>
</tr>
<tr>
<td>ZMAPSAP6</td>
<td>ZMAPSTR6</td>
<td>37</td>
<td>37</td>
<td>1.00</td>
<td>0.090</td>
<td>20.477</td>
<td>.0</td>
<td>24.4</td>
<td>25.5</td>
</tr>
<tr>
<td>ZMAPSAP3</td>
<td>ZMAPSTR3</td>
<td>64</td>
<td>64</td>
<td>1.00</td>
<td>0.084</td>
<td>8.916</td>
<td>.0</td>
<td>26.6</td>
<td>11.4</td>
</tr>
<tr>
<td>*<em>Grand</em></td>
<td>*<em>TOTALS</em></td>
<td>874</td>
<td>877</td>
<td>1.00</td>
<td>0.081</td>
<td>10.461</td>
<td>.0</td>
<td>27.2</td>
<td>13.5</td>
</tr>
</tbody>
</table>

IMS Performance Analyzer for z/OS also provides much more detailed reporting for this type of data through its Program Analysis reports:

- Program Activity Detail
- Program Trace
- Batch VSAM Statistics

**IWAIT elapsed time**

This time can be calculated by using the IMS Monitor reports.

- IMS Monitor report: PROGRAM I/O
- IMS Monitor report: PROGRAM SUMMARY

You can calculate by using the IWAIT TIME TOTAL (PSB TOTAL) column of the PROGRAM I/O report (Example 6-20 on page 259), which shows a total IWAIT time for a PSB, and TRANS.DEQ column of the PROGRAM SUMMARY report (Example 6-21 on page 259), which shows the number of transactions.

IWAIT elapsed time = IWAIT TIME TOTAL (PSB TOTAL) / TRANS.DEQ
### Example 6-20  IMS Monitor report: PROGRAM I/O

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>PCB NAME</th>
<th>IWAITS</th>
<th>TOTAL</th>
<th>MEAN</th>
<th>MAXIMUM</th>
<th>DDN/FUNC</th>
<th>MODULE</th>
<th>NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSIVP2</td>
<td>I/O PCB</td>
<td>3</td>
<td>3059</td>
<td>1019</td>
<td>1037</td>
<td>DFSIVD2</td>
<td>VBH</td>
<td>49</td>
</tr>
<tr>
<td>PCB TOTAL</td>
<td></td>
<td>3</td>
<td>3059</td>
<td>1019</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IVPDB2</td>
<td></td>
<td>3</td>
<td>40025</td>
<td>13341</td>
<td>15149</td>
<td>DFSIVD2</td>
<td>VBH</td>
<td>50</td>
</tr>
<tr>
<td>PCB TOTAL</td>
<td></td>
<td>3</td>
<td>40025</td>
<td>13341</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB TOTAL</td>
<td></td>
<td>6</td>
<td>43084</td>
<td>7180</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Example 6-21  IMS Monitor report: PROGRAM SUMMARY

<table>
<thead>
<tr>
<th>PSBNAME</th>
<th>SCHEDS.</th>
<th>DEQ.</th>
<th>CALLS /TRAN</th>
<th>I/O IWAIT /CALL</th>
<th>CPU TIME /SCH.</th>
<th>ELAPSED TIME /SCHED.</th>
<th>SCHED TO</th>
<th>1ST CALL DISTR.</th>
<th>Elapsed TIME /TRANS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSBDUMMY</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANDB</td>
<td>3</td>
<td>1</td>
<td>36 12.0</td>
<td>0</td>
<td>1.0</td>
<td>1256</td>
<td>42A,B</td>
<td>2569</td>
<td>44A,B 2569</td>
</tr>
<tr>
<td>DFSIVP2</td>
<td>5</td>
<td>5</td>
<td>43 8.6</td>
<td>6</td>
<td>1.0</td>
<td>743</td>
<td>46A,B</td>
<td>37447</td>
<td>48A,B 37447</td>
</tr>
<tr>
<td>DFSIVP1</td>
<td>5</td>
<td>5</td>
<td>45 9.0</td>
<td>14</td>
<td>1.0</td>
<td>816</td>
<td>52A,B</td>
<td>68665</td>
<td>54A,B 68665</td>
</tr>
<tr>
<td>**TOTALS</td>
<td>14</td>
<td>13</td>
<td>124 9.5</td>
<td>20</td>
<td>0</td>
<td>828</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**I/O IWAIT elapsed time**

This time can be calculated by using the IMS Monitor reports or IMS Performance Analyzer for z/OS:

- You can calculate by using the IWAIT TIME MEAN column of the PROGRAM I/O report (Example 6-22), and CALLS/TRAN and I/O IWAIT/CALL columns of the PROGRAM SUMMARY report (Example 6-23).

\[
\text{I/O IWAIT elapsed time} = \text{IWAIT TIME MEAN} \times \text{CALLS/TRAN} \times \text{I/O IWAITS/CALL}
\]
IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis

You can calculate by using the Elapsed column and IWT (Pct of Tran Elap) column of the PROGRAM SUMMARY report (Example 6-24):

\[ I/O \text{ IWAIT elapsed time} = \frac{\text{Elapsed} \times \text{IWT} \text{ (Pct of Tran Elap)}}{100} \]

**Example 6-24**  IMS Performance Analyzer for z/OS: PROGRAM SUMMARY, Program Analysis


**Program Analysis**  (Sorted by Total IWAIT Elapsed time)

<table>
<thead>
<tr>
<th>PSBname</th>
<th>TranCode</th>
<th>Scheds</th>
<th>Trans /Sched</th>
<th>Scd-OLI</th>
<th>DLI-Term Sched</th>
<th>Elapsed</th>
<th>CPUtime</th>
<th>Pct of Tran Elap</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSAP1</td>
<td>ZMAPSTR1</td>
<td>294</td>
<td>296</td>
<td>1.01</td>
<td>0.084</td>
<td>9.280</td>
<td>0.0</td>
<td>21.9</td>
</tr>
<tr>
<td>ZMAPSAP5</td>
<td>ZMAPSTR5</td>
<td>172</td>
<td>172</td>
<td>1.00</td>
<td>0.074</td>
<td>14.387</td>
<td>0.0</td>
<td>33.4</td>
</tr>
<tr>
<td>ZMAPSAP4</td>
<td>ZMAPSTR4</td>
<td>109</td>
<td>110</td>
<td>1.00</td>
<td>0.084</td>
<td>13.410</td>
<td>0.0</td>
<td>33.5</td>
</tr>
<tr>
<td>ZMAPSAP2</td>
<td>ZMAPSTR2</td>
<td>198</td>
<td>198</td>
<td>1.00</td>
<td>0.075</td>
<td>5.808</td>
<td>0.0</td>
<td>23.0</td>
</tr>
<tr>
<td>ZMAPSAP6</td>
<td>ZMAPSTR6</td>
<td>37</td>
<td>37</td>
<td>1.00</td>
<td>0.059</td>
<td>20.477</td>
<td>0.0</td>
<td>44.5</td>
</tr>
<tr>
<td>ZMAPSAP3</td>
<td>ZMAPSTR3</td>
<td>64</td>
<td>64</td>
<td>1.00</td>
<td>0.084</td>
<td>8.916</td>
<td>0.0</td>
<td>26.6</td>
</tr>
</tbody>
</table>

**Grand TOTALS**

<table>
<thead>
<tr>
<th>PSBname</th>
<th>TranCode</th>
<th>Scheds</th>
<th>Trans /Sched</th>
<th>Scd-OLI</th>
<th>DLI-Term Sched</th>
<th>Elapsed</th>
<th>CPUtime</th>
<th>Pct of Tran Elap</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSAP1</td>
<td>ZMAPSTR1</td>
<td>874</td>
<td>877</td>
<td>1.00</td>
<td>0.081</td>
<td>10.461</td>
<td>0.0</td>
<td>27.2</td>
</tr>
</tbody>
</table>

This time can also be monitored using the X’56FA’ log which introduced in IMS V10. The X’56FA’ log is optional which records the transaction level statistics and provides statistics for each commit or message. It is written in the following timing:

- **Commit**
  - MODE=SNGL transactions
  - Non-message driven BMPs and JBPs

- **Input message processed**
  - MODE=MULT transactions

This log includes the information of the X’07’ log record, and the following fields are added:

- VSAM I/O reads and writes
- OSAM I/O reads and writes
- Total ESAF (DB2) calls
- CPU time in TOD clock format
- Elapsed time of database I/Os
- Elapsed wait time for database locks

For a mapping of the all X’56’ subcodes, see the DFSETPCP macro.

You can use IMS Performance Analyzer for z/OS forms report. See Chapter 3, “Performance monitoring tools” on page 43 to find the details of forms report.

Example 6-25 shows the IMS Performance Analyzer for z/OS: Transaction Transit Log report. You can see I/O IWAIT elapsed time in the DB IO Time column and DB lock wait time (approximately the NON I/O IWAIT elapsed time) in the LOCKTIME column.

**Example 6-25**  IMS Performance Analyzer for z/OS: Transaction Transit Log

<table>
<thead>
<tr>
<th>Org</th>
<th>LTERM</th>
<th>TranCode Program</th>
<th>DB Call</th>
<th>Total IO</th>
<th>DB IO</th>
<th>DB Wait</th>
<th>DC Call</th>
<th>CPU Comp</th>
<th>DB Update</th>
<th>ESAFcall</th>
<th>DB Lock</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS009</td>
<td>IYTN0</td>
<td>DFSIVP1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IMS009</td>
<td>IYTN0</td>
<td>DFSIVP1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IMS009</td>
<td>IYTN0</td>
<td>DFSIVP1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

You can use IMS Performance Analyzer for z/OS forms report. See Chapter 3, “Performance monitoring tools” on page 43 to find the details of forms report.

Example 6-25 shows the IMS Performance Analyzer for z/OS: Transaction Transit Log report. You can see I/O IWAIT elapsed time in the DB IO Time column and DB lock wait time (approximately the NON I/O IWAIT elapsed time) in the LOCKTIME column.
Chapter 6. Transaction manager performance: Application considerations

Considerations

There are several tuning considerations:

- Processing time of scheduling to the first DL/I call
- Application processing time

Consider the processing time of scheduling to first DL/I call

Consider reducing program loading time and initialization time:

- Reduce the program loading time by doing the following tasks:
  - Preloading the module.
  - Using the VLF function of z/OS.
  - Reducing the I/O to the PGMLIB directory:
    - Defining the MPP EXEC parameter DBLDL as enough.
    - Using the LLA function of z/OS.
  - Reducing the I/O to the libraries by doing the following tasks:
    - Reconsidering the concatenation sequence of STEPLIB on MPP execution JCL.
    - Checking the usage of DASD containing the PGMLIB.
- Reduce the initialization time in the application program.

Consider the application processing time

Consider reducing the following items:

- Reduce the I/O IWAIT elapsed time by doing the following tasks:
  - Reducing each I/O IWAIT elapsed time:
    - Check the accessing rate to the DASD that contains the databases and change the allocation if needed.
    - If the I/O rate to the message queue data set is high, consider increasing the size of message queue buffer (QBUF).
  - Reducing the number of I/O IWAIT per CALL:
    - Reorganize the database to improve the space utilization.
    - Check the DBD and change the CI size or other parameters if needed.
    - Check the logic of the randomizer, if the database is HDAM or PHDAM.
    - Increase the database buffers, because the buffer shortage results in the re-reading of the CIs or blocks, which were already read in the previous calls.
- Reduce the NON I/O IWAIT elapsed time.
  - Reducing the lock wait time
    - One way you might shorten the length of your lock wait time is by shortening the processing time of transaction applications or increasing the frequency of CHKPK/SYNC calls in batch applications.
    - Reduce the occurrence of lock conflicts by allocating the transactions, which access the same databases, to different classes so they are not scheduled simultaneously, which can increase the execution time of BMPs.
    - Divide the IMS resources (such as databases) so you can schedule the applications in parallel.
    - Reduce the lock conflicts by changing the PROCOPT parameter to Gx if the application does not update the database.
Reduce the CPU time per transaction by doing the following tasks:
- Reducing the I/O IWAIT/CALL numbers.
- Reducing the I/O of program loading by using the preload function or static link-editing of the subroutines if possible.

Reduce the number of DL/I calls per transaction by doing the following tasks:
- Issuing the PATH call to get multiple segments per a call.
- Reducing the number of message segments using MFS screen, which changes the multi-segment message into single segment message.
- Changing the non-qualified GN calls into qualified GN to limit the scope of search.

In addition, system overhead can also degrade the application processing time, as in the following examples:
- CPU wait
- Frequent paging
- Conflict of DASD or channel path

6.1.6 Output queue time analysis

In this section, we analyze the time from the program termination to the completion of the preparation of the sending task. See Figure 6-9.

The processing times involve preparing the send task:
- IMS creates the ITASK to send an output message.
- A buffer space to place the output message (OUTBUF) is acquired in HIOP.
- IMS writes a type X’31’ log to record the CNT GU.

6.1.7 Monitoring

The output queue time can be monitored from the following reports:
- IMS Performance Analyzer for z/OS: Transit Time Analysis report (Example 6-26 on page 263)
  This report shows the average output queue time for each transaction code, including the time from the program termination (X’07’) to the CNT GU (X’31’).
Example 6-26   IMS Performance Analyzer for z/OS: Transit Time Analysis

IMS Performance Analyzer
Transit Time Analysis By Transaction Code


Min ****** Average Transit Time (msec) ******  ****** 90% Peak Transit Time (msec) *****   Max   Avg
Transact  PSB          Resp  Tran  Input Swit   Pgm  ----- Output ---- Total  Input Swit   Pgm  ----- Output ---- Total  Tran   CPU
Code                 Count  Time  Queue Queue  Exec Queue   CQS Local        Queue Queue  Exec Queue   CQS Local        Time  Time
ZMAPSTR1 ZMAPSAP1       349     7      1     0    13     0     0     0    14     33     0    47     0     0     0    76   858     1
ZMAPSTR2 ZMAPSAP2       223     4      0     0     8     0     0     0     8      0     0    23     0     0     0   24   121     1
ZMAPSTR3 ZMAPSAP3        76     7      5     0    15     0     0     0    20     55     0    68     0     0     0   121   695     2
ZMAPSTR4 ZMAPSAP4       133    11      2     0    20     0     0     0    22     24     0    67     0     0     0    89   188     2
ZMAPSTR5 ZMAPSAP5       187    10      0     0    17     0     0     0    17      1     0    40     0     0     0   41   188     3
ZMAPSTR6 ZMAPSAP6        41    12      0     0    23     0     0     0    24      1     0    46     0     0     0   47   135     3

**System Totals**     1,009     4      1     0    14     0     0     0    15     25     0    47     0     0     0   69   858     2

▶ Log Transaction Analysis utility, DFSILTA0 (Example 6-27)

The OUT Q column of this report shows the output queue time per each one transaction.
Therefore, if you want to know the average time for each transaction code, some
calculation is needed by using this output. The report shows the time from the enqueueing
of the output message (X’35’) to the CNT GU (X’31’), so it includes some time needed to
process the sync point.

Example 6-27   Log Transaction Analysis utility (DFSILTA0)

Considerations

There are several tuning considerations:

▶ The invocation of selective dispatching caused by the shortage of dynamic SAP

If there is not enough dynamic SAP, IMS cannot create a new send ITASK and the send
process has to wait.

▶ Terminal disconnection

Terminal disconnection results in the failure of new send ITASK creation.

▶ Prevention of ITASK creation by the other IMS processes

If the other IMS processes are too busy, ITASK creation can delay.

▶ Delay of dispatching by z/OS

IMS control region has to wait for CPU and this leads to the delay of preparation of
sending.
6.1.8 Output time analysis

In this section, we analyze the time from the start of sending to the dequeueing of the message from the message queue. See Figure 6-10.

The processing times are as follows:

1. Log write ahead
   - IMS issues a wait write request to write the sync point log.
   - IMS writes logs to DASD.
2. MOD/DOF fetching
   - If the required MOD or DOF is not in the format block pool, IMS reads it from the format library.
3. MFS editing
   - An MFS workspace is acquired in the high I/O pool (HIOP).
   - An output message segment is created by using the field information included in MID.
4. VTAM processing
   - IMS issues the VTAM SEND request and passes the output data to VTAM.
   - VTAM sends a message to the terminal.
   - VTAM receives an acknowledgement (+DR) from the terminal.
5. Dequeueing of the output message
   - IMS writes a type X’36’ log to record dequeueing of the message.
   - IMS releases the QBLKS record and writes type X’33’ log to record the releasing of the records.
   - The output message record is released.

Monitoring
The output message processing time can be calculated by using following reports:

- Statistical Analysis utility (DFSISTS0): TRANSACTION RESPONSE REPORT
  (Example 6-28 on page 265)

You can calculate the time by subtracting the number of lower line (host response time) from the number of upper line (user response time).
Example 6-28  Statistical Analysis utility (DFSISTS0): TRANSACTION RESPONSE REPORT

<table>
<thead>
<tr>
<th>TRANSACTION CODE</th>
<th>TOTAL RESPONSES</th>
<th>LONGEST RESPONSE</th>
<th>95% RESPONSE</th>
<th>75% RESPONSE</th>
<th>50% RESPONSE</th>
<th>25% RESPONSE</th>
<th>SHORTEST RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZMAPSTR1</td>
<td>349</td>
<td>.85S</td>
<td>.16S</td>
<td>.01S</td>
<td>.01S</td>
<td>.00S</td>
<td>.00S</td>
</tr>
<tr>
<td>ZMAPSTR2</td>
<td>223</td>
<td>.32S</td>
<td>.14S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
</tr>
<tr>
<td>ZMAPSTR3</td>
<td>76</td>
<td>.05S</td>
<td>.10S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
<td>.00S</td>
</tr>
<tr>
<td>ZMAPSTR4</td>
<td>133</td>
<td>.39S</td>
<td>.13S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
</tr>
<tr>
<td>ZMAPSTR5</td>
<td>187</td>
<td>.30S</td>
<td>.16S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
<td>.01S</td>
</tr>
<tr>
<td>ZMAPSTR6</td>
<td>41</td>
<td>.26S</td>
<td>.14S</td>
<td>.02S</td>
<td>.02S</td>
<td>.01S</td>
<td>.01S</td>
</tr>
<tr>
<td>TOTAL RESPONSES</td>
<td>1009</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example 6-29  Log Transaction Analysis utility (DFSILTA0)

You can calculate the approximate time by subtracting the number of IN Q and PROC and OUT Q from the number of TOTAL column of this report.

Consideration

Additional tuning considerations are as follows:

- I/O to the format library
  
  This I/O can be monitored by using IMS Performance Analyzer for z/OS: Internal Resource Usage -Message Format Buffer Pool Statistics, or the IMS Monitor Buffer Pool Statistics: Message Format Buffer Pool.

- I/O to the message queue data set
  
  This can be monitored by using IMS Performance Analyzer for z/OS: Internal Resource Usage -Message Queue Pool Statistics or the IMS Monitor Buffer Pool Statistics: Message Queue Pool.

- I/O to the log data sets
  
  Check the I/O rate of the DASDs, which contains OLDS or WADS.

- Logging rate
  
  The log write requests that are issued during the output message processing time is wait write, so the log writing to disk can delay when the system logging rate is low.
6.2 Fast Path transaction

Figure 6-11 shows the flow of Fast Path transaction, we discuss in this chapter.

6.2.1 Transaction response time

The transaction response time of Fast Path transaction can be divided into processes similar to those of the full-function transaction. Input queue time can be calculated by using the time stamp of type X'5901' log, and program execution time can be calculated by X'5903' log. Output queue time and output time can be calculated by the X'5936' log. See Figure 6-12.
Transit time means the time spent before the scheduling of the FP router to send the output message, and so, it does not include the output time. This time is usually considered as a target of the tuning.

The following examples show the average response time for each transaction as displayed by several IMS tools:

- **IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis (Example 6-30)**

  The Total column under the Average Transit Time category shows the average transit time. IMS Performance Analyzer for z/OS calculates the average host response time per transaction code. The example shows the report ordered by the transaction code, and it can also be ordered by the routing code, user ID, and time of sync point.

  **Example 6-30 IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis**

  ```
  From 17Aug2012 14:21:03:04 To 17Aug2012 14:23:18:53 Elapsed= 0 Hrs 2 Mins 15.482.032 Secs
  Min --Average Transit Time-- --90% Peak Transit Time- Max --- Shared EMHQ Time ---
  Transact Code Routing Code Resp Count Tran Time Input Queue Exec Queue Total Input Queue Exec Queue Total Tran Time Global Count Average 90% Peak
  Code Code Time Pgm Outpt Input Pgm Outpt Total Time Count Input Outpt Input Outpt
  __________ ________ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ _______ i
  DFSIVP4  *IFP  1 0 0 0 0 0 0 0 0 0 0 0 0 0
  DFSIVPS  *IFP  1 0 0 0 0 0 0 0 0 0 0 0 0 0
  IVTFD    IVTFD  10 1 0 25 21 46 0 89 100 177 319 0
  IVTFM    IVTFM  13 0 0 0 1 0 1 1 1 1 1 0
  **System Totals**  25 0 0 10 9 19 0 52 59 104 319 0
  ```

- **Example 6-31 Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF TRANSIT TIMES (Example 6-31)**

  The TOTAL-AVG column of this report also shows the average transit time.

  **Example 6-31 Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF TRANSIT TIMES**

  ```
  OVERALL SUMMARY OF TRANSIT TIMES IN MILLI-SECONDS
  TRANS CODE    -NO.OF-    ---TOTAL---    --INPUT Q --    --PROCESS --    --OUTPUT Q--    LENG  LENG
  CODE        -TRANS-    AVG    MAX    AVG    MAX    AVG    MAX    AVG    MAX
  ___________    _______    _____    _____    _____    _____    _____    _____    _____    _____
  IVTFM        13  0  1  0  1  0  1  0  1  59  59  90  90
  IVTFD        10  44  312  0  0  24  120  20  192  59  59  90  90
  DFSIVPS      1  0  0  0  0  0  0  0  0  0  0  0  0
  DFSIVP4      1  0  0  0  0  0  0  0  0  0  0  0  0
  ```

- **IMS Performance Analyzer for z/OS: Fast Path (EMH) Transaction Transit Log (Example 6-32 on page 268)**

  The Total Time column of this report shows you the total time for each one transaction. Therefore, if you want to know the average time for each transaction code, some calculation is needed by using this output.
6.2.2 The analysis of response time

In this chapter, we show the method to analyze the transaction processing time and discuss tuning. We divide the processing time into the same five parts as we do for full-function transactions (see 6.1.2, “The analysis of response time” on page 244 for full-function transaction).

6.2.3 Input time analysis

In this part, we analyze the time from the receiving of the input message at the VTAM buffer to the time it is moved into EMHB. The first through fourth processes are same as those of the full-function transaction; the fifth process differs. See Figure 6-13.

The input processing time is required for the following IMS actions:
1. Receiving the transaction message and putting it into IOBUF by VTAM
2. Placing the message into a receive-any (RECANY) buffer by VTAM
3. Fetching of MID/DIF

If the required MID or DIF is not in the format block pool, IMS reads it from the format library.
4. Editing of MFS
   - IMS acquires an MFS workspace in the high I/O pool (HIOP).
   - IMS builds an input message segment by using field information included in the MID.

5. Moving the data into EMHB
   - IMS determines that the input transaction is Fast Path, and calls the Fast Path Input Edit/Routing exit routine (DBFHAGU0).
   - IMS moves the input message into EMHB.
   - If all of the IFP regions to be scheduled are occupied, IMS enqueues the EMHB into BALG.

**Monitoring and consideration**
Although there is no utility to report the length of this time, you can determine the length of time by using one of the following methods:

- Use of RECANY buffer
  Buffer can be reported using IMS Performance Analyzer for z/OS Internal Resource Usage: Miscellaneous Statistics report or the type X’450B’ log.

- The invocation of selective dispatching caused by the shortage of dynamic SAP
  The usage of SAPs can be monitored by using IMS Performance Analyzer for z/OS Internal Resource Usage: Dynamic SAP Statistics or the type X’450F’ log.

- I/O to the format library

### 6.2.4 Input queue time analysis

In this section, we analyze the period from the scheduling of the Fast Path Input Edit/Routing exit routine (DBFHAGU0) to the issuing of the GU call by the application program. See Figure 6-14.

![Input queue time](image)

The processing times for GU IOPCB is because of the following moves:

- IMS moves the EMHB to ESRT.
- IMS moves the input message from ESRT to I/O area of the user application.
Monitoring
The input queue time can be monitored from the following reports:

- IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis (Example 6-33)

  The Input Queue column under the Average Transit Time category shows the average input queue time for each transaction code.

Example 6-33  IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis

<table>
<thead>
<tr>
<th>Transact Code</th>
<th>Routing Code</th>
<th>Resp Count</th>
<th>Tran Time</th>
<th>Input Queue</th>
<th>Pgm Outpt</th>
<th>Total</th>
<th>Input Queue</th>
<th>Pgm Outpt</th>
<th>Total</th>
<th>Tran Time</th>
<th>Global Count</th>
<th>Average</th>
<th>90% Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSIVP4</td>
<td>*IFP</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVP5</td>
<td>*IFP</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IVTFD</td>
<td>IVTFD</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>25</td>
<td>21</td>
<td>46</td>
<td>0</td>
<td>89</td>
<td>100</td>
<td>177</td>
<td>319</td>
<td>0</td>
</tr>
<tr>
<td>IVTFM</td>
<td>IVTFM</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>System Totals</strong></td>
<td></td>
<td>25</td>
<td>0</td>
<td>10</td>
<td>9</td>
<td>19</td>
<td>0</td>
<td>52</td>
<td>59</td>
<td>104</td>
<td>319</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Considerations
A Fast Path transaction is processed in the same way as a wait for input (WFI) transaction, so the processing time of scheduling is not included in the input queue time. In the case of Fast Path transaction, input queue time equals the time waiting for IFP region scheduling. Consider the following questions:

- Are there enough dependent regions to process the arriving transactions?
- How high is the transaction arriving rate?
- How long is the average transaction processing time?
- How many IFPs are available for scheduling?

You can acquire the information about the usage of IFP region using the IMS Performance Analyzer for z/OS log report. If the utilization of a region is high, input queue time can be long.

Example 6-35 on page 271 shows the IMS Performance Analyzer for z/OS: Fast Path (IFP) Region Occupancy report. The Processing column of this report shows the sum of application processing time and phase 1 and phase 2 sync point processing time. The Idle column shows...
the time during which the region is available but waiting to process a transaction. Therefore, the regions with a high processing time compared to idle time have high utilization.

**Example 6-35  IMS Performance Analyzer for z/OS: Fast Path (IFP) Region Occupancy**

<table>
<thead>
<tr>
<th>Region</th>
<th>IMID</th>
<th>Program Name</th>
<th>Tran Count</th>
<th>&lt;- Processing -&gt; HH.MM.SS</th>
<th>&lt;---- Idle ----&gt; HH.MM.SS</th>
<th>&lt; Not Active -&gt; HH/MM.SS %</th>
<th>Elapsed Time 0 Hrs 2 Mins 42.451.266 Secs</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMSP   1</td>
<td>DFSIVP4</td>
<td>11</td>
<td>0</td>
<td>0.2</td>
<td>1.25</td>
<td>52.1</td>
<td>1.18 47.8</td>
</tr>
<tr>
<td>IMSP   2</td>
<td>DFSIVP5</td>
<td>14</td>
<td>0</td>
<td>0.0</td>
<td>2.11</td>
<td>80.8</td>
<td>31 19.1</td>
</tr>
</tbody>
</table>

### 6.2.5 Processing time analysis

In this section, we analyze the time period from the GU call of the application program to the sync point. See Figure 6-15.

![Processing diagram](image.png)

The following events occur during this time frame:

1. **Updating the database** (in case the program updates the database)
   - Application program issues the GHU call.
   - IMS passes the required segment to the I/O area of the program.
   - Program issues the REPL call and updates the segment.
   - IMS writes the content of I/O area into Fast Path DB buffer pool.
   - Program issues the ISRT IOPCB call.

2. **Sync point processing**
   - Program process returns to the GU IOPCB call to read the next input message.
   - The sync point process starts triggered by the next GU IOPCB.
   - Logical logger moves the logs related to the transaction (X’5901’, X’5903’, X’5950’, X’5937’) into OLDS buffer.
   - If the next input message arrives, program continues processing. If no message is to be scheduled, it waits for next input.
Monitoring

The program execution time can be monitored from the following reports:
- IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis (Example 6-36)

The Pgm Exec column, under the Average Transit Time category, shows the average program execution time for each transaction code.

**Example 6-36**  IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis

<table>
<thead>
<tr>
<th>Transact Code</th>
<th>Routing Code</th>
<th>Resp Count</th>
<th>Average Transit Time</th>
<th>-90% Peak Transit Time</th>
<th>Global Average</th>
<th>90% Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSIVP4</td>
<td>*IFP</td>
<td>1</td>
<td>0 0 0 0 0 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVP5</td>
<td>*IFP</td>
<td>1</td>
<td>0 0 0 0 0 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IVTFD</td>
<td>IVTFD</td>
<td>10</td>
<td>1 0 25 21 46 0</td>
<td>89 100 177 319</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IVTFM</td>
<td>IVTFM</td>
<td>13</td>
<td>0 0 0 1 1 0</td>
<td>1 1 1 1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**System Totals**

25 0 10 9 19 0 52 59 104 319 0

- Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF TRANSIT TIMES (Example 6-37)

The PROCESS-AVG column of this report also shows the average program execution time for each transaction code.

**Example 6-37**  Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF TRANSIT TIMES

<table>
<thead>
<tr>
<th>Transact Code</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IVTFM</td>
<td>13</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>59</td>
<td>59</td>
</tr>
<tr>
<td>IVTFD</td>
<td>10</td>
<td>44</td>
<td>312</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>120</td>
<td>20</td>
<td>192</td>
</tr>
<tr>
<td>DFSIVP5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVP4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**System Totals**

25 0 1 1 1 3 0 1 0 0 0 1 1 0 1 0 0 0 0 0 2 0

Considerations

The program execution time is mainly occupied by the time to access the databases. Therefore, you can pick up the transactions to be analyzed by using the following reports.

- IMS Performance Analyzer for z/OS: Fast Path Resource Usage and Contention (Example 6-38)

Be sure to check the DEDB Calls, ADS I/O Reads, Common Buffer Usage, and Contentions columns.

**Example 6-38**  IMS Performance Analyzer for z/OS: Fast Path Resource Usage and Contention

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSIVP4</td>
<td>*SF=L</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>DFSIVP5</td>
<td>*SF=L</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>IVTFD</td>
<td>IVTFD</td>
<td>10</td>
<td>1 1 0 1</td>
<td>1 3 0 1</td>
<td>0 0 0 0</td>
<td>1 1 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IVTFM</td>
<td>IVTFM</td>
<td>13</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**System Totals**

25 0 1 1 1 3 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 2 0
Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF RESOURCE USAGE AND CONTENTIONS (Example 6-39)

Be sure to check the DEDB CALLS TOTAL, ADS I/O RDS, COMMON BUFFER USAGE, and CONTENTIONS columns.

Example 6-39  Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF RESOURCE USAGE AND CONTENTIONS

<table>
<thead>
<tr>
<th>TRANCODE</th>
<th>--NO.--</th>
<th>DEDB CALLS</th>
<th>---MSDB--</th>
<th>--ADS I/O--</th>
<th>--VSO ACT--</th>
<th>--COMMON BUFFER--</th>
<th>TOTAL CONTENTIONS</th>
<th>TRAN LGNR</th>
<th>STA</th>
<th>TS</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVTFM</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>IVTFD</td>
<td>10</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVP5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVP4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The following tips describe how to interpret information from the reports:

- **Transactions that access the ADS more than the number of DEDB calls:**
  - Use the calculation with the IMS Performance Analyzer for z/OS report:
    \[\text{DEDB Calls Reads Avg} + \text{DEDB Calls Updates Avg} < \text{ADS I/O Reads Avg}\]
  - Use the calculation with the DBFULTA0 report:
    \[\text{DEDB CALLS TOTAL AVG} < \text{ADS I/O RDS AVG}\]

  In this situation the segment is contained in the overflow area (dependent or independent) or the database has long synonym chains. The reorganization of the database accessed by the transaction can improve the processing time.

- **Transactions that include the process of CI contention:**
  - Contentions CI/Sec > 0

  This situation indicates that the segment is contained in the overflow area (dependent or independent) or indicates the increasing of the synonyms. This situation might also be caused by multiple programs that access the same root segment and can run simultaneously. The reorganization of the database accessed by the transaction can improve the processing time.

- **Transactions that include the process of buffer steal:**
  - Common Buffer Usage Stl > 0

  This statement indicates the invocation of buffer steal caused by the shortage of NBA buffer. We suggest that you increase the number of NBAs.

- **Transactions that include the process of buffer wait:**
  - Common Buffer Usage Wts > 0

  This statement indicates the invocation of buffer steal caused by the shortage of Fast Path buffers. We suggest that you increase the allocation of DBFX.

- **Transactions that includes the process of buffer wait:**
  - Contentions Tot OBA > 0

  This statement indicates the invocation of OBA wait caused by the shortage of NBA. We suggest that you increase the allocation of NBA.
6.2.6 Output queue time analysis

In this section, we analyze the period from sync point to entry to the output router. See Figure 6-16.

![Figure 6-16 Output queue time](image)

The processing times are as follows:

1. Writing log records
   
   Physical logger writes logs X'5901', X'5903', X'5950', X'5937' into OLDS/WADS, which were placed in the OLDS buffer.

2. Scheduling the output processing
   
   IMS creates the output thread to update database, and schedules the FP DC router to send the output message.

**Monitoring**

The output queue time can be monitored from following reports:

- IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis (Example 6-40)

  The Output Queue column under the Average Transit Time category shows you the average output queue time per each transaction code.

**Example 6-40 IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis**

<table>
<thead>
<tr>
<th>Transact Code</th>
<th>Routing Code</th>
<th>Resp Count</th>
<th>Tran Time</th>
<th>Input Queue Time</th>
<th>Pgm Outpt Time</th>
<th>Avg Qnt</th>
<th>90% Peak Qnt</th>
<th>Max Trans Time</th>
<th>Global Avg Time</th>
<th>Shared EMHQ Time</th>
<th>Avg Output Time</th>
<th>90% Peak Output Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSIVP4</td>
<td>*IFP</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVPS</td>
<td>*IFP</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IVTFD</td>
<td>IVTFD</td>
<td>10</td>
<td>1</td>
<td>25</td>
<td>21</td>
<td>46</td>
<td>0</td>
<td>89</td>
<td>100</td>
<td>177</td>
<td>319</td>
<td>0</td>
</tr>
<tr>
<td>IVTFM</td>
<td>IVTFM</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td><strong>System Totals</strong></td>
<td></td>
<td>25</td>
<td>0</td>
<td>10</td>
<td>9</td>
<td>19</td>
<td>0</td>
<td>52</td>
<td>59</td>
<td>104</td>
<td>319</td>
<td>0</td>
</tr>
</tbody>
</table>

Example 6-40 IMS Performance Analyzer for z/OS: Fast Path (EMH) Transit Time Analysis
Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF TRANSIT TIMES (Example 6-41)

The OUTPUT Q-AVG column of this report also shows the average output queue time per each transaction code.

### Example 6-41  Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF TRANSIT TIMES

<table>
<thead>
<tr>
<th>TRANSACTION CODE</th>
<th>NO. OF TRANS.</th>
<th>TOTAL AVG</th>
<th>MAX</th>
<th>INPUT AVG</th>
<th>MAX</th>
<th>PROCESS AVG</th>
<th>MAX</th>
<th>OUTPUT AVG</th>
<th>MAX</th>
<th>LENG (CH)</th>
<th>LENG (CH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVTFM</td>
<td>13</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>59</td>
<td>90</td>
</tr>
<tr>
<td>IVTFD</td>
<td>10</td>
<td>44</td>
<td>312</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>120</td>
<td>20</td>
<td>192</td>
<td>59</td>
<td>90</td>
</tr>
<tr>
<td>DFSIVP5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DFSIVP4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Considerations

Output queue time is mainly the time needed to write the logs to disk, so the time depends on the disk where the WADS and OLDS are located.

When the logging rate of the system is low, the output queue time can lengthen because the log write requests from the Fast Path application are *wait write* type and often relies on the logging time. Running another application in parallel, which writes logs to disk or issues the *check write* type requests constantly, can improve the performance.

### 6.2.7 Output time analysis

In this section, we analyze the period from output router entry to dequeue time. See Figure 6-17.

#### Figure 6-17  Output time

- **MSG SEND**
  - **Input Time**
  - **Input Queue Time**
  - **Processing Time**
  - **Output Queue Time**
  - **Output Time**

The processing times are as follows:

1. **MOD/DOF fetch**
   - If the required MOD or DOF is not in the format block pool, IMS reads it from the format library.

2. **MFS editing**
   - The MFS routine acquires an MFS workspace in the high I/O pool (HIOP).
   - IMS moves the output message from EMHB to HIOP after the completion of writing of the sync point log.
   - The MFS routine builds an output message segment by using the field information included in DOF.
3. VTAM processing
IMS issues the VTAM SEND request and pass the output data to VTAM.

4. Updating the DEDB physically
OTHREADS writes the updated Fast Path DB buffer into the DEDB data set. (This step is processed asynchronously with the message response process.)

Monitoring
The output time can be calculated by using following reports:

- IMS Performance Analyzer for z/OS: Fast Path (EMH) Transaction Transit Log (Example 6-42)

The Output Time column shows the average output time for each one transaction. Therefore, if you want to know the average time for each transaction code, some calculation is needed by using this output.

Example 6-42   IMS Performance Analyzer for z/OS: Fast Path (EMH) Transaction Transit Log

<table>
<thead>
<tr>
<th>Sync Point Time</th>
<th>Transact Code</th>
<th>Routing Code</th>
<th>Logical Terminal</th>
<th>User ID</th>
<th>IMID PST</th>
<th>DEDB MSDB</th>
<th>In-Q</th>
<th>Proc Out-Q</th>
<th>Total</th>
<th>Output</th>
<th>Total</th>
<th>-SEMHQ Time-</th>
</tr>
</thead>
<tbody>
<tr>
<td>14:21:03.04</td>
<td>DFSIVP4</td>
<td>*IFP</td>
<td>IMSP</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14:21:07.21</td>
<td>DFSIVP5</td>
<td>*IFP</td>
<td>IMSP</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14:21:50.66</td>
<td>IVTFD</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>123</td>
<td>197</td>
<td>319</td>
<td>6</td>
<td>319</td>
<td>6</td>
</tr>
<tr>
<td>14:21:53.39</td>
<td>IVTFD</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>14:21:55.55</td>
<td>IVTFD</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>14:22:00.19</td>
<td>IVTFD</td>
<td>IVTFD</td>
<td>SASAKI1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Example 6-43   Fast Path Log Analysis utility (DBFULTA0): DETAIL LISTING OF EXCEPTION TRANSACTIONS (Example 6-43)

The OUT column also shows you the output time per each one transaction.

Example 6-43   Fast Path Log Analysis utility (DBFULTA0): DETAIL LISTING OF EXCEPTION TRANSACTIONS

<table>
<thead>
<tr>
<th>SEQ NO.</th>
<th>TRANSCODE</th>
<th>SYNC POINT</th>
<th>SROUTING</th>
<th>LOGICAL TERMINAL</th>
<th>PST QUEUE</th>
<th>TRANSIT TIMES(MSEC)</th>
<th>-OUT-</th>
<th>DEBD</th>
<th>ADS...</th>
<th>VSO...</th>
<th>MSDB</th>
<th>BUF CONTENTIONS</th>
<th>R P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DFSIVP4</td>
<td>14:21:03:04</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DFSIVP5</td>
<td>14:21:07:21</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>IVTFD</td>
<td>14:21:03:04</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>IVTFD</td>
<td>14:21:50.66</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>IVTFD</td>
<td>14:21:53.39</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>IVTFD</td>
<td>14:21:55.55</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>IVTFD</td>
<td>14:21:57.25</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>IVTFD</td>
<td>14:22:00.19</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>IVTFD</td>
<td>14:22:05.03</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>IVTFD</td>
<td>14:22:10.18</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>IVTFD</td>
<td>14:22:13.74</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>IVTFD</td>
<td>14:22:17.86</td>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Considerations
Additional tuning considerations are as follows:

- **I/O to the format library**
  
  I/O to the format library can be monitored by using IMS Performance Analyzer for z/OS: Internal Resource Usage - Message Format Buffer Pool Statistics or the IMS Monitor: Buffer Pool Statistics - Message Format Buffer Pool.

- **Output message length**
  
  You can check the message length by using the IMS Performance Analyzer for z/OS: Fast Path (EMH) Message Statistics report (Example 6-44).

  The Out-Length column shows the output message length for each transaction code.

### Example 6-44  IMS Performance Analyzer for z/OS: Fast Path (EMH) Message Statistics

<table>
<thead>
<tr>
<th>Transact Code</th>
<th>Routing Code</th>
<th>Count</th>
<th>In-Length</th>
<th>Avg Max</th>
<th>Out-Length</th>
<th>Avg Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVTFD</td>
<td>IVTFD</td>
<td>10</td>
<td>59 90</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IVTFM</td>
<td>IVTFM</td>
<td>13</td>
<td>59 90</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**System Totals**

<table>
<thead>
<tr>
<th>Count</th>
<th>In-Length</th>
<th>Avg Max</th>
<th>Out-Length</th>
<th>Avg Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>59 90</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **The speed of the network**
- **The timing of the response from the terminal (In case of programmable terminal)**

### 6.3 Application considerations

In this section, we discuss the language interface, provide application considerations for performance when writing applications that operate in a IMS environment, and IMS and Language Environment.

#### 6.3.1 IMS language interface

IMS supports application programs that are written in Assembler, COBOL, Enterprise COBOL for z/OS, PL/I for z/OS, z/OS C/C++, Pascal, REXX, and Java.

To protect IMS and shield the developer from internal complexities, IMS uses an application programming interface referred to as the IMS language interface. The language interface consists of both the language-specific interface and the non-language-specific interface. The inherent difference is that the former is language-dependent and therefore supports only the language being used. The latter is language-independent and therefore can be called from any of the supported languages in IMS.

For simplicity, we use COBOL as the preferred programming language and show examples of the language dependent interface only.

Figure 6-18 on page 278 shows the structure of an application program as it relates to IMS and the language interface. A simple call to CBLTDLI with the required parameters ensures that IMS can obtain the data or message that is required by the application for processing. In its simplest form, we can either retrieve or send a message, or alternatively, retrieve and send data to a database in IMS.
Structure of a typical program using the language interface

The structure of any IMS program must take into account several issues:

- The hierarchy of all the databases that the application uses, which is called the DL/I hierarchy. Be sure you understand the root and dependent segment relationships in detail.
- IMS separates the program from the physical characteristics of the database. This separation is possible by providing the program with access to the program specification block (PSB). The program does not need to be concerned with the physical characteristics of the database description (DBD).
- The application program can access only the database that is using its program communication block (PCB) as defined in the PSB. The application therefore has the view of the data only as defined in the PCB.
- The call to the language interface is actually using the alias CBLTDLI. At linkage time, the linkage editor resolves the reference to DFSLI000.

Essentially, an IMS COBOL program includes the following processes:

- Program entry showing all the various sections of the program
- PCB definitions as per the Procedure Division mapping to the PSB
- I/O area definitions as required to retrieve and send messages and database segments
- DL/I calls as required by IMS
- SSA definitions as required by the database hierarchy
- Call to the language interface using CALL CBLTDLI
- Termination of program
6.3.2 Performance and programming considerations

Accessing data from a hierarchic database requires that we understand certain basic rules for Segment Search Arguments (SSAs). With an SSA the application program can access data as specified by the hierarchy. Using the correct DL/I retrieval call makes the retrieval process a lot more efficient. Consider the following information:

- If you need to process all segments sequentially in ascending sequence, within a database, a database record, or a specific segment type, use unqualified SSAs.
- If you need to process segments in a skip-sequential order in ascending sequence, always fully qualify the root segment.
- If you need to process database records directly, as in a random order by online transactions, use a qualified SSA.
- Use command codes to eliminate the overhead of doing multiple DL/I calls.
- Use programming standards for efficiency and reusability. Issues pertaining to standards are as follows:
  - Use the COPY facility in COBOL to copy any predefined areas that are shareable across multiple programs. This way prevents duplicating code. Function codes, segment descriptions, SSAs, and PCB masks are some of the common copybooks that you require.
  - Keep your I/O routines separate so that they can be easily changed if required. Checking status codes should be performed by a subroutine. In this way, any new status codes that are introduced to be easily adapted.
  - Use a modular approach to design your application. Keep generic functions as subroutines that can be called by any program that uses this service.
  - With batch processing, try to meet all your business requirements through a single pass of your data. If you require data for input into another process, create flat files with the required data rather than reading the database again.

SSA considerations
SSAs can be either qualified or unqualified. A qualified SSA identifies a specific occurrence of a segment. With a unqualified SSA, you identify only the segment type. When incorporated with command codes, the number of DL/I calls that are required to retrieve a segment or occurrences of a specific segment type can be reduced. Assume we have a hierarchy as depicted in Figure 6-19 on page 280.
We compare the differences between standard DL/I calls and those using command codes. For purposes of illustration, we use only the “F” command code. The input data consists of names. For example, assume we want to produce a listing of all names that match the input data. The program should check each dependent EDUC segment for the given name; if an MBA degree is detected for the person, the program can return to the first EDUC segment and list all degrees that person has. Table 6-1 shows the calls that are required.

<table>
<thead>
<tr>
<th>Standard DL/I call logic</th>
<th>Using command codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUbb NAMEbbb(LASTNMbbb=ADAMS)</td>
<td>GUbb NAMEbbb(LASTNMbbb=ADAMS)</td>
</tr>
<tr>
<td>GNPb EDUCbbb(DEGREEbbb=MBA)</td>
<td>GNPb EDUCbbb(DEGREEbbb=MBA)</td>
</tr>
<tr>
<td>GUbb NAMEbbb(LASTNMbbb=ADAMS)</td>
<td>GNPb EDUCbbb*Fb</td>
</tr>
<tr>
<td>GNPb EDUCbbb</td>
<td>GNPb EDUCbbb</td>
</tr>
</tbody>
</table>

The first solution without command codes is to issue a GU call to the NAME segment that is qualified by the input data. A GNP call to the EDUC segment that is qualified on the degree of MBA initiates a search of all dependent segments for a segment that meets that qualification. If none is found, the next input record can be read. If the person has an MBA degree, a GU call for the NAME, qualified again by the input data, backs up the database position to the root segment. A series of GNP calls that specifies the EDUC segment retrieves all those dependent segments, which then are listed.

The second solution uses the “F” command code after a hit has occurred on the degree of MBA. For GN or GNP calls, the “F” command code allows backing up to the first occurrence of this segment type within a database record. Therefore, unlike solution one, a GU call is not required to reestablish position at the root level for subsequent GN type calls, thus, saving an additional call and simplifying the programming.
Figure 6-20 shows the various command codes and which DL/I call type is suited for the command code. If a command code is used with a function for which it is not suitable, it is ignored. There are no status codes for invalid combinations of command codes. IMS assumes certain defaults. If two command codes are in direct conflict, the last one within the set is effective. However, “F” or “L” override “U” or “V” in the same SSA.

<table>
<thead>
<tr>
<th>Command Code</th>
<th>GU GHU</th>
<th>GN GHN</th>
<th>GNP GHNP</th>
<th>DLET</th>
<th>REPL</th>
<th>ISRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>D</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>F</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>L</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>U</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>V</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

- There is no status code for invalid combinations
- “F” or “L” will override “U” or “V” in the same SSA

Single versus multiple positioning

When single positioning (the default) is specified for a PCB, DL/I maintains only one position in that database for that PCB. This position is used to attempt to satisfy all subsequent GN calls. If an application requires parallel processing of segment types, a better solution is to use multiple positioning. Multiple positioning is specified in the PSB as Example 6-45 shows.

Example 6-45  Single and multiple positioning

```
PCB... POS=S  
PCB... POS=M  
```

The use of multiple positioning allows you greater independence for dependent segments. It allows the programmer to use GN or GNP calls without regard to the relative order of segment types within the DBD. If the relative order of segment types was reversed and multiple positioning was specified, the change does not affect application programs that use multiple positioning. However, multiple positioning requires the programmer to keep track of all positions maintained by DL/I; therefore, this approach requires additional coding. In any case, the program coding between single and multiple positioning differs. See Figure 6-21 on page 282.
Figure 6-21 shows the differences between a GNP call that uses single and multiple positioning. A GE status code is returned for single positioning when a GNP call is made to retrieve the second ORDER segment, because a GNP call can move forward only within the database. With multiple positioning, a position is maintained at each hierarchical leg within the database record until all segment occurrences are exhausted.

If we now replace the GNP call with a GN call, for single positioning, the second call to the ORDER segment retrieves the first ORDER segment of the next root segment, the third call the next root segment, and so on. Each call moves forward in the database. With multiple positioning, the results of a GN call will be the same as the results of the GNP calls, except that the GE status on the next to last call retrieves the first ORDER segment of the next root segment.

There are also restrictions when using multiple positioning and a mixture of qualified and unqualified SSAs. A programmer should not mix qualified and unqualified calls with multiple positioning, because the results can be unpredictable.

**Variable length segments**
You must make many decisions during the application design and the database design. Variable length segments must be considered when the size of the segment varies dependent on the data. Variable length segments save space in the database, and the programmer is responsible for ensuring that the size is defined correctly as part of the segment data to be inserted.

The minimum and maximum sizes are specified in the DBD with the following parameter:

\[ \text{BYTES}=(\text{max}, \text{min}) \]
The inserted size can never exceed the max size. If it does, a “V1” status code is returned to the application.

The key field of a variable length segment type must be contained within the minimum size as defined during DBD generation. Other non-key search fields that can be used in SSAs might appear anywhere within the segment. If a search is initiated on a non-key field and the field does not appear within a specific segment occurrence, a blank or zero is assumed for that data depending on how the data was originally described.

When used correctly, variable length segments can improve the efficiency of the database. However, they do involve overhead, especially if they are updated frequently. If you are using variable length segments, ensure that their length does not change frequently.

**Secondary indexing**

Secondary indexing (SI) was designed to provide two basic facilities:

- Processing of a database in sequences other than the prime root key sequence
- A direct search capability into a database record other than key field of the root

Secondary indexes are automatically maintained by DL/I, when the program inserts, replaces, or deletes data within a database.

When you use secondary indexes, consider the following information:

- Secondary indexing is a database structuring technique and normally concerns the database administrator. It should be transparent to the majority of the application programmers. However, because the same logical structure can yield two results in the PCB key feedback area, the programmer should be aware of these differences.
- The index target segment is the segment in the prime database that is pointed to by the SI segment. It is the segment that DL/I selects to satisfy the call made by the application.
- **XDFLD** is a special field within the DBD that contains the name of the key field data of the SI. The XDFLD name can be used in SSAs for the target segment. When the SI is used, the XDFLD is considered the key field of the target segment, and thus its value is placed in the key feedback area on any access to that segment.
- The pointer segment is the segment in the secondary index database that allows the SI to be used as a database.

**Program reusability considerations**

When coding an IMS COBOL program, the manner in which you set up and initialize working storage plays an important role in the eventual performance of the application. A program is truly reusable if it is capable of executing multiple transactions through a single copy of the program. In an IMS world, online transactions execute in a dependent region. If a copy of the program executes for the first time in a dependent region, the programmer must ensure that the logic of the program is not compromised after subsequent iterations of the same transaction in the same dependent region. Therefore, working storage areas should be initialized prior to retrieving a message.

**Processing options and the PROCOPT statement**

The PROCOPT parameter can be coded on the PCB and the SENSEG statement. If the PROCOPT parameter is not coded on the PCB statement, it defaults to PROCOPT=A. When PROCOPT is coded on the SENSEG statement, it overrides what is specified on the PCB statement, assuming they are compatible. IMS always locks a database record based on the PROCOPT specified on the PCB statement. In Example 6-46 on page 284, the PCB PROCOPT defaults to A.
When a program accesses the SENSEG, the GO applies, because the SENSEG overrides the PCB when they are compatible (GO is a subset of A). However, IMS locks on the PCB statement. Therefore, if the program was read only, IMS will still use the PROCOPT=A default on the PCB to lock the entire database record. This way affects the performance of a read-only program, so, a PROCOPT must always be coded on the PCB statement.

**Read only programs**
For those programs that are read-only on a database, two options can be specified. The first is read with integrity, which means that the read-only program must retrieve data that is 100% up-to-date. If this is the case, then specify PROCOPT=G. The second option is read without integrity, which means that the read-only program needs information that is not necessarily 100% current. If this is the case, then specify PROCOPT=GO, which tells IMS not to enqueue on the database record. As a result, the read-only program might run concurrently with an update program and not cause database record lockouts.

**Care when using PROCOPT=GO:** From a business perspective, supplying information based on aggregating data (such as number of employees) with PROCOPT=GO calls is usually acceptable, because minor errors are probably not significant. Supplying information based on a single record or a small number of possibly erroneous records by using PROCOPT=GO can compromise business integrity. *Use PROCOPT=GO with care.*

However, a PROCOPT=GO call might cause a program to abend with a U0853, because of IMS pointers that are not 100% up-to-date. Therefore, use PROCOPT=GON for read-only jobs that do not require 100% integrity (see “PROCOPT=GOT with DBRC SHARECTL” on page 284 for more information about GOT). When GOT is used and IMS encounters a bad pointer, IMS reads the segment a second time. If the pointer is still bad on the second read, then IMS returns a “GG” status code to the program. The program can then choose whether to issue the read again, to abend, or bypass that segment. Using GOx reduces the number of U085x abends.

**PROCOPT=GOT with DBRC SHARECTL**
We strongly prefer using DBRC SHARECTL and registering all databases in a production environment.

**Performance degradation:** be aware that IMS performance degradation can occur when using PROCOPT=GOT and DBRC SHARECTL. This degradation applies to all users and might apply to databases other than the one that is accessed with PROCOPT=GOT. This degradation applies to IMS DB/TM, DBCTL, CICS, and batch sharing environments.

If an invalid pointer is detected when using PROCOPT=GON or GOT, then processing is retried. With PROCOPT=GON, retry processing simply retries the call and returns a GG status code if the problem is still present.
The retry processing for PROCOPT=GOT consists of two actions:

- If a locking environment exists, a test lock on the database record containing the segment being retrieved is requested. (A locking environment exists for an IMS DB/TM, IMS DBCTL, or CICS system and also for IMS batch jobs if block-level data sharing is used.)
  
  The test lock causes call processing to wait until conflicting holders of locks on the database record release their locks.

- The last block read for the call is reread.
  
  This action is an attempt to get the latest image of the block.

In many cases, this retry succeeds in correcting the problem that produced the original invalid pointer condition. However, it can also cause serious performance degradation in the IMS subsystem. The degradation might affect databases for which PROCOPT=GOT is not used.

In addition to causing a reread of the last block, buffer invalidation takes place. This invalidation causes all the buffers for all databases that are registered at a share level greater than 0 (1, 2, or 3) to be invalidated. The lookaside process does not find any blocks in the pool for these databases until they are reread from DASD. OSAM sequential buffering buffers are invalidated, and also other buffers.

If the use of PROCOPT=GOT causes unacceptable performance degradation, consider changing PROCOPT=GOT to PROCOPT=GON.

Use PROCOPT=GOT only if contention with updaters, although possible, is highly unlikely.

**Use of checkpointing in batch**

Batch workload is extremely intensive and generally processes large amounts of data. When compared to an online transaction, a batch workload can be looked at as processing multiple messages in a single batch job. The problem is how to manage batch differently than online and at the same time allow both batch and online to coexist using the same limited resources. The second issue is how to restart a batch job after it runs for a few hours. We certainly do not want to restore and rerun the batch job. IMS checkpointing and restart both address these problems.

The two types of checkpointing in IMS are as follows:

- **Basic checkpointing**
  
  With basic checkpointing, the restart and repositioning become the responsibility of the application.

- **Symbolic checkpointing**
  
  With symbolic checkpointing, IMS restores the application work areas that are specified in the XRST call and does the repositioning. The application then restarts processing from the last successful checkpoint.

Several performance considerations are related to checkpointing in a batch process. We discuss several critical issues regarding the design of batch processes:

- A checkpointing frequency plays an important role, especially if you have a mixture of online and batch workload that runs concurrently. We suggest that you make the checkpointing frequency an externally controlled variable either through a control data set or by using the IMS APARM capability on the EXEC statement. An optimum balance must be found and is usually a factor of how much work is done by your batch process. Assume that the driver to the batch process is an input file with two million records. Each record requires complex processing across multiple databases. This unit of work per record must be evaluated in terms of its profile. When you understand the profile and the time duration
for processing a single record, you can begin to work out the checkpoint frequency. Be sure that your checkpoint frequency reflects the type of work that is being performed. Too few checkpoints can hold locks and might cause online transactions to wait until the locks are released. Too many checkpoints result in long batch run times, because the batch job spends most of its time checkpointing. A balance between batch and online must be found; there is no single good general rule for checkpointing, because it is a direct correlation to the complexity of the batch process.

▶ If your batch process updates Fast Path data entry databases (DEDB), ensure that your application logic revolves around the use of Fast Path buffers, namely normal buffer allocation (NBA) and overflow buffer allocation (OBA). When updating records in a Fast Path database, your JCL normally overrides the NBA and OBA buffer allocation. After you use all your NBA buffers, IMS returns the “FW” status code to your application. On receipt of the status code, you must set an indicator in your application to checkpoint at the next opportunity. Do not continue processing, because the result can be an “FR” status code, resulting in your batch job abending. When deciding on fair NBA and OBA buffer allocations, as a general rule, try to keep your NBA buffers to a reasonable size that allows you to checkpoint often. Remember that the NBA and OBA allocations come from your total Fast Path pool, defined at IMS startup time. A high NBA value might cause other jobs or online transactions to abend if you are running a number of batch jobs and online transactions concurrently.

Multi-streaming your batch processes
If your environment performs all its major consolidation of financial transactions at night by using a batch process, you most likely have the problem of trying to complete the batch before your next online day. To complicate matters further, if you need your online system to be available 24x7x365, a bigger problem can be that you must ensure that your batch processes do not conflict with your online processes.

A common approach is to ensure that you run your batch workload in parallel through multiple partitions or areas. If you use flat files as drivers for your batch process, ensure that you sort and split the files in the sequence of the database. This approach means that if you are dealing with a PHIDAM database, then sort and split the files by using the Partition Selection Exit or high keys, as specified in the RECON, in ascending sequence. If you use DEDBs, then sort and split the files in randomizer sequence. The sorts and splits can be easily achieved through specialized E15 and E35 exits by using the ICEMAN utility or alternatively, using ICETOOL.

If batch processes are intensive and time is of the essence, we prefer to use OSAM sequential buffering for PHDAM (with a sequential randomizer), or use PHIDAM, to sequentially process through the database, even if we are not updating each record. PHDAM (without a sequential randomizer) benefits from OSAM sequential buffering if the processing is consecutive (in relative byte address order), as with GN.

Why online programs are serially reusable
When IMS receives a message for a message processing program (MPP) in the input queue, it goes through several steps to process the request. Consider each run of a program or transaction as a message request to IMS. To keep this discussion in context, the program we are discussing has not yet been processed today. The first event that happens is that IMS determines if a message region is available in which to run this program.
If only one message region is available and there are two messages eligible to execute in that message region, IMS processes in the following order:

1. IMS looks at the current priority of the two messages and picks the one with the highest priority.
2. IMS loads the program-load module and the application control block (ACB). The ACB is a relationship between the program specification block (PSB) and the database definition (DBD). These steps consume time and affect the performance of the transaction.
3. After the program and ACB are loaded, IMS passes the message to the program and passes it control. When the program is finished processing and sends a message back to the user, control is given back to IMS.
4. IMS does not do any more work than necessary; it looks at the input queue for any different transactions with an equal or higher priority to process. If there are, IMS flushes the ACB and program from memory and proceeds with the new transaction.
5. However, if there are no different transactions with an equal or higher priority to process, and there are messages on the input queue for this same transaction to process, IMS uses the same load of the program and ACB to process the next message on the input queue. For this reason, the program has to reinitialize all of the working storage before it does a get unique (GU) operation to the input/output program communication block (I/O PCB) of the PSB. IMS continues to do this step until the processing limit count of the transaction is reached.
6. After the processing limit count is reached, IMS looks to see if it needs the message region for some other transaction:
   - If it does, the message region gets flushed and the new transaction is processed.
   - If not and there is more work for this transaction to process, IMS resets the processing limit count, writes some accounting records, and continues to use the same program load and ACB. This is called a quick reschedule.
7. IMS continues to do this process until it needs the message region or there are no messages for this transaction to work on. Now the process becomes more complex. If IMS does not need this message region to process other transactions and there are no more messages for the current transaction to process, IMS keeps everything in the message region. The program and ACB stay there and wait for work to do. What IMS is “hoping for” is that another message for this program is on the input queue before IMS needs this message region for a different program. This step is referred to as pseudo wait-for-input.

A parameter, named parallel limit count, determines how many messages must be on the input queue for a transaction before IMS schedules it in another message region. In this way, multiple transactions of the same name can run concurrently. A value of zero sets no limit. Generally, the parallel limit count is set to five, which means there must be five messages on the input queue before IMS schedules another message region. However, now that there are two message regions running the same program, there must be 10 messages on the input queue before IMS schedules another message region and so on. Although it is possible for a single transaction to occupy every message region available, a maximum region count parameter prevent this.
6.3.3 IBM Language Environment

The IBM Language Environment provides a single runtime environment for high level languages. It consists of the following features:

- Basic routines that support starting and stopping programs, allocating storage, communicating with programs in different languages, and handling conditions
- Callable services to handle generic services, such as math, and date and time services
- Language-specific portions of the library that ensure that the behavior is consistent across languages

The following languages are supported:
- C/C++
- Cobol
- PL/I
- Fortran (but not in IMS)
- Assembler

Functional overview
The functional overview of the Language Environment is shown in Figure 6-22.
Application and performance considerations in Language Environment

Performance of an application in a Language Environment is governed by many variables, some of which exist within Language Environment and others depend on the application itself. Consider the following information when you implement Language Environment in your environment:

- Review and understand exactly what is meant by migrating to Language Environment. This task involves understanding your applications, and reviewing the requirements and differences between working storage and local storage and how uninitialized variables affect the logic in your program when you convert to the Language Environment. The single important factor in your decision is the amount of work that is necessary to change programs if they are not Language Environment compliant. The result is that the logic or program flow must not be affected when you move to Language Environment.

- Investigate the use of Library Retention Routine (LRR) in your environment, because it allows performance improvement for Language Environment resources. Specify CEELRRIN in the DFSINT.xx member in PROCLIB.

- Review runtime options related to HEAP and STACK storage. The initial stack segment must be large enough for all requests for stack storage. Stack storage is essentially used for routine linkage and acquiring dynamic storage areas for the application. Storage for data items that are declared in the COBOL LOCAL-STORAGE-SECTION is allocated by using STACK storage. Heap storage, however, has a lifetime that is unrelated to the current execution of the current routine. It remains allocated until the enclave terminates. An example is a WFI region being terminated. For best performance, the initial HEAP size must be large enough to satisfy all requests for HEAP storage. All working storage variables are obtained from HEAP storage; therefore, be sure that your working storage is initialized on each invocation.

  Review the following runtime options:
  - ANYHEAP, BELOWHEAP, HEAP, and THREADHEAP
  - LIBSTACK, STACK, and THREADSTACK
  - ALL31(ON)

- The defaults cause the z/OS operating system to issue multiple GETMAIN macros for STACK and HEAP storage for the application and result in badly performing applications. Changes in the profile of a transaction, especially CPU, is noticeable when you review the tracking and trending reports of your workloads.

- With ALL31, you must ensure that all your applications are AMODE=31 programs. If they are AMODE=24 programs that coexist with AMODE=31, then you must use ALL31(OFF) and STACK(xx,xx,B ELOW). Try to ensure that all programs are AMODE=31.

- Setting the runtime options requires that you customize the runtime options by using either CEEDOPT (installation-wide defaults), CEEROPT (IMS region-wide default), or CEEUOPT (application specific options).

Extensive testing is of prime importance. Testing must follow a strict methodology to identify any problems that are related to Language Environment. A combination of both positive and negative testing must be done to detect possible errors in the logic of programs. Performance testing must also be done to gauge the savings or overhead that is associated with Language Environment.
Performance considerations for managing distributed workloads

In this chapter, we provide an overview of the specific performance considerations for managing workloads that originate from distributed platforms. We limit the concept of distributed workloads to the direct interfaces between distributed platforms and IMS, with an emphasis on online transactions and database access through the IMS Open Database architecture.

The chapter contains the following topics:

► General considerations
► TCP/IP connectivity through IMS Connect
► Open Database
► Synchronous callout
► Identifying and managing performance blind spots
7.1 General considerations

Real-time connectivity to and from IMS and distributed platforms is more critical today than at any other time in the history of IMS. This connectivity presents a number of challenges:

- How do you account for difference between the business conception of the transaction and the IMS-centric view of a transaction?
- How do you monitor performance in a way that is meaningful to the business?
- How do you assist application developers on distributed platforms to optimize for performance?

There are also concerns at the organizational level:

- How do you triage and assign problems that can exist across multiple stakeholders?
- How do you create meaningful test and production checks for performance?
- How do you communicate performance issues (and successes) to parts of your organization that are not familiar with IMS?

An effective performance management strategy depends on accounting for these challenges by understanding the types of workloads that must be handled, and by understanding how to monitor the various components that provide connectivity to IMS. One of these core components is providing direct TCP/IP connectivity to IMS transactions data.

7.2 TCP/IP connectivity through IMS Connect

IMS Connect provides a TCP/IP gateway into OTMA and IMS database (through IMS Open Database Manager). Because we discuss Open Database access in 7.3, “Open Database” on page 300, this section is limited only to managing transactional requests.

In this context, IMS Connect connects TCP/IP clients to IMS applications. It provides a simple pathway for you to create non-SNA (and often non-mainframe) applications that connect to IMS applications, and it is the pathway that other products, such as WebSphere Application Server and the IMS SOAP Gateway, use to connect to IMS.

IMS Connect clients use a protocol that is specific to IMS Connect to connect to IMS Connect. The protocol is the IMS Request Message (IRM) protocol. IMS Connect then uses message exits to do the following functions:

- Convert IRM requests to OTMA.
- Convert IMS Transaction Resource Adapter (ITRA) requests to OTMA. These are requests originating from Java clients running under WebSphere.
- Translate between EBCDIC and ASCII if required.
- Provide user authentication and authorization.

In addition to these functions, IMS Connect can be customized to provide site-specific functions such as additional security services, routing, and handling of proprietary payload formats.
7.2.1 Possible performance benefits of IMS Connect

IMS Connect helps you reduce network management costs by using more ubiquitous TCP/IP infrastructure and skill sets and also simplify your enterprise architecture by providing a more direct means of connecting to IMS.

In the broader context of the IMS Connect ecosystem, various IMS users find that the overall process of migrating to IMS Connect provides significantly improved transaction performance, more predictable scaling, and greater levels of availability.

This chapter explores ways in which these customers achieved these outcomes.

7.2.2 Routing messages to improve performance

The IMS Connect architecture provides opportunities for which parallelism can help improve transaction performance, throughput, and reliability by increasing the number of the following items:

- Ports that service a given IMS Connect instance
- IMS Connect instances that serve a given IMS
- TCBs between a given IMS Connect and IMS

The actual benefits (if any) of using these approaches depends on factors such as the use of security, the type of message payload, the dominant programming model, and other resource considerations. Fundamentally, determining which approach to use depends on trial and error, and monitoring.

**Increasing the number of IMS Connect instances**

By increasing the number of IMS Connect instances, you can more easily provide additional parallel pathways into OTMA. You can use a product such as SYSPLEX DISTRIBUTOR to distribute incoming TCP/IP session requests across the front ends.

The disadvantages of using multiple IMS Connect instances are the increased management overhead, and address space utilization.

Because the amount of CPU used for gross IMS Connect processing is typically low, you might be able to achieve similar advantages by using multiple OTMA TCBs (see “Increasing the number of OTMA TCBs” on page 293).

**Increasing the number of OTMA TCBs**

When a message comes in to IMS Connect, it specifies a target data store. The target data store maps to an XCF TMEMBER(i.e. IMS system). If there are any bottlenecks in OTMA, this aspect of transactional processing can provide room for improving transactional performance.

To improve parallelism here, you can create multiple data store definitions that each point to the same XCF TMEMBER, and then distribute messages between these data stores by using any of the following approaches:

- Client side: Implement a battery of target data stores on the client that each map to the same underlying IMS instance. The client alternates the various target data store values in the input message stream.
- Customized user message exit: Customize the IMS Connect message exit to map a single logical data store into multiple target data stores.
- Available tool: Use an IMS Tool such as IMS Connect Extensions for z/OS to define routing rules to provide the improved performance.
The main advantage of the first two approaches is that no additional tooling is necessary. The disadvantage is that in the long term such schemes are hard to maintain. In the case of client side batteries, changes to client collections require rollout to any number of potential clients, synchronization between distributed and IMS teams, and also introduce the potential for intermittent errors.

Customizing message exits is a better approach because they are easier to centrally manage, and allows the distributed clients to maintain a simple programming model.

You can customize the exits; you can use two tables that are available to IMS Connect Message exits:

- The INIT TABLE, which allows user data to be stored
- The DATASTORE TABLE, which contains data store IDs, statuses, and, optionally, user data.

You can then customize the user initialization exit routine (HWSUINIT) to load the tables, obtain the needed storage, and add any required user data to build your routing scheme. Modification of the message exit becomes more difficult if acknowledgement of transaction responses is required. In this case, some technique is necessary for remembering the original transaction routing so that the acknowledgment can be sent to the same data store. The use of commit mode 1 sync level NONE transactions does not totally eliminate the need for the remote client to provide acknowledgement for some IMS messages that might be received by the client.

Alternatively, you can use a tools such as IMS Connect Extensions for z/OS. The advantage of using this tool is that it makes this definition process simple and easy to update, including making real-time updates.

As shown in Figure 7-1, with IMS Connect Extensions for z/OS, you can do the following tasks:

- Map a single DESTID value into multiple values, each mapping to a separate TCB.
- Provide a degree of redundancy. Recall that if the IMS system fails, all data stores will also be unavailable. IMS Connect Extensions for z/OS can detect this condition so a fallback on a remote system can be done.

![Figure 7-1](Figure 7-1) Parallelism can help improve OTMA performance
Figure 7-2 shows how to do this configuration in IMS Connect Extensions for z/OS.

![Figure 7-2: Specifying routing rules to improve parallelism]

### 7.2.3 Recording IMS Connect events

Recording IMS Connect activity introduces additional overhead in terms of CPU and I/O. The extent of the overhead depends on site-specific factors and is difficult to predict. Although event collection can become a significant proportion of IMS Connect CPU utilization, it is typically only an incidental factor in the overall resource usage of the transaction. Again however, these numbers are highly site-specific.

In real-world cases, many customers learn that the benefits of providing event collection, in terms of reporting and problem determination, outweighs the cost in CPU utilization. However, recording events can provide additional motivation for minimizing the overall number of IMS Connect systems that process messages.

### 7.2.4 Real-time capacity management

Another aspect of managing TCP/IP performance is to use the capacity of IMS systems to dictate the distribution of messages from IMS Connect. This aspect helps tune performance, improve availability, and optimize resource utilization.

Consider the following scenarios that you might want to do:

- Send the majority of messages to the local IMS system to improve performance and availability but route some messages to a remote IMS to use its spare capacity.
- Send the majority of messages to the LPAR with the greatest capacity.
- Redirect messages to another IMS, because its default IMS performance is degraded or because you want to stop it for maintenance.
These kinds of scenarios can be implemented through more sophisticated routing schemes (those that account for system capacity). By using IMS Connect Extensions for z/OS, you can add a capacity weight to each target data store. The capacity weight dictates the probability that IMS Connect Extensions for z/OS will pick a particular data store relative to other data stores.

A key advantage of using IMS Connect Extensions for z/OS is that you can change these capacity weights in real time.

7.2.5 Message expiration and its impact on performance

For IMS Connect clients that access IMS TM, you can specify timeout values in the TCP/IP and DATASTORE configuration statements, and also in the IRM header of input messages.

On the TCP/IP configuration statement, the TIMEOUT keyword controls two aspects of communication wait time:

- How long IMS Connect keeps a connection open if the client does not send any input after the connection is first established
- For input messages that do not specify a timeout interval in the IRM, how long IMS Connect waits for a response from IMS before notifying the client of the timeout and returning the socket connection to a RECV state

IMS Connect can also adjust the expiration time for IMS transactions to match the timeout value of the socket connection on which the transaction is submitted.

If an expiration time is specified in IMS, transactions can expire and be discarded if IMS does not process them before the expiration time is exceeded. The expiration time is set in the IMS definition of the transaction by the EXPRTIME parameter on either the TRANSACT stage-1 system definition macro or either of the dynamic resource definition type-2 commands, CREATE TRAN or UPDATE TRAN. A transaction expiration time set by IMS Connect overrides any transaction expiration time specified in the definition of the transaction in IMS.

Setting the expiration time to match the socket timeout is generally preferred.

You might find that setting the timeout value requires more tweaking:

- Setting a timeout value that is too short results in having the additional overhead of clients needlessly reconnecting.
- Setting a timeout value that is too high results in needlessly holding resources on transactions that are not likely to complete.

Depending on the nature of the workload, both these errors can have a significant impact on performance.

We prefer setting the timeout value in the IRM giving more flexibility to change the setting over time and to make it dependent on the client that is connecting. One alternative is to use IMS Connect Extensions for z/OS to set these values for you. The benefits are that you do not
Chapter 7. Performance considerations for managing distributed workloads

need to modify clients, and that you can tweak these settings dynamically. Figure 7-3 shows such settings.

![Figure 7-3 Setting transaction timer options and activating transaction expiration]

### 7.2.6 User ID caching

With IMS 12, you can now cache user IDs and passwords. This helps improve performance and reduce processing overhead if you are performing user ID validation in IMS Connect. After being activated, IMS Connect will cache the user credentials and not require IBM RACF® for further re-authentications.

You activate user ID caching in the HWSCFGxx member by using the RACF, UIDCACHE, and UIDAGE parameters.

When IMS Connect is running, you can issue a type 2 command to refresh the details of a specific user:

```
UPDATE IMSCON TYPE(RACFUID) NAME(userid) OPTION(REFRESH)
```

You can also enable or disable the cache when IMS Connect is running with by issuing the following type 2 command:

```
UPDATE IMSCON TYPE(CONFIG) SET(UIDCACHE(ON | OFF))
```

If you have IMS Connect Extensions for z/OS, you can cache user IDs with any supported IMS version. In this case, you must not use IMS Connect validation, and only use IMS Connect Extensions for z/OS to do user validation; that is, if you activate IMS Connect Extensions for z/OS security, disable IMS Connect validation.

If you are using IMS 12, the main advantage of using IMS Connect Extensions for z/OS is that it provides additional security options, such as validations the port and IP address to connect on a given IMS Connect instance.
7.2.7 Specifying NODELAYACK

An ACK delay can be a significant contributor to your total end-to-end response time. It is particularly important for clients that do multiple writes, although can significantly add to the total response time.

Figure 7-4 demonstrates the issue. It shows the delay that is introduced for an IMS Connect input message when the TCP/IP 'ACK DELAY' parameter is defaulted. In the figure, port 7771 uses a default of approximately 250 ms for ACK delay; however, port 7772 specifies NODELAYACK.

![Figure 7-4: The effect of ACK delay on overall IMS Connect performance](image)

**Figure 7-4**  The effect of ACK delay on overall IMS Connect performance
Chapter 7. Performance considerations for managing distributed workloads

Figure 7-5 shows a report that is produced by the IMS Performance Analyzer for z/OS. IMS Performance Analyzer for z/OS is reporting from the IMS Connect Extensions for z/OS journal event records. This example of the IMS Connect trace report shows significant IMS Connect events and the elapsed time delta between the events. The first event for an IMS Connect transaction is normally the READ PREPARE event. This trace is for port 7771 and it shows a time delta of 265 ms between the READ PREPARE and READ SOCKET events. This time is close to the ACK DELAY default of 250 ms.

Because this transaction is a SYNCLEVEL=CONFIRM type of transaction, the ACK DELAY also affects the client acknowledgement. The time delta between the first and second READ SOCKET events to read the acknowledgement is 265 ms. Again, this number is close to the ACK DELAY default.

---

**Figure 7-5** Report demonstrating slow performance from ACK delay

---

**Table:**

<table>
<thead>
<tr>
<th>Start/Relative</th>
<th>Elapsed</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>09.48.08.036414</td>
<td>265.264</td>
<td>49</td>
<td>Read Socket</td>
</tr>
<tr>
<td>+0.265264</td>
<td>6.104</td>
<td>49</td>
<td>Read Socket</td>
</tr>
<tr>
<td>+0.265536</td>
<td>0.029</td>
<td>3E</td>
<td>Message Exit called for READ</td>
</tr>
<tr>
<td>+0.265480</td>
<td>0.082</td>
<td>3E</td>
<td>Message Exit return for READ</td>
</tr>
<tr>
<td>Time for client to acknowledge: 0.182</td>
<td>41</td>
<td>Message sent to OTMA</td>
<td></td>
</tr>
<tr>
<td>+0.399146</td>
<td>124.442</td>
<td>42</td>
<td>Message received from OTMA</td>
</tr>
<tr>
<td>+0.399246</td>
<td>0.148</td>
<td>3D</td>
<td>Message Exit called for XMIT</td>
</tr>
<tr>
<td>+0.399289</td>
<td>0.043</td>
<td>3E</td>
<td>Message Exit return for XMIT</td>
</tr>
<tr>
<td>Also affects read socket for Ack/Nak: 0.187</td>
<td>41</td>
<td>Message sent to OTMA</td>
<td></td>
</tr>
</tbody>
</table>

---

IMS Performance Analyzer 3.3
IMS Connect Trace - hwsinst
Trace from 10May2005 09.48.08.
Figure 7-6 shows the same IMS Performance Analyzer for z/OS report, but for port 7772 (that was defined with NODELAYACK). The elapsed time between the READ PREPARE and READ SOCKET events is now less than 1 ms. Similarly, the time for reading the clients ACK is close to 1 ms.

### 7.3 Open Database

Open Database, introduced in IMS 11, provides a significant innovation that allows greatly simplified access to IMS data from distributed platforms. Although Open Database presents a number of architectures, including local connectivity, we focus solely on distributed access in this section.

In this section, Open Database calls involve IMS Connect as the TCP/IP front end and acts as the DRDA Application Server (AS), a new address space named Open Database Manager (ODBM) that manages the requests and forwards them to IMS.
From a functional perspective, with Open Database, programmers on distributed platforms can access IMS data in a standards-based way by using either DL/I or SQL. See Figure 7-7.

### Open Database Environment

![Diagram of Open Database Environment]

**Figure 7-7  Open Database distributed architecture**

#### 7.3.1 Challenges of Open Database performance management

The Open Database architecture presents a number of new performance challenges that you must address:

- Access to IMS database by programmers who are not familiar with IMS
- Unexpected DL/I call trees emerging from SQL queries with unexpected performance implications
- Multiple calls for the same logical requests (which cannot easily be tied together)
- Performance that depends on multiple address spaces such as ODBM and IMS Connect
7.3.2 Open Database call-flows and timings

Open Database requests are based on open standards and can therefore be analyzed by standard tools. Figure 7-8 shows a Wireshark analysis of a DRDA request to IMS. Note the calls generated by the client, and the timings for each response.

![Wireshark analysis of a DRDA request](image)

Also notice the following information:
- Although Wireshark identifies each of the stages in the call flow and understands many of the code points, several code points are IMS-specific.
- Wireshark can show the DRDA flow but cannot interpret what is being requested.
Open Database (DRDA) analysis using IMS Connect Extensions for z/OS data

Consider the following features:

- IMS Connect Extensions for z/OS event journaling provides a comprehensive trace of every Distributed Relational Database Architecture (DRDA) call that is issued by the application (Figure 7-9).

<table>
<thead>
<tr>
<th>Command</th>
<th>Tracking active</th>
</tr>
</thead>
<tbody>
<tr>
<td>BROWSE CEXX000.QADATA.REDBOOK.ORDAT111.ICON.D1003</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code Description</th>
<th>Date 2010-03-31 Wednesday Time (GMT)</th>
</tr>
</thead>
</table>

- A03C Prepare READ Socket 05.46.43.550438
- A049 READ Socket 05.46.43.550569
- **A05B DRDA 1041 EXCSAT-Exchange Server Attributes** 05.46.43.550577
- A049 READ Socket 05.46.43.550601
- A049 READ Socket 05.46.43.550623
- A05B DRDA 106D ACCSEC-Access Security 05.46.43.550630
- A05C DRDA 1443 EXCSATRD-Server Attributes Reply data 05.46.43.550653
- A04A WRITE Socket 05.46.43.550715
- A049 READ Socket 05.46.43.550727
- **A05B DRDA 106E SECCHK-Security Check** 05.46.44.166264
- A063 ODBM Security Exit called 05.46.44.166264
- A064 ODBM Security Exit returned 05.46.44.166264
- A05C DRDA 1219 SECCHKRM-Security Check Reply Message 05.46.44.166264
- A04A WRITE Socket 05.46.44.166264
- A049 READ Socket 05.46.44.794050
- **A05B DRDA 2001 ACCRDB-Access RDB** 05.46.44.794057
- A05D ODBM begin Allocate PSB (APSB) Program=AUTPSB11 05.46.44.794080
- A061 ODBM Routing Exit called 05.46.44.794090
- A062 ODBM Routing Exit returned 05.46.44.794258
- A05A ODBM Trace: Message sent to ODBM 05.46.44.794699
- A05A ODBM Trace: Message received from ODBM 05.46.45.067979
- A05A Message received from ODBM 05.46.45.067979
- A05E ODBM end Allocate PSB (APSB) Program=AUTPSB11 05.46.45.068131
- A05C DRDA 2201 ACCRDBRM-Access RDB Reply Message 05.46.45.068134
- A04A WRITE Socket 05.46.45.068647
- A048 Trigger Event for ODBMSMG 05.46.45.068680
- A03C Prepare READ Socket 05.46.47.094684
- A049 READ Socket 05.46.47.094762
- **A05B DRDA CC05 DLIFUNC-DL/I function** 05.46.47.094832

*IMSPI ISREDDE2 SDSF -DMLIST

---

Figure 7-9 IMS Problem Investigator: ODBM Open Database formatting
Provides easy-to-read formatting of all DRDA code points, for both the open standard (Figure 7-10) and for IMS (Figure 7-11).

**Figure 7-10** IMS Problem Investigator: DRDA open source code-point formatting

**Figure 7-11** IMS Problem Investigator: DRDA IMS code-point formatting
Tracks application calls that are associated with a single thread to identify bottlenecks.

Analyzes DL/I call results including I/O and feedback areas (Figure 7-12).

Figure 7-12  IMS Problem Investigator: DL/I call analysis
Merges the IMS Connect Extensions for z/OS journal with the IMS log to see the complete end-to-end picture of the session thread of a distributed transaction (Figure 7-13).

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Date</th>
<th>Time (GMT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>08</td>
<td>Application Start Program=AUTPSB11 Region=0003</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.066230</td>
</tr>
<tr>
<td>5607</td>
<td>Start of UOR Program=AUTPSB11 Region=0003</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.066232</td>
</tr>
<tr>
<td>5616</td>
<td>Start of protected UOW Region=0003</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.066530</td>
</tr>
<tr>
<td>A06A</td>
<td>ODBM Trace: Message received from ODBM</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.067979</td>
</tr>
<tr>
<td>A06E</td>
<td>Message received from ODBM</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.067997</td>
</tr>
<tr>
<td>A05E</td>
<td>ODBM end Allocate PSB (APSBB) Program=AUTPSB11</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.068131</td>
</tr>
<tr>
<td>A05C</td>
<td>DRDA 2201 ACCRDBRM-Access RDB Reply Message Program=AUTPSB11</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.068154</td>
</tr>
<tr>
<td>A04A</td>
<td>WRITE Socket</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.068647</td>
</tr>
<tr>
<td>A048</td>
<td>Trigger Event for ODBMMSG</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.068680</td>
</tr>
<tr>
<td>A03C</td>
<td>Prepare READ Socket</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.068762</td>
</tr>
<tr>
<td>A049</td>
<td>READ Socket</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.068925</td>
</tr>
<tr>
<td>A05B</td>
<td>DRDA CC05 DLIFUNC-DL/I function</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.069131</td>
</tr>
<tr>
<td>A049</td>
<td>READ Socket</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.069231</td>
</tr>
<tr>
<td>A048</td>
<td>Trigger Event for ODBMMSG</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.0695015</td>
</tr>
<tr>
<td>A049</td>
<td>READ Socket</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.0695038</td>
</tr>
<tr>
<td>A05B</td>
<td>DRDA CC06 SSALIST-List of segment search argument</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.0695045</td>
</tr>
<tr>
<td>A06A</td>
<td>ODBM Trace: Message sent to ODBM</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.0695985</td>
</tr>
<tr>
<td>A069</td>
<td>Message sent to ODBM</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.45.0696016</td>
</tr>
<tr>
<td>20</td>
<td>Database Open Database=EMPLDB2 Region=0003</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.47.143647</td>
</tr>
<tr>
<td>20</td>
<td>Database Open Database=EMPLDB2 Region=0003</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.47.143647</td>
</tr>
<tr>
<td>B021</td>
<td>DLI Database Trace Database=EMPLDB2 Func=GHU</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.47.192378</td>
</tr>
<tr>
<td>A06A</td>
<td>ODBM Trace: Message received from ODBM</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.47.192881</td>
</tr>
<tr>
<td>A06A</td>
<td>Message received from ODBM</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.47.193099</td>
</tr>
<tr>
<td>A05C</td>
<td>DRDA 2205 OPNQRYRM-Open Query Complete</td>
<td>2010-03-31 Wednesday</td>
<td>05.46.47.193186</td>
</tr>
</tbody>
</table>
7.4 Synchronous callout

Synchronous callout allows an IMS application to call out to a web service, receive a response, and continue processing the transaction. As a feature, synchronous callout allows IMS to participate in a service-oriented architecture (SOA).

As shown in Figure 7-14, a callout is processed by IMS Connect and is passed to a web service either through the SOAP gateway, WebSphere, or a custom application.

![Figure 7-14  A synchronous callout allows an IMS transaction to reach web services](image)

As shown in Figure 7-15 on page 308, the call flow for a synchronous callout involves a number of steps. From a performance perspective, the RESUME TPIPE that is used by IMS Connect to first initialize a connection with a web service is not significant. However, the actual call flow can have significantly affect performance.

Moreover, understanding and reporting synchronous callout performance requires information from additional points in the transaction flow. For example, if the synchronous callout takes a long time to complete, how do you eliminate IMS and IMS Connect as potential sources of the problem?
IMS Connect provides communication between IMS and the callout server. Understanding and analyzing synchronous callout requires information from both IMS and IMS Connect.

Using the data that is provided by IMS Connect Extensions for z/OS or your own collection in IMS, you can report on the synchronous callout performance by using IMS Performance Analyzer for z/OS.

Figure 7-16 shows an IMS Performance Analyzer for z/OS report. The sync callout response time is a large percentage of the overall processing time, indicative of a delay.

The sync callout response time is a large percentage of the overall processing time.
Although the previous report shows the synchronous callout characteristics, performance tuning requires a deeper investigation into the callout.

As Figure 7-17 shows, you can use IMS Problem Investigator for z/OS to display the flow of a single synchronous callout, and use an elapsed time view to identify timing delays in the lifecycle of the synchronous callout request.

![Figure 7-17  Synchronous callout event flow](image)
7.5 Identifying and managing performance blind spots

The scenarios and issues raised in this chapter share a common theme: the more complicated a transaction is, the more likely it is to involve areas for which you have limited performance information. Traditional IMS performance reporting, which is centered around the time between when the input message is received by IMS and then is put on the output queue by IMS, can be insufficient because too many scenarios are possible, in which the client’s perspective of the problem can imply significant performance issues. However, the actual input-output queue processing in IMS is optimal. See Figure 7-18.

You can illuminate a performance issue by eliminating alternative explanations for which we do have performance information.

For example, in Figure 7-19 on page 311, IMS is showing rapid response times, IMS Connect response times are slow, but is still fast before the message enters OTMA. Therefore, OTMA is the source of the problem.
Notice that although there is no direct measurement of OTMA performance in the report, the performance of OTMA can be deduced by covering the performance for the transaction endpoints. However, without IMS Connect information, diagnosing and tracing the performance problem to its source becomes more difficult.

Figure 7-19 shows that OTMA is responsible for the majority of processing time.

![Table of IMS Performance Analyzer](image)

**Figure 7-19** IMS performance is optimal; most of the time is spent in IMS Connect
Fast Path performance considerations

This chapter explores Fast Path processing. IMS 11 introduced Fast Path 64-bit buffer pools. IMS 12 introduces Fast Path secondary index (FPSI) and Fast Path log reduction.

First, we describe general performance considerations and then proceed to the new functions:

- **Fast Path 64-bit buffer manager**
  This function was introduced in IMS 11 and enhanced in IMS 12. The 64-bit buffer manager creates database buffers above the 2 GB bar, in 64-bit virtual storage. Those buffer pools are pre-expanded in anticipation of future needs and automatically compressed when the use of a subpool drops.

- **Fast Path secondary index (FPSI) and performance comparison with HDAM**
  IMS 12 provides support for secondary indexes for DEDBs. A secondary index database provides an alternate path to access its corresponding primary DEDB and can be processed as a separate database. IMS supports two database structures for FPSI: hierarchical indexed sequential access method (HISAM) and simple hierarchical indexed sequential access method (SHISAM).

- **Fast Path log reduction**
  IMS 12 allows users to specify that the before-image record of type ‘99’ log is not to be written. It can be a benefit for asynchronous data capture users who want only after-image log records with reducing logging volumes and improving performance.
8.1 IMS Fast Path databases

IMS Fast Path includes two database organizations:

- Data entry database (DEDB)
- Main storage database (MSDB)

DEDBs are hierarchical databases similar to HDAM but have significant differences that provide even higher performance, capacity, and availability. MSDBs reside in storage, enabling application programs to avoid the I/O activity to access them, and also provide high performance. MSDBs cannot be used in a data sharing environment and cannot be updated from OTMA transaction.

8.2 DEDB general performance considerations

When the performance problems occur with DEDBs, major analysis and tuning points are as follows:

- I/O IWAIT/CALL
- Elapse time for I/O IWAIT
- Usage of Fast Path resources

Figure 8-1 shows two patterns of DL/I call access to DEDB. If the DB is well organized, only one I/O occurs for one segment search request (Case1). But, if the segments in a record are contained in multiple CIs, multiple I/O can occur for one segment request (Case2).
8.2.1 I/O IWAIT/CALL

This I/O IWAIT/CALL is the number of I/O IWAITs that occur per one DL/I call. The DASD I/O occurs when the CI, including the searched segment, is not in the buffer. If the segment is included in the CI that reads first, the program can get the segment with one DASD I/O. But the program needs to read multiple CIs per DL/I call if there are synonym chains or overflows of segments into the Independent Overflow (IOVF) or the Dependent Overflow (DOVF) parts. When there are many dependent segments for one root segment, the DB record can be divided into multiple CIs; and multiple I/Os are also needed to reach the dependent segment that is placed in the latter part of the record. These instances can make the I/O IWAIT rate higher.

The IWAITs/Call section in IMS Performance Analyzer Database IWAIT Summary report gives you the number of I/O IWAIT/CALL. It is reported for each database data sets. The rate under 1.0 is ideal. See Example 8-1.

Example 8-1 IMS Performance Analyzer for z/OS: Database IWAIT Summary

<table>
<thead>
<tr>
<th>DDname</th>
<th>Type</th>
<th>IWAITs</th>
<th>Sc.Mil.Mic</th>
<th>X Avg</th>
<th>Sc.Mil.Mic</th>
<th>Waiting</th>
<th>/Call</th>
<th>Calls</th>
<th>IWTElp</th>
<th>D LAElp</th>
<th>Elapsed=</th>
<th>0 Hrs</th>
<th>2 Mins</th>
<th>19.800.185 Secs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDOPER01</td>
<td>DEDB</td>
<td>1,866</td>
<td>0.454</td>
<td>.367</td>
<td>6.930</td>
<td>1,492</td>
<td>1.25</td>
<td>6.27%</td>
<td>15.80%</td>
<td>13.535%</td>
<td>10.171%</td>
<td>0.454</td>
<td>.367</td>
<td>6.930</td>
</tr>
<tr>
<td>DDTRMC01</td>
<td>DEDB</td>
<td>1,436</td>
<td>0.453</td>
<td>.620</td>
<td>9.839</td>
<td>1,336</td>
<td>1.07</td>
<td>5.61%</td>
<td>12.16%</td>
<td>10.378%</td>
<td>7.799%</td>
<td>0.453</td>
<td>.620</td>
<td>9.839</td>
</tr>
<tr>
<td>DDBOMN01</td>
<td>DEDB</td>
<td>1,077</td>
<td>0.460</td>
<td>.119</td>
<td>1.114</td>
<td>980</td>
<td>1.10</td>
<td>4.12%</td>
<td>9.12%</td>
<td>7.901%</td>
<td>5.938%</td>
<td>0.460</td>
<td>.119</td>
<td>1.114</td>
</tr>
<tr>
<td>DDSSEQ01</td>
<td>DEDB</td>
<td>903</td>
<td>0.487</td>
<td>.102</td>
<td>1.009</td>
<td>903</td>
<td>1.00</td>
<td>3.79%</td>
<td>7.64%</td>
<td>7.015%</td>
<td>5.272%</td>
<td>0.487</td>
<td>.102</td>
<td>1.009</td>
</tr>
<tr>
<td>DDCODE01</td>
<td>DEDB</td>
<td>731</td>
<td>0.414</td>
<td>.071</td>
<td>0.783</td>
<td>679</td>
<td>1.08</td>
<td>2.85%</td>
<td>6.19%</td>
<td>4.831%</td>
<td>3.631%</td>
<td>0.414</td>
<td>.071</td>
<td>0.783</td>
</tr>
<tr>
<td>DDDEPO02</td>
<td>DEDB</td>
<td>180</td>
<td>0.734</td>
<td>1.245</td>
<td>12.923</td>
<td>128</td>
<td>1.41</td>
<td>.54%</td>
<td>1.52%</td>
<td>2.108%</td>
<td>1.584%</td>
<td>0.734</td>
<td>1.245</td>
<td>12.923</td>
</tr>
<tr>
<td>DDDEPO06</td>
<td>DEDB</td>
<td>192</td>
<td>0.671</td>
<td>.079</td>
<td>1.090</td>
<td>132</td>
<td>1.45</td>
<td>.55%</td>
<td>1.63%</td>
<td>2.059%</td>
<td>1.546%</td>
<td>0.671</td>
<td>.079</td>
<td>1.090</td>
</tr>
<tr>
<td>DDDEPO15</td>
<td>DEDB</td>
<td>319</td>
<td>0.431</td>
<td>.071</td>
<td>0.783</td>
<td>679</td>
<td>1.08</td>
<td>2.85%</td>
<td>6.19%</td>
<td>4.831%</td>
<td>3.631%</td>
<td>0.431</td>
<td>.071</td>
<td>0.783</td>
</tr>
<tr>
<td>DDBOCN01</td>
<td>DEDB</td>
<td>198</td>
<td>0.564</td>
<td>.077</td>
<td>0.982</td>
<td>181</td>
<td>1.09</td>
<td>.76%</td>
<td>1.68%</td>
<td>1.783%</td>
<td>1.340%</td>
<td>0.564</td>
<td>.077</td>
<td>0.982</td>
</tr>
<tr>
<td>DDDEPO05</td>
<td>VSAM</td>
<td>163</td>
<td>0.682</td>
<td>.095</td>
<td>0.986</td>
<td>114</td>
<td>1.43</td>
<td>.48%</td>
<td>1.38%</td>
<td>1.776%</td>
<td>1.334%</td>
<td>0.682</td>
<td>.095</td>
<td>0.986</td>
</tr>
<tr>
<td>DDDEPO10</td>
<td>VSAM</td>
<td>163</td>
<td>0.677</td>
<td>.096</td>
<td>1.118</td>
<td>122</td>
<td>1.34</td>
<td>.51%</td>
<td>1.38%</td>
<td>1.762%</td>
<td>1.324%</td>
<td>0.677</td>
<td>.096</td>
<td>1.118</td>
</tr>
</tbody>
</table>

If the I/O IWAIT/CALL rate is high, be sure to check the existence of synonym chains or the overflows of segments into IOVF/DOVF by using, for example, High Performance Fast Path Utilities -DEDB Pointer Checker. Then, if necessary, you can reorganize the database or change the structure of it (the sizes of CI, RAA, and UOW).

If there are many dependent segments for one root segment, you can change only the logical structure of the database.

8.2.2 Elapse time for I/O IWAIT

The elapse time is of DASD I/O processing. The Elap/IWAIT section, in the “IMS Performance Analyzer for z/OS: Database IWAIT Summary report,” gives information of the length of I/O IWAIT. The proper value varies on the disc model, generally under 10 ms is preferable.

It might be possible to identify the parts of the database that are heavily accessed (hot spots) and move these records either to another existing area or to a new area. We suggest that you spread the hot spots around, if possible, so that the activity is evenly dispersed among the area data sets on fast DASD devices. Or, if the reduction of the elapse is essential, a solution might be to use Virtual Storage Option (VSO) or Shared Virtual Storage Option (SVSO). See 8.4, “VSO and SVSO” on page 322.
8.2.3 Usage of Fast Path resources

Fast Path databases are accessed by using dedicated buffers, so checking these resources is important. The following parameters are related to Fast Path resources. Some of them can be defined in IMS control region EXEC parameters and others in the dependent region:

- **Control region EXEC parameters (DFSPBxx member of IMS.PROCLIB)**
  - **DBBF**  Maximum number of Fast Path buffers
  - **DBFX**  Number of buffers out of the DBBF to be set aside and page-fixed at control region initialization to use for DEDB writes.
  - **BSIZ**  Fast Path database buffer size
  - **OTHR**  Number of concurrent output threads that Fast Path is to support for the entire system

- **Dependent region EXEC parameters (IFP, MPP, BMP)**
  - **NBA**  Normal buffer allocation
  - **OBA**  Overflow buffer allocation

**Fast Path 64-bit manager:** If you enable the Fast Path 64-bit manager, you can specify the buffer pool usage in DFSDFxxx member of IMS.PROCLIB, and some of those parameters are ignored. See 8.5, “Fast Path 64-bit buffer manager” on page 331.

You can display the entire current Fast Path buffer usage with the `/DIS POOL FPDB` command. See Example 8-2.

**Example 8-2 /DIS POOL FPDB command response**

```
/DIS POOL FPDB
DFS4445I CMD FROM MCS/E-MCS CONSOLE USERID=IMSR4: DIS POOL FPDB IMS12A
DFS4444I DISPLAY FROM ID=IMS12A 781
FPDB BUFFER POOL:
+ AVAIL =   20  WRITING =      0  PGMUSE =      0  UNFIXED = 280
  ** NO DEDB PRIVATE POOLS DEFINED  **
*12203/084520*
```

When the UNFIXED is less than the NBA required by the starting dependent region, the region initialization fails. If it happens, you should increase the DBBF.

**DBBF**

Use the following formula to calculate the number of Fast Path database buffers required:

\[
DBBF = \text{Number of open areas that have SDEP segments} + \text{Sum of NBA for all concurrently active FP programs} + \text{Largest OBA allocation for any of the concurrently active FP programs, including any specified by CICS for DBCTL} + \text{DBFX} + \text{Sum of all Fast Path buffers used by CICS(CNBA)} + \text{(Some margin for IMS internal tasks)}
\]

If the number of database buffers requested by DBBF is not large enough, then an area open or a region initialization fails.
When the new NBA request occurs after using all DBFX buffers, new buffers are page-fixed; this operation can influence the transaction or batch performance. The number of buffer waits are reported in either of the following locations; the ideal number is 0 (zero):

- The Common Buffer Usage Wts section in IMS Performance Analyzer for z/OS: Fast Path Resource Usage and Contention (Example 8-3)
- The COMMON BUFFER USAGE WTS section in Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF RESOURCE USAGE AND CONTENTIONS report (Example 8-4)

If the buffer wait occurs, increase DBFX and DBBF. The long response time of database update I/O can also lead to the delay of buffer release, and cause buffer wait. When that happens, you should also investigate the DASD response time.

### NBA and OBA

When a dependent region is started with NBA specified in its execution parameters, it causes the NBA number of buffers to be made available for the region in the Fast Path buffer pool. This number of buffers must be sufficient to handle the processing of the vast majority of programs running in that region. These buffers are page-fixed when the region starts.

All CIs that are locked at the exclusive level remain locked until the buffer is released. Buffers that were not updated are released when either of the following items is reached:

- The NBA limit is reached (and buffer stealing occurs).
- The program reaches sync point.

Updated buffers are released only when the OTHREADs are completed.

If your program requires more than its NBA, IMS can provide additional buffers. The number allowed is specified by the OBA parameter on the region procedure. However, IMS permits only a single program to access its OBA buffers at any point in time and uses the OBA latch to enforce this requirements (generally, OBA is required only by update programs).
The OBA latch is released when the holding program reaches sync point. If the latch is unavailable because another program is using its OBA buffers, the region waits until the latch becomes available. At any time, only the largest OBA requested by a region is page-fixed in the Fast Path buffer pool. Be sure to allocate sufficient NBA for the majority of work units, so that OBA is rarely used.

The IMS Performance Analyzer for z/OS: Fast Path Resource Usage and Contention (Example 8-5) and Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF RESOURCE USAGE AND CONTENTIONS report (Example 8-6) show the number of NBAs actually used and the number of OBA latch conflicts by each Fast Path transaction.

**Example 8-5  IMS Performance Analyzer for z/OS: Fast Path Resource Usage and Contention**

**Example 8-6  Fast Path Log Analysis utility (DBFULTA0): OVERALL SUMMARY OF RESOURCE USAGE AND CONTENTIONS**

You can also use the IMS Performance Analyzer for z/OS: Fast Path Transaction Exception Log report (Example 8-7). This report can show similar information to the DBFULTA0 report, but enhanced data filtering, time precision, and additional fields, such as USERID, are available.

**Example 8-7  IMS Performance Analyzer for z/OS: Fast Path Transaction Exception log**
The number that is reported under the Buf Use column, on the first line for every transaction, is the total number of buffers used, irrespective of whether they were only NBA, or NBA plus OBA. The Buffers line in the report gives the details of the number used within NBA and OBA.

The report can provide details of buffer usage by type. It details the number of NBA, OBA, NRDB (non-related buffers for SDEP and MSDB use), number of times buffer stealing was invoked, number of times the program waited for a buffer to become available, number of buffers written with OTHREADs, and also the number of buffer sets that are used by HSSP and the high-speed reorganization utility.

You can monitor the OBA latch contention by means of the Latch Statistics section of IMS Performance Analyzer for z/OS: Internal Resource Usage report. Be sure to investigate any value different from 0 (zero) total IWAITs in the OBA latch line, and increase the NBA enough to make it to 0. The fragmentation can lead to the increase of buffer usage, so you can investigate the DEDBs that are accessed by the transaction, using High Performance Fast Path Utilities -DEDB Pointer Checker, and reorganize them if needed.

**OTHRE**

When buffers that are waiting to be written start queuing for OTHREADs, buffer contention increases because the locks on the data in the buffers are not released until the buffers are written to DASD. If the problem is on the buffer side, we suggest that you split the area across several DASD volumes.

However, the problem might be an output thread shortage. We also suggest that you set the OTHR system execution parameter to a value big enough so that the write buffers are not queued because of insufficient numbers of SRBs.

The IMS Performance Analyzer for z/OS: Fast Path OTHREAD Analysis report gives you some useful information about the value to specify for the OTHR parameter (Example 8-8 on page 320). The Max Value column of the Active OTHREADs section shows the maximum number of SRBs scheduled for OTHREAD requests; if the value is close or equal to the OTHR parameter, increase this parameter value.
The output thread processing for VSO and SVSO DEDBs is somewhat different. See “VSO output threads” on page 323 for details.

**Example 8-8  IMS Performance Analyzer for z/OS: Fast Path OTHREAD Analysis**

Report from 14Jun2012  18.22.05.26 IMS 12.1.0 IMS Performance Analyzer 4.3 Report to 14Jun2012  18.52.30.76

**Fast Path OTHREAD Analysis**

From 14Jun2012 18.30.29.14 To 14Jun2012  18.44.19.38 Elapsed= 0 Hrs 13 Mins 50.236.850 Secs

<table>
<thead>
<tr>
<th>Enq Counts</th>
<th>OTHREAD/Enq StDev Max value</th>
<th>Area/Enq StDev Max value</th>
<th>Buff/Enq StDev Max value</th>
</tr>
</thead>
<tbody>
<tr>
<td>731</td>
<td>0.03</td>
<td>6.444</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td>12.025</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3.20</td>
<td>1.066</td>
<td>16</td>
</tr>
</tbody>
</table>

**Active OTHREAD / Enq**

<table>
<thead>
<tr>
<th>Average Std-Avg Max Value</th>
<th>Average Std-Avg Max Value</th>
<th>Average Std-Avg Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.03</td>
<td>6.444</td>
<td>2</td>
</tr>
<tr>
<td>0.01</td>
<td>12.025</td>
<td>1</td>
</tr>
<tr>
<td>3.20</td>
<td>1.066</td>
<td>16</td>
</tr>
</tbody>
</table>

**Waiting Area / Enq**

**Buffer Count / Enq**

<table>
<thead>
<tr>
<th>Range Count in To Maximum</th>
<th>Range Count in To Maximum</th>
<th>Range Count in To Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>730</td>
<td>1</td>
</tr>
</tbody>
</table>

**Total**

<table>
<thead>
<tr>
<th>731</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>731</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**8.3 Lock wait time**

Be sure to monitor the lock contention, because it can last several seconds or minutes and can greatly affect performance even though it happens only a few times.

IMS Performance Analyzer for z/OS: Fast Path DEDB Resource Contention Summary report can show you detailed information about lock contentions (Example 8-9 on page 321). The Area Name column shows the name of the area where lock contention occurs; the Counts column shows the number of locks; the Elap/Count column shows the average elapsed time to wait per request; and the Max IWAIT column shows the maximum elapsed time to wait per request.
Example 8-9  IMS Performance Analyzer for z/OS: Fast Path DEDB Res. Contention Summary

Report from 09Jun2012 14.25.56.36 IMS 12.1.0 IMS Performance Analyzer 4.3 Report to 09Jun2012 14.30.06.71
Fast Path DEDB Resource Contention Summary

From 09Jun2012 14.26.11.74 To 09Jun2012 14.29.21.57 Elapsed= 0 Hrs 0 Mins 09.836.240 Secs

**** CI Lock IWAIT ****
Sharing Types:

<table>
<thead>
<tr>
<th>Area</th>
<th>Sharing</th>
<th>Name</th>
<th>Type</th>
<th>Counts</th>
<th>Elap/Count</th>
<th>Max IWAIT</th>
<th>Pct Tot</th>
<th>Pct Tot</th>
<th>A : Area / Non Level Share</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>DB23AR0</td>
<td>A</td>
<td>3</td>
<td>3.313</td>
<td>0.466</td>
<td>5.498</td>
<td>9.0%</td>
<td>0.05%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB23AR1</td>
<td>A</td>
<td>4</td>
<td>2.222</td>
<td>0.551</td>
<td>3.308</td>
<td>12.12%</td>
<td>0.04%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB23AR3</td>
<td>A</td>
<td>1</td>
<td>4.871.974</td>
<td>0.000</td>
<td>4.871.974</td>
<td>3.0%</td>
<td>24.50%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB23AR4</td>
<td>A</td>
<td>1</td>
<td>0.257</td>
<td>0.000</td>
<td>0.257</td>
<td>3.03%</td>
<td>0.00%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB23AR5</td>
<td>A</td>
<td>11</td>
<td>1.358.286</td>
<td>1.620</td>
<td>4.981.761</td>
<td>33.33%</td>
<td>75.15%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D001AR0</td>
<td>A</td>
<td>13</td>
<td>3.880</td>
<td>0.499</td>
<td>6.863</td>
<td>39.39%</td>
<td>0.25%</td>
</tr>
<tr>
<td>** Total</td>
<td></td>
<td></td>
<td></td>
<td>33</td>
<td>602.504</td>
<td>2.668</td>
<td>4.981.761</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

**** Area Lock IWAIT ****
Sharing Types:

<table>
<thead>
<tr>
<th>Area</th>
<th>Sharing</th>
<th>Name</th>
<th>Type</th>
<th>Counts</th>
<th>Elap/Count</th>
<th>Max IWAIT</th>
<th>Pct Tot</th>
<th>Pct Tot</th>
<th>A : Area / Non Level Share</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>BANKC01</td>
<td>C</td>
<td>17</td>
<td>68.036</td>
<td>2.828</td>
<td>837.022</td>
<td>60.71%</td>
<td>84.82%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BANKC00</td>
<td>C</td>
<td>11</td>
<td>18.813</td>
<td>0.129</td>
<td>22.795</td>
<td>39.29%</td>
<td>15.18%</td>
</tr>
<tr>
<td>C  :  2 IRLM Block Level Share</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**** CI Lock IWAIT ****
Sharing Types:

<table>
<thead>
<tr>
<th>Range</th>
<th>Sc Mil Mic</th>
<th>Count in Range</th>
<th>Elap/Count</th>
<th>Max IWAIT</th>
<th>Counts</th>
<th>IM Elp</th>
<th>A : Area / Non Level Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>Count in Range</td>
<td>Elap/Count</td>
<td>Max IWAIT</td>
<td>Counts</td>
<td>IM Elp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>--------------</td>
<td>------------</td>
<td>-----------</td>
<td>--------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.000</td>
<td>1</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.000</td>
<td>2</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.000</td>
<td>4</td>
<td>4.000</td>
<td>4.000</td>
<td>4.000</td>
<td>4.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.000</td>
<td>8</td>
<td>8.000</td>
<td>8.000</td>
<td>8.000</td>
<td>8.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>16</td>
<td>16.000</td>
<td>16.000</td>
<td>16.000</td>
<td>16.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32.000</td>
<td>32</td>
<td>32.000</td>
<td>32.000</td>
<td>32.000</td>
<td>32.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64.000</td>
<td>64</td>
<td>64.000</td>
<td>64.000</td>
<td>64.000</td>
<td>64.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.000</td>
<td>128</td>
<td>128.000</td>
<td>128.000</td>
<td>128.000</td>
<td>128.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256.000</td>
<td>256</td>
<td>256.000</td>
<td>256.000</td>
<td>256.000</td>
<td>256.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>** Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**** Area Lock IWAIT ****
Sharing Types:

<table>
<thead>
<tr>
<th>Range</th>
<th>Sc Mil Mic</th>
<th>Count in Range</th>
<th>Elap/Count</th>
<th>Max IWAIT</th>
<th>Counts</th>
<th>IM Elp</th>
<th>A : Area / Non Level Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>Count in Range</td>
<td>Elap/Count</td>
<td>Max IWAIT</td>
<td>Counts</td>
<td>IM Elp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>--------------</td>
<td>------------</td>
<td>-----------</td>
<td>--------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.000</td>
<td>1</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.000</td>
<td>2</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.000</td>
<td>4</td>
<td>4.000</td>
<td>4.000</td>
<td>4.000</td>
<td>4.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.000</td>
<td>8</td>
<td>8.000</td>
<td>8.000</td>
<td>8.000</td>
<td>8.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>16</td>
<td>16.000</td>
<td>16.000</td>
<td>16.000</td>
<td>16.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32.000</td>
<td>32</td>
<td>32.000</td>
<td>32.000</td>
<td>32.000</td>
<td>32.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64.000</td>
<td>64</td>
<td>64.000</td>
<td>64.000</td>
<td>64.000</td>
<td>64.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.000</td>
<td>128</td>
<td>128.000</td>
<td>128.000</td>
<td>128.000</td>
<td>128.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256.000</td>
<td>256</td>
<td>256.000</td>
<td>256.000</td>
<td>256.000</td>
<td>256.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>** Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**** CI Lock IWAIT ****
Sharing Types:

<table>
<thead>
<tr>
<th>Range</th>
<th>Sc Mil Mic</th>
<th>Count in Range</th>
<th>Elap/Count</th>
<th>Max IWAIT</th>
<th>Counts</th>
<th>IM Elp</th>
<th>A : Area / Non Level Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>Count in Range</td>
<td>Elap/Count</td>
<td>Max IWAIT</td>
<td>Counts</td>
<td>IM Elp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>--------------</td>
<td>------------</td>
<td>-----------</td>
<td>--------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.000</td>
<td>1</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.000</td>
<td>2</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td>2.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.000</td>
<td>4</td>
<td>4.000</td>
<td>4.000</td>
<td>4.000</td>
<td>4.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.000</td>
<td>8</td>
<td>8.000</td>
<td>8.000</td>
<td>8.000</td>
<td>8.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.000</td>
<td>16</td>
<td>16.000</td>
<td>16.000</td>
<td>16.000</td>
<td>16.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32.000</td>
<td>32</td>
<td>32.000</td>
<td>32.000</td>
<td>32.000</td>
<td>32.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64.000</td>
<td>64</td>
<td>64.000</td>
<td>64.000</td>
<td>64.000</td>
<td>64.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.000</td>
<td>128</td>
<td>128.000</td>
<td>128.000</td>
<td>128.000</td>
<td>128.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256.000</td>
<td>256</td>
<td>256.000</td>
<td>256.000</td>
<td>256.000</td>
<td>256.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>** Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Chapter 8. Fast Path performance considerations 321
8.4 VSO and SVSO

Virtual Storage Option (VSO) and Shared VSO (SVSO) are a high-performance option for DEDBs and the preferred alternative to MSDBs for holding data in memory. IMS offers the user the best features of DEDBs and the response time and parallelism of MSDBs.

8.4.1 Virtual Storage Option (VSO)

VSO data resides in a z/OS data space in virtual storage, so response times for DL/I calls are close to memory access times. The definition, access, and operations on a DEDB using the VSO function are exactly the same as they are for an ordinary DEDB.

In this section, we provide performance tips related to the following items:

- Segment level locking
- Lock contention
- I/O reduction
- VSO output threads
- VSO PRELOAD options
- IMS system checkpoint

Segment level locking

Locking at the segment level is implemented for any VSO DEDB whose characteristics exactly match those of an MSDB:

- Root-only hierarchy
- Fixed length segment
- PCB PROCOPT=G or R
- VSO option in DBRC
- No compression

If you implement the VSO option for an existing DEDB, be sure you consider modifying the DEDB, when possible, so that it matches the segment level locking requirements. However, taking an existing DEDB and modifying it so that it meets the requirements of segment level locking might not be possible.

Lock contention

One of the benefits obtained from a VSO DEDB when compared to a non-VSO DEDB is that a CI lock held for updating can be released earlier because of the differences in the output thread processing:

- With a non-VSO DEDB, a CI lock has to be held until the CI is written to DASD. Because Fast Path writes out the CIs asynchronously for performance reasons, the CI lock can be held for a relatively long time.
- With VSO areas, the lock is released as soon as the updated CI is copied to the data space, which occurs early in sync point phase 2. This way considerably reduces lock contention.
Table 8-1 summarizes the lock level in effect for every case.

### Table 8-1 Locking level for GU and GN DL/I calls on VSO DEDB

<table>
<thead>
<tr>
<th>PROCOPT</th>
<th>Segment level locking</th>
<th>VIEW=MSDB</th>
<th>Lock level</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>N</td>
<td>N</td>
<td>EXCLUSIVE</td>
</tr>
<tr>
<td>A</td>
<td>N</td>
<td>Y</td>
<td>READ</td>
</tr>
<tr>
<td>A</td>
<td>Y</td>
<td>N</td>
<td>EXCLUSIVE</td>
</tr>
<tr>
<td>A</td>
<td>Y</td>
<td>Y</td>
<td>EXCLUSIVE</td>
</tr>
<tr>
<td>G</td>
<td>N</td>
<td>N</td>
<td>READ</td>
</tr>
<tr>
<td>G</td>
<td>N</td>
<td>Y</td>
<td>READ</td>
</tr>
<tr>
<td>G</td>
<td>Y</td>
<td>N</td>
<td>READ</td>
</tr>
<tr>
<td>G</td>
<td>Y</td>
<td>Y</td>
<td>READ</td>
</tr>
<tr>
<td>GR</td>
<td>N</td>
<td>N</td>
<td>EXCLUSIVE</td>
</tr>
<tr>
<td>GR</td>
<td>N</td>
<td>Y</td>
<td>READ</td>
</tr>
<tr>
<td>GR</td>
<td>Y</td>
<td>N</td>
<td>EXCLUSIVE</td>
</tr>
<tr>
<td>GR</td>
<td>Y</td>
<td>Y</td>
<td>READ</td>
</tr>
</tbody>
</table>

If you migrate an MSDB to VSO DEDB, be sure to include VIEW=MSDB in the corresponding PCBs, to avoid possible deadlock situations and performance impact.

If you implement the VSO option for an existing non-VSO DEDB, remember that if you code VIEW=MSDB in the PCB, the READ locks are released soon after the GU calls complete. This behavior is quite different from the non-VSO DEDB method, and your application programs might not be prepared for this event or they might experience an integrity exposure.

With VIEW=MSDB PCBs holding locks on VSO DEDBs, as MSDBs do, and locking at the segment level, the difference between VSO DEDB locking rules and the ones used with MSDBs is not significant. Therefore, from a lock contention and parallelism point of view, VSO DEDBs are comparable to MSDBs and considerably better than non-VSO DEDBs.

The FLD call works for a VSO in the same way as for an MSDB. That is, FLD calls are processed at sync point and are sorted by resource.

### I/O reduction

Some IMS systems have DEDB areas that experience high I/O rates. Implementing such areas with VSO provides major benefits, because the read I/Os are eliminated and the write I/Os are optimized; updates to a CI from multiple transactions are applied with a single I/O. We strongly suggest that you implement VSO for small and highly volatile DEDBs. If your system has small and highly volatile databases implemented as MSDBs, we suggest that you migrate them to VSO DEDBs also.

### VSO output threads

Periodically, all updated CIs are written out from the data spaces to the area data sets during a process that is also referred to as output thread (OThread).

This OThread process for VSO DEDB is more efficient than the non-VSO DEDB process. Locks on CIs are released earlier during sync point processing, and the I/Os are performed in
chains of 200 KB. Also, multiple updates to the same CI are written back to DASD only once. See Table 8-2 for a comparison of OTHREAD processing.

At IMS system checkpoint, the OTHREAD process is started too, but it is also asynchronous, and so it does not affect IMS system checkpoint duration. If you migrate your MSDB to VSO DEDB and specify your system to no longer use MSDBs (MSDB parameter left to null in DFSPBxx member of IMS.PROCLIB), then your IMS system checkpoint elapsed time should be reduced.

Table 8-2  OTHREAD processing for VSO and non-VSO DEDBs

<table>
<thead>
<tr>
<th>Non-VSO DEDB</th>
<th>VSO DEDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updated CI in FP buffer</td>
<td>Updated CI in FP buffer</td>
</tr>
<tr>
<td>SYNC</td>
<td>SYNC</td>
</tr>
<tr>
<td></td>
<td>CI Write to Data Space</td>
</tr>
<tr>
<td></td>
<td>Lock Release</td>
</tr>
<tr>
<td>Asynchronous Physical Logging (OTHREAD)</td>
<td>Asynchronous Physical Logging</td>
</tr>
<tr>
<td>Database Data set Write I/Os</td>
<td>Asynchronous OTHREAD (Wait for Logging)</td>
</tr>
<tr>
<td>Lock Release</td>
<td>Database Data set Write I/Os</td>
</tr>
</tbody>
</table>

**VSO PRELOAD options**

Any DEDB can be moved to virtual storage by specifying the VSO keyword on the DBRC commands: `INIT.DBDS` or `CHANGE.DBDS`.

VSO areas are mapped linearly to z/OS data spaces. A data space can contain the data of multiple areas, but the data of an area cannot be divided into multiple data spaces. During IMS startup, two 2 GB data spaces are acquired:

- One has the disabled reference (DREF data space) option specified, which means that its pages can reside in memory but are never paged out to DASD.
- The other data space does not have the DREF option (non-DREF data space) and might experience paging.

An area can be defined to DBRC with either the PRELOAD or the NOPREL attributes:

- Any area defined with the PRELOAD option is read into the DREF data space following the IMS initialization system checkpoint. Only the direct part of the area is loaded.
- If an area has the NOPREL option, then each direct part CI is allocated a position in the non-DREF but no automatic load takes place. Instead, the first time a CI is requested after the area open, IMS reads it from DASD and copies it to the data space.

SDEP CIs are never loaded into the data space.

CIs from PRELOAD areas always remain in central storage. For medium or small areas with a high access rate, where most of the CIs in the area are accessed with similar frequency (there are zero or negligible hot spots), we suggest implementing the PRELOAD option.

When choosing the PRELOAD option, know that PREOPEN is implicitly assumed for this option. This means that VSAM area data sets are allocated and opened immediately when IMS starts or a `/START AREA` command is issued.
CIs from NONPRELOAD areas can be paged out to page data sets if there is a storage constraint. If you have big VSO areas where only a small part of the area is actually accessed, implement the NOPREL option so that only space for needed CIs is allocated.

VSO DEDB performance will more likely degrade with paging than non-VSO DEDBs doing I/Os to the actual data sets. Be sure to make use of the VSO option for DEDBs, that is, for those databases where the area is not too big and that need high performance.

**IMS system checkpoint**

When VSO DEDBs databases are used, the content of the data space is written to DASD at IMS system checkpoint time only for those VSO DEDBs that had update processing. To do so, an output thread is started at every system checkpoint. However, this output thread is an asynchronous process, and its elapsed time does not affect the IMS system checkpoint duration.

A considerable reduction in the IMS system checkpoint elapsed time (tens of seconds, depending how large the MSDBs are) can be achieved by migrating MSDBs to VSO DEDBs and specifying the MSDB parameter, in the DFSPBxx member of IMS.PROCLIB, as null.

During IMS system checkpoint, transaction processing almost stops, which has several negative consequences. If this is the case for your system, we suggest that you migrate from MSDBs to VSO DEDBs.

### 8.4.2 Shared Virtual Storage Option (SVSO)

Block-level data sharing of VSO DEDB areas allows multiple IMS subsystems to concurrently read and update VSO DEDB data. VSO DEDBs that support block-level data sharing are commonly known as *Shared VSO (SVSO) DEDBs*.

The following main elements participate in the SVSO implementation:

- **Cache structures in the coupling facility**
  - Store-in cache structures are used to contain copies of the control intervals accessible to each IMS in the data sharing group.

- **Local cache buffers**
  - Each IMS has a set of buffers that contains a local copy of the control intervals.

- **Permanent storage**
  - DEDB area data sets are needed to contain a non-volatile copy of the data in DASD.

When the cache structures contain the data required, IMS reads the data from CF. Else, when they do not, IMS reads from DASD and puts the data into CF. When updating the shared VSO DEDB, IMS writes the data on CF and the changed CIs are written into DASD periodically.
Local buffer pool definitions
The private local buffer pools can be defined by means of the DEDB statement of member DFSVSMxx on IMS.PROCLIB. Multiple DEDB statements can be used to define various buffer pools, each one with different attributes. The following attributes can be specified in the DEDB statement:

- Name of the buffer pool
- Number of buffers for the primary and secondary allocations
- Maximum number of buffers that can be allocated
- Buffer pool buffer size
- Whether to assign the buffer pool to a certain area
- LKASID option

An area can use a certain buffer pool if the CI size for the area matches the buffer pool buffer size, it has the same LKASID option specified in the RECON, and the buffer pool is not implicitly assigned to another area.

Two or more areas with the same characteristics can also share a buffer pool.

When starting an area, if no buffer pool is defined or no available buffer pool matches the CI size for the area, IMS creates a default buffer pool for it. Default buffer pools can also be shared among different areas.

Consider the following suggestions when you define local buffer pools:

- Do not allow the default buffer pools to be created. Include at least one DEDB statement for each different CI size in the DFSVSMxx member of IMS.PROCLIB.
- Specify a number of buffers large enough to hold those needed for the maximum number of concurrent requests. Specify them as primary allocation buffers, so that they are page-fixed. Then, specify a secondary allocation to cover unexpected high load situations, and a maximum number of buffers large enough so that the maximum is never reached (if a buffer is not available, IMS waits for a buffer to become free). Paging of primary or secondary buffers does not occur. When secondary buffers are allocated, they are page-fixed.

If your applications use PCB with PROCOPT=GO to access SVSO areas, Fast Path might steal local buffers. That means that extra buffers must be defined for the local buffer pool.

Also, if the buffer pool is not large enough, the LKASID option can be ineffective. So, it is important for the primary allocation to be large enough.

A buffer containing committed changes that was not written to the coupling facility (CF) cannot be reused until an output thread writes the buffer to the CF and completes. Therefore, a lack of OTHREADs can result in more buffers being needed for the local buffer pools. Set the OTHR parameter to the maximum value (32,767) if there is any doubt about how to fine-tune their number, because they are cheap to define.

- If the application’s access rate pattern for different SVSO areas is similar, you might allow two or more areas to share the same buffer pool if they have the same CI size and LKASID options, which is better from an administration point of view. However, if the two areas both have high access rates, you should give the high-access-rate areas dedicated buffer pools and allow only the low-access-rate areas to share pools.

If one of the areas sharing a pool is accessed much more than the others, then it consumes most of the buffers, and the performance of the accesses to the other areas can be degraded. If the access profiles are not similar for the different areas in your system, we suggest that you assign dedicated buffer pools for each area.

- Normally, we prefer the LKASID option unless the area is highly updated from different IMS instances in the data sharing group.
For those areas where the read/write ratio is high, we prefer assigning to them a dedicated buffer pool and enabling the LKASID option, so that the major part of the reads are resolved within the local IMS (without accessing the CF).

We also suggest that you monitor the usage of these buffer pools and adjust the allocation values and the LKASID option appropriately.

**LKASID option**

IMS maintains the buffers in the local buffer pool that are chained off three or four queues:

- **Available queue**
  These buffers are available for use. It does not matter what they contain, because they are considered as empty buffers.

- **Requestor queue**
  These buffers are in use by an application program. These buffers count toward the NBA/OBA limit.

- **Output Thread queue**
  These buffers have committed updates that are waiting for an OTHREAD to be written to the CF.

- **LKASID queue (optional)**
  This queue applies only for pools with the LKASID option, which we discuss in this section. When the LKASID option is the choice, the local buffers that are used by the application are not returned to the available queue after application sync point. Instead, they are chained off the lookaside queue.

We recommend that you select the LKASID option. It provides better performance, because CF accesses are saved, especially if the read/write ratio is high and most of the requests for CI are resolved locally without accessing the CF.

To take advantage of the LKASID option, the buffer pool must be large enough (primary plus secondary) to hold all of the current requests for buffers from executing transactions (requestor queue), for scheduled but incomplete output threads (output thread queue), and with enough additional buffers to provide a sufficient number of buffers that remain on the lookaside queue.

How efficiently the LKASID option performs depends on the ratio of valid hits and number of searches. If this ratio is not a big number, searching the lookaside queue might be worth doing, and you might consider choosing NOLKASID. This ratio can be obtained by means of either IMS Performance Analyzer VSO Statistics report, the `/DIS POOL FPDB` command, or the DBFULTA0 utility.

The LKASID option can be inefficient for three reasons:

- A shortage in the buffer pool
- The application database access profile
- INOPREL area and CF structure size shortage

Be aware that the Valid ratio shown in the output for the `/DIS POOL FPDB` command has a slightly different meaning from the Hits Valid ratio shown in the IMS Performance Analyzer VSO Statistics report. For the command, Valid means the percentage of times a buffer found in the pool had valid data. Therefore, you must use the displayed values for Hits and Valid together to obtain the ratio of efficiency for the LKASID option. For example, if Hits is 40%, and Valid is 75%, a buffer was found in the pool 40% of the time. Of that 40%, 75% of the buffers found had valid data, that is, 30% of the requests found buffers on the LKASID queue.
with valid data. So, IMS had to read data from CF approximately 70% of the time. The Hits Valid ratio shown in IMS Performance Analyzer VSO Statistics report is 30%.

DBFULTA0 gives the number of cross-invalidated hit buffers instead, which is the percentage of invalid buffers from those that were found in the lookaside queue.

Monitoring the local buffer pools is necessary to set the local buffer pool allocations and the LKASID option appropriately.

**Coupling facility structure definitions**

Each VSO DEDB cache structure in the shared storage of a coupling facility represents one or more VSO DEDB areas. A VSO DEDB cache structure can be a single-area structure or a multi-area structure. A single-area structure can contain data from only one DEDB area. A multi-area cache structure can hold data from multiple areas. Both single-area and multi-area cache structures conform to the characteristics of the areas for which they are created. Both types of cache structures are also non-persistent: they are deleted after you close the last area connected to them.

To determine the structure size, you can use the z/OS coupling facility structure sizer tool (CFSizer). CFSizer is a web-based application that calculates the structure size based on the input data that you provide. To use the CFSizer tool, go to the following web page:


**PRELOAD or NOPREL option**

The PRELOAD option causes the whole area to be read from DASD and written to the CF as soon as IMS restarts or an /STA AREA command is issued. When this is the choice, the CF structure size must be large enough to contain all the CIs of the direct portion of the area; otherwise, the area cannot be started.

As a general rule, we prefer to use NOPREL and specify a CF structure size large enough to contain the CIs of the area that are actually active. If there are hot spots in the database, then as soon as the applications start accessing those CIs, they are placed in the CF and remain there for the remainder of the IMS session (or until the area is closed or a /VUNLOAD command is issued). No unnecessary space is wasted in the CF with NOPREL and an appropriate CF structure size. The first reference to each CI might have a worse response time because DASD accesses are also needed (CF structures are just a cache).

PRELOAD is used only if almost all the CIs in the area contain data, they are all referenced with similar frequency, and it is required to have good performance from the start.

Also, you might consider using PRELOAD for those areas with medium load but where extremely high performance is a requirement and all the CIs are randomly accessed. In this case, PRELOAD maintains all the CIs in the CF.

The preloading process differs slightly between multi-area structure (MAS) and single area structure (SAS). MAS is the option introduced in IMS V9, which enables multiple DEDB areas that have same CI size to share a single coupling facility structure. In the case of MAS, IMS does not check a cache structure size for preloading MAS SVSO areas. However, in the case of SAS, IMS checks a cache structure size for SAS SVSO, when the area is opened for preloading. Although this checking occurs, IMS sometimes is unable to load whole CI images because of inaccurate calculation of the checking macro.
Block level locking and root-only DEDBs

As the following conditions are fulfilled, local VSO DEDB areas use segment level locking, which makes them equivalent to MSDBs:

▶ ROOT segment only
▶ Fixed length segment
▶ PROCOPT=G or R

This behavior means that two segments can be accessed simultaneously from separate dependent regions without incurring lock contention even if they were in the same CI and the access intent was exclusive.

With SVSO areas, the locking unit is the CI. If, after moving from local VSO to SVSO, you experience an increase in lock contention, try to reduce contention problems at a maximum in the following ways:

▶ Reduce the CI size
  
  Different CI sizes can be specified for each area and local buffer pool. Try to adjust the CI size to the segment size where possible, so that only one segment fits in a CI.

▶ Expand the area
  
  You can also increase the size of the area to have more available RAP CIs. This way causes segments to be more sparsely distributed in the database and reduces the chances of having two or more segments in the same CI. If you do this step, be aware that, if PRELOAD option is the choice, the CF structure size must be appropriately increased. Increasing the area size can result in many empty CIs, therefore, consider whether PRELOAD is the best option in this case.

We also suggest that you adjust the PROCOPT values of the database PCBs to the actual type of access that the programs perform. If you overly specify the PROCOPTs, you can experience lock-contention problems. The only time locking differs between VSO and SVSO is if VSO is eligible for segment level locking. If you have lock contention with VSO, you will have the same contention with SVSO. The difference is that with SVSO, you use the IRLM to resolve the contention. If there is contention, then using the IRLM to resolve it is not appreciably less efficient than using PI to resolve it if data sharing is not used at all.

If you implement the VSO option for an already existing SHARELEVEL(3) DEDB, no particular increase in lock contention is expected to occur.
8.4.3 Monitoring VSO and SVSO performance

The VSO Activity Summary section of IMS Performance Analyzer for z/OS: VSO Statistics report provides information that can be used to monitor the performance of the VSO areas. Example 8-10 shows this report. You can determine how well VSO is performing by comparing the number of requests that are performed against the data space to the number of requests that need DASD access.

**Example 8-10  IMS Performance Analyzer for z/OS: VSO Statistics (VSO Activity Summary)**

You can also monitor the areas that are actually loaded into data spaces and the amount of storage they use. The AREASIZE column shows the number of 4K pages reserved for a certain area in the data space. This amount is the maximum space needed to allocate the whole area (all its CIs) and depends on the DBD. If the area is not preloaded, the actual amount of storage allocated for the area can be sensibly less.

**Example 8-11 shows a /DIS FPV command.**

**Example 8-11 /DIS FPV command**

IM1BDIS FPV
DF544441 DISPLAY FROM ID=IM1B
DATASPACE MAXSIZE(4K) AREANAME AREASIZE(4K) OPTION
001 524238 DREF
NO AREAS LOADED INTO DREF DATASPACE 001.
+ AREANAME STRUCTURE ENTRIES CHANGED AREA CI? POOLNAME OPTIONS
+ AREAWH00 IMOB_AREAWH00A 0000075 0000020 00000075 WAREPOOL PREO, PREL
+ AREAWH00 IMOB_AREAWH00B 0000075 0000020 00000075 WAREPOOL PREO, PREL
+ AREAIT01 IMOB_AREAIT01A 0002448 0000000 00002520 TEMPOOL PREO, PREL
+ AREAIT01 IMOB_AREAIT01B 0002448 0000000 00002520 TEMPOOL PREO, PREL
+ AREAD01 IMOB_AREAD01A 0000440 0000185 00000440 DISTPOOL PREO, PREL
+ AREAD01 IMOB_AREAD01B 0000440 0000185 00000440 DISTPOOL PREO, PREL
*2006271/171202*
8.5 Fast Path 64-bit buffer manager

The Fast Path 64-bit buffer manager autonomically controls the number and size of Fast Path buffer pools, including buffer pools for DEDBs, MSDBs, and system services. It also places the DEDB buffer pools above the bar in 64-bit common storage in IMS 12, which was placed in ECSA in the previous versions, so it provides ECSA constraint relief.

8.5.1 Fast Path 64-bit buffer manager overview

If you are using the Fast Path 64-bit buffer manager, IMS creates and manages the Fast Path buffer pools for you and places DEDB buffers in 64-bit storage. The buffers are placed in virtual storage above 2 GB bar. The 64-bit buffer manager creates multiple subpools with different buffer sizes. It creates an initial allocation of subpools based on the number of areas of each CI size, and automatically creates more buffers in a subpool as they were needed. Obviously this approach is possible only when you have the z/Architecture enabled.

When the Fast Path 64-bit manager is enabled, you do not need to design DEDB buffer pools. The parameters that define the sizes of buffer pools, DBBF, DBFX, and BSIZ, are ignored when you use 64-bit buffer manager.

With IMS 12, the user can specify the initial amount of 64-bit storage used for the buffer pool. Buffer pools are pre-expanded, that is, expanded in anticipation of future needs. They are compressed when the use of a subpool drops. IMS 12 moved some buffers that were still in ECSA to 64-bit storage, which are the buffers that are used for SDEP inserts and FLD calls.

When the 64 bit buffer manager is used, the use of OBA buffers is not serialized. Multiple dependent regions or threads can be using their OBA allocations at the same time. This way eliminates a potential bottleneck in buffer use. In practice, it means that each region or thread can be using the number of buffers equal to its NBA plus OBA specification.

8.5.2 Monitoring of Fast Path 64-bit buffer usage

You can capture usage statistics for the Fast Path 64-bit buffers by issuing the following command:

```
UPDATE IMS SET(LCLPARM(FPBP64STAT(Y)))
```

IMS captures the usage statistics for each unit of work in a dependent region, and writes the statistics to the online log data set as X’5945’ log records, which are mapped by the DBFL5945 and DBFBPND6 macros. These log records are read and processed by the Fast Path Log Analysis utility (DBFULTA0).
The type-2 QUERY POOL command is enhanced to show information about the 64-bit buffer manager. Example 8-12 and Example 8-13 shows responses. The SHOW(STATISTICS) option is available in IMS 12 or later.

**Example 8-12  Response of QUERY POOL TYPE(FPBP64) SHOW(ALL) command**

<table>
<thead>
<tr>
<th>Subpool MbrName</th>
<th>CC</th>
<th>Size</th>
<th>Type</th>
<th>Status</th>
<th>T_id</th>
<th>Tot_Buf</th>
<th>Buf_Use</th>
<th>Buf_Avl</th>
<th>Buf_A</th>
<th>%Use</th>
<th>Ext</th>
<th>Qui_Buf</th>
<th>Buf_Q</th>
<th>EPVT_Tot</th>
<th>EPVT_T</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBF_MAXB I12A</td>
<td>G</td>
<td>88</td>
<td>0</td>
<td>88</td>
<td>0</td>
<td>1K</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBF_TOTB I12A</td>
<td>512</td>
<td>Tot</td>
<td>10</td>
<td>32</td>
<td>0</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>156</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFC0001 I12A</td>
<td>Base</td>
<td>15</td>
<td>32</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>156</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFC0001 I12A</td>
<td>1024</td>
<td>Tot</td>
<td>10</td>
<td>16</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>156</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFS0001 I12A</td>
<td>Base</td>
<td>15</td>
<td>32</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>156</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFS0001 I12A</td>
<td>1024</td>
<td>Tot</td>
<td>10</td>
<td>16</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>156</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Example 8-13  Response of QUERY POOL TYPE(FPBP64) SHOW(STATISTICS) command**

<table>
<thead>
<tr>
<th>Buf_Size</th>
<th>MbrName</th>
<th>CC</th>
<th>SPT</th>
<th>Tot_Buf</th>
<th>Buf_Use</th>
<th>Buf_Avl</th>
<th>%Use</th>
<th>HWM</th>
<th>EPVT_Tot</th>
<th>ECSA_Tot</th>
<th>64b_Tot</th>
<th>64b_Buf</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td>88</td>
<td>0</td>
<td>88</td>
<td>0</td>
<td>1K</td>
<td>71K</td>
<td>32K</td>
<td></td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>I12A</td>
<td>0</td>
<td>C</td>
<td>32</td>
<td>0</td>
<td>32</td>
<td>0</td>
<td>156</td>
<td>15K</td>
<td>16K</td>
<td>16K</td>
<td>16K</td>
</tr>
<tr>
<td>1024</td>
<td>I12A</td>
<td>0</td>
<td>C</td>
<td>16</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>156</td>
<td>8K</td>
<td>16K</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note that the 64-bit buffer usage cannot be reported by using the type-1 command, such as in Example 8-14.

**Example 8-14  /DIS POOL FPDB command response**

/DIS POOL FPDB
DFS44451 CMD FROM MCS/E-MCS CONSOLE USERID=IMS008: DIS POOL FPDB I12A
DFS44441 DISPLAY FROM ID=I12A 478
FPDB BUFFER POOL:

** NO DEDB PRIVATE POOLS DEFINED **
Example 8-15  IMS Performance Analyzer for z/OS: Internal Resource Usage report, Fast Path 64-bit Buffer Statistic


Example 8-16  Fast Path Log Analysis utility (DBFULTA0): DETAIL LISTING OF EXCEPTION TRANSACTIONS

DBFULTA0 was also enhanced to report the usage of FP 64-bit buffer. When you write the keyword FPBP64 in the control statement, you can obtain the information, such as in Example 8-16.
8.6 Fast Path secondary index (FPSI) and performance comparison with HDAM

Secondary indexing provides a way to meet the processing requirements of various applications. With secondary indexing, you can have an index based on any field in the database, not only the key field in the root segment.

IMS 12 supports secondary indexing on a primary DEDB to process a segment type in a sequence other than the one that is defined by the segment’s key.

The secondary index databases must be root-only VSAM-based HISAM or SHISAM databases. HISAM databases must be used if you are confronted with duplicate secondary index values. A SHISAM database is composed of one KSDS data set; a HISAM database can have overflow in an ESDS data set.

At IBM Silicon Valley Laboratory, a comparison test was run between HDAM with secondary index and DEDB with secondary index. The environment, scenarios, and results are reported in this section.

Test environment
The test environment configuration used the following hardware and software:

- **Hardware**
  - z196 processor, 2 CPs for 1-IMS, 3 CPs for TPNS
  - DASD DS8700

- **Software**
  - z/OS V1R12
  - IRLM V2.3
  - TPNS V3R5

Test scenarios
We used three scenarios:

- **Scenario 1:** Determine the performance improvement of running with DEDB with two secondary indexes.

  We use one area of DEDB, with two secondary indexes in the ROOT segment; the application updates DEDB roots randomly with 100% GHU/GHN/REPL calls and 10% issuing ISRT/DLET calls, which caused secondary index maintenance.

- **Scenario 2:** Determine the performance improvement of running with DEDB with secondary index case.

  We use a BMP to access DEDB with secondary index sequence (PROCDEQD), issuing GU to identify one particular primary key, and GHN/REPL next 10 records sequentially.

- **Scenario 3:** Determine the performance of running with DEDB with two and four secondary indexes, but not using the secondary indexes. Determine the base performance of running with DEDB without secondary index case.

  We use DEDB with two and four secondary indexes definition. The applications update DEDB roots randomly with GHU/GHN/REPL DL/I calls, which do not cause secondary index maintenance.
## Results

Table 8-3 shows results of scenario 1.

### Table 8-3  Scenario 1: HDAM/VSAM with two secondary indexes versus one area DEDB with two secondary indexes

<table>
<thead>
<tr>
<th>Metric</th>
<th>HDAM VSAM with two secondary indexes</th>
<th>DEDB with two secondary indexes</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETR</td>
<td>658</td>
<td>642</td>
<td>-16 (-2%)</td>
</tr>
<tr>
<td>CPU utilization</td>
<td>17%</td>
<td>14%</td>
<td>-3% (-18%)</td>
</tr>
<tr>
<td>ITR</td>
<td>3871</td>
<td>4586</td>
<td>+715 (+18%)</td>
</tr>
<tr>
<td>Processing time (in seconds)</td>
<td>0.008</td>
<td>0.013</td>
<td>+0.005 (+63%)</td>
</tr>
</tbody>
</table>

**I/O activity (per second)**

<table>
<thead>
<tr>
<th>Metric</th>
<th>HDAM VSAM with two secondary indexes</th>
<th>DEDB with two secondary indexes</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main DB</td>
<td>3800</td>
<td>2055</td>
<td>-1,745 (-46%)</td>
</tr>
<tr>
<td>Secondary index</td>
<td>1578</td>
<td>1496</td>
<td>-82 (-5%)</td>
</tr>
</tbody>
</table>

**DLI statistics (per second)**

<table>
<thead>
<tr>
<th>Metric</th>
<th>HDAM VSAM with two secondary indexes</th>
<th>DEDB with two secondary indexes</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>GHU</td>
<td>29.74%</td>
<td>29.86%</td>
<td></td>
</tr>
<tr>
<td>GHN</td>
<td>10.79%</td>
<td>10.41%</td>
<td></td>
</tr>
<tr>
<td>ISRT</td>
<td>6.31%</td>
<td>6.49%</td>
<td></td>
</tr>
<tr>
<td>DLET</td>
<td>6.32%</td>
<td>6.48%</td>
<td></td>
</tr>
<tr>
<td>REPL</td>
<td>21.58%</td>
<td>20.82%</td>
<td></td>
</tr>
</tbody>
</table>

Table 8-4 shows results of scenario 2.

### Table 8-4  Scenario 2: Access DEDB with secondary index sequence versus no secondary index sequence

<table>
<thead>
<tr>
<th>Metric</th>
<th>DEDB without secondary index</th>
<th>DEDB with secondary index (using PROCSEQD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed time</td>
<td>0:00:41</td>
<td>0:00:01</td>
</tr>
<tr>
<td>CPU</td>
<td>5.44%</td>
<td>2.22%</td>
</tr>
</tbody>
</table>

**I/O activity (per second)**

<table>
<thead>
<tr>
<th>Metric</th>
<th>DEDB without secondary index</th>
<th>DEDB with secondary index (using PROCSEQD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEDB</td>
<td>2,106.53</td>
<td>1.978</td>
</tr>
<tr>
<td>Secondary index</td>
<td>0</td>
<td>2.182</td>
</tr>
</tbody>
</table>

**DLI statistics (per second)**

<table>
<thead>
<tr>
<th>Metric</th>
<th>DEDB without secondary index</th>
<th>DEDB with secondary index (using PROCSEQD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB GU</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>DB GHU</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>DB GHN</td>
<td>534,990</td>
<td>10</td>
</tr>
<tr>
<td>DB REPL</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Total DLI DB Calls</td>
<td>535,001</td>
<td>21</td>
</tr>
</tbody>
</table>
Table 8-5 shows results of scenario 3.

Table 8-5  Scenario 3: Access DEDB with 0, 2, and 4 secondary indexes

<table>
<thead>
<tr>
<th>Metric</th>
<th>DEDB with 0 secondary indexes</th>
<th>DEDB with 2 secondary indexes</th>
<th>DEDB with 4 secondary indexes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETR</td>
<td>278.54</td>
<td>278.93</td>
<td>280.86</td>
</tr>
<tr>
<td>CPU</td>
<td>12%</td>
<td>13%</td>
<td>13%</td>
</tr>
<tr>
<td>ITR</td>
<td>2321.17</td>
<td>2145.62</td>
<td>2160.46</td>
</tr>
<tr>
<td>DLI statistics (per second)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GHU</td>
<td>2.33%</td>
<td>2.33%</td>
<td>2.33%</td>
</tr>
<tr>
<td>GHN</td>
<td>46.51%</td>
<td>46.51%</td>
<td>46.51%</td>
</tr>
<tr>
<td>REPL</td>
<td>48.84%</td>
<td>48.84%</td>
<td>48.84%</td>
</tr>
</tbody>
</table>

8.7 Fast Path log reduction

You might want to use asynchronous changed data capture, but do not want to write log records for the before-images. Prior to IMS 12, asynchronous changed-data capture wrote before- and after-image log records (X’99’). In IMS 12, you can specify that these before-image log records not be written. To specify this information, use the following values on the EXIT= parameter of the DBD or SEGM macro of DBDGEN:

- **DLET**: Writes the before-image log record for DLET calls. This value is the default and is also the action taken by previous versions of IMS.
- **NODLET**: Does not write the before-image log record for DLET calls.
- **BEFORE**: Writes the before-image log record for REPL calls. This value is the default and is also the action taken by previous versions of IMS.
- **NOBEFORE**: Does not write the before-image log record for REPL calls.

The benefit of this enhancement is to reduce logging volumes for asynchronous data capture for users who want only after-image log records.
DBRC

In this chapter, we provide background information to help familiarize or remind you about key aspects of DBRC and its components.

We also explore the main Database Recovery Control (DBRC) enhancements, which are included in IMS 12, that can be used to manage the Recovery Control (RECON) data sets, and affect DBRC.

We review the new DBRC functions that you can use to perform a deeper cleanup when removing obsolete information, and to better control change accumulation (CA) records.

We briefly explore use of parallel RECON access and its performance benefits, and the use of DBRC under Base Primitives Environment (BPE), and show the comparable performance for both execution environments.

This chapter contains the following topics:
- DBRC background information
- DBRC performance considerations
- DBRC RECON maintenance
- The CLEANUP command parameters
- DBRC parallel RECON access
- DBRC execution on Base Primitives Environment (BPE)
9.1 DBRC background information

DBRC forms an integral part of IMS. IMS relies on DBRC to record and manage recovery-related information about all aspects of IMS. DBRC keeps this information in a set of VSAM data sets that are collectively called the Recovery Control (RECON) data set. DBRC also provides information to IMS about how to proceed for certain IMS actions. DBRC specifically helps with the following tasks:

- Helps you ensure IMS system and database integrity by recording and managing information that is associated with the logging process.
- Assists IMS in the restart process by notifying IMS which logs to use for restart.
- Assists IMS to allow or prevent access to databases in data-sharing environments by recording and managing database authorization information.
- Facilitates database and log recovery as follows:
  - Controls the use and integrity of the information in the logs.
  - Records and maintains information about the databases and logs in the RECON data set.
  - Generates and verifies the job control language (JCL) for various IMS utility programs.
- Supports Extended Recovery Facility (XRF) by identifying (in the RECON data set) whether the subsystem is XRF-capable.
- Supports Remote Site Recovery (RSR) by containing the RSR complex definition in the RECON data set and providing other services that are associated with controlling RSR.
- Supports IMSplexes by notifying all DBRCs in the same IMSplex when one of the DBRCs performs a RECON data set reconfiguration.
- Provides enhanced database function support for the following items:
  - HALDB
  - Unrecoverable databases
  - Concurrent Image Copy
  - Image Copy 2
  - Fast Path VSO options
  - Online reorganization
9.2 DBRC performance considerations

There are a number of performance considerations with regard to DBRC and the management of the RECON data sets. *IMS Version 12: System Administration*, SC19-3020, provides detailed information about this topic. We highlight several major performance considerations with DBRC.

Recognize the need to reorganize the RECON data sets periodically. Many of the record keys in the RECON data set include the date and time. DBRC recording of IMS log and database activity can cause control interval (CI) and control area (CA) splits, which can degrade performance. In addition, deleting unnecessary records might not prevent the RECON data set from filling up because VSAM does not always reuse the space freed.

IMS Version 12 adds maintenance commands to help with removing records and maintaining the RECON data sets. When significant changes the RECON contents occur, review the RECON data set statistics as part of this record management.

9.2.1 Defining the RECON data sets

All information pertaining to DBRC is kept on a VSAM key-sequenced data set (KSDS). DBRC uses two RECON data sets to increase availability and recoverability. They contain identical information. The data sets are identified by the data definition (DD) names RECON1 and RECON2. We strongly suggest that you define a third RECON data set (RECON3) as a spare data set. This spare data set is empty until either an error occurs on one of the two active RECON data sets, or you issue the CHANGE.RECON REPLACE(XXX) command. Then, DBRC copies the good RECON to the spare data set.
Naturally, there are performance issues pertaining to VSAM and the RECON definition:

- Use the same index control interval (CI) size and data CI size for all RECON data sets. Ensure that the specified data CI size exceeds the index CI size by at least 2048 bytes, or RECON performance degrades.

- Carefully review space allocation of the RECON data sets.
  
  Make the three RECONs different sizes (make the spare the largest), so that they do not all run out of space at the same time.

- Use secondary allocation in the event that your RECON data set runs out of space.

- We suggest using the following keywords on the DEFINE CLUSTER command:
  
  - CONTROLINTERVALSIZE: Our suggestion is 8192 KB for data and 4096 KB for the index. Remember that the index CI size must be at least 2048 bytes smaller than the data size and that all index and data components must have the same CI size.
  
  - FREESPACE: Our suggestion is that you specify at least 50% free space, and you can lower this amount of free space when you are satisfied with the usage of the RECON data sets.

  Make the FREESPACE characteristics different on each RECON so that they do not take CI and CA splits at the same time.

  - KEYS: Must be specified as KEYS(32,0).
  
  - NONSPANNED: Use because DBRC always writes records smaller than the VSAM CI size. DBRC divides its own records into segments, each of which is always smaller than a single control interval and each of which is seen by VSAM as a complete physical record. VSAM record spanning is not used. Segmenting allows a logical RECON record to be as large as 16 MB, independent of the VSAM RECORDSIZE parameter.

  However, this approach does not mean that the RECON records can grow indefinitely. A fast growing PRILOG record can indicate incorrect image copy frequency, because of an unexpected increased volume of logging. So, readjusting your threshold values might be useful:

  - LOGALERT(dsnum,volnum), which triggers the DSP0287W warning message
  
  - SIZALERT(dsnum,volnum,percent), which triggers the DSP0387W warning message or the DSP0007I informational message

  These two parameters give you the opportunity to react, such as deleting inactive logs, executing Image Copy, or ensuring that log compression is not inhibited, and to correct the situation before the IMS system abends. The two parameters give you time to determine what is causing the extremely large PRILOG record. You probably want to issue the messages much earlier than the default values do, so the value settings might need to be readjusted. You can find examples of more reasonable values in *IMS Version 8 Implementation Guide: A Technical Overview of the New Features*, SG24-6594.

  - RECORDSIZE: Be sure it is CI size, minus 7 bytes. If the CI size was defined as 8192 bytes, then RECORDSIZE (4086,8185) applies.
  
  - SHAREOPTIONS: Must be specified as SHAREOPTIONS (3,3).

  - SPEED: Should be used because the initial load is faster.

  - NOWRITECHECK: Should be used because IMS manages dual copies of the RECON, which eliminates the need for writing checks.
Use dynamic allocation for all your RECON data sets across all your IMS systems. RECON data set names must not be coded in JCL. It is also easier to deallocate the data set if required.

Ensure that the RECON data sets are monitored and reorganized when required. When doing a reorganization, always cycle through all RECON data sets to make sure that RECON1 and RECON2 are always indicated as COPY1 and COPY2. Remember to back up your RECON data sets before doing any reorganization.

To avoid deadlock situations, carefully consider the placement of RECON data sets that are shared among multiple processors. During a physical open operation, DBRC reserves RECON1, RECON2, and RECON3. DBRC determines which are available and which are Copy1, Copy2, and spare. DBRC then closes and dequeues the spare (if it exists) and any unusable RECON data sets. So, during the use of DBRC, two RECON data sets are reserved most of the time. DBRC always reserves both RECON data sets in this order: RECON1 and RECON2. If RECON1 and RECON2 are specified consistently throughout jobs, DBRC does not encounter deadlock.

9.2.2 Resolving data set contention problems

The availability and performance of the RECON data sets are dependent on whether global resource serialization (GRS) is running in a star or ring configuration. GRS provides methods to convert a RESERVE into an ENQ request. The ENQ or DEQ identifies a resource by its symbolic name, containing major name, minor name, and scope. To ensure that resources are treated as you want them to be treated without changes to your applications, GRS provides three resource name lists (RNLs):

- **SYSTEMS EXCLUSION RNL**
  A list of resources that are requested with a scope of systems that you want GRS to treat as local resources

- **RESERVE CONVERSION RNL**
  A list of resources that are requested on RESERVE macros for which you want global resource serialization to suppress the RESERVE

- **SYSTEM INCLUSION RNL**
  A list of resources that are requested with a scope of system that you want GRS to treat as global resources

Use the following steps to avoid contention problems:

- In a GRS star configuration, a RESERVE CONVERSION RNL should be implemented for DSPURI01 if all systems accessing the RECON data sets are within the sysplex (or GRSPlex). If the RECON data sets are accessed by systems that are outside of the sysplex, the RESERVE must not be converted. A SYSTEMS EXCLUSION RNL must be implemented instead. If you implement GRS RNL CONVERSION by adding the QNAME for the RECON data set (DSPURI01) to the conversion list, the hardware reserve is eliminated and replaced by a GRS enqueue that is communicated to all other sharing z/OS systems. Other data sets on the same DASD volume can be used while the RECON data set is reserved; this is the benefit of performing the RNL conversion. GRS RNL CONVERSION uses CPU and storage and affects system performance positively. The performance (in terms of the least CPU time used, the least storage used, and the least elapsed time) is best by using this option in a GRS STAR configuration.

In an IMS Fast Path two-way data sharing environment comparison done in a controlled environment (at the Silicon Valley Laboratory) in which 6000 IMS Fast Path areas were opened in parallel using BMP, a significant reduction in total elapsed time was observed when the reserve is converted. In a GRS star configuration, placing DSPURI01 in the
conversion list, and also both SYSZVVDS and SYSVTOC (these two should always be in the same RNL), produced a roughly 3.5% reduction in total BMP elapsed time.

- In a GRS ring configuration, a SYSTEMS EXCLUSION RNL should be implemented for DSPURI01 so that the RECON is serialized by the hardware reserve. In this case, the following items also apply:
  - The RECON data sets must be the only objects in their respective catalogs.
  - The RECON data sets must be on the same device as the catalog and isolated from other data sets.

Minor name: Batch jobs also enqueue on a minor name so that they serialize on each other and avoid monopolizing the RECON to the detriment of online IMS systems.

9.3 DBRC RECON maintenance

Monitoring the status of IMS RECON data sets should be a routine task of the IMS database system administrator. IMS RECON data sets can be considered as the heart of the IMS system, especially when data sharing is enabled. Even without data sharing, RECON data sets are used to store recovery-related information about subsystems, logs, and utility executions, along with other pertinent information. The RECON data sets are critical resources for IMS. If all RECON data sets are lost, IMS cannot continue processing and it terminates abnormally.

RECON data sets are VSAM key-sequenced data sets (KSDS) that should be periodically reorganized. For direction and guidance about allocating RECON data sets, see the IMS Version 12: System Administration, SC19-3020

You can enter the following Time Sharing Option (TSO) command to determine the number of records, CI splits, and CA splits, to ultimately determine whether the RECON data sets might require reorganization:

LISTC ENT('IMSA.RECON1') ALL

When one of the RECON data sets is in a DISCARDED state, take advantage of the situation and perform a reorganization on that one data set. A RECON data set can become discarded by IMS because of a problem or by issuing a DBRC CHANGE.RECON REPLACE() command.

RECON data sets can be reorganized by both an online process and a batch process. Run these processes only during a slow period of activity in the IMS region because copying the RECON data sets from one data set to another might affect the performance of IMS.

If a RECON data set is discarded by IMS, you should, at a minimum, perform the delete and define step for the data set that is discarded, to maintain two RECON data sets and one spare. The other RECON data sets can then be reorganized at a slower processing period.
You may use the following process to reorganize the RECON data sets. To execute the process while online IMS systems are accessing the RECONs, the RECON data sets need to be dynamically allocated.

1. Enter the DBRC command to determine the status of the RECON data sets. If you enter DBRC commands from the z/OS MCS or E-MCS console, be sure to add a period at the end of the command or the IMS system issues the following message:

   DFS972A *IMS AWAITING MORE INPUT*

   Use the following syntax when entering the command as an IMS command:

   /RML DBRC='RECON STATUS'.

   You can also execute this command by submitting a job similar to the one in Example 9-1. Use the LIST.RECON STATUS, which is provided in the SYSIN stream.

   You might also want to delete unnecessary log information from the RECON by issuing the DELETE.LOG INACTIVE command before starting the reorganization. This can be done in the same job, as in Example 9-1.

   **Example 9-1  Job for executing DBRC command**

   //YOURJOBL JOB CLASS=A,MSGCLASS=X,REGION=4M
   //******************************************
  //*     USE: RECON COMMANDS
   //******************************************
   //*
   //STEP010   EXEC PGM=DSPURX00
   //*
   //STEPLIB   DD  DSN=IMS.SDFSRESL, DISP=(SHR,KEEP,KEEP)
   //*
   //SYSPRINT  DD  SYSOUT=* 
   //SYSUDUMP  DD  SYSOUT=*
   //*
   //SYSIN     DD  *
   LIST.RECON STATUS
   DELETE.LOG INACTIVE
   /*

   Example 9-2 shows the most interesting part of the output from the command.

   **Example 9-2  LIST.RECON STATUS before starting the reorganization**

<table>
<thead>
<tr>
<th>DDNAME-</th>
<th>STATUS-</th>
<th>DATA SET NAME-</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECON1</td>
<td>COPY1</td>
<td>IMSA.RECON1</td>
</tr>
<tr>
<td>RECON2</td>
<td>COPY2</td>
<td>IMSB.RECON2</td>
</tr>
<tr>
<td>RECON3</td>
<td>SPARE</td>
<td>IMSC.RECON3</td>
</tr>
</tbody>
</table>

   2. Issue the CHANGE.RECON REPLACE(RECON1) command to discard RECON1. RECON2 is copied to RECON3, and RECON1 is discarded. Example 9-3 shows the output.

   **Example 9-3  Output of the CHANGE.RECON REPLACE(RECON1) command**

   CHANGE.RECON REPLACE(RECON1)
   DSP0380I RECON2 COPY TO RECON3 STARTED
   DSP0380I SSID=IMSG FOUND
   DSP0380I 0001 SYS RECORD(S) IN THE RECON AT RECONFIGURATION
   DSP0381I COPY COMPLETE, RC = 000
   DSP0242I RECON1 DSN=IMSA.RECON1
After the `CHANGE.RECON REPLACE(RECON1)` command, RECON2 is the new COPY1 and RECON3 is the new COPY2, and RECON1 is in DISCARDED status, such as in Example 9-4.

**Example 9-4**  LIST.RECON STATUS after replacing RECON1

<table>
<thead>
<tr>
<th>DDNAME-</th>
<th>-STATUS-</th>
<th>-DATA SET NAME-</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECON2</td>
<td>COPY1</td>
<td>IMSB.RECON2</td>
</tr>
<tr>
<td>RECON3</td>
<td>COPY2</td>
<td>IMSC.RECON3</td>
</tr>
<tr>
<td>RECON1</td>
<td>DISCARDED</td>
<td>IMSA.RECON1</td>
</tr>
</tbody>
</table>

3. Before you can delete the cluster, ensure that no other users, such as batch jobs, are using the data set. If you are using the automatic RECON loss notification feature, all DBRC instances, which were registered with the Structured Call Interface (SCI), are automatically notified about the reconfiguration and the discarded RECON data set. The SCI is used for the communication between all registered DBRC instances across your IMSplex. The unavailable RECON data set gets discarded instantly from all notified DBRC instances. As long as you are using dynamic allocation for your RECON data sets, all involved DBRC instances also deallocate the discarded RECON data set.

If you are not using automatic RECON loss notification feature, then online IMS subsystems, when they next access the RECONs, release the RECON that was discarded. You can force IMS to access the RECON by issuing either of the following commands:

- `/RML DBRC='RECON STATUS'.`
- `/DIS OLDS`

**Simplify the process:** The automatic RECON loss notification feature can be used to simplify the process. The automatic RECON loss notification feature is optional and is automatically enabled if the SCI address space is up and running on the same z/OS image and the DBRC instance registers with SCI as a member of an IMSplex.

4. Now delete and define the VSAM cluster for the discarded RECON by using the appropriate attributes (from 9.2.1, “Defining the RECON data sets” on page 339). A good practice is to store the IDCAMS statements for each RECON, because you might want to use the same attributes that were used when the RECON was previously allocated.

For the previous RECON listing, delete and define RECON1, which is currently discarded. If you now display the RECON again, you see that RECON1 is shown as a SPARE (Example 9-5).

**Example 9-5**  LIST.RECON STATUS after redefining the VSAM cluster for RECON1

<table>
<thead>
<tr>
<th>DDNAME-</th>
<th>-STATUS-</th>
<th>-DATA SET NAME-</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECON2</td>
<td>COPY1</td>
<td>IMSB.RECON2</td>
</tr>
<tr>
<td>RECON3</td>
<td>COPY2</td>
<td>IMSC.RECON3</td>
</tr>
<tr>
<td>RECON1</td>
<td>SPARE</td>
<td>IMSA.RECON1</td>
</tr>
</tbody>
</table>

5. Issue the `CHANGE.RECON REPLACE(RECON2)` command to discard RECON2. RECON3 is copied to RECON1 and RECON2 is discarded.
6. Now delete and define the VSAM cluster for the discarded RECON2. Before you can delete the cluster, ensure that no other users are using the data set. See step 3 on page 344 for details.

7. Delete and define the discarded RECON2 by using the appropriate parameters (from 9.2.1, “Defining the RECON data sets” on page 339). RECON2 becomes the new spare.

8. Issue the `CHANGE.RECON REPLACE(RECON3)` command to discard RECON2. RECON3 is copied to RECON1 and RECON2 is discarded.

9. Now delete and define the VSAM cluster for the discarded RECON3. Before you can delete the cluster, ensure that no other users are using the data set. See step 3 on page 344 for details.

10. Delete and define the discarded RECON3 by using the appropriate attributes (from 9.2.1, “Defining the RECON data sets” on page 339). RECON3 becomes the new spare. All VSAM clusters are now redefined, and RECON1 is now the COPY1, and RECON2 is COPY2, as they were at the starting point.

You can automate this process by putting all the previous steps in the same job, but you need to schedule the job at a time when no other subsystems are using the RECON. You must also ensure that the discarded RECON is released by all the sharing online systems. With only one online subsystem and no other users (batch jobs or utilities), you can use the `/DIS OLDS` command by executing an automated operator BMP. With multiple systems sharing the RECON, you must propagate the command to the other systems. You can set up the automatic RECON loss notification feature for this purpose.

### 9.4 The CLEANUP command parameters

IMS 11 introduced the `CLEANUP.RECON` command to delete old or expired recovery-related information from the RECON data set, including image copy, allocation, reorganization, and recovery records and log information.

IMS Version 12 enhanced the features with the following items to further manage the records on the RECONS:

- **New CLEANUP command parameters**
  - With IMS 12, either of the following commands can be processed when the LOGALL record does not exist:
    - `DELETE.LOG INACTIVE`
    - `TOTIME`

- **New change accumulation retention period**
  - IMS 12 introduces the retention period keyword into the CA group type of record. The RECOVPD optional keyword is added to `INIT.CAGRP` or `CHANGE.CAGRP` commands to control DBRC keeping of CA execution records. When GRPMAX is exceeded, CA execution record is kept if the RECOVPD value is not exceeded.

IMS 12 adds the option to delete CA execution records information by the new CAGRANGE, CAONLY, and LASTCA keywords. In addition, IMS 12 issues the DSP0115I message instead of setting a fatal return code when a `DELETE.LOG INACTIVE` or `TOTIME` command was issued and a LOGALL record did not exist for the inactive log.

The following commands are affected:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CLEANUP.RECON</code></td>
<td>Deletes obsolete recovery-related CA execution and log records.</td>
</tr>
<tr>
<td><code>DELETE.LOG</code></td>
<td>Deletes LOG records.</td>
</tr>
</tbody>
</table>
Figure 9-2 shows the syntax of the `CLEANUP.RECON` command. For the syntax of other DBRC commands, see *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

![Figure 9-2 The CLEANUP.RECON command syntax](image_url)

The `CLEANUP.RECON` command has the following optional keywords:

**CAGRANGE(firstcag,lastcag | firstcag |,lastcag)**  
Specifies that CA data sets for a range of CA groups are to have cleanup done. If this keyword is not specified, all CA groups are processed. Use this keyword to clean up CA data sets for multiple CA groups or to resume cleanup after a `CLEANUP.RECON CAGRANGE` command is stopped.

- **firstcag**, **lastcag**  
  Upper and lower limits of the range of CA groups. These parameters do not need to match the name of a CA group in the RECON data set. They are used as the beginning and ending arguments in an alphabetic search.

- **firstcag**  
  Name of the CA group from which RECON cleanup begins processing the CA execution data sets. This CA group and all subsequent CA groups are processed until the last group in the RECON data set is reached or until the CA group specified as the lastcag value is reached.

- **lastcag**  
  Name of the CA group up to which RECON cleanup continues to process CA groups. RECON cleanup starts with the first CA group defined in the RECON data set and stops after this CA group is processed. If lastcag is specified, it must be preceded by a comma (,).

**CAONLY**  
Specifies that you want to delete only CA execution records. (No database recovery-related information or log information is processed.) However, if the DBONLY keyword is also included in the command, database recovery-related information is also processed, and only the log information remains unprocessed.

**LASTCA**  
Specifies that the last CA execution record for a CA group can be deleted as part of the RECON cleanup process if it meets the criteria for deletion. If this keyword is not specified, the last CA execution record for a CA group is retained.
Using the CLEANUP and DELETE commands

Example 9-6 shows the output from a CLEANUP.RECON command, submitted with LASTIC and LASTCA optional keywords. These keywords mean that IC and CA records can be deleted as part of the RECON cleanup process if they meet the criteria for deletion, including the retention period specified.

**Example 9-6 The CLEANUP.RECON command**

```
IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
CLEANUP.RECON LASTCA LASTIC RETPRD(365)
DSP0123I NO PREDEFINED IMAGE COPY DATA SETS REMAINING
DSP0123I DBDNAME=AUTODB     DDNAME=DFSDLR
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=AUTODB     DDN=DFSDLR
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=DI21PART  DDN=DI21PARO
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=DI21PART  DDN=DI21PARO WERE DELETED
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=DI21PART  DDN=DI21PART
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=DI21PART  DDN=DI21PART WERE DELETED
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=DI21PART  DDN=DI21PART
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=DI21PART  DDN=DI21PART WERE DELETED
DSP1203I NO PREDEFINED IMAGE COPY DATA SETS REMAINING
DSP1203I DBDNAME=EMPDB2     DDNAME=DFSEML
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=EMPDB2     DDN=DFSEML
DSP1203I NO PREDEFINED IMAGE COPY DATA SETS REMAINING
DSP1203I DBDNAME=IPDB1     DDNAME=IPDB1A
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IPDB1     DDN=IPDB1A
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB1I   DDN=IPDB1A
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB1     DDN=IPDB1
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=IVPDB1     DDN=IPDB1 WERE DELETED
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB1I   DDN=IPDB1I
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=IVPDB1I   DDN=IPDB1I WERE DELETED
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB2     DDN=IPDB2
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=IVPDB2     DDN=IPDB2 WERE DELETED
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB2     DDN=IPDB2
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB2I   DDN=IPDB2I
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=IVPDB2I   DDN=IPDB2I WERE DELETED
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=IVPDB2I   DDN=IPDB2I
DSP1212W ALL EXISTING IMAGE COPIES FOR DBNAME=IVPDB2I   DDN=IPDB2I WERE DELETED
DSP1203I NO PREDEFINED IMAGE COPY DATA SETS REMAINING
DSP1203I DBDNAME=SINDEX11  DDNAME=SINDEX1P
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=SINDEX11  DDN=SINDEX1P
DSP1203I NO PREDEFINED IMAGE COPY DATA SETS REMAINING
DSP1203I DBDNAME=SINDEX22  DDNAME=SINDEX2P
DSP1214I RECON INFORMATION WAS DELETED FOR DBNAME=SINDEX22  DDN=SINDEX2P
DSP1203I NO CHANGE ACCUMULATION INFORMATION WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.053 14:51:54.800000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.053 14:51:54.800000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.060 19:40:46.400000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.060 19:40:46.400000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.083 11:40:22.500000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.083 11:40:22.500000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.084 21:07:05.300000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.084 21:07:05.300000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.091 07:49:56.400000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.091 07:49:56.400000 AND SSID=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.092 08:57:01.300000 AND SSD=IMSB WAS DELETED
DSP1216I THE PRILOG FAMILY WITH TIME=10.092 08:57:01.300000 AND SSID=IMSB WAS DELETED
DSP1203I NUMBER OF INACTIVE PRILOG RECORDS DELETED WAS 00006
DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00
```

For more examples of the CLEANUP.RECON command, see *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

Example 9-7 shows the DELETE.LOG command, when specifying that the records of all inactive recovery log data sets (RLDSs) and system log data sets (SLDSs), which have a start time older than the time specified with the TOTIME keyword, are to be deleted.

**Example 9-7 The DELETE.LOG command**

```
//SYSIN DD *
DELETE.LOG TOTIME(10293213950100000)
/*
```
Considerations for the CLEANUP command

After you delete amounts of data, you might want to reorganize your RECONs to ensure that the freed space can be reused. You must issue the CLEANUP.RECON command through a batch DBRC job or the DBRC API. This command is not available from IMS through an /RMx command.

Evaluate the use of the CLEANUP.RECON command first on a copy of the RECONs, to avoid unintended deletions and to estimate the amount of time to process the command.

The last CA execution record for a CA group is deleted only when specifically requested.

New change accumulation retention period

Earlier versions of IMS use the GRPMAX parameter to control how many versions of CA execution should be kept in the RECON data sets. When the number of times you run the CA utility for the specified group exceeds the GRPMAX value, the record with the earliest CA stop time for the group is deleted for CA groups.

To improve the control of relevant information stored in the CA group record, IMS 12 provides the RECOVPD optional keyword set the recovery period for a specified CA group record. The recovery period is the amount of time before the current date for which DBRC maintains CA information in the RECON data set.

The CHANGE.CAGRP and INIT.CAGRP commands, which you use to modify information contained in a specified CA group record, are affected. Figure 9-3 shows the new syntax of CHANGE.CAGRP command. For the syntax of other DBRC commands, see IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011.

Figure 9-3  Syntax of the CHANGE.CAGRP command

The CHANGE.CAGRP command has the following option:

RECOVPD(0 | value)

Using the RECOVPD keyword

You use this optional keyword to modify the recovery period for a specified CA group. The recovery period is the amount of time before the current date for which DBRC maintains CA information in the RECON data set. For example, if the recovery period of a CA group is 14 days, DBRC maintains sufficient CA execution records for at least 14 days.
To determine whether the CA execution record falls within the recovery period, subtract the RECOVPD value from the current time. Any CA execution records with stop times that are newer than the calculated time is to be kept in the RECON data set.

For value, specify a decimal number in the range 0–999 that represents the number of days that the CA execution records are to be kept in the RECON data set. If you specify 0 (the default), there is no recovery period.

In Example 9-8, the retention period for the CA data sets for CA group GRPIVP is being changed to 7 days. This change accommodates new business requirements because the user started doing more than one CA process per day.

Example 9-8 The CHANGE.CAGRP command with the RECOVPD keyword

<table>
<thead>
<tr>
<th>IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL</th>
<th>PAGE 0001</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGE.CAGRP GRPNAME(GRPIVP) RECOVPD(7)</td>
<td></td>
</tr>
<tr>
<td>DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00</td>
<td></td>
</tr>
<tr>
<td>DSP0220I COMMAND COMPLETION TIME 11.208 17:05:49.648625</td>
<td></td>
</tr>
<tr>
<td>IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL</td>
<td>PAGE 0002</td>
</tr>
<tr>
<td>LIST.CAGRP GRPNAME(GRPIVP)</td>
<td></td>
</tr>
<tr>
<td>11.208 17:05:49.408264</td>
<td></td>
</tr>
<tr>
<td>LISTING OF RECON</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>CAGRP</td>
<td></td>
</tr>
<tr>
<td>GRPNAME=GRPIVP GRPMAX=7 CA AVAIL=0 CA USED=0</td>
<td></td>
</tr>
<tr>
<td>NOREUSE CAJCL=CAJCL DEFJTJCL=<strong>NULL</strong> RECOVPD=7</td>
<td></td>
</tr>
<tr>
<td>#MEMBERS=1 -DBD- -DDN- IVPDB1 DFSIVD1</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>DSP0180I NUMBER OF RECORDS LISTED IS 1</td>
<td></td>
</tr>
<tr>
<td>DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00</td>
<td></td>
</tr>
</tbody>
</table>

Considerations for the RECOVPD keyword

The recovery period value is a decimal number, in the range of 0 - 999, that represents the quantity of days that the CA execution records are kept in the RECON data set. If you specify 0 (zero, the default), there is no recovery period. If you specify 7, DBRC maintains sufficient CA execution records for at least 7 days based on the stop time of the CA execution record.

If you issue the CHANGE.CAGRP command and specify the GRPMAX and RECOVPD values that are less than the existing values, any used CA data sets with stop times that are beyond the recovery period are deleted. These data sets remain deleted until the number of remaining CA data sets equals the specified GRPMAX value.

If you issue the DELETE.CA command, any specified CA data set record is deleted, regardless of the RECOVPD or GRPMAX values.

If the GRPMAX limit is reached, but the RECOVPD for the oldest CA record has not expired, DBRC issues an informational message (DSP1232I) and does not discard the record. If the DSP1232I message appears frequently, you might need to tune the GRPMAX or RECOVPD values by using the CHANGE.CAGRP command.

**GRPMAX value:** If the GRPMAX value is lowered by using the CHANGE.CAGRP command, the GRPMAX value is recorded regardless of whether the oldest CA data sets can be deleted, because they are within the recovery period.
9.5 DBRC parallel RECON access

IMS Version 10 introduced an optional function named Parallel Recon Access (PRA), which affects how DBRC accesses its RECON data sets. To maintain integrity and provide backout in the event of an error, DBRC always provided an internal mechanism for locking and backing out changes if necessary. Basically, the locking mechanism consisted of issuing a RESERVE macro for each of the data sets and tracking updates within the RECON data sets. This locking can be converted potentially to a global enqueue by the GRSRNLxx member of SYS1.PARMLIB. Either way, the RECON access was serialized across all the IMS images in the sysplex.

With the PRA function introduced in Version 10, DBRC uses the functions of transactional VSAM to provide locking and logging for backout. The DBRC implementation of PRA itself can be easy, with a few simple DBRC commands. However, implementation of the prerequisite transactional VSAM function (TVS) is much more complex.

More details about this implementation is in IBM IMS Version 10 Implementation Guide: A Technical Overview, SG24-7526. Although implementing PRA can require significant planning and tuning to make it perform well, the results can prove valuable. We conducted several measurements, opening 5,000 databases by using various methods of locking; the results are shown in Figure 9-4.

As you see, there was little difference in using the RESERVE or enqueue functions with serial access. However, after tuning transactional VSAM, these time could be significantly improved. The first column shows the time when the DBDs were non-resident in order to compare with the other columns, which represented the measurements with RESIDENT. The only reason this first measurement was done was to determine whether loading the DMBs had any significant affect on the overall time.

Figure 9-4  DBRC access times for serial and parallel RECON access

<table>
<thead>
<tr>
<th>Option</th>
<th>Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-res, grs conv</td>
<td>261</td>
</tr>
<tr>
<td>serial, grs excl</td>
<td>260</td>
</tr>
<tr>
<td>serial, grs conv</td>
<td>249</td>
</tr>
<tr>
<td>parallel</td>
<td>121</td>
</tr>
</tbody>
</table>
9.6 DBRC execution on Base Primitives Environment (BPE)

Starting with IMS 11, DBRC provides the optional ability to run on Base Primitive Environment (BPE) and also offers serviceability options for the RECON data sets. A BPE-based DBRC provides improved user exit management and trace support. This support is totally optional. You can currently start DBRC with current procedures or use a new startup JCL member that uses BPE. One of the advantages of running DBRC on BPE is the ability to update user exits without shutting IMS or DBRC down.

Another feature of DBRC under BPE is the additional trace information and common systems services that the BPE environment provides.

As shown in Figure 9-5, BPE offers further benefits and studies of DBRC processing, indicating that performance is comparable to DBRC executing in a non-BPE configuration.

### IMS BPE DBRC

- **BPE DBRC Observations**
  - Allows DBRC to run as a BPE based region in order to take advantage of common services and allows for greater levels of tracing.
  - Comparing total elapsed time during IMS startup with FP DEDB Area PREOPEN. Measurements were obtained from IMS messages DFS3715I and DFS3719I. PREOPEN process started and PREOPEN process completed respectively.
  - Performance characteristics for both DBRC Regions, BPE based and without BPE, are very similar. There was no significant degradation observed for total elapsed time for this comparison.

<table>
<thead>
<tr>
<th>Pre-Opens FP DEDB Areas</th>
<th>10,000</th>
<th>20,000</th>
<th>30,000</th>
<th>40,000</th>
<th>50,000</th>
<th>60,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBRC with BPE Based</td>
<td>00:04:50</td>
<td>00:11:25</td>
<td>00:20:44</td>
<td>00:31:01</td>
<td>00:43:48</td>
<td>00:58:22</td>
</tr>
<tr>
<td>DBRC without BPE</td>
<td>00:04:51</td>
<td>00:11:04</td>
<td>00:20:43</td>
<td>00:31:02</td>
<td>00:44:03</td>
<td>00:58:41</td>
</tr>
</tbody>
</table>

The table demonstrates the total elapsed time observed for PREOPEN of an FP DEDB area and compares BPE based DBRC vs. non-BPE based DBRC.

Figure 9-5  DBRC BPE/non-BPE processing benefits and performance comparison

Information about implementation of DBRC under BPE is in *IMS Version 12: System Administration*, SC19-3020.
Parallel Sysplex considerations

IBM Parallel Sysplex technology can harness the power of up to 32 z/OS systems and make them behave like a single, logical computing facility. The underlying structure of the Parallel Sysplex remains virtually transparent to users, networks, applications, and even operations.

With IMS and Parallel Sysplex, there are several factors which you must additionally take into consideration for IMS performance. Some of these are typically outside the control of the IMS systems programmer but must be understood anyway. We document these considerations in this chapter.

This chapter covers the following topics:

- Coupling facility hardware and microcode
- Structure sizing
- IRLM considerations
- IRLM lock structure
- VSAM cache structure
- OSAM cache structure
- DEDB considerations
- Application considerations
- Shared queues
10.1 Coupling facility hardware and microcode

The coupling facility (CF) can have a significant impact on IMS performance. Depending on your particular hardware configuration, you might need to use certain functions to maintain high availability, which are less than ideal for performance. These facilities include system-managed duplexing and the use of staging data sets for the MVS system logger. These items are discussed more in this chapter.

10.1.1 Coupling facility configuration

Coupling facilities can be contained within the same physical system as the z/OS LPAR or they can be stand-alone systems. If they are stand-alone systems, you can generally achieve high availability without having to use system-managed duplexing or staging data sets. If the CF LPARs are on the same physical system as the z/OS images running IMS, then using these functions to prevent a loss of integrity might be necessary. In a production environment, review the following considerations:

- Dynamic dispatching for the CF LPAR might cause poor or erratic CF response times.
- Multiple CF engines, even when utilization is relatively low, can help keep response times more stable.
- Multiple CF links help both availability and benefit performance.

10.1.2 Coupling facility microcode

New levels of CF microcode generally introduce new capabilities. Some of these new functions can take up additional control storage so that there might be less storage available for user data in the same size structure. The good news is that except for preloaded VSO, the microcode changes rarely cause any failures and IMS continues to run. The bad news is that your IMS performance can suffer because of some issue you might not even know about. For this reason, you be aware any time there is a change to the CF microcode or the configuration. Make sure you know how to obtain and read the RMF CF Structure Activity reports and keep a repository of these reports for future reference in the event of problems.

10.2 Structure sizing

Individual structure sizing considerations are described later in this chapter, but for all IMS structures and also other system and subsystem structures, see the IBM CFSizer website: http://www.ibm.com/servers/eserver/zseries/cfsizer

Click IMS, and then select any or all of the IMS-related structures to estimate the size. The associated coupling facility sizing application provides an easy-to-use interface to calculate the structure sizes for you based on minimum input data that you provide to represent your expected usage of the structure-owning product. You supply input that corresponds to your expected peak usage of the product. In general, a good practice is to slightly over specify peak values, to produce a sizing recommendation slightly larger than absolutely necessary. This way provides room for growth, and can help avoid failures caused by insufficient structure sizes. This website normally calculates the size based on the most current CFLEVEL available, which is displayed in the results page. Structure sizes are always a multiple of 256K, so if you specify a size, which is not a multiple of 256K, the system rounds it up to the next 256K increment.
10.3 IRLM considerations

Consider several IRLM startup parameters for IRLM performance:

- **PC=YES or NO**
  Specifies control blocks in private storage. IMS continues to support the use of IRLM 2.1 at this time, which allows the specification of either value. IRLM 2.2 allows either value to be specified but always runs with PC=YES regardless of the setting. Many years ago, specifying PC=NO might have affected IRLM CPU usage. With the processors of today, however, there is little difference in CPU and removing the lock table from ECSA helps free valuable common storage.

- **DEADLOK=**
  Specifies deadlock detection timing. Using values less than one second for DEADLOK have little impact on CPU but might be beneficial by resolving deadlocks quickly, because hundreds of transactions typically are running every second. Waiting one second or more to resolve a deadlock can back up transaction processing significantly, which might result in other undesirable conditions. If you are using DB2, the IMS IRLM deadlock resolution can reduce the number of long running queued DB2 transactions that are waiting for DB2 deadlock or timeout processing, which is typically a much higher value.

- **MAXUSRS=**
  Specifies maximum number of users in the group. The MAXUSRS value determines the size of each entry in the lock table, which can be 2, 4, or 8 bytes. MAXUSRS represents the number of IRLMs that you expect to have in the data sharing group. Any number up to 7 causes each entry to be 2 bytes; numbers 8 - 23 create 4-byte entries, and numbers greater than 23 create 8-byte entries. As you see, this number directly affects the number of lock entries in a given size structure, so use the smallest number (or a number in the ranges mentioned) that meets your needs.

- **LTE=**
  Specifies lock structure table entries. With the LTE value, you can override the IRLM default calculation of half of the lock structure for LTEs. The number specified is multiplied by 1048576. Be careful when you specify this value, because severely restricting the number of record list entries is possible, potentially causing tasks to fail. If more LTEs are necessary to reduce false contention, then increasing the size of the lock structure is a better approach.

The `F irlmproc,SET,TIMEOUT=nnnn,dbms` command

Use this command to establish a time after which IMS is notified that some tasks are waiting for a lock longer than the time specified, and also to create a type 79, subtype15 SMF record and with a DXR162I message that is issued by the IRLM. If no command is issued, then the default is 300 seconds (five minutes). Note that you must issue this command for each IMS, and it must be after IMS is identified to the IRLM.

`LOCKTIME=` in the DFSVSMxx member of IMS PROCLIB

This value interacts with the TIMEOUT value. LOCKTIME, if specified, overrides the default value of 300 seconds; however, there is an important difference between specifying here as opposed to using the IRLM command. If LOCKTIME is specified, then any applications waiting longer than the specified time are abended with a U3310 or given a BA status code. This result might or might not be what you want. You can change the time by using the IRLM SET command, but you cannot change the action (abend or status code) without restarting IMS.
Set TRACE=NO from a pure performance perspective. If there are locking problems, having the trace information is probably worth the cost of the trace collecting information.

**WLM for IRLM**

WLM should be set up to put IRLM in a high priority system task level, such as SYSSTC, SYSTEM, or a performance group with similar settings. Most IRLM locking code is run under the dependent region’s TCB, and the times when IRLM must be dispatched are for contention and deadlock resolution. In this case, you want IRLM to be dispatched as quickly as possible, because any delay can have a global (sysplex-wide) effect.

### 10.4 IRLM lock structure

IMS uses the IRLM to manage locks in a block-level data sharing environment. The IRLM uses a lock structure to manage lock requests from its IMS clients.

#### 10.4.1 Lock structure size

The size of the lock structure is defined in the CFRM policy. By default, the IRLM makes half of the structure for LTEs. Therefore, we suggest that the size specified is also a power of 2 and that any increases then are double or four times the original amount. A reasonable starting size for production is probably 64 MB, which means 32 MB for LTEs, and if there are two-byte entries, then there are 16 million LTEs.

#### 10.4.2 False contention

Monitor false contention with RMF and adjust the size of the structure. We suggest that the number for false contention is one-tenth of 1%. This number might or might not be achievable, because much is dependent on the workload. If higher than 0.1%, try increasing the size of the structure to reduce this number. However, remember that you should always increase by doubling the size. There is normally a point of diminishing (or no) return; so if increasing the size has little or no impact, you probably reached that point and do not need to increase the size further.

Remember that any contention, real or false, causes the IRLM to do additional work to resolve that contention. This work involves the IRLM itself being dispatched, and XCF signalling activity with the other lock managers, which has a negative impact on performance.

#### 10.4.3 Automatic rebuild

If you initially specify MAXUSRS as 7 or fewer so that you have 2-byte LTEs, then the IRLM initiates a structure rebuild if a seventh IRLM (yes, the seventh) joins the group. This way causes the number of LTEs to be cut in half, because they are now 4-byte entries instead of 2 (or perhaps 8, instead of 4). Consider this information when you set the MAXUSRS and the size of the structure, so that you are not surprised by a sudden increase in false contention because of a reduction in the number of LTEs. If this happens, you can, of course, alter the size of the structure and then rebuild to get back to the optimal number of LTEs.
10.4.4 System-managed duplexing

From a performance perspective, you want to avoid this function, because it can add significantly to the CF structure response time. The IRLM can rebuild the lock structure if something happens, unless both the lock structure and an IRLM fail concurrently (such as when they are both on the same physical hardware). In that case, you must restart IRLM (and IMS if it also stopped) before the lock structure can be rebuilt and data sharing can continue. This event is unlikely but might have to be considered over performance.

10.5 VSAM cache structure

The VSAM cache structure is a directory-only structure, so it must be large enough to have a directory entry for each buffer in every IMS. It actually needs to have only enough entries to handle all of the unique blocks in all of the IMS images (including batch data sharing), but it is impractical and a waste of effort to try to figure out this number. To determine the size, you need to determine the total number of VSAM buffers in all of the IMS images, which might join the data sharing group. This number is then used as input to the CFSizer program. A quick calculation can be done by taking the number of buffers, multiplying by 300, and adding another 256,000 for CF overhead. Always estimate high. It is better to have too much storage than not enough.

If the size is too small to support all of the buffers in the data sharing group, there are directory reclaims, which can affect performance. In this case, some buffers in IMS cannot be used and any existing data in them must be discarded.

IMS supports the dynamic ALTER of the VSAM cache structure size, if you determine it is too small. However, beware of using the AUTOALTER facility, because it is possible for the system to decrease the size of the structure if it needs storage for something else.

10.6 OSAM cache structure

The OSAM structure might be a directory-only structure or might additionally cache data. If you use the structure only for buffer tracking and invalidation, as with the VSAM structure, the sizing considerations are basically the same. Simply add all the OSAM buffers in all the IMS images in the data sharing group, including sequential buffer sets. Provide that number to the CFSizer utility, or manually calculate as described previously.

For data caching, IMS allows you to specify what is to be cached, if anything, by subpool. Because you can isolate database data sets to specific subpools, you can effectively cache by database data set. In the DFSVSMxx member that you use to define OSAM subpools, you can specify the caching option (co) in one of the following ways:

- omitted No data caching. Caching is not active for the subpool.
- N No data caching. Caching is not active for the subpool.
- A Cache all data. Write all data read from DASD and all changed data to the coupling facility.
- C Cache only changed data. Write all changed data written to DASD to the coupling facility.
The following considerations can help you decide whether caching is a good idea:

- For the CACHE ALL option, consider that every block that is read into IMS is also written to the structure. This option can cost CPU time, and to a lesser extent, elapsed time. This option is probably best for small databases (or small locality of reference) that are heavily referenced, and all the blocks can remain in the structure.
- Full-function database blocks are always written to DASD at sync point, regardless of the cache option.
- For the CACHE CHANGED option, blocks are only written to the structure if updated. If many IMS images update the same set of blocks frequently, then this option might be appropriate, because it prevents each sharing IMS from rereading updated blocks from DASD.

10.7 DEDB considerations

DEDBs do not use a cache structure except for shared VSO. With Parallel Sysplex data sharing, there is additional overhead however. This overhead is in accessing the lock structure. Shared DEDBs get a global lock to update. Because the IRLM lock structure typically has fast response time, this is not usually a concern; however, additional contention can occur depending on the access pattern to these databases. Database contention should be closely monitored and corrective action taken to reduce the contention. Most often this is accomplished by using a smaller CI size.

Shared VSO uses a cache structure or many cache structures. Its use is specified on an area-by-area basis. Consider the following information:

- Use the PRELOAD option for small, highly referenced areas. With this option, the entire area must fit into the CF structure. NOPREL accommodates larger areas and can be as effective as PRELOAD in some cases, but it depends on the data access pattern.
- The LKASID option is normally best. You can monitor the \%hits and \%valid numbers; if these numbers are extremely low, consider turning off this option. Otherwise, keep it on.
- Private buffer pools are actually located in ECSA. Keep this in mind, because common storage is a limited resource. If the \%hits is not good, but \%valid is almost equal, it is an indication that additional buffers might help.
- SVSO updates are written asynchronously to sync point processing. The castout process as it is referred to, is initiated at IMS checkpoint time. IMS reads the updated blocks from the CF and writes them to DASD. For this reason, it is important that SVSO areas with lots of updates physically reside on the best performing DASD.
- IMS duplexing is more efficient than system-managed duplexing. A general guideline is that IMS duplexing is twice the CPU overhead of a single structure with little or no response time increase. System-managed duplexing increases CPU overhead also but also increases the response time to the structure by three to five times.
10.8 Application considerations

Application considerations are as follows:

- Review PSB PROCOPTs to ensure only the necessary processing options are used. Use the minimal access needed for processing. Specify the lowest level of access to reduce the levels of locks, and contention.

  Do this step for PSBs that are used by the high volume transaction. If a particular high volume transaction does only read operations, then changing the PROCOPT from A to G (or even, if business integrity allows, GOT or GON) might have significantly affect reducing contention.

- Review DEADLOCK situations. In some cases, the number of deadlocks can actually be reduced with data sharing, but that is a rare situation. If deadlocks are an impact without data sharing, then they most likely get worse when Parallel Sysplex data sharing is implemented.

- Look for areas of contention, which might be a simple change of PROCOPT, as previously mentioned, or perhaps an application change is necessary. This case is especially true where a single segment or set of segments are updated by many high volume transactions. As an interim solution, it is sometimes necessary to limit the number of regions which can process a given transaction, or possibly process that transaction type on a single IMS. Although this solution is not permanent for availability, it might be a temporary patch for performance.

- PSBs should be generated for each application and not reused or shared. Reuse can lead to changes in one application, requiring changes to other applications that are not necessary, resulting in resource utilization where it is unnecessary. The PSB should access only what is required. This approach can also reduce storage consumption and CPU utilization, managing PSB pools, and so on.

- Remove obsolete resources. Applications should remove obsolete resources, such as databases from the application code, so the PSB defines only what is required. This step reduces the number of items, such as “DUMMY” databases defined, and reduces IMS pool storage requirements.

- Resources should be accessed consistently and in the same order between application programs. This step reduces the potential for deadlock situations. A result is a reduction in resource utilization for deadlock processing, application dump or abend processing, and for pseudo abends, and application rescheduling.

- IMS batch processing should, in most cases, be BMPs. BMPs run under the control of the IMS control regions, which coordinate resource utilization, backout processing, and so on. This way allows the online transactions to always be available. BMPs should include checkpoint and restart logic to recover from error situations. In a data sharing environment, DL/I processing can introduce many places for errors, problems, and corrupted data.

- Applications should take frequent checkpoints to free locks. The frequency should be adjustable externally to the application code. The application should be able to define a valid unit of work, at which point the checkpoint frequency can be tested and a checkpoint taken if required. The checkpointed application should be restartable. IMS provides extended checkpoint and restart functionality in the application API.

- Application should be serially reusable and follow the standard message processing structure to process multiple messages (Initialize, GU IOPCB, process, loop). This way allows message processing regions to be defined as WFI and PWFI based on review by IMS and performance teams. It can significantly reduce resources to schedule, and increase thread reuse in DB2 connected applications. DB2 thread creation is costly, and
for hi-volume transactions, the reduction can be significant. Additionally, this consideration increases the overall performance of these applications. Failure to follow proper coding techniques in a WFI and PWFI environment can lead to unexpected results (they are expected abends, bad data, incorrect logic execution, and so on).

- Applications should review the IMS status codes and include appropriate checking for status codes that can affect them in the future, for example, Partition Not Available. This review prepares the application for further availability when features such as HALDB and online reorganization are used.

- Applications should be designed to be independent of the environment in which they are executing. They should be independent of system-specific resources that tie them to a system or application software component.

- Applications should be loosely coupled. When information must be exchanged and updated between applications, service and request processing should be used, for example message switching or other mechanisms. Applications should be segregated so they do not directly access each other's databases. This way reduces the ability to move applications within the environment independently. If applications each access the data of the other, they are coupled, creating an affinity to each other.

- Applications should not be designed to depend on a single control record within a database that must be held during processing. This dependency essentially creates a serial processing path for all applications that are dependant on this access to this record. An example is a single control record that must be attained, held for the duration of processing, and updated or incremented. Aside from causing serialization, if the resources are not accessed in the same order, deadlock situations occur, which costs processing, dump processing, and rescheduling overhead.

### 10.9 Shared queues

Significant advantages to shared queues exists, which are primarily automatic workload balancing, increased capacity, and availability.

From a performance perspective, shared queues have the following impacts:

- Increase CPU usage. How much increase depends to a large degree on the speed of the coupling facility and the links.

- Increase log data and I/O operations, because the CQS logging is in addition to IMS logging.

- Increase elapsed times. This increase is most apparent if the synchronous APPC/OTMA function is enabled, which necessitates the use of MVS RRS.

- Facilitate a decrease in log data for any given IMS image by spreading the transaction load.
10.9.1 IMS parameters

The following IMS parameters can directly affect full-function shared-queues performance:

- **QBUF**
  IMS dynamically expands and contracts the number of QBUFs, but it is best to specify a reasonable value. If no value is specified, then the value on the BUFFERS keyword of the MSGQUEUE macro is used, or 255, if BUFFERS is not specified. Although the default of 255 is probably not a bad starting place, you should specify a value for QBUF so there is no question about what value is being used.

- **LGMSGSZ**
  LGMSGSZ size should be large enough to accommodate most, if not all, of your largest messages, if possible. Any messages larger than this size cause additional interaction with the coupling facility, which affects performance.

- **QBUFSZ**
  QBUFSZ is the size of each queue buffer. It should be the same as LGMSGSZ or some multiple. From a performance perspective, there is no real advantage for making this size larger than LGMSGSZ.

- **SHMSGSZ**
  Set SHMSGSZ value to something evenly divisible into QBUFSZ. The basic principle is to have this value large enough to hold most of the messages, which are smaller than those going into LGMSGSZ to make the best possible use of storage in the queue pool.

- **OBJAVGSZ**
  The OBJAVGSZ value sets the ratio of list entries to data elements in the coupling facility, which can impact overflow processing. If ever in doubt, set this value to 512, which yields a 1:1 ratio. We cover this more in the structure size section.

The other parameters, such as QBUFMAX, QBUFHITH, and so on, are normally fine when you use the default.

10.9.2 Structure size

As with other structures, the best place to get sizing information is with the CFSizer website. It is best to avoid overflow processing under normal conditions. The website asks for the SHMSGSZ and LGMSGSZ values and also the high used DRRN for short and long messages. These values allow for normal activity, but it is probably best to consider making the size much larger, because sometimes delays happen for various reasons. Consider factoring in how long you might want to hold messages if, for some reason, you had a locking problem hanging up your regions, or if some destination, perhaps a remote MSC link, is unavailable. Although these cases might be considered abnormal where overflow processing can help, it might be better to be able to handle some period of time before invoking overflow. Also, consider specifying a SIZE and also INITSIZE for the queue structure. This specification allows the structure to be altered to a larger value when it is becoming full. When this happens, a number of messages can be used for automatic notification that there might be a problem.

10.9.3 Structure duplexing

IMS does not provide software duplexing of the queue structures. The same considerations apply to the queue structures as they do to the lock structure. That means from a performance perspective, you want to avoid system-managed duplexing (SMD), but if you
want to be able to rebuild a queue structure without having to also restart CQS in the case of a concurrent failure, you would probably have no choice. Consider how likely it is that you would have a double failure such as that before using SMD.

10.9.4 Overflow

Overflow processing occurs when the threshold specified in CQSSGxxx is met. CQS first tries to alter the primary structure to a larger size if the SIZE value in the CFRM policy is not yet reached. If the primary structure cannot be increased, then the overflow structure, if specified, is allocated and some queues moved to decrease the primary structure usage by 20%. Consider the following information:

- OBJAVGSZ: Only data elements are monitored by CQS, so be sure to set this value low enough to achieve the proper LE/EL ratio. If IMS stops processing because of a structure full condition, that is generally not considered good performance.
- SIZE of overflow structure: Consider making this the same or larger than the primary structure. Doing so can allow the movement of a queue name in the case where one particular queue name is causing the full condition.
- Structure activity is quiesced during overflow processing: Be sure this is only invoked in rare circumstances.

10.9.5 Structure checkpoint

The speed of processors and links keeps improving, which tends to reduce the impact of taking structure checkpoints, but depending on the amount of data in the structure this time can still be several seconds.

- Regarding activity quiesced while in progress, time depends on amount of data.
- Frequency affects time to rebuild queues but also interacts with the amount of MVS logger, which must be kept.
- CFRM couple data sets are accessed by all systems to quiesce and resume, so their performance is extremely important to the checkpoint process.

10.9.6 MVS logger

Note the following performance considerations for the CQS log streams:

- Estimate the volume by looking at the IMS 01 and 03 records. There are other records also, but these make up the large majority of the volume.
- Avoid the use of staging data sets if possible. Use of a structure with duplexing in a data space is the recommended configuration for best performance.
- When duplexing to a data space, be aware that the amount of data, therefore, storage used, in the data space is affected by the frequency of structure checkpoints and the size of the log structure. A low checkpoint frequency and large structure size can cause a high storage demand.
- Set the logger lowoffload value to 0 (zero).
- Use large blocksize such as 64 KB for the log streams.

CQS uses the MVS logger, which in turn uses a CF structure and staging data sets. Depending on your hardware configuration, this might not always be possible, as we discuss with other structures. If you must use staging data sets or if it is even a possibility in the case
of a configuration change, make sure these data sets are on the best possible performing DASD, because they are even more affected than the WADS.

The amount of data that is written to the MVS log can be estimated by looking at the IMS 01 and 03 log records. There are other records also, but these make up the majority of the volume.

Set your logger lowoffload value to 0 (zero).

### 10.9.7 FF scheduling differences

In a shared queues environment, IMS currently has no easy way to know how many messages are on the shared queue. As a result, the standard PARLIM rules cannot be applied and an event called *false scheduling* can occur. The best way to reduce the cost of false scheduling is to try to avoid scheduling in the first place.

Consider the following tips:

- Use (P)WFI when possible to avoid scheduling.
- Put high volume transactions in specific classes.
- Specify MAXRGN, especially for those transactions that are not in unique classes.
- Use PARLIM > 1 to make IMS less reactive and potentially reduce scheduling. Take the average processing time for the transaction into account to be sure it does not cause unnecessary delays, however.

### 10.9.8 FP Parallel Sysplex processing options

The DBFHAGU0 exit routine can be used to set a Parallel Sysplex processing code, which can affect how CQSs process work in a shared queues environment. There are three possible options:

- **Local first**
  
  This option is the default, and most likely the best option. With this option, there is virtually no shared-queues overhead in the case where the transaction can be processed on the inputting system.

- **Local only**
  
  This option avoids any shared-queues overhead but limits availability because there is no queue sharing at all.

- **Global only**
  
  This option always places the message on the shared queue, so you get all the overhead all the time. This option is not the best for performance.

### 10.9.9 OTMA and APPC synchronous shared queues enhancements

IMS 12 adds support for using z/OS cross-system coupling facility (XCF) for communicating between a front-end IMS and a back-end IMS in a shared-queue group for APPC synchronous conversations or OTMA send-then-commit (CM1) transactions with a synchronization level of NONE or CONFIRM. In these situations, IMS is the synchronization point manager instead of z/OS Resource Recovery Services (RRS).
**OTMA**

The IMS 12 synchronous shared queues enhancement provides support for using cross-system coupling facility (XCF) for commit mode 1 (CM1, send-then-commit) transactions with a sync-level of NONE or CONFIRM in the shared queues environment.

This capability removes the Resource Recovery Services (RRS) dependency and provides a simplified solution by using IMS as the sync-point manager for shared queues transactions enhancing performance.

Up to a 40 - 50% internal transaction rate improvement (CPU efficiency) were observed with IMS 12 APPC/OTMA synchronous shared queues environments using XCF, compared with the same environments using RRS.

The IMS 12 OTMA synchronous shared queues enhancement with XCF for CM1 (send-then-commit) provides significant CPU efficiency improvements leading to increased throughput compared to RRS for CM1 transactions with a synchronization level of NONE or CONFIRM. Based on performance evaluations, significant improvements can be seen, such as 63% ITR path-length improvement for SYNCLVL=NONE by using XCFIMS.

**APPC**

Also implemented in IMS 12 is XCF communication within IMS APPC synchronous conversations, with synchronization levels of NONE or CONFIRM, in IMS Shared Queues (SQ) environments.

With this enhancement, you might be able to eliminate Resource Recovery Services (RRS) as a requirement in their shared queue environments, because XCF can be used for the communications between IMS systems in a shared queue group.

As with the OTMA enhancement, also in IMS 12, performance evaluation results have demonstrated that by using XCF communications compared to RRS, significant CPU efficiency improvements leading to increased throughput can be achieved.

**Implementation**

Existing AOS= options in the DFSDCxxx member of IMS PROCLIB are as follows:

- **AOS=N**: Specifies that shared message queue support for synchronous APPC/OTMA is not active. The default is inactive (N).
- **AOS=Y**: Specifies that shared message queue support for synchronous APPC/OTMA is active (Y) and uses RRS.
- **AOS=F**: Activates the function by force (F) even if another member in the IMSplex cannot activate the function.

IMS 12 adds the following options to support XCF (the existing parameters of Y, N, and F are still available and their functionalities are unchanged.):

- **AOS=B**: Requests the use of XCF to communicate between the front-end and the back-end systems for synchronous transactions with synchronization level of NONE and CONFIRM. The processing of requests that are associated with synchronization level of SYNCPNT depends on the RRS specification. If RRS=Y, then transactions such as the synchronization level SYNCPNT transaction can be processed at either a front-end or back-end IMS system using RRS. If RRS=N then transactions with synchronization level SYNCPNT are only processed at the front-end IMS.

- **AOS=S**: Also requests the use of XCF to communicate between the front-end and the back-end systems for synchronous transactions with synchronization level of NONE and CONFIRM along with RRS Multisystem Cascaded Transaction support for synchronous
transaction with synchronization level of SYNCPT. Under this option, SYNCPT request processing is equivalent to AOS=F, which means that if RRS is not active on one system, the systems that specified “S” (force) will still queue incoming transactions globally (without any affinity). Therefore, if a system without RRS tries to process one of these transactions, IMS abends the application with a U711 code.

- **AOS=X**: As with AOS=B and AOS=S, requests the use of XCF to communicate between the front-end and the back-end systems for synchronous transaction with synchronization level of NONE and CONFIRM.

RRS will not be used for synchronous transaction with synchronization level of SYNCPT. In this case, the functionality of AOS=X specification is equivalent to the AOS=N specification.

The B and X parameters are only applicable to the front-end system. The back-end system is not required to specify this parameter. The back-end will invoke either the new XCF message processing or the existing RRS message processing depending upon the message having the XCF indicator or the RRS indicator, respectively.

Choice of B, S, or X is dependent on how syncpoint messages are to be processed. An IMS restart (NRE/ERE) is required to change the AOS= and AOSLOG= settings. DBRC MINVERS of 12.1 is required and all IMS systems in a shared queues group must be at this level to enable the enhancement. IMS systems at an earlier version, using the same RECON data sets, are not able to join the shared queue group. IMS systems at lower versions using a different RECON dataset will still be able to join the shared queue group, but might experience U711. Different AOS parameter settings in different IMS systems might complicate the environment.

IMS 12 also provides an AOSLOG=Y or N parameter in the DFSDCxxx member of proclib to request that a 6701 log record be written for the following cases:

- Response messages that are returned from the back-end system through XCF for transactions with all synchronization levels of NONE, CONFIRM, and SYNCPT
- Error messages that are returned from the back-end system through XCF for transaction with all synchronization levels of NONE, CONFIRM, and SYNCPT
Figure 10-1 shows the flow using XCF and synclevel=NONE processing.

When a message is received, the front-end (FE) IMS control region determines whether the input transaction specifies the synchronization level of NONE or CONFIRM, that XCF enablement is specified (AOS specification is B, S, or X), and queues the APPC synchronous conversation or OTMA CM1 input transaction message into the shared queue structure with a newly defined indicator. Any back-end (BE) IMS control region that processes this input will recognize the new indicator and start using XCF to send and receive messages to and from the originating front-end IMS control region. The front-end IMS system is notified when a transaction response is available.

The following steps describe the flow:

1. A client application enters an OTMA CM1 or APPC synchronous conversation transaction with synchronization level NONE to the front-end IMS system.
2. The input transaction message with the XCF indicator is put on the shared queue by using the existing transaction-ready queue type.
3. A dependent region on the back-end IMS system retrieves and processes the input message.
4. The back-end IMS system sends the transaction response to the front end and waits for confirmation.
5. The front end forwards the transaction response to the client application.
6. The front end sends the receive-confirmation to the back end and waits for the commit completion.
7. The back end receives the receive-confirmation, completes the commit process and sends back the commit completion message to the front-end.

8. The front end forwards the commit completion to the client application.

In Figure 10-2 you can follow processing for XCF, using synclevel=CONFIRM.

![Diagram](image)

The FE IMS queues the synchronous synclevel CONFIRM message into the SQ structure with a newly defined indicator and waits to be notified when the response is available.

Synchronous requests with synclevel=CONFIRM differ slightly from the previous flow because of an extra flow from the client. The following steps describe this scenario:

1. A client application enters OTMA CM1 or APPC synchronous conversation transaction with synclevel=CONFIRM from the front-end IMS system.

2. The input transaction message with the XCF indicator is put on the shared queue.

3. A dependent region on the back-end IMS system retrieves and processes the input message.

4. The back-end IMS system sends the transaction response to the front-end and waits for an ACK.

5. The front end forwards the transaction response to the client application.

6. The client application returns an ACK to acknowledge receipt of the output response.

7. The front end forwards the ACK to the back end and waits for a commit completion.

8. The back end receives the ACK, completes the commit process and sends back the commit completion message to the front end.

9. The front end forwards the commit completion message to the client application.
The AOSLOG parameter is applicable only to the front-end system. The back-end system is not required to specify this parameter.

The /DIAGNOSE command, retrieves diagnostic information for system resources such as IMS control blocks, user-defined nodes, or user-defined transactions at any time without taking a console dump. In IMS 12, the command is enhanced to allow or disallow the capture of events that are related to APPC and OTMA synchronous transactions in a shared-queues environment. When enabled, the AOSLOG events are written to the OLDS as type X'6701' records. If AOSLOG(ON) is specified in a non-shared queues environment or when AOS=N is specified in the DFSDCxxx PROCLIB member, the command is rejected with a DFS2859I message.

The use of XCF rather than RRS in a shared queues environment for synchronous CM1 synchronization level of NONE or CONFIRM messages simplifies syncpoint processing and eliminates unnecessary RRS overhead.
Capturing documentation for performance diagnosis

This chapter describes ways of capturing documentation for IMS performance analysis. The chapter contains the following topics:

- Documentation for defect-related software issues
- Procedures to capture IMS documentation
- Procedure to request a IMS performance health check
11.1 Documentation for defect-related software issues

The IMS Technical Support Center assists with diagnosis of defect related software issues. This most recent information is at the following website:


Assistance with non-defect performance and tuning aspects of a system is available through our IMS Technical Advocate team on a contractual basis.

If you suspect that a code defect might be causing performance issues, be prepared to define the performance issue seen in your IMS environment as follows:

- A clear definition of the problem being reported along with the exact time periods of the performance issue:
  - Start times
  - End times
  - Duration of events
- The name of the tool, report, fields, field definitions, field contents, and so on, that are used by the customer to determine that there was a performance related issue.
- Performance data collected from one of the performance related windows. This data should be from the same time period for all data sets.

The raw data sets are required along with any formatted reports in all of the following cases. Provide the data set names for the following recommended data:

- SLDS during performance issue with at least two statistics checkpoints, which is required for IMSPA report analysis. The SLDS should also span a period of time without the slowdown.
- SMF data dump (all SMF records) for the slow performance period. The SMF data should also span a period of time without the slowdown.
- A dump from during the period of poor performance. The IMS Technical Support team can assist with creation of a specialized dump command if one hasn't already been set up for your system.
- IMS Monitor data set (if it can be re-created). Although this is suggested, it might not be obtainable in all cases. The monitor should be run during the same period as the identified slowdown.
- Exported Application Performance Analyzer for z/OS (APA) sample data from the relevant time frame.

Simple checkpoint: If locking issues are suspected then a simple checkpoint versus a statistics checkpoint should be used to capture the IRLM statistics.

Raw monitor data sets: In addition to the monitor report examples showing the problem, we need the raw monitor data set (or sets) so that we can use the data set (or sets) as input to a variety of tools.

If a tool with similar function to APA is used, such as STROBE, send STROBE reports for the region experiencing the performance issue during the problem period. Include a STROBE report of the same region outside the problem period.
Chapter 11. Capturing documentation for performance diagnosis

11.2 Procedures to capture IMS documentation

The section can assist you with capturing IMS performance diagnostic documentation.

11.2.1 The IMS Monitor trace

Capture IMS Monitor tracing data as follows:

- Use the ALL option for the trace command, it includes the following settings:
  - APDB
    - Monitors the activity between application programs and databases, including Fast Path activity. Monitoring includes all application program requests to external subsystem databases.
  - APMQ
    - Monitors the activity between application programs and message queues, including Fast Path activity.
  - LA
    - Monitors the line and logical link events. Immediately before starting the IMS Monitor, take an IMS simple checkpoint (/CHE).
  - SCHD
    - Monitors the scheduling and termination events, including Fast Path activities.
  - Immediately after stopping the IMS Monitor, take another IMS simple checkpoint (/CHE).

For more detail about running this trace, see IBM Technical Support Technote Providing documentation to IMS technical support for system-wide defect-related performance issues:


This Technote also includes information regarding useful SLIP traps and GTF traces.

- If specific transactions are involved, provide the following information:
  - The transaction name
  - The transaction classes
  - The IMS recovery token for the affected transactions (available from the IMS log records)

Requests for non-defect performance and tuning assistance on a contractual basis from IMS Technical Advocate team can be sent through email to Jeff Hook (mailto:jhook@us.ibm.com).
Example 11-1 shows how to start and stop the IMS Monitor.

Example 11-1   Starting and stopping the IMS Monitor

/ CHE
/TRACE SET ON MON ALL
(trace interval)
/TRACE SET OFF MON
/ CHE
or use the following sequence of commands
/ CHE
/TRACE SET ON MON ALL INTERVAL #seconds
/ CHE

11.2.2 IMS in storage table traces

Turn on the IMS dispatcher, scheduler, DL/I, and lock traces to capture general system diagnostics and database access history.

Specify the following options in the IMS PROCLIB member DFSVSMxx:

DISP=ON,SCHD=ON,DL/I=ON,LOCK=ON

Otherwise, use the following IMS command, where nnnn is alternately DISP, SCHD, DL/I, and LOCK:

/TRA SET ON TABLE nnnn

11.2.3 Running and externalizing IMS lock and DL/I traces

Allocate DFSTRA01 and DFSTRA02 data sets:

DFSTRA01 and DFSTRA02 are the external trace data sets that are used by an IMS online system. These two data sets are used when the trace table OUT parameter is used on the OPTIONS,nnnn=OUT statement of the DFSVSMxx control cards or when the LOG parameter is used on the /TRACE SET ON TABLE nnnn OPTION LOG command. The data sets are used in a wrap-around fashion:
– When DFSTRA01 fills, DFSTRA02 is used
– When DFSTRA02 fills, DFSTRA01 is used.

Use unique trace data set names for each IMS online system:
– hlq.imsid.DFSTRA01
– hlq.imsid.DFSTRA02

Allocate the DFSTRA01 and DFSTRA02 trace data sets for each IMS online system as a single extent (do not use a secondary space allocation) by using the following attributes:
– DSORG Sequential
– RECFM VB
– LRECL 4016
– BLKSIZE 20084*

The BLKSIZE attribute must be a multiple of the LRECL with an additional 4 bytes for the block descriptor word.
For a large production IMS system, allocate each data set the size of a full 3390 pack so as to minimize the number of switches during an IMS online day. If an IMS system has a lower activity rate, then half of the 3390 pack might be plenty of space for a trace data set. In other words, you should size the trace data sets based on the activity rate in your IMS online system. Do not allocate any other high activity data sets on the same volumes as the trace data sets.

Set up an DFSMDA dynamic allocation member for each trace data set in an appropriate system library.

**Activate lock and DL/I tracing**

To verify and test that external tracing will work, manually activate DL/I and LOCK external tracing and manually stop the traces after a 10 - 15 minute sampling period. It might be useful to observe how full the trace data set is and extrapolate to get an idea of how long it might take to fill a 3000-plus cylinder trace data set.

Issue the following commands:

/TRACE SET ON TABLE DL/I OPTION LOG VOLUME HIGH
/TRACE SET ON TABLE LOCK OPTION LOG VOLUME HIGH
/DISPLAY TRACE XTRC

(trace interval)

/TRACE SET OFF TABLE DL/I OPTION LOG
/TRACE SET OFF TABLE LOCK OPTION LOG
/DISPLAY TRACE XTRC

**External DL/I and lock traces in an IMS online system**

The OPTIONS statement of the DFDSVSMxx parameter allows you to specify which IMS traces should be activated automatically by the IMS system during initialization. Or you can choose to activate the traces with the /TRACE SET ON TABLE nnn OPTION LOG command after IMS system startup is completed.

To activate the external traces at IMS initialization time, modify the DFSVSM** OPTION statement to include the following information:

OPTIONS,DLI=OUTHIGH,LOCK=OUTHIGH

Or, if you prefer, an automated operator (AO) rule can be implemented for each of the IMS online systems to activate the DL/I and LOCK traces immediately after the control region STARTUP COMPLETED message is issued:

In the following line, rtype is the restart type and imsid is the IMS identifier:

DFS994I rtype START COMPLETED. imsid

Issue the following commands:

/TRACE SET ON TABLE DL/I OPTION LOG VOLUME HIGH
/TRACE SET ON TABLE LOCK OPTION LOG VOLUME HIGH
/DISPLAY TRACE XTRC

As a safety measure, an AO rule should be implemented to issue a display command to check the external trace status approximately every 90 minutes to verify that external tracing is functioning.
Issue the following command:

/DISP TRACE XTRC
VERIFY: DFS4444I DISPLAY FROM ID=DBD1
IMS ACTIVE TRACES
IMS EXTERNAL TRACE IS TRACING TO DISK
XTRC DDNAME ALLOC STATUS STATUS
DFSTRA01 DYNA ALLOCATED ACTIVE
DFSTRA02 DYNA ALLOCATED OPEN
DFSTRAOT DYNA UNALLOCATED CLOSED
IF: DFSTRA01 and DFSTRA02 both show "UNALLOCATED"
THEN ISSUE:
/TRACE SET ON TABLE DL/I OPTION LOG VOLUME HIGH
/TRACE SET ON TABLE LOCK OPTION LOG VOLUME HIGH
/DISP TRACE XTRC

Archive trace data
A GDG can be defined for a trace archive data set for each IMS online system. Define the
GDG with a LIMIT to retain a minimum of at least 24 hours worth of trace data for each IMS
online system. If a virtual tape server (VTS) is available, allocate the data set to a VTS
storage class; if VTS is not available, allocate the data set to a storage class, which
immediately goes to hierarchical storage management (HSM) migration level 2 (ML2).

Use unique trace archive data set names for each IMS online system:

hlq.imsid.TRACE.ARCHIVE(0)

An AO rule can be implemented for each IMS online system to automatically initiate an
archive of the filled DFSTRAxx data set when tracing is switched to the alternate trace data
set. The rule should save a “switched to” trace data set identifier in a global AO variable (that
is "01" or "02") for use at system shutdown. The filled DFSTRAxx data set can be copied to
the GDG data set by using a simple IEBGENER or another copy utility. An AO rule is shown in
the next example:

WHEN:
DFS2864I EXTERNAL TRACE DATA SET xxxxxxxx FULL - SWITCHING TO yyyyyyyy. imsid
START: AO Rule to initiate job to copy using iebgener, filled trace data set off
to trace archive data set

Create a started task or a JOB which executes a copy utility to archive a filled trace data set.
Trace data set information from the DFS2864I message can be used for substitution as part
of the input data set name. The IMSID from the message can be used as part of the output
data set name.
**IMS shutdown**

The IMS online system shutdown procedure can be modified to display and stop the DL/I and LOCK traces before the last checkpoint. The following rule is an example:

WHEN: IMS Online Shutdown starts  
ISSUE:/DISPLAY TRACE XTRC  
VERIFY: DFS4444I DISPLAY FROM ID=imsid  
IMS ACTIVE TRACES  
IMS EXTERNAL TRACE IS TRACING TO DISK  
XTRC DDNAME ALLOC STATUS STATUS  
DFSTRA01 DYNALLOCATED ACTIVE  
DFSTRA02 DYNALLOCATED OPEN  
DFSTRA0T UNALLOCATED CLOSED  
ISSUE:  
/TRACE SET OFF TABLE DL/I OPTION LOG  
/TRACE SET OFF TABLE LOCK OPTION LOG  
/DISPLAY TRACE XTRC

An AO rule can be implemented to kick off an archive started task or JOB for the last used trace data set after the trace data sets are DEALLOCATED. Using the saved “switched to” global AO variable can help you to archive the last active DFSTRAxx data set so that no trace records are lost.

WHEN: DFS2500I DATASET DFSTRA01 SUCCESSFULLY DEALLOCATED  
DFS2500I DATASET DFSTRA02 SUCCESSFULLY DEALLOCATED  
START: AO Rule to initiate job to gener 'switched to' trace dataset off to trace archive dataset

**Important:** Do not start external tracing in the restarted IMS online system if the previous trace archive started task or job does not complete successfully. When IMS is started or restarted, it will always begin collecting externalized trace records on DFSTRA01. There is a small window where you might loose externalized trace records by overwriting a DFSTRA01 trace data set which still needs to be archived.

**IMS abend**

An AO rule can be implemented to start an archive-started task or job for the last used trace data set when the IMS online system abends. Using the saved “switched to” global AO variable will enable you to archive the last active DFSTRA** data set so that no trace records are lost.

WHEN: DFS3896I ATTEMPTING TO PURGE OLDS BUFFERS  
START: AO Rule to initiate job to gener 'switched to' trace dataset off to trace archive dataset

**Important:** Do not start external tracing in the restarted IMS online system if the previous trace archive started task or job does not complete successfully. When IMS is started or restarted, it will always begin collecting externalized trace records on DFSTRA01. There is a small window where you might loose externalized trace records by overwriting a DFSTRA01 trace data set, which still needs to be archived.
z/OS crash (and running FDBR across LPARs)
An AO rule can be implemented to start an archive started task or job for the last used trace data set if the MVS system crashes. Using the saved “switched to” global AO variable will enable you to archive the last active DFSTRAxx data set so that no trace records are lost.

WHEN: DFS4166I FDR FOR (imsid) DB RECOVERY PROCESS STARTED, REASON = reasonid
START: AO Rule to initiate job to generate 'switched to' trace dataset off to trace archive dataset

**Important:** Do not start external tracing in the restarted IMS online system if the previous trace archive started task or job does not complete successfully. When IMS is started or restarted, it will always begin collecting externalized trace records on DFSTRA01. There is a small window where you could loose externalized trace records by writing over a DFSTRA01 trace data set which still needs to be archived.

**Tracing for DL/I batch jobs**
Some customers run DL/I batch jobs. If these jobs have update capability, then a good plan is to capture DL/I and LOCK traces for them also, especially if they run in an IMS n-way data-sharing environment. The trace data can be collected in the IEFRDER data set (IMS Log).

To activate DL/I and LOCK tracing for a DL/I batch job, modify the OPTIONS statement in DFSVSAMP to include the following information:

```plaintext
OPTIONS,DL/I=OUTHIGH,LOCK=OUTHIGH
```

**Suggested performance benchmark test**
Work with your performance analyst to set up a benchmark test to measure the overhead and evaluate the impact of running the DL/I and LOCK TRACEs in a busy IMS online system:

1. Run a five-minute Application Performance Analyzer for z/OS or similar product sampling for IMS control region (busiest IMS), and review the reports to identify the base resource usage.
2. Turn on Application Performance Analyzer for z/OS or similar product sampling for five minutes or 30000 samples.
   Start IMS Monitor, run the monitor for five minutes, stop the monitor, run the monitor report:
   ```plaintext
   /TRACE SET ON MONITOR ALL
   /TRACE SET OFF MONITOR ALL
   ```
   Review the reports to identify the resource usage when the IMS Monitor is active.
3. Turn on Application Performance Analyzer for z/OS or similar product sampling for five minutes or 30000 samples.
   a. Activate DL/I and lock traces:
      ```plaintext
      /TRACE SET ON TABLE DL/I OPTION LOG VOLUME HIGH
      /TRACE SET ON TABLE LOCK OPTION LOG VOLUME HIGH
      /DISPLAY TRACE XTRC  (Use this command to display TRACE status)
      ```
   b. Start IMS Monitor, run monitor for five minutes, stop monitor, and run the monitor report:
      ```plaintext
      /TRACE SET ON MONITOR ALL
      /DISPLAY TRACE ALL
      /TRACE SET OFF MONITOR ALL
      ```
c. Deactivate DL/I and lock traces:
   /TRACE SET OFF TABLE DL/I OPTION LOG
   /TRACE SET OFF TABLE LOCK OPTION LOG

d. Review the reports to identify the resource usage when the IMS Monitor and Tracing is active.

4. Turn on Application Performance Analyzer for z/OS or similar product sampling for five minutes/30000 samples
   a. Activate DL/I and lock traces
      /TRACE SET ON TABLE DL/I OPTION LOG VOLUME HIGH
      /TRACE SET ON TABLE LOCK OPTION LOG VOLUME HIGH
      /DISPLAY TRACE XTRC (Use this command to display TRACE status)
   b. Deactivate DL/I and lock traces
      /TRACE SET OFF TABLE DL/I OPTION LOG
      /TRACE SET OFF TABLE LOCK OPTION LOG
   c. Review the reports to identify the resource usage when the IMS Tracing is active.

5. Review IMS Monitor reports
   Review the reports for IMS Control Region sampling.

   The actual tracing overhead on each IMS online system can vary, depending a lot on the workload profile.

### 11.2.4 Storage as a moving target

Storage can be a moving target while being dumped, especially when multiple address spaces interact. Consider the following information:

- If the z/OS CHNGDUMP SDUMP option (Q=Y) is specified, all address spaces that are not exempt from system-wide non-dispatchability are set as non-dispatchable while global storage (CSA, SQA) is dumped.
  
  This way can increase dump capture time and should be used with careful consideration.

- In all cases (Q=Y or Q=N), dumped address spaces are made non-dispatchable while global storage and their own private region storage is being dumped.

- There is no intentional dispatching co-ordination between requested address spaces in a dump.

  All private storage is dumped through scheduling of service request blocks (SRBs) to the requested address spaces:
  
  - This SRB activity is asynchronous.
  - Multiple SRBs to multiple address spaces run independently and loosely in parallel.
11.2.5 The z/OS DUMP command

There are various ways to get an IMS SVC dump:

- Use DUMP command parmlib member:
  
  z/OS SYS1.PARMLIB member IEADMCxx

- Use SLIP command parmlib.
  
  z/OS SYS1.PARMLIB member IEASLPxx

- Manual z/OS DUMP command from the console.

DUMP command parmlib member

Consider the following information:

- Create members called IEADMCxx for each customized dump command in the z/OS SYS1.PARMLIB.
- The command can be used to customize IMS dumps prior to error event and has a simpler operator interface.
- All jobs must be active or no dump is produced on the system on which the dump command is entered. Remote systems without a job name match will dump the DUMPSRV address space.
- Wildcard characters can be used to specify the job names:
  - * (multiple characters)
  - ? (single character)
- More information is in the following sources:
  - z/OS MVS Initialization and Tuning Guide, SA22-7591
  - z/OS MVS System Commands, SA22-7627.
- The following example is of a SYS1.PARMLIB member named IEADMCI1, which contains the following DUMP parameters:

  JOBNAME=(j1,j2,j3,j4,j5,j6,j7),
  SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,ALLNUC,LPA,GRSQ)

  The parameter values are as follows:

  j1 IMS Control Region Jobname
  j2 DBRC Region Jobname
  j3 IMS DLI Region Jobname
  j4 IRLM Region Jobname
  j5 IMS CCTL Region 1
  j6 IMS CCTL Region 2
  j7 IMS CCTL Region 3

- To request a dump from using IEADMCI1 parmlib member, enter the following z/OS command:

  DUMP TITLE=(DUMP OF IMS and CCTL Regions ),PARMLIB=(I1)

- A dump data set will be created on the z/OS image from which the dump command is entered, containing the specified address spaces.
SLIP command parmlib member

Consider the following information:

- Create a SYS1.PARMLIB member called IEASLPxx with the SLIP commands.
- Jobs do not need to be active for dump to be produced.
- Wildcards characters can be used to specify the job names:
  - * (multiple characters)
  - ? (single character)
- Can be used to customize IMS dumps prior to error event.
- Has a simple operator interface.
- Must ensure no other PER SLIP is active on all systems before activating.

The SYS1.PARMLIB member, named IIEASLPI1, contains the following SLIP entries:

```
SLIP SET,IF,N=(IEAVEDS00,00,FF),A=(SVCD),ID=IMS1,
SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,ALLNUC,LPA,GRSQ),
JL=(j1,j2,j3,j4,j5,j6,j7),D,END
```

The parameter values are as follows:

- **j1** IMS Control Region Jobname
- **j3** DBRC Region Jobname
- **j2** IMS DLI Region Jobname
- **j4** IRLM Region Jobname
- **j5** CCTL Region 1
- **j6** CCTL Region 2
- **j7** CCTL Region 3

Before activating the SLIP, be sure to disable any existing PER SLIP:

```
SLIP MOD,D,DISABLE,ID=trapid
```

To activate SLIP trap and trigger the associated SVC dumps, enter the following z/OS commands:

```
SET SLIP=I1
SLIP MOD,D,ENABLE,ID=IMS1
```

A dump data set will be created on the z/OS image from which the SLIP command is entered, containing the specified address spaces.

Manual z/OS DUMP command from the console

This example issues the DUMP command from the z/OS console.

To capture an SVC Dump, issue the following command:

```
DUMP COMM=('dump title')
```

In the command, 'dump title' is a descriptive comment of your choice.

Use the REPLY command to respond to message IEE094D:

```
R xxx,JOBNAME=(j1,j2,j3,j4),SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,ALLNUC,LPA,GRSQ),END
```

The command uses the following values:

- **xxx** = Reply number for the IEE094D message
- **j1** IMS Control Region Jobname
- **j2** DBRC Region Jobname
- **j3** IMS DLI Region Jobname
- **j4** IRLM Region Jobname
IMS SYSPLEX dump considerations

In IMS SYSPLEX implementations, consider the possibility that a performance problem on one IMSplex member might be because of a problem originating from another member. It should include SVC dumps from other members of the IMSplex as part of the documentation.

The challenge is to ensure that a dump is used for all necessary address spaces on each system.

The following example shows the creation of IEADMCI1 and IEADMCI2 members in SYS1.PARMLIB to capture two dump data sets on each z/OS image in the sysplex, matching the REMOTE specifications for the JOBNAMEs:

member IEADMCI1
 JOBNAME=(job1,job2,job3,job4),
 SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ),
 REMOTE=(SYSLIST='job1','job2','job3','job4',SDATA)
   job1 = IMS Control Region Jobname
 job2 = IMS DLI Region Jobname
 job3 = DBRC Region Jobname
 job4 = IRLM Region Jobname (If IRLM DB Locking is used)

member IEADMCI1
 JOBNAME=(job5,job6,job7),
 SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQ,GRSQR),
 REMOTE=(SYSLIST='job5','job6','job7',SDATA)
   job5 = CCTL Region 1
 job6 = CCTL Region 2
 job7 = CCTL Region 3

To activate, use the following command:

DUMP TITLE=(IMS/CCTL sysplex DUMPS),PARMLIB=(I1,I2)

The following example creates IEASLPxx member in SYS1.PARMLIB to capture two dump data sets on each z/OS image in the sysplex, matching the REMOTE specifications:

member IEASLPxx #1
 SLIP SET,IF,N=(IEAVEDS0,00,FF),A=(SYNSVCDD,TARGETID),
 SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRS),
 JOBLIST=(job1,job2,job3,job4), ID=IMS1,TARGETID=(IMS2),
 REMOTE=(JOBLIST,SDATA),D,END
   job1 = IMS Control Region Jobname
 job2 = IMS DLI Region Jobname
 job3 = DBRC Region Jobname
 job4 = IRLM Region Jobname (If IRLM DB Locking is used)

member IEASLPxx #1
 SLIP SET,IF,N=(IEAVEDS0,00,FF),
 JOBLIST=(job5,job6,job7), ID=IMS2,
 SDATA=(CSA,PSA,RGN,SQA,SUM,TRT,GRSQR),
 REMOTE=(JOBLIST,SDATA),D,END
To activate, use the following steps:

1. Ensure all PER SLIPS are disabled by issuing the following command:
   
   ```
   ROUTE *ALL,SLIP,MOD,DISABLE,ID=trapid
   ```

2. Issue the following command:

   ```
   SET SLIP=xx
   SLIP MOD,ENABLE,ID=IMS1
   ```

### Avoid obtaining a partial dump

Ensure adequate CHNGDUMP MAXSPACE is specified to hold the internal SVC dump. Use the COMMNDxx SYS1.PARMLIB member to issue the appropriate CHNGDUMP command at IPL time:

```
CD SET,SDUMP,MAXSPACE=3000M - Default size is 500M
```

The value of 3000M is standard for large multi-address space SVC Dumps. See the following resources for additional details:

- /OS MVS Diagnosis: Tools and Service Aids, SY28-1085
- /OS MVS Initialization and Tuning Guide, SA22-7591
- /OS MVS System Commands, SA22-7627

Ensure local page data sets are large enough to contain their normal peak load plus additional SVC dumps.

#### 11.2.6 GTF trace

When you activate GTF, it operates as a system task, in its own address space. The only way to activate GTF is to enter a START GTF command from a console with master authority. Using this command, the operator selects either IBM or your cataloged procedure for GTF. The cataloged procedure defines the GTF operation; you can accept the defaults that the procedure establishes, or change the defaults by having the operator specify certain parameters on the START GTF command.

Because GTF sends messages to a console with master authority, enter the command only on a console that is eligible to be a console with master authority. Otherwise, you cannot view the messages from GTF that verify trace options and other operating information.

IBM supplies the GTF cataloged procedure, which resides in SYS1.PROCLIB. Such a procedure is shown in the following example:

```
//GTF PROC MEMBER=GTFPARM
//IEFPROC EXEC PGM=AHLGTF,PARM='MODE=EXT,DEBUG=NO,TIME=YES',
// TIME=1440,REGION=2880K
//IEFRDER DD DSN=SYS1.TRACE,UNIT=SYSDA,SPACE=(TRK,20),
// DISP=(NEW,KEEP)
//SYSLIB DD DSN=SYS1.PARMLIB(&MEMBER),DISP=SHR
```

To invoke GTF, enter the START command:

```{START|S}{GTF|membername}.identifier.
```

To stop GTF processing, enter the STOP command. The STOP command must include either the GTF identifier specified in the START command, or the device number of the GTF trace data set if you specified MODE=EXT or MODE=DEFER to direct output to a data set.
If you are not sure of the identifier, or the device number of the trace data set, enter the 
**DISPLAY A, LIST** command. The results of the command are as follows:

```
DISPLAY A, LIST
IEE114I 14.51.49 1996.181 ACTIVITY FRAME LAST F E SYS=SY1
JOBS M/S TS USERS SYSAS INITS ACTIVE/ MAX VTAM OAS
00000 00000 00016 00000 00000/00000 00000
LLA LLA LLA NSW S VLF VLF VLF NSW S
JES2 JES2 IEFPROC NSW S
GTF EVENT1 IEFPROC NSW S
```
The STOP command has the following general syntax:

```
{STOP|P} identifier
```

**Input/output supervisor (IOS) GTF Trace**

Start GTF with a 1000 cylinder data set by using the following options:

```
TRACE=SSCHP, IOXP, CCWP, MSCH, HSCH, XSCH, CSCH, PCI
Prompt: SSCH=IO=(xxxx, yyyy), CCW=(SI, IOSB, DATA=50, CCWN=10), END
```

In the second line, `xxxx, yyyy` specifies the IMS WADS device numbers for the IMS being traced (for example: 0900,0960).

The following steps start the trace:

1. Issue the `S GTF.TRACE` command.
2. GTF issues the following prompt on the console:

   `*01 AHL125A RESPECIFY TRACE OPTIONS OR REPLY U`

3. Issue the following command to the outstanding reply number 0:

   ```
   R 01, TRACE=SSCHP, IOXP, CCWP, MSCH, HSCH, XSCH, CSCH, PCI
   ```

4. GTF issues the following message on the console:

   `*02 AHL101A SPECIFY TRACE EVENT KEYWORDS -- IO=, SSCH=, CCW=`

5. Issue the following command to the outstanding reply number 02:

   ```
   R 02, SSCH=IO=(xxxx, yyyy), CCW=(SI, IOSB, DATA=50, CCWN=10), END
   ```

   In the command, `xxxx, yyyy` specifies the IMS WADS device numbers for the IMS that is being traced (for example: 0900,0960).

6. GTF now issues the following messages on the console:

   ```
   AHL103I TRACE OPTIONS SELECTED -- PCI, MSCH, HSCH, CSCH
   AHL103I IOXP, XSCH, IO=SSCH=(xxxx, yyyy)
   AHL103I CCW=(SI, IOSB, CCWN=10, DATA=50, PCITAB=1)
   AHL125A RESPECIFY TRACE OPTIONS OR REPLY U.
   *03 AHL125A RESPECIFY TRACE OPTIONS OR REPLY U
   AHL031I GTF INITIALIZATION COMPLETE
   ```
11.2.7 The /DIAGNOSE command

The /DIAGNOSE command retrieves diagnostic information for system resources, such as IMS control blocks and user-defined resources, at any time without creating a disruptive situation.

IMS 12 improves the reliability of diagnostic information by supporting additional control blocks for /DIAG SNAP command. The overall benefit of these enhancements is an improved reliability of diagnostic information, and, ultimately, to streamline the problem determination process.

Overview of the /DIAGNOSE command

Users should be able to gather diagnostic information without affecting the normal business operations. In most cases, gathering a console dump in a productive environment is disruptive and can negatively affect the service-level agreement (SLA). Therefore, users are obligated to produce these dumps off the business prime hours.

The /DIAGNOSE command alleviates this situation by allowing you to take a snapshot of IMS system resources at any time without affecting system availability. The /DIAGNOSE command enables users to retrieve diagnostic information for system resources, such as IMS control blocks, user-defined nodes, or user-defined transactions, at any time without taking a console dump.

The /DIAGNOSE command SNAP function

The /DIAGNOSE command SNAP function takes a current snapshot of system resources and displays the response into the issuing LTERM. Optionally, the resource information can be sent to either an online log data set (OLDS) or trace data sets as type X'6701' log records.

The SNAP function captures information for the following resources:

- A specific IMS control block
- A user-defined resource
- Primary control blocks for a dependent region
- Any area of storage within the region address space
- Prolog information for an IMS load module
- A user-defined shared queues structure

The SNAP function provides a non-intrusive way to capture IMS resources and control blocks. Using this command can decrease the time needed to generate problem determination data.

The SNAP function takes a current snapshot of system resources at any time without negatively impacting the IMS system. The SNAP function of the /DIAGNOSE command captures storage information and shows information about the issuing LTERM. Optionally, the resource information can be sent to either an OLDS or trace data sets as type X'6701' log records.

The /DIAGNOSE command is a standard type-1 command. It can be issued from an IMS terminal, a console WTOR, APPC and OTMA clients, an AOI program, MCS/EMCS consoles, and any OM command clients including SPOC.

The SNAP function captures information for the following areas:

- A specific IMS control block. The /DIAGNOSE SNAP BLOCK(CSCD) command captures storage information for the APPC/OTMA SMQ SCD Extension control block.
- A user-defined database, communication line, logical link, node, program, transaction, logical terminal (LTERM), or USER.
- Primary control blocks for a dependent region.
Any area of storage within the control region address space (by specifying the address of that storage area).

Prolog information for an IMS load module. The /DIAGNOSE SNAP MODULE(modname) command identifies the entry point address and captures prolog information for the specified IMS module. The prolog information contains the current maintenance level for a module on your system, which can help you to determine if any maintenance is missing.

A user-defined shared queues structure. The /DIAGNOSE SNAP STRUCTURE(structurename) command captures storage information for the DFSSQS control block storage for the specified shared queue structure.

The SNAP function of the command captures storage, both addresses and raw data, for the requested IMS control blocks and resources. The information in the blocks is copied to a copy storage area to avoid holding enqueues, locks, latches, and others. The environment is further protected by a separate ESTAE routine that protects the copy process and also prevents an IMS failure.

You can also use the SNAP function to do the following tasks:

- Show filtered resource information captured by the SNAP function.
- Specify a limit for the number of lines to display.
- Specify the control blocks to be captured by the SNAP function.

Valid environments for commands and keywords
Table 11-1 shows the valid environments for commands and keywords.

<table>
<thead>
<tr>
<th>Commands or keywords</th>
<th>DB/DC</th>
<th>DBCTL</th>
<th>DCCTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>/DIAGNOSE</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>ADDRESS</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>AOSLOG</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>AREA</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>BLOCK</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DB</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>JOBNAME</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LINE</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>LINK</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>LTERM</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>MODULE</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>NODE</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>OPTION</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PGM</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>REGION</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SET</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>SHOW</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
The /DIAGNOSE SNAP command is enhanced in IMS 12 to improve the reliability of diagnostic information and to streamline the problem determination process. The /DIAGNOSE command supports the following additional control blocks:

- **AREA(areaname)** Captures control block information for the Fast Path area.
- **DB(dbname)** Captures control block information for the database.
- **LINE(linenumber)** Captures control block information for the communication line.
- **LINK(linknumber)** Captures control block information for the logical link specified.
- **PGM(pgmname)** Captures control block information for the program.
- **REGION(regionnumber)** Captures control block information for the dependent region.

The /DIAGNOSE command SNAP function has the following options:

- A DISPLAY option to route output back to the issuing LTERM
- A LIMIT option to restrict the number of lines of output going to LTERM
- A SHOW parameter to control the type and amount of output produced

For detailed information, see *IMS Version 12 Commands, Volume 1: IMS Commands A-M*, SC19-3009.

### Using the /DIAGNOSE command and keywords

Use the /DIAGNOSE command to retrieve diagnostic information about system resources at any time. Also use it to enable and disable diagnostic features such as the facility to capture the events that are related to APPC and OTMA synchronous transactions in a shared-queues environment.

The keywords that are introduced by IMS 12 are listed in this section.

### The /DIAGNOSE SNAP AREA command

The /DIAGNOSE SNAP AREA command captures control block information for the Fast Path area that is specified in the `areaname` parameter. The `areaname` parameter that is specified must be alphanumeric, no longer than eight characters, and identify a currently defined Fast Path area. Multiple `areaname` parameters can be specified with each parameter, separated by a comma or a blank.

The /DIAGNOSE SNAP AREA() command is available in a DB/DC or DBCTL environment where Fast Path is defined. The DEDB extended area control block (EMAC) is available only in an RSR tracker environment.
Table 11-2 lists the /DIAGNOSE SNAP AREA() control blocks.

Table 11-2 The /DIAGNOSE SNAP AREA() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADSC</td>
<td>DEDB area data set control block</td>
<td>DBFADSC</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>ALDS</td>
<td>DEDB area name list entry</td>
<td>DBFAREA</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DDIR</td>
<td>Database directory block</td>
<td>DFSDDIR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMAC</td>
<td>DEDB area control block</td>
<td>DBFDMAC</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMAX</td>
<td>DMAC ERE extension block</td>
<td>DBFDMHV</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMCB</td>
<td>DEDB master control block</td>
<td>DBFDMCB</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMHR</td>
<td>DEDB buffer header (SDEP)</td>
<td>DBFDHR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMSL</td>
<td>Data space map list</td>
<td>DBFDMSL</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DSME</td>
<td>Data space mapping entry</td>
<td>DBFDSME</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EMAC</td>
<td>DEDB extended area control block</td>
<td>DBFEMAC</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>MRMB</td>
<td>DEDB randomizing module block</td>
<td>DBFDMRB</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

The /DIAGNOSE SNAP DB command

The /DIAGNOSE SNAP DB command captures control block information for the database that is specified in the dbname parameter. The dbname parameter specified must be alphanumeric, no longer than eight characters, and identify a currently defined database. Multiple dbname parameters can be specified with each parameter separated by a comma or a blank.

The /DIAGNOSE SNAP DB command is available in a DB/DC or DBCTL environment. If the /DIAGNOSE SNAP DB command is issued in a DCCTL environment, a DFS110I error message is issued in response.

Table 11-3 lists /DIAGNOSE SNAP DB control blocks.

Table 11-3 The /DIAGNOSE SNAP DB() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDIR</td>
<td>Database directory block</td>
<td>DFSDDIR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMB</td>
<td>Data management block</td>
<td>DFSDBM</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMCB</td>
<td>DEDB master control block</td>
<td>DBFDMCB</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>BHDR</td>
<td>MSDB header</td>
<td>DBFDMDB</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

The /DIAGNOSE SNAP LINE command

The /DIAGNOSE SNAP LINE command captures control block information for the communication line that is specified in the linenumber parameter. The linenumber parameter specified must be numeric, in the range 1 - 1000, and identify a currently defined communication line. Multiple linenumber parameters can be specified, with each parameter separated by a comma or a blank.

The /DIAGNOSE SNAP LINE command is available in a DB/DC or DCCTL environment. If the /DIAGNOSE SNAP LINE command is issued in a DCCTL environment, a DFS110I error
message is issued in response. The extended communication name table (ECNT) is available only in an IMS system where Fast Path is defined.

Table 11-4 lists /DIAGNOSE SNAP LINE control blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECB</td>
<td>Event control block</td>
<td>IEPF</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CLB</td>
<td>Communication line block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CTB</td>
<td>Communication terminal block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CTT</td>
<td>Communication translate table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CRB</td>
<td>Communications restart block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SPQB</td>
<td>Subpool queue block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SPQBEXT</td>
<td>Subpool queue extension block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EMHB</td>
<td>Expedited message handler block</td>
<td>DBFEMHB</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CNT</td>
<td>Communication name table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ECNT</td>
<td>Extended communication name table</td>
<td>DBFECNT</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CCB</td>
<td>Conversational control block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CIB</td>
<td>Communication interface block</td>
<td>ICLI(CIB)</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>INBUF</td>
<td>Input line buffer</td>
<td>ICLI(CIB)</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>OUTBUF</td>
<td>Output line buffer</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

**The /DIAGNOSE SNAP LINK command**

The /DIAGNOSE SNAP LINK command captures control block information for the logical link that is specified in the linknumber parameter. The linknumber parameter specified must be numeric, in the range 1 - 675, and identify a currently defined logical link. Multiple linknumber parameters can be specified, with each parameter separated by a comma or a blank.

The /DIAGNOSE SNAP LINK command is available in a DB/DC or DCCTL environment. If the /DIAGNOSE SNAP LINK command is issued in a DBCTL environment, a DFS110I error message is issued in response.

Table 11-5 lists the /DIAGNOSE SNAP LINK control blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECB</td>
<td>Event control block</td>
<td>IEPF</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LLB</td>
<td>Link link block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LTB</td>
<td>Link terminal block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CTT</td>
<td>Communication translate table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CRB</td>
<td>Communications restart block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SPQB</td>
<td>Subpool queue block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
The /DIAGNOSE SNAP PGM command

The /DIAGNOSE SNAP PGM command captures control block information for the program that is specified in the pgmname parameter. The pgmname parameter that is specified must be alphanumeric, no longer than eight characters, and identify a currently defined program. Multiple pgmname parameters can be specified, with each parameter separated by a comma or a blank.

Table 11-6 lists /DIAGNOSE SNAP PGM control blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>Work</th>
<th>No work</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPQBEXT</td>
<td>Subpool queue extension block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EMHB</td>
<td>Expeditd message handler block</td>
<td>DBFEMHB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LNB</td>
<td>Link name table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECNT</td>
<td>Extended communication name table</td>
<td>DBFECNT</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCB</td>
<td>Conversational control block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CIB</td>
<td>Communication interface block</td>
<td>ICLI(CIB)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INBUF</td>
<td>Input line buffer</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTBUF</td>
<td>Output line buffer</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCB</td>
<td>Link control block</td>
<td>LCB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LXB</td>
<td>Link extension block</td>
<td>LXB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 11-6 lists /DIAGNOSE SNAP PGM control blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>Work</th>
<th>No work</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDIR</td>
<td>Program directory block</td>
<td>DFSPDIR</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSCX</td>
<td>Resource extension block</td>
<td>DFSRSCX</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTLIST</td>
<td>Intent list</td>
<td>INTLIST</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB</td>
<td>Program specification block</td>
<td>DFSPSB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PST</td>
<td>Partition specification table</td>
<td>IPST</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNT</td>
<td>Communication name table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMB</td>
<td>Scheduler message block</td>
<td>IAPS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMBL</td>
<td>Data management block list</td>
<td>DFSDMBL</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XPCB</td>
<td>Program communication block index</td>
<td>DFSPCBS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCB</td>
<td>Program communication block</td>
<td>DFSPCBS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCBX</td>
<td>Program communication extension block</td>
<td>DFSPCBS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPCB</td>
<td>Program communication block extension</td>
<td>DBFEPB</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MP62</td>
<td>Message prefix (LU62)</td>
<td>DFS62PRE</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPOT</td>
<td>Message prefix (OTMA)</td>
<td>DFSYPRE</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSBPRM</td>
<td>User parameter list block</td>
<td>IDLI</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The `/DIAGNOSE SNAP REGION` command

The `/DIAGNOSE SNAP REGION` command captures control block information for the dependent region specified in the `regionnumber` parameter. The `regionnumber` parameter specified must be numeric, in the range 1 - 999, and identify a currently active dependent region. Multiple `regionnumber` parameters can be specified, with each parameter separated by a comma or a blank.

The dependent region might also be identified by using the `SNAP JOBNAME(jobname)` format of the SNAP REGION() resource type. The `jobname` parameter specified must be alphanumeric, no longer than eight characters, and identify a currently active dependent region. Multiple `jobname` parameters can be specified, with each parameter separated by a comma or a blank.

The `regionnumber` and `jobname` formats can both be specified on the same command.

The `SNAP JOBNAME(jobname)` format of the SNAP REGION() resource type cannot be used to identify a CCTL thread. All CCTL threads have the same job name and the CICS region, and therefore it is impossible to identify the correct thread by the job name. If multiple regions are started with the same job name, only the first region will be found using the `SNAP JOBNAME(jobname)` format of the SNAP REGION() resource type.

Table 11-7 lists the `/DIAGNOSE SNAP REGION` control blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>System</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>WKCDS</td>
<td>Data capture segment work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKNDX</td>
<td>Index maintenance work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKKXIO</td>
<td>Index I/O work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKSEG</td>
<td>Segment work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKOIA</td>
<td>I/O work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKSSA</td>
<td>Segment search argument work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKFIP</td>
<td>Fast Path control block work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 11-7: The `/DIAGNOSE SNAP REGION()` control blocks
<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>System</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDT</td>
<td>Identify table entry</td>
<td>DFSIDT</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCB</td>
<td>MVS task control block (IDT)</td>
<td>IKJTCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCB</td>
<td>Task control block (PST)</td>
<td>IKJTCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAP</td>
<td>Save area prefix block</td>
<td>ISAP</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK1</td>
<td>Dispatch block: work area part 1 (current)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK2</td>
<td>Dispatch block: work area part 2 (current)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCB</td>
<td>MVS task control block (CDSP)</td>
<td>IKJTCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB</td>
<td>MVS associated request block (CDSP)</td>
<td>IHARB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RBP</td>
<td>MVS associated request block prefix (CDSP)</td>
<td>IHARB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XSB</td>
<td>MVS extended status block (CDSP)</td>
<td>IHAXSB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK1</td>
<td>Dispatcher block: work area part 1 (home)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK2</td>
<td>Dispatcher block: work area part 2 (home)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPPST</td>
<td>PST dispatching control block</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XMCI</td>
<td>Cross-memory control block, ITASK level</td>
<td>DFSXMC</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSVPL</td>
<td>System service parameter list block</td>
<td>DFSSSVPL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMIB</td>
<td>Directed message manager interface block</td>
<td>DFSDMIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CULE</td>
<td>Common use list element block</td>
<td>DFSCULE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLLE</td>
<td>Common latch list element block</td>
<td>DFSCLLE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSMB</td>
<td>Logging secondary master block</td>
<td>DFSLSMB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSIDX</td>
<td>Subsystem status index entry</td>
<td>DFSSSIE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LCRE</td>
<td>Local current recovery entry</td>
<td>DFSLCRE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RRE</td>
<td>Residual recovery element block (LCRE)</td>
<td>DFSRRE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIB</td>
<td>APPC transaction instance block</td>
<td>DFSTIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>YTIB</td>
<td>OTMA transaction instance block</td>
<td>DFSYTIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCENTRY</td>
<td>Protected conversation task table entry</td>
<td>DFSRRSIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RRE</td>
<td>Residual recovery element block (PC)</td>
<td>DFSRRE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The /DIAGNOSE SET AOSLOG command

The /DIAGNOSE SET AOSLOG command updates the attributes of the specified facility. The SET keyword is mutually exclusive with the SNAP keyword.

The AOSLOG parameter enables or disables logging of events related to APPC and OTMA synchronous transactions in a shared queues environment. When enabled, the events are written to the OLDS as type X'6701' records.

If AOSLOG(ON) is specified in a non-shared-queues environment or when AOS=N is specified in the DFSDCxxx PROCLIB member, the command is rejected with a DFS2859I message.

| ON    | Enables AOS logging |
| OFF   | Disables AOS logging |

Using the /DIAG SNAP command

Example 11-2 shows the /DIAG SNAP LINE command being issued and its results.

Example 11-2 The /DIAG SNAP commands

DIAG SNAP LINE(3) SHOW(ALL) I12A
44I DISPLAY FROM ID=I12A 982
/DIAGNOSE SNAP STORAGE DISPLAY

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>System</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>PST</td>
<td>Partition specification table</td>
<td>IPST</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>CNT</td>
<td>Communication name table</td>
<td>ICLI</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMB</td>
<td>Scheduler message block</td>
<td>IAPS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQPST</td>
<td>Scheduler queue element</td>
<td>ISQPST</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UOW</td>
<td>Unit of work value (QMGR)</td>
<td>DFSUOWE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UOWE</td>
<td>Unit of work table entry</td>
<td>DFSUOWE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The /DIAGNOSE SET AOSLOG command

The /DIAGNOSE SET AOSLOG command updates the attributes of the specified facility. The SET keyword is mutually exclusive with the SNAP keyword.

The AOSLOG parameter enables or disables logging of events related to APPC and OTMA synchronous transactions in a shared queues environment. When enabled, the events are written to the OLDS as type X'6701' records.

If AOSLOG(ON) is specified in a non-shared-queues environment or when AOS=N is specified in the DFSDCxxx PROCLIB member, the command is rejected with a DFS2859I message.

| ON    | Enables AOS logging |
| OFF   | Disables AOS logging |

Using the /DIAG SNAP command

Example 11-2 shows the /DIAG SNAP LINE command being issued and its results.
<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>00D0</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>00E0</td>
<td>00000000</td>
<td>40404040 40404040 00000000</td>
<td>............</td>
</tr>
<tr>
<td>00F0</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0100</td>
<td>0FC4E2D7</td>
<td>00000000 00000000 00000000</td>
<td>.DSP .............</td>
</tr>
<tr>
<td>0110</td>
<td>00400000</td>
<td>20FF4820 8004FCF8 00000000</td>
<td>........8....</td>
</tr>
</tbody>
</table>

CTB  Communication Terminal Block  Loc: 004CEF0

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>004E88B8</td>
<td>00487000 004F0040 00080000</td>
<td>...Yh...000 00000000 00000000 00000000</td>
</tr>
<tr>
<td>0010</td>
<td>00040000</td>
<td>00000004 01350000 25C5FB98</td>
<td>............E.q</td>
</tr>
<tr>
<td>0020</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0030</td>
<td>F0007FFF</td>
<td>F0040E3 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>0040</td>
<td>00000000</td>
<td>25C5FB98 00000000 00000000</td>
<td>............E.q</td>
</tr>
<tr>
<td>0050</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>2590386 00000000 00000000</td>
</tr>
<tr>
<td>0060</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000003 00000001</td>
</tr>
<tr>
<td>0070</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>0080</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00040000 00000000 00000000</td>
</tr>
<tr>
<td>0090</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>00A0</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>00B0</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>00C0</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>00D0</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
</tbody>
</table>

---

CTT  Communication Translate Table  Loc: 004E88B8

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>004E88B8</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
<tr>
<td>0010</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>002000A8 67200000</td>
</tr>
<tr>
<td>0020</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>00000000 00000000 00000000</td>
</tr>
</tbody>
</table>

CRB  Communications Restart Block  Loc: 00000000

Storage unavailable

SPQB  Subpool Queue Block  Loc: 00000000

Storage unavailable

SPQBEXT  Subpool Queue Extension Block  Loc: 00000000

Storage unavailable

EMHB  Expedited Message Handler Block  Loc: 00000000

Storage unavailable

CNT  Communication Name Table  Loc: 25C5FB98

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>000A875C</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0010</td>
<td>000A875C</td>
<td>00200171 0020022 2E24C1E2</td>
<td>............</td>
</tr>
<tr>
<td>0020</td>
<td>003C9319</td>
<td>400000001 0004CEF0 25C5FB32</td>
<td>............</td>
</tr>
<tr>
<td>0030</td>
<td>00000000</td>
<td>0004400 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0040</td>
<td>25C5FB10</td>
<td>00000000 00048780 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0050</td>
<td>00010001</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0060</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
</tbody>
</table>

---

CNT  Communication Name Table  Loc: 25C5FB32

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>000A875C</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0010</td>
<td>000A875C</td>
<td>00200171 0020022 2E24C1E2</td>
<td>............</td>
</tr>
<tr>
<td>0020</td>
<td>003C9319</td>
<td>400000001 0004CEF0 25C5FB32</td>
<td>............</td>
</tr>
<tr>
<td>0030</td>
<td>00000000</td>
<td>0004400 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0040</td>
<td>25C5FB10</td>
<td>00000000 00048780 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0050</td>
<td>00010001</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
<tr>
<td>0060</td>
<td>00000000</td>
<td>00000000 00000000 00000000</td>
<td>............</td>
</tr>
</tbody>
</table>

---

392  IMS 12 Selected Performance Topics
Example 11-3 shows the sample JCL to extract these DIAG records with DFSERA10 and print them with exit DFSERA30.

**Example 11-3 JCL for printing DIAG records with exit DFSERA30**

```plaintext
//P010 EXEC PGM=DFSERA10
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSUT1 DD DISP=SHR,DSN=IMS12Q.IMS12A.OLP05
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *
CONTROL CNTL
OPTION PRINT O=5,V=6701,L=2,C=M,E=DFSERA30
OPTION PRINT O=9,V=DIAG,L=4,T=C,C=E,E=DFSERA30
END
/*

**Considerations for the /DIAGNOSE command**

To capture data from a series of /DIAGNOSE commands in a trace data set, issue the commands in the following order:

/TRACE SET ON TABLE DIAG OPTION LOG
/DIAGNOSE
/TRACE SET OFF TABLE DIAG
11.3 Procedure to request a IMS performance health check

The IMS Technical Specialist team consists of globally dispersed IMS specialists who are part of the IMS Development organization of the Silicon Valley Lab. Their mission is devoted to ensuring a high level of customer satisfaction and partnering through ongoing communication, counseling and coordination of customer IMS issues.

The team's focus is fostering collaboration across IMS communities, sharing strategies, new capabilities and directions of IMS. Benefits to the customer are direct contact with lab management and development to resolve business related activities from design requirements, education, hands on assistance and up to date communication on IMS and related technologies and directions.

They team offers fee-based IMS support services including IMS system and application performance health check services.

For more information, consult the following website:
http://ibm.com/software/data/services/imsofferings.html
## Abbreviations and acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACB</td>
<td>application control block</td>
<td>CP</td>
<td>central processor</td>
</tr>
<tr>
<td>ACBLIB</td>
<td>ACB library</td>
<td>CQS</td>
<td>Common Queue Server</td>
</tr>
<tr>
<td>ACEE</td>
<td>access control environment element</td>
<td>CRL</td>
<td>certificate revocation list</td>
</tr>
<tr>
<td>ACK</td>
<td>acknowledgement</td>
<td>CRSS</td>
<td>Conditional Reorganization Support Service</td>
</tr>
<tr>
<td>AIB</td>
<td>application interface block</td>
<td>CSA</td>
<td>common storage area</td>
</tr>
<tr>
<td>AOI</td>
<td>Automated Operator Interface</td>
<td>CSL</td>
<td>Common Service Layer</td>
</tr>
<tr>
<td>AOP</td>
<td>automated operator program</td>
<td>CSM</td>
<td>Complete Status Message</td>
</tr>
<tr>
<td>AOR</td>
<td>application-owning region</td>
<td>CST</td>
<td>Consolidated Service Test</td>
</tr>
<tr>
<td>AOS</td>
<td>APPC/OTMA SMQ</td>
<td>CSV</td>
<td>comma-separated value</td>
</tr>
<tr>
<td>APF</td>
<td>authorized program facility</td>
<td>CTG</td>
<td>CICS Transaction Gateway</td>
</tr>
<tr>
<td>API</td>
<td>application programming interface</td>
<td>DB2</td>
<td>Database 2</td>
</tr>
<tr>
<td>APPC</td>
<td>Advanced Program-to-Program Communication</td>
<td>DBCTL</td>
<td>Database Control</td>
</tr>
<tr>
<td>APSB</td>
<td>allocate PSB</td>
<td>DBD</td>
<td>database description</td>
</tr>
<tr>
<td>ARM</td>
<td>automatic restart manager</td>
<td>DBDS</td>
<td>database data sets</td>
</tr>
<tr>
<td>ASN</td>
<td>abend search and notification</td>
<td>DBHDA</td>
<td>DB Historical Data Analyzer</td>
</tr>
<tr>
<td>BMP</td>
<td>batch messaging program</td>
<td>DBMS</td>
<td>database management system</td>
</tr>
<tr>
<td>BPE</td>
<td>Base Primitives Environment</td>
<td>DBPCB</td>
<td>database PCB</td>
</tr>
<tr>
<td>BSAM</td>
<td>Basic Sequential Access Method</td>
<td>DBRA</td>
<td>database resource adapter</td>
</tr>
<tr>
<td>CA</td>
<td>certificate authority</td>
<td>DBRC</td>
<td>Database Recovery Control</td>
</tr>
<tr>
<td>CA</td>
<td>change accumulation</td>
<td>DBSR</td>
<td>Database Segment Restructure</td>
</tr>
<tr>
<td>CA</td>
<td>control area</td>
<td>DCCTL</td>
<td>Data Communications Control</td>
</tr>
<tr>
<td>CBM</td>
<td>Component Business Modeling</td>
<td>DDEP</td>
<td>direct dependent</td>
</tr>
<tr>
<td>CBPDO</td>
<td>Custom-built Product Delivery Option</td>
<td>DDIR</td>
<td>database directory</td>
</tr>
<tr>
<td>CCF</td>
<td>Common Connector Framework</td>
<td>DDM</td>
<td>Distributed Data Management</td>
</tr>
<tr>
<td>CCI</td>
<td>Common Client Interface</td>
<td>DEBs</td>
<td>data extent blocks</td>
</tr>
<tr>
<td>CCTL</td>
<td>coordinator controller</td>
<td>DEDB</td>
<td>data entry database</td>
</tr>
<tr>
<td>CDE</td>
<td>contents directory entry</td>
<td>DLDLP</td>
<td>DMB pool</td>
</tr>
<tr>
<td>CDSP</td>
<td>MVS control block</td>
<td>DLET</td>
<td>delete</td>
</tr>
<tr>
<td>CEX</td>
<td>connect extension</td>
<td>DMB</td>
<td>data management block</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
<td>DRA</td>
<td>database resource adapter</td>
</tr>
<tr>
<td>CI</td>
<td>control interval</td>
<td>DRD</td>
<td>dynamic resource definition</td>
</tr>
<tr>
<td>CICS</td>
<td>Customer Information Control System</td>
<td>DRDA</td>
<td>Distributed Relational Database Architecture</td>
</tr>
<tr>
<td>CIOP</td>
<td>Communication I/O pool</td>
<td>DRF</td>
<td>Database Recovery Facility</td>
</tr>
<tr>
<td>CLI</td>
<td>command-line interface</td>
<td>DSN</td>
<td>data source name</td>
</tr>
<tr>
<td>CLP</td>
<td>command-line processor</td>
<td>DTP</td>
<td>distributed transaction processing</td>
</tr>
<tr>
<td>COMPID</td>
<td>component IDs</td>
<td>EAB</td>
<td>Enterprise Access Builder</td>
</tr>
<tr>
<td>CPC</td>
<td>central processor complex</td>
<td>EAI</td>
<td>enterprise application integration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EAS</td>
<td>extended addressing space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EAV</td>
<td>extended address volume</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECB</td>
<td>event control block</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECNT</td>
<td>extended communication name table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECSA</td>
<td>extended common service area</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EINIT</td>
<td>Early Initialization exit routine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EIS</td>
<td>enterprise information system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EJB</td>
<td>Enterprise JavaBeans</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EMCS</td>
<td>extended multiple console support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EMF</td>
<td>Eclipse Modeling Framework</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOM</td>
<td>end-of-memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPS</td>
<td>extended pointer set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPVT</td>
<td>extended private</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESS</td>
<td>IBM Enterprise Storage Server®</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EWLM</td>
<td>Enterprise Workload Manager</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EX6</td>
<td>execution phase 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXE</td>
<td>execution</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDBR</td>
<td>Fast Database Recovery</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FIXCAT</td>
<td>fix category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMID</td>
<td>function modification ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FPA</td>
<td>Fast Path Advanced</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FPB</td>
<td>Fast Path Basic Tools</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FPO</td>
<td>Fast Path Online</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FPSI</td>
<td>Fast Path Secondary Index</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FT3</td>
<td>file tailoring phase 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GA</td>
<td>General Availability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GDG</td>
<td>generation data set group</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GEF</td>
<td>Graphical Editing Framework</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GN</td>
<td>get next</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GNP</td>
<td>get next within parent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GSAM</td>
<td>generalized sequential access method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GU</td>
<td>get unique</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GUI</td>
<td>graphical user interface</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HALDB</td>
<td>high availability large databases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDAM</td>
<td>hierarchical direct access method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDC</td>
<td>Hardware Data Compression</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDPC</td>
<td>HD Pointer Checker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDTA</td>
<td>HD Tuning Aid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIDAM</td>
<td>hierarchical indexed direct access method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIOP</td>
<td>high I/O pool</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HPIC</td>
<td>High Performance Image Copy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HPPC</td>
<td>High Performance Pointer Checker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HSSR</td>
<td>High Speed Sequential Retrieval</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**HTTP** | Hypertext Transfer Protocol

**HTTPS** | HTTP Secure

**IDE** | integrated development environment

**IDRC** | improved data recording capability

**IDp** | identity provider

**IETF** | Internet Engineering Task Force

**IFCC** | interface control checks

**ILDS** | indirect list data set

**ILK** | indirect list key

**ILS** | isolated log sender

**IMS** | Information Management System

**IPCS** | Interactive Problem Control System

**IPL** | initial program load

**IRLM** | internal resource lock manager

**IRM** | IMS request message

**IRUR** | Internal Resource Usage Report

**ISC** | intersystem communication

**ISPF** | Interactive System Productivity Facility

**ISRT** | insert

**ISV** | independent software vendor

**ITKB** | IMS Tools Knowledge Base

**ITSO** | International Technical Support Organization

**IVP** | Installation Verification Program

**IVPEX** | IVP Export Utility

**JBP** | Java batch processing

**JCL** | job control language

**JDBC** | Java Database Connectivity

**JMP** | Java Message Program

**Java EE** | Java Platform, Enterprise Edition

**KFBA** | key feedback area

**KSDS** | key-sequenced data set

**LAN** | local area network

**LDAP** | Lightweight Directory Access Protocol

**LDS** | linear data set

**LIU** | Library Integrity Utility

**LLLLL** | length value

**LPAR** | logical partition

**LSQA** | local system queue area

**LU** | logical unit

**LU2** | logical unit 2

**LUNAME** | logical unit name
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAS</td>
<td>multi-area structure</td>
</tr>
<tr>
<td>MDA</td>
<td>MVS dynamic allocation</td>
</tr>
<tr>
<td>MDB</td>
<td>message-driven bean</td>
</tr>
<tr>
<td>MFBP</td>
<td>message format buffer pool</td>
</tr>
<tr>
<td>MFS</td>
<td>Message Format Service</td>
</tr>
<tr>
<td>MOD</td>
<td>message output descriptor</td>
</tr>
<tr>
<td>MPP</td>
<td>message processing program</td>
</tr>
<tr>
<td>MSC</td>
<td>Multiple Systems Coupling</td>
</tr>
<tr>
<td>MSDB</td>
<td>main storage database</td>
</tr>
<tr>
<td>MTO</td>
<td>master terminal operator</td>
</tr>
<tr>
<td>MVS</td>
<td>Multiple Virtual System</td>
</tr>
<tr>
<td>NAK</td>
<td>negative acknowledgment</td>
</tr>
<tr>
<td>ODBA</td>
<td>Open Database Access</td>
</tr>
<tr>
<td>ODBC</td>
<td>Open Database Connectivity</td>
</tr>
<tr>
<td>ODBM</td>
<td>Open Database Manager</td>
</tr>
<tr>
<td>OLC</td>
<td>online change</td>
</tr>
<tr>
<td>OLDS</td>
<td>online log data set</td>
</tr>
<tr>
<td>OLR</td>
<td>online reorganization</td>
</tr>
<tr>
<td>OM</td>
<td>Operations Manager</td>
</tr>
<tr>
<td>OO</td>
<td>object-oriented</td>
</tr>
<tr>
<td>OSAM</td>
<td>overflow sequential access method</td>
</tr>
<tr>
<td>OTE</td>
<td>Open Transaction Environment</td>
</tr>
<tr>
<td>OTMA C/I</td>
<td>OTMA Callable Interface</td>
</tr>
<tr>
<td>OTMA</td>
<td>Open Transaction Manager Access</td>
</tr>
<tr>
<td>OTT</td>
<td>Open Thread TCB</td>
</tr>
<tr>
<td>PA</td>
<td>Performance Analyzer</td>
</tr>
<tr>
<td>PC</td>
<td>personal computer</td>
</tr>
<tr>
<td>PCB</td>
<td>program communication block</td>
</tr>
<tr>
<td>PCBs</td>
<td>program control blocks</td>
</tr>
<tr>
<td>PDIR</td>
<td>program directory</td>
</tr>
<tr>
<td>PDS</td>
<td>partitioned data set</td>
</tr>
<tr>
<td>PDU</td>
<td>Partition Definition Utility</td>
</tr>
<tr>
<td>PHIDAM</td>
<td>partitioned HIDAM</td>
</tr>
<tr>
<td>PI</td>
<td>problem investigator</td>
</tr>
<tr>
<td>PPT</td>
<td>program properties table</td>
</tr>
<tr>
<td>PRA</td>
<td>parallel RECON access</td>
</tr>
<tr>
<td>PSB</td>
<td>program specification block</td>
</tr>
<tr>
<td>PSP</td>
<td>Preventive Service Planning</td>
</tr>
<tr>
<td>PSSR</td>
<td>Physical Sequence Sort for Reload</td>
</tr>
<tr>
<td>PST</td>
<td>program specification table</td>
</tr>
<tr>
<td>PTF</td>
<td>program temporary fix</td>
</tr>
<tr>
<td>QBUF</td>
<td>message queue buffers</td>
</tr>
<tr>
<td>QoS</td>
<td>quality of service</td>
</tr>
<tr>
<td>RAR</td>
<td>resource adapter archive</td>
</tr>
<tr>
<td>RAS</td>
<td>Resource Access Security</td>
</tr>
<tr>
<td>RBA</td>
<td>relative byte address</td>
</tr>
<tr>
<td>RDDS</td>
<td>resource definition data set</td>
</tr>
<tr>
<td>RDS</td>
<td>repository data set</td>
</tr>
<tr>
<td>RDS</td>
<td>restart data set</td>
</tr>
<tr>
<td>RECA</td>
<td>receive any pool</td>
</tr>
<tr>
<td>RECON</td>
<td>recovery control</td>
</tr>
<tr>
<td>RLDS</td>
<td>recovery log data set</td>
</tr>
<tr>
<td>RM</td>
<td>Resource Manager</td>
</tr>
<tr>
<td>RMF</td>
<td>Resource Measurement Facility</td>
</tr>
<tr>
<td>RMM</td>
<td>Request MOD Message</td>
</tr>
<tr>
<td>RRDS</td>
<td>relative record data set</td>
</tr>
<tr>
<td>RRS</td>
<td>Resource Recovery Services</td>
</tr>
<tr>
<td>RRS/MVS</td>
<td>Resource Recovery Services/MVS</td>
</tr>
<tr>
<td>RS</td>
<td>Repository Server</td>
</tr>
<tr>
<td>RSM</td>
<td>request status message</td>
</tr>
<tr>
<td>RSR</td>
<td>Remote Site Recovery</td>
</tr>
<tr>
<td>RVA</td>
<td>RAMAC Virtual Array</td>
</tr>
<tr>
<td>SAF</td>
<td>System Authorization Facility</td>
</tr>
<tr>
<td>SAM</td>
<td>sequential access method</td>
</tr>
<tr>
<td>SAML</td>
<td>Security Assertion Markup Language</td>
</tr>
<tr>
<td>SAS</td>
<td>secondary address space</td>
</tr>
<tr>
<td>SCI</td>
<td>Structured Call Interface</td>
</tr>
<tr>
<td>SDEP</td>
<td>sequential dependent segment</td>
</tr>
<tr>
<td>SDK</td>
<td>software development kit</td>
</tr>
<tr>
<td>SEVT</td>
<td>structure event trace table</td>
</tr>
<tr>
<td>SHISAM</td>
<td>simple hierarchical sequential access method</td>
</tr>
<tr>
<td>SLA</td>
<td>service-level agreement</td>
</tr>
<tr>
<td>SLDS</td>
<td>system log data set</td>
</tr>
<tr>
<td>SLSB</td>
<td>stateless session bean</td>
</tr>
<tr>
<td>SMB</td>
<td>scheduler message block</td>
</tr>
<tr>
<td>SMF</td>
<td>System Management Facilities</td>
</tr>
<tr>
<td>SMQ</td>
<td>shared message queue</td>
</tr>
<tr>
<td>SMU</td>
<td>Security Maintenance Utility</td>
</tr>
<tr>
<td>SNA</td>
<td>Systems Network Architecture</td>
</tr>
<tr>
<td>SOA</td>
<td>service-oriented architecture</td>
</tr>
<tr>
<td>SOMA</td>
<td>Service Oriented Modeling and Architecture</td>
</tr>
<tr>
<td>SPE</td>
<td>small programming enhancement</td>
</tr>
<tr>
<td>SPI</td>
<td>system programming interface</td>
</tr>
<tr>
<td>SPMN</td>
<td>Space Monitor</td>
</tr>
<tr>
<td>SPOC</td>
<td>single point of control</td>
</tr>
</tbody>
</table>
SQ  shared queues
SQLJ  SQL for Java
SRB  service request block
SSA  segment search argument
SSPM  sysplex serialized program management
STE  storage tracking element
STR  structure trace table
STSN  set and test sequence numbers
SVL  Silicon Valley Laboratories
SXPL  Standard User Exit Parameter List
SYSDEF  system definition
SYSID  system identifier
TCB  task control block
TCB  thread task control block
TCO  Time-Controlled Operations
TCP/IP  Transmission Control Protocol/Internet Protocol
TM  Transaction Manager
TMRA  Transaction Manager Resource Adapter
TMS  Transport Manager System
TOR  terminal-owning region
TOSI  Tools Online System Interface
TPIPE  transaction pipe
TSO  Time Sharing Option
TTL  time to live
UI  user interface
UOR  unit of recovery
UOW  unit of work
VSAM  Virtual Storage Access Method
VTAM  Virtual Telecommunications Access Method
WADS  write-ahead data set
WFI  wait for input
WLM  Workload Manager
WSDL  Web Services Description Language
WTOR  write to operator with reply
XCF  cross-system coupling facility
XML  Extensible Markup Language
zAAP  System z Application Assist Processor
zIIP  System z Integrated Information Processors
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- IBM IMS Version 12 Technical Overview, SG24-7972
- IMS 11 Open Database, SG24-7856
- IMS Performance and Tuning Guide, SG24-7324
- IMS Version 7 Performance Monitoring and Tuning Update, SG24-6404
- System z Parallel Sysplex Best Practices, SG24-7817

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:
ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- IMS Version 12 Commands, Volume 1: IMS Commands A-M, SC19-3009
- IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011
- IMS Version 12 System Utilities, SC19-3023
- IMS Version 12 System Administration, SC19-3020
- IMS Problem Investigator for z/OS User’s Guide V3R3, SC19-3635
- IMS Version 12 System Definition, GC19-3021
- IMS Version 12 Operations and Automation, SC19-3018
- IMS Version 12 Database Utilities, SC19-3014
- IMS Version 12 System Utilities, SC19-3023
- IMS Performance Analyzer for z/OS User’s Guide V4R3, SC19-3633
- IMS Connect Extensions for z/OS User's Guide V2R3, SC19-3632
- IMS Buffer Pool Analyzer User's Guide V1R3, SC19-2477
- Transaction Analysis Workbench for System z User's Guide V1R1, SC19-2920
Online resources

These websites are also relevant as further information sources:

- IMS family
  http://www.ibm.com/software/data/ims/
- IMS tools

Help from IBM

IBM Support and downloads
ibm.com/support

IBM Global Services
ibm.com/services
IMS 12 Selected Performance Topics

Learn about IMS system and application performance

IBM Information Management System (IMS) provides leadership in performance, reliability, and security to help you implement the most strategic and critical enterprise applications. IMS, IMS utilities, and IMS tools continue to evolve to provide value and meet the needs of enterprise customers.

With IMS 12, integration and open access improvements provide flexibility and support business growth requirements. Scalability improvements have been made to the well-known performance, efficiency, availability, and resilience of IMS by using 64-bit storage.

In this IBM Redpaper Redbooks publication we provide IMS performance monitoring and tuning information by describing the key IMS performance functions and by showing how to monitor and tune them with traditional and new strategic applications. This book is for database administrators and system programmers.

We summarize methods and tools for monitoring and tuning IMS systems, describe IMS system-wide performance, database, and transaction considerations.

Based on lab measurements, we provide information about recent performance enhancements that are available with IMS 12, and advice about setting performance-related parameters.

Explore methods and tools for monitoring

Examine recent performance enhancements

For more information: ibm.com/redbooks