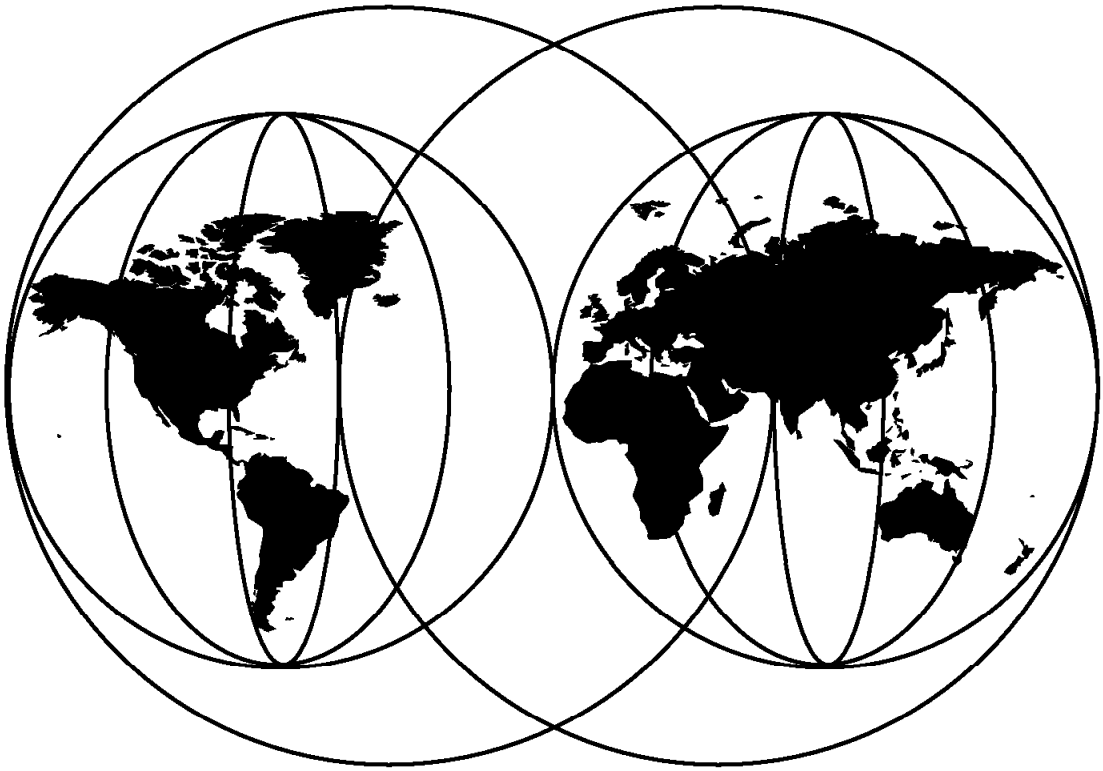
IMS/ESA Shared Queues: A Planning Guide

Glen Battershell, Luigi Cetorelli, Hannelore Nestinger
Geoff Nicholls, Pete Sadler, Siew Tay



International Technical Support Organization

http://www.redbooks.ibm.com

This book was printed at 240 dpi (dots per inch). The final production redbook with the RED cover will be printed at 1200 dpi and will provide superior graphics resolution. Please see "How to Get ITSO Redbooks" at the back of this book for ordering instructions.

SG24-5257-00

International Technical Support Organization

SG24-5257-00

IMS/ESA Shared Queues: A Planning Guide

December 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix G, “Special Notices” on page 163.

First Edition (December 1998)

This edition applies to the IBM Information Management System (IMS), Transaction and Database Server for System/390 (Program Number 5655-158), and the IMS/ESA Performance Analyzer Version 1 Release 2 (Program Number 5697-B89) for use with MVS/ESA SP Version 4 Release 3, OS/390 Version 1 or later Operating System.

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

Contents

Figures

Preface

This redbook presents a case study of the planning and preparation for the implementation of IMS shared queues using IMS/ESA Version 6 at IBM Global Services Australia (IGSA) for Telstra in Melbourne, Australia.

The steps that were taken to plan, prepare, and implement this project are outlined, with details on the challenges presented to IGSA during the migration and implementation of the solutions.

This redbook provides IMS system programmers, systems and applications designers, database administrators, and applications programmers who are responsible for the implementation of IMS shared queues with information regarding a case study of a migration to that environment.

Chapter 1 contains an introduction to Telstra and the IBM Global Services Australia environment, and the rest of the book contains a summary of our conclusions and outlines the tasks required.

Some knowledge of IMS shared queues, sysplex environments, and the coupling facility, of recovery procedures for failures in complex IMS environments, and of the role of the coupling facility in support of IMS are assumed. This information is available from several sources, including IMS Education and Training class CM61, “IMS/ESA Version 6 Shared Queues.”

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at Telstra in Melbourne, Australia for the International Technical Support Organization San Jose Center.

Glen Battershell is an IMS Systems Programmer for IBM Global Services, Australia. Glen has 11 years of experience in the information technology industry, the last 3 focusing on IMS. He holds a Certificate in Computer Programming from Box Hill College, Melbourne, Australia. Glen′s areas of expertise include IMS Transaction Manager and automated operations.

Luigi Cetorelli is an IMS Systems Specialist with IBM Italy. Luigi has many years of experience with IMS, and has worked for IBM for 14 years. His area of expertise includes IMS data sharing, shared queues, and the Parallel Sysplex architecture. Since joining IBM, Luigi has supported several customers throughout Italy He has written extensively on IMS.

Hanne Nestinger is an IMS Systems Specialist with IBM Global Services Germany. Hanne has 20 years of experience with IMS, 13 of those with IBM. She holds a science degree in Mathematics from Justus Liebig University Giessen, Germany. Her areas of expertise include both stand-alone and Parallel Sysplex environments. She worked as an application programmer, database administrator, and instructor. Since joining IBM in 1985, Hanne has supported several large customers mainly in the banking industry and German government.

Geoff Nicholls is an Advisory Systems Engineer at the International Technical Support Organization, San Jose Center. Geoff has a Science degree from the University of Melbourne, Australia, where he majored in Computer Science. He

worked as an application programmer and database administrator for several insurance companies before specializing in database and transaction management systems with Unisys and IBM. Since joining IBM in 1989, Geoff has worked extensively with IMS customers in Australia and throughout Asia.

Pete Sadler is an IMS Product Specialist working across Europe. Pete has over 25 years of experience in IMS and related fields as an Instructor, Systems Engineer and IMS Developer. He is a coauthor of *Client/Server Computing Using APPC/IMS*, GG24-3981, *IMS Version 6 Guide*, SG24-2228, and has contributed to several other Redbooks. He holds a degree in Chemical Engineering from the University of London and is a Chartered Engineer.

Siew Tay is an IMS Specialist with IBM Global Services Australia based in Perth, Australia. Siew has been an IMS systems programmer for several customers in Australia for many years, and now has responsibility for developing the IMS Automation software.

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

Alan Tippett
International Technical Support Organization, San Jose Center

Ken Browning
Robert Hain
Peter Kellett
Anthony Larinok
Peter Phillips
Luke Wilby
IBM Global Services Australia

Patrick DelCourt
Bob Guttman
Darren Schembri
Telstra, Australia

Dick Hannan
Betty Patterson
Pat Smith
Sandy Stoob
Santa Teresa Laboratory, IBM San Jose

Ulrich Müller
IBM Germany

Jennifer Pearcey
IBM South Africa

David Mierowsky
Fundi Software

Comments Welcome

Your comments are important to us!

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

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 177 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

  For Internet users `http://www.redbooks.ibm.com`
  For IBM Intranet users `http://w3.itso.ibm.com`

- Send us a note at the following address:

  `redbook@vnet.ibm.com`

Part 1. Introduction

Chapter 1. Introduction to Telstra and IBM Global Services Australia

In this chapter we describe the Telstra business environment and the IBM Global Services Australia (IGSA) information technology environment.

1.1 Telstra Corporate History

Telstra′s origins date back to 1901, when the Postmaster-General′s Department (PMG) was established to manage all domestic telephone, telegraph and postal services. The Overseas Telecommunications Commission OTC was established in 1946 to manage Australia′s international telecommunications.

The Australian Telecommunications Commission, trading as Telecom Australia, was created as a separate entity in July 1975, after the breakup of the PMG. While continuing to trade as Telecom Australia, the Commission became the Australian Telecommunications Corporation in January 1989. OTC and Telecom Australia became the Australian and Overseas Telecommunications Corporation Limited, after merging in February 1992.

Telstra Corporation Limited became the legal corporate name of the merged entity in April 1993. The domestic trading name, Telecom Australia, was changed to Telstra on July 1, 1995, to distinguish Telstra from other telecommunications companies in increasingly competitive and deregulated markets. The company has been trading as Telstra internationally since 1993.

The markets in which Telstra operates have undergone significant liberalization in recent years as Australia has moved toward open competition. Telstra has been subject to competition in the national long distance and international telephone service markets since 1991 and in the mobile telephone service market since 1992. On July 1, 1997, Australia′s telecommunications markets were opened to full competition, with no limit on the number of carriers owning transmission infrastructure that could enter the market.

1.2 Telstra and IGSA Data Processing Environment

In July 1997, an agreement was reached between IGSA and Telstra whereby IGSA would manage Telstra′s operational environment for 10 years. Software currency, a cornerstone of that agreement, provided the impetus to install IMS Version 6 and possible use of shared queues, a new feature delivered with IMS Version 6. The use of shared queues was seen as necessary because multiple systems coupling (MSC) links, with their limited capacity, were a potential bottleneck. Because the business workload profile can change very rapidly, for example, with the introduction of Internet access to mainframes, the MSC links could cause response time problems.

1.2.1 Production Environment

All applications using IMS data base and data communication (DB/DC) in the Telstra-IGSA environment are billing related. The Flexcab application holds all customer, call, and charge data to enable production of customer bills. The production workload profile consists of both batch and online components running 24 hours a day, 6.5 days a week, spread across two IMSs. The online transaction volume of 250 transactions per second is generated through

approximately eight thousand 3270-type terminals. A small number of transactions enter IMS through advanced program-to-program communication (APPC), message queueing (MQ), and intersystem communication (ISC) links. The batch load consists of up to 80 batch message processing regions (BMPs) running in parallel with the online transaction load. The data is stored in 200 IMS full-function databases, 1200 data entry databases (DEDBs) and 7000 DB2 tables. DB2 data (but not IMS data) is shared in a data-sharing configuration. IMS data is shared at a transaction level between 2 IMSs only. Transactions are passed from a front-end IMS system to a back-end IMS system using MSC links. Seven logical links are defined to support approximately 50 transactions per second.

The workload is run on two large mainframe processors. One runs 700 million instructions per second (MIPS), with 2.0 GB of central storage and 4.0 GB of expanded storage. The other processor runs 620 MIPS with 1.5 GB of central storage and 2.5 GB of expanded storage. They are combined with two other processors to form a four-way sysplex.

Logging rates on the back-end IMS are 1.5 MB/s sustained average over 24 hours and rising to more than 3 MB/s for a 15 minute interval. The front-end system logging rate is about 50% that of the back-end system. The higher logging rate on the back-end system is almost entirely due to the volume of simultaneous batch updates being performed. To sustain these rates and meet service level agreements (SLAs) for online transaction response time, the write-ahead data storage (WADS) has been placed on fully cached direct access storage device (DASD). The dedicated cache ensures that we never go into back-end destage; this provides a stable WADS response time. The resulting configuration provides for consistent MSC link response times. Before the WADS revamp we found that inconsistent WADS response times produced inconsistent MSC link response times because of the high number of check writes requested by the MSC links.

1.2.2 Development Environments

To support the production environment, three levels of development environment exist. The first level is known as the *Aplex*. This is a sysplex environment used exclusively by the systems programming team for configuration testing, software shakeouts and the like. IMS Support runs five IMSs on the Aplex, two of which are in a full data-sharing sysplex environment. The two IMSs in the data-sharing environment are known as *IMA1* and *IMA2*. (The A does not relate to the MVS sysplex name; it is a coincidence in this case). Performance is not considered to be important. All system software, including IMS is distributed from the Aplex.

The second level of the development environment is made up of approximately 15 unrelated IMS systems. These environments are used by application developers for all aspects of their development. Extensive use is also made of BTS and an inhouse-written and supported application environment duplicator called*ACE*. The *ACE* product enables multiple copies of the same application environment to be combined within a single IMS. None of this second IMS tier was used during the residency that produced this redbook.

A stress and volume testing environment makes up the third level of the development environment. It is contained within a sysplex called the *Cplex*. Two IMSs run here: IME1 and IME2. These IMSs are not in a data-sharing environment. Their environment is used for application performance tuning and monitoring. It is not available to system or application personnel to abend and

fail as they choose. Instead, the sizing and configuration of this environment is more closely matched to production needs. Fully tested applications are introduced here and stress tested using the teleprocessing network simulator (TPNS). The scripts used with TPNS are created from production traces and matching production data.

1.2.3 Software Levels

As of August 1998, the following key product software levels were installed on the systems used during the residency:

Software	Release Level
MVS OS/390	2.4
IMS	Version 6.1
BMC Delta/VT	V6.0.7
DB2	V5.0
VTAM	V4.4
RACF	V1.9

1.2.4 Software Maintenance Environment

IMS Support has responsibility for providing a stable IMS software environment to the user community. The installation of all software and fixes into the System Modification Program (SMP) distribution libraries is carried out by a Product Installation Group at the request of the IMS group. This installation is always done on the machines used by the systems programmers, the Apex. To ensure that correct versions, maintenance levels and fixes are installed, close coordination is required between the Product Installation group and the IMS group. Thus the Product installation Group and IMS Support both have well documented and well understood procedures.

IMS system generations are carried out as required, using the distribution libraries containing the appropriate software levels. The distribution library is selected by means of an ISPF panel that forms the front-end of an automated IMSGEN process. Before distribution to application development environments the new software is installed into systems programming run-time libraries and tested using the IBM-supplied Installation Verification Program (IVP) application. Any new software is always installed into new SMP libraries so its introduction into application environments can be strictly controlled. After a suitable shakedown period, the new software is rolled out to the next IMSs. This technique removes the need to make multiple maintenance runs of the same fixes. A fix needs to be applied and accepted only once. This process is described further in Appendix E, “System Maintenance Procedures” on page 147.

1.3 Understanding the IMS Environment in which Flexcab Runs

IMS systems IME1 and IME2, are the two IMS systems that are used by Flexcab for stress and volume testing. IME1 and IME2 were the targeted environments for the shared-queue implementation. IME1 and IME2 are IMS systems that are copies of the two production IMSs, IMFD and IMFK. A successful implementation of shared queues in the IME1 and IME2 environments will lead to implementing shared-queues in the IMFD and IMFK environments later. IME1 and IME2 are the

two IMSs used as a stress test environment for the Flexcab application. CPU constraints make it impossible to replicate the workload that is put through the IMFD and IMFK environments in IME1 and IME2. IMFD is the front-end production IMS and it normally has about 5000 users connected to it, processing up to 320 full-function transactions a second.

IME1 is the front-end IMS in the stress and test environment that the users connect to, and a subset of the transactions are routed to IME2 via an MSC link for processing. The routing to IME2 is required because no data sharing is performed between the two environments, although plans for data sharing are under way.

IME1 has 2794 transactions defined to it, while IME2 has 159 transactions defined to it. A few transactions are defined on both systems that require special attention. A small number of transactions are received from APPC and MQSeries. The APPC transactions are explicit-mode transactions. Most of the Flexcab databases are DB2 databases, connected to both IME1 and IME2. Transaction volume varies depending on when a new release of the application is being rolled out or fixes are being tested.

Flexcab management's is charged with producing a document detailing how IME1 and IME2 systems will be put into a shared-queue environment. The move to shared-queues must not effect reliability and must cause only minor performance degradation. A plan to fall back to a non-shared-queues environment needs to be prepared. Such fallback would need to be performed within 60 minutes.

Chapter 2. IMS/ESA Shared Queues in a Parallel Sysplex Environment

Many IMS users are looking to manage their increasing application workloads efficiently now and in the future. In addition, they want to develop strategies to improve their current service levels and move toward continuous operations while maintaining or improving acceptable response times.

IMS users recognize the requirement to efficiently distribute and balance the total workload across multiple systems. The introduction of shared-queues, a new function in IMS/ESA Version 6, assists users in meeting these requirements by:

- Providing dynamic load balancing among multiple IMS/ESA systems in a Parallel Sysplex environment
- Optimizing overall throughput across the sysplex, so that no single IMS remains underutilized while others are overloaded
- Achieving improvements in continuous availability for applications
- Improving reliability and providing better failure isolation, so that any remaining IMS can continue processing the shared workload
- Adding on new IMS systems as workload increases

With previous releases of IMS, each IMS system has its own queues for both input and output messages, and its own expedited message handler for Fast Path messages. Messages are processed in the local IMS system, unless that IMS sends the message to another IMS system.

With IMS Version 6, all IMS systems in a Parallel Sysplex can share a common set of message queues stored in the coupling facility. Full-function and Fast-Path input and output messages can be processed by any IMS system that has access to the shared-queues and is capable of processing the message. Any IMS system running in a Parallel Sysplex can participate in processing application workload as defined in its IMS system definition according to available CPU capacity and resources.

With a shared-queues implementation, new IMS subsystems can easily be added to the sysplex as workload increases or as extra load must be processed. The content of shared-queues is not affected by hardware or software failures. Remaining IMSs continue to schedule their applications and process messages stored in the coupling facility.

In this chapter, we introduce shared queues; we describe their design, components, and message flow; and we provide some information to consider when planning and implementing a shared-queues environment.

We assume a general knowledge of IMS message processing.

2.1 Overview of Shared-Queues Message Processing

Before IMS Version 6, workload distribution was defined statically through such methods as network balancing, the MSC facility, intersystem communication (ISC) links, advanced program-to-program communication (APPC) and the workload router (WLR).

With IMS/ESA Version 6, messages can be placed on a shared-queue that can be processed by any IMS subsystem in the shared-queues group. A shared-queues environment has two sets of queue types; one for full-function and Fast Path. The-full function shared queues contain all input and output messages for full function shared queue processing. The EMH shared queues are used for Fast Path messages. The queues are maintained as list structures on a coupling facility and can therefore potentially be processed by any IMS in the Parallel Sysplex. Any IMS transaction manager (TM) system has the potential to process any message on its shared-queues. The list structures are persistent and recoverable. A shared-queues environment can also be seen as a single-image view of a defined set of multiple IMS system members in a sysplex.

2.1.1 An IMS Shared-Queues Environment

The shared-queues function manages and distributes, on demand, all workload effectively and efficiently to any IMS system in a shared-queues group. Transactions that are entered on one IMS can potentially be processed by any other IMS that has access to the shared-queues. In this redbook, we call this single-image view of multiple IMS systems connected through their coupling links to a common set of list structures in the coupling facility an *IMS sysplex*. Examples of IMS sysplex groupings are all IMS systems in your system test environment, an IMS sysplex for inquiry applications only, or a production IMS sysplex where a set of single IMS DB/DC production systems are set up as a production sysplex environment. Each customer's approach to sysplex configuration will differ according to individual needs.

2.1.2 The Common Queue Server

The transactions entered on an IMS in an IMS shared-queues environment are put on shared-queues by a new interface between IMS and the shared queues: the *Common Queue Server* (CQS). CQS is a generalized queuing facility that manages data objects on a shared queue on behalf of its clients. CQS runs in its own address space and is started by an IMS system. Every IMS system communicates with it's own CQS to manipulate IMS messages sent to and received from shared-queues. There is a one-to-one relationship between each IMS and its associated CQS. Each CQS is connected to the common set of list structures in the coupling facility; up to 32 CQSs can be connected to these structures.

2.1.3 Shared Queues Group

An IMS that shares a defined set of list structures with other IMSs is a member of what is called a *shared-queues group*. A shared-queues group defines a set of IMSs, that can put their messages to the list structures, or get their messages from the list structures. Members can be added to or removed from the shared-queues group dynamically. You can remove membership by shutting down the IMS or by changing the IMS SHAREDQ startup parameter to reference the

name of the procedure library (proclib) member containing the shared queue parameters.

The relationships between the structure names, the structures, and the various proclib members are shown in Figure 1 on page 10.

2.1.4 Registering Interest in Queues

All IMSs in their shared-queues group register interest in particular transactions or LTERMs corresponding to their system definition in shared queues. Each CQS notifies its associated client (IMS) that messages exist for it to process. If that IMS has an available dependent region in which to execute a transaction, IMS asks the CQS to get the message from the shared queues. Multiple IMSs can register interest through their CQS in the same transaction. The first CQS that requests a message from the shared queue receives it and passes the message to its IMS.

IMS also registers interest in output messages for its defined LTERMs to its CQS. If a terminal is logged on to an IMS and messages are held for this terminal, CQS delivers the message to its IMS for terminal output transmission. Messages with unknown LTERM destinations stay on shared queues.

Figure 2 shows a sample-shared queues configuration.

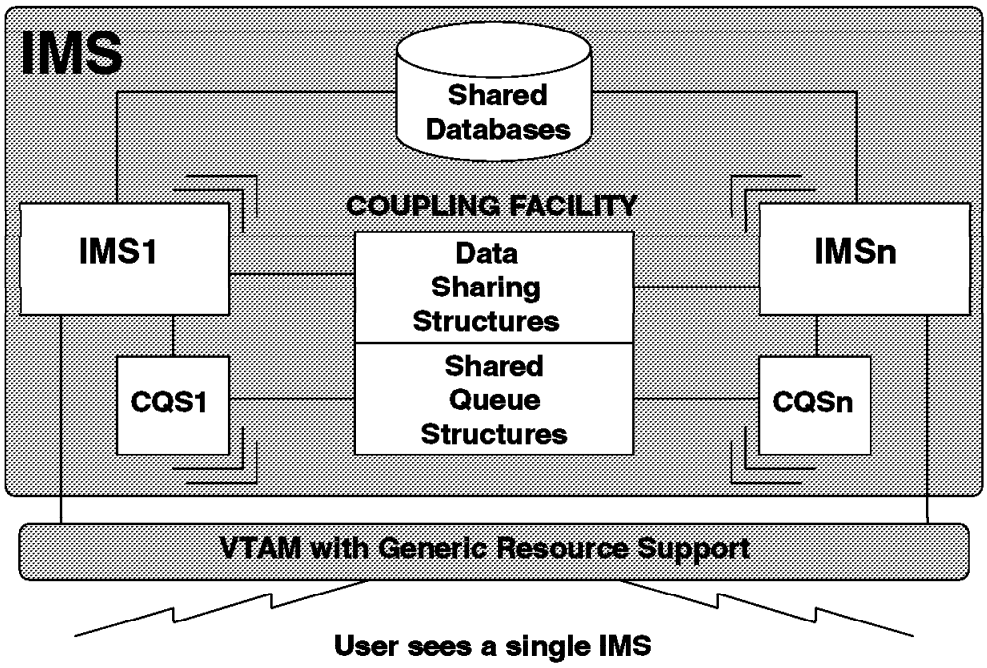


Figure 2. A Shared-Queues Configuration

Any IMS installation can take advantage of the enhanced functionality of shared-queues among multiple IMSs. *IMS/ESA Version 6 Shared Queues*, provides you with information to evaluate and set up efficient IMS workload management in your production environment. The shared-queues function is optional in IMS/ESA Version 6; you can still use conventional message queuing with local queues.

When you implement shared-queues, the IMS sysplex manages your incremental capacity needs with enhanced availability for your application systems. The

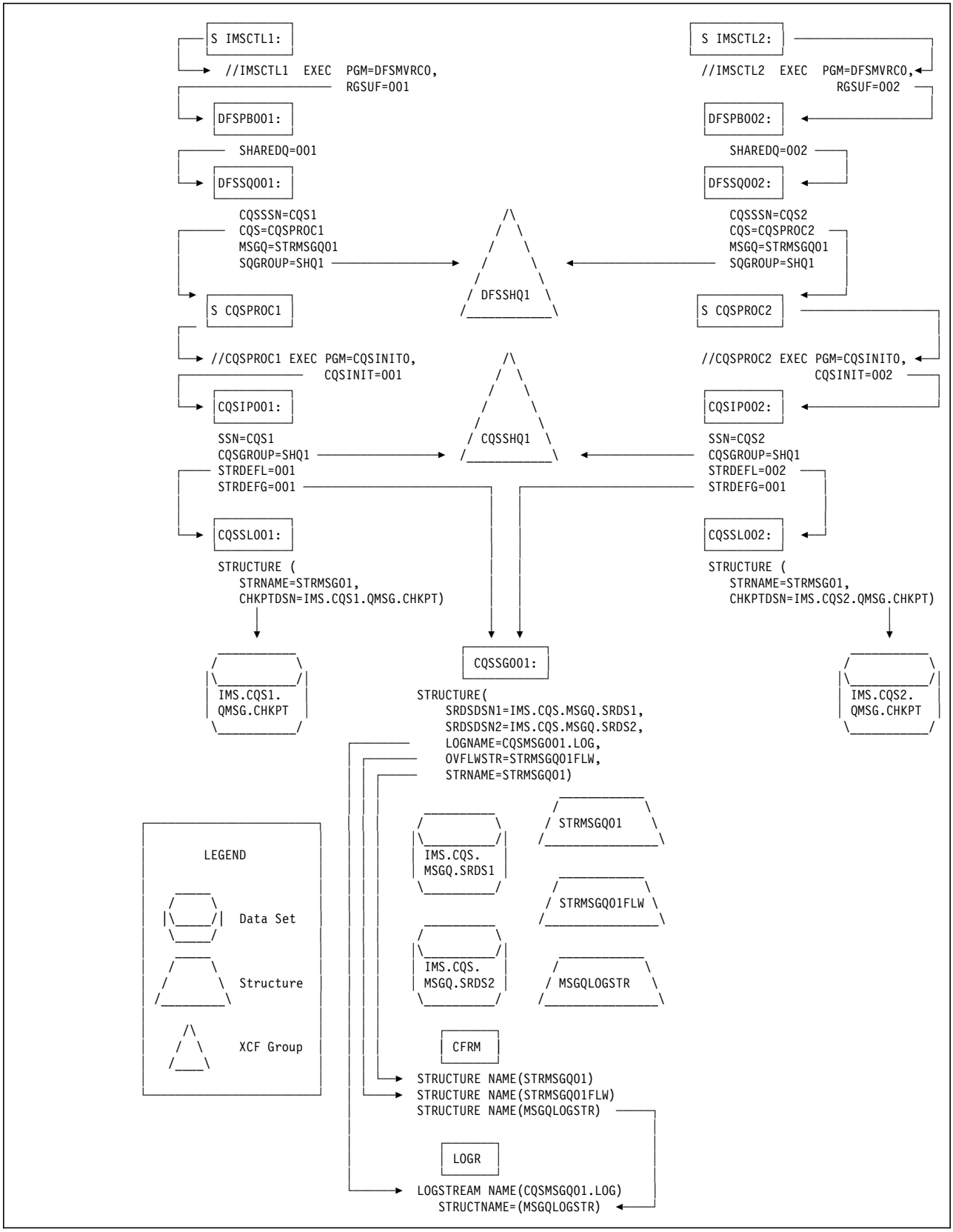


Figure 1. Shared Queue Objects and PROCLIB Member Name Relationships

shared-queue function also reduces the impact of system failures by continuing the message processing on remaining or additional IMSs. It can also eliminate

message traffic overhead and bottlenecks for multiple MSC links. When possible, cloning your IMSs may be an attractive alternative. It additionally reduces the system maintenance effort and allows any IMS in the shared-queues group to process all transactions.

2.2 Message Flow Within a Shared-Queues Environment

Message flow in a shared-queues environment differs from that in a local queuing environment. In this section, we briefly review the general message flow for full-function and Fast Path messages according to the major components used in a shared-queues environment and explain the steps required to process a transaction with its associated output message.

2.2.1 Full-Function Message Queuing

Whenever a full-function transaction arrives in an IMS that has all the resources available to immediately process the message, it puts it on the shared-queue list structure in the coupling facility. The IMS processes the received message. If the IMS does not have the resources available to process the message, then it makes the message available for processing to all other clients in the shared queues group. For further details, see *IMS/ESA Version 6 Shared Queues*, SG24-5088. Figure 3 illustrates how full-function transactions are processed when the message is put on the shared queue.

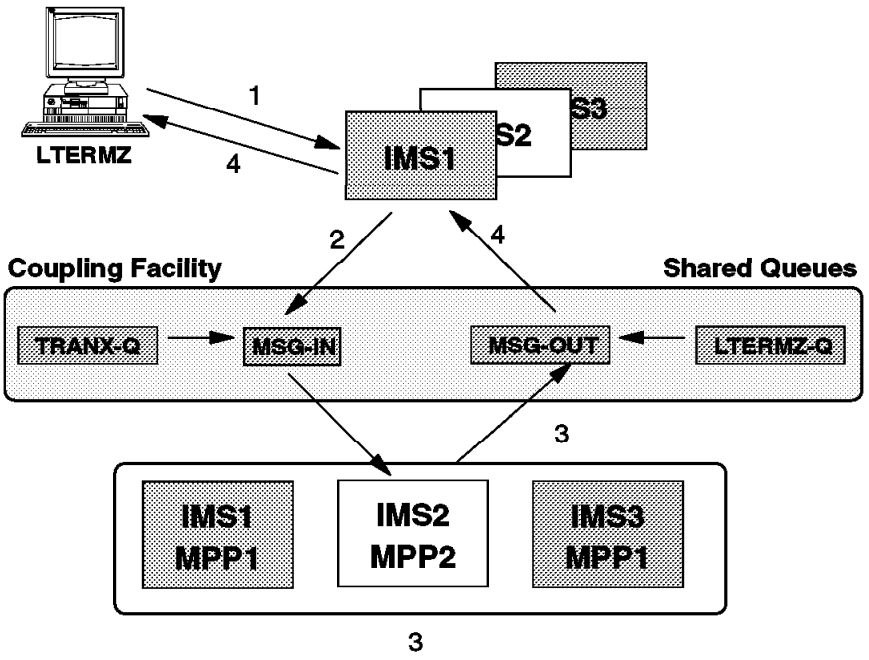


Figure 3. Full-Function Message Processing in a Shared-Queues Environment

The steps are these:

1. LTERMZ establishes a session with IMS1.
2. LTERMZ enters a message (MSG-IN), which is placed on the coupling facility and queued on an MSGQ structure for TRANX.
3. MPP2 starts on IMS2 requests and processes MSG-IN. The output message response (MSG-OUT) is placed in the coupling facility and queued on an MSGQ structure for LTERMZ.

4. MSG-OUT is delivered to LTERMZ by IMS1, as IMS1 has the session with LTERMZ.

2.2.2 Message Queuing for Expedited Message Handling

In general, Fast Path messages are processed in their local system, to avoid access to the shared EMH queues. This mode of processing takes place when the option **local first** (the default) is specified in the Fast Path input/edit routine and an IMS Fast Path (IFP) program processing the program specification block (PSB) is waiting for work in the IMS Fast Path system. Otherwise the message is placed on the shared queue for workload distribution. For additional information about processing a Fast Path transaction, see *IMS/ESA Version 6 Shared Queues*, SG24-5088.

Figure 4 illustrates EMH processing in a shared-queues environment.

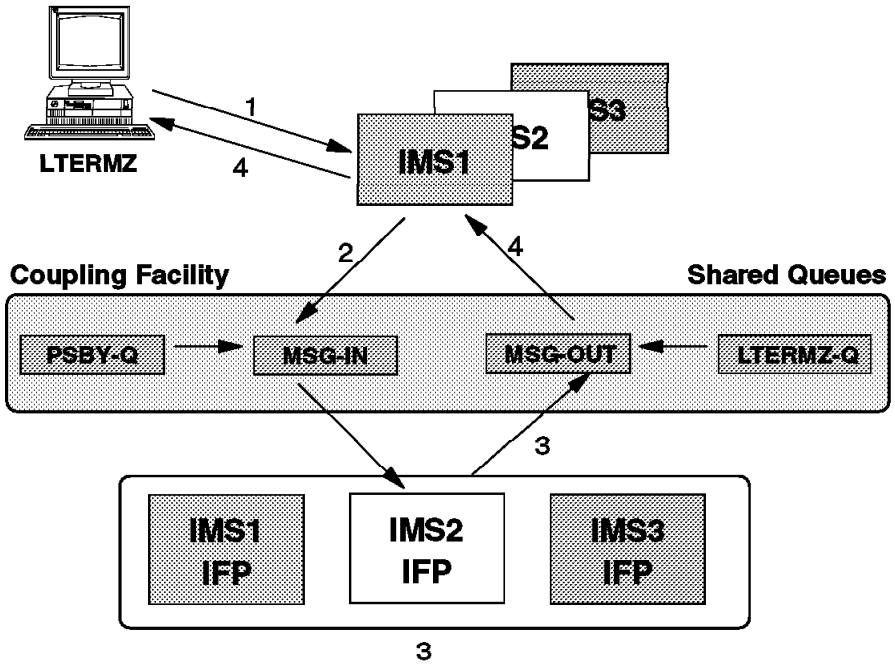


Figure 4. EMH Processing in a Shared-Queues Environment

The steps are these:

1. LTERMZ establishes a session with IMS1.
2. LTERMZ enters an EMH transaction (MSG-IN), which is placed in the coupling facility and queued on an EMHQ structure for PSBY.
3. IFP belonging to IMS2 requests and processes MSG-IN and creates an output message (MSG-OUT), which is placed in the coupling facility and queued on an EMHQ structure for LTERMZ.
4. MSG-OUT is delivered to LTERMZ by IMS1, as IMS1 has the session with LTERMZ.

In a shared-queues environment,

- Messages are queued on coupling facility structures accessible by all IMS systems in the shared-queues group.

- Any IMS that has registered interest in a PSB can request and subsequently process messages queued for that PSB.
- The transaction output is ultimately delivered by the IMS that is in session with the originating terminal.

Chapter 3. Planning for Shared Queues

Planning for the use of shared queues is an activity that involves the analysis of the type of transactions that are processed, the number and the size of the input and output messages that are processed, and the amount of data sharing that exists. This is all required to determine whether it is best to move to a shared queue environment, the optimum size for the message queue buffers and how large the message queue and logging structures need to be.

There are some situations in which it would be best not to move to a shared-queue environment, for example, if you had a large load of APPC, Open Transaction Manager Access (OTMA) or serial transactions. Because these types of transactions must be processed in the local IMS, they would not benefit from being in a shared-queue environment.

This chapter covers the issues that relate to the system definition for IMS when planning to implement shared-queues.

The issues that relate to the IMS system-definition deck that need to be addressed are these:

- Analyzing the IMS system-definition deck
- Duplicate transactions
- Serial transactions
- MSC
- Transaction classes
- Message sizes
- Transactions initiated through APPC and OTMA

3.1 Analyzing the IMS System-Definition Deck

The first step that was performed in the planning phase was to look at how we were going to put IME1 and IME2 into a shared-queue environment without affecting the reliability of the systems and how to provide an easy fallback. The analysis of the two IMxxAPP members found that out of the 3000 or so transaction definitions, there were 40 duplicate transaction names. Each one of these would need to be carefully looked at.

In order to provide a quick fallback, which may last for a long period, we decided that the same IMS system-definition decks would be used in a shared-queue environment as would be used in a non-shared-queue environment. Decks would need to be revisited when data sharing was introduced because this would then allow true workload sharing.

3.2 Duplicate Transactions

To determine what duplicate transactions we had in the two IMxxAPP members, a REXX utility that had previously been written to convert the IMS system definition macros into a flat file was used. The TRANSACT and APPLCTN macros were expanded out in a format that made it easy to sort on transaction name and then scroll through the list and pick up each duplicate that appeared

in column 1. If the number of transactions was a large amount, we would have written another REXX utility to pick the duplicates out of the flat file.

The 40 duplicate transactions fell into two categories:

- Redundant transactions still being used

  A transaction that was called by application code in order to output trace information. This transaction was then processed by a wait-for-input (WFI) BMP which merged the trace information into a single file.

  There are two transactions that appear in both the IME1 and the IME2 decks that are both being used. Each of these transactions has a partner transaction, which is suffixed with the IMS identifier to make it unique. The following is an example of how TRANA has been renamed on both IME1 and IME2 to have a suffix of the IMS ID, which then makes transaction TRANA redundant:

  – Transaction TRANA is defined in IME1 and IME2
  – Transaction TRANA1 is also defined in IME1
  – Transaction TRANA2 is also defined in IME2

  When the IMS-suffixed transactions are defined, the base transaction (TRANA) is meant to have been deleted from the IMS system definition decks. The base transaction is still being used on a very small scale on both IME1 and IME2. Speaking to the application support team made it clear that either the base transaction is still being called somewhere in the application code, or a user is running the transaction manually. Further analysis is required to determine where it is being called from and, if necessary, what changes to make to the application. When this has been determined (and, if required, rectified), the two base transactions can be deleted from the IMS system definition deck.

  The transaction being called by the application code which is used to record trace information was being processed by two WFI BMPs (one on IME and one on IME2). It was more advantageous from an application perspective to merge the trace information from both IME1 and IME2 into a single file. This could easily be achieved in a shared-queue environment by having the transaction processed by only a single WFI BMP running on either IMS system. The transaction TCFLOG was the only transaction defined to run in class 99.

- Redundant transactions.

  The remaining redundant transactions were deleted from both system-definition decks.

3.3 Serial Transactions

The Flexcab application did not have any transactions defined with the SERIAL=YES parameter of the TRANSACT macro. There is a small number of transaction types that do not have to be processed in any order, but they do need to be processed one at a time. The main reason for this requirement is to avoid database contention. We refer to this type of transaction as *pseudo serial* This is achieved by having only one message-processing region (MPR) running on both IME1 and IME2 servicing the class that the pseudo-serial transactions run in. When we go to data sharing, there will be just one MPR servicing the class in the sharing group, rather than one to each IMS. This is because they

access the same database when the database is shared. Pseudo-serial transactions try to prevent simultaneous updates of the same database. They do not have to be processed in a particular order.

3.4 Multiple Systems Coupling

Transactions that routed across the MSC link between IME1 and IME2 in a non-SMQ environment are now transported by means of the message queue structures. No changes were required to the IMS system-definition deck in order to achieve this. This provided us an ideal situation where we could move to a shared-queue environment and the workload would not alter. If we fall back to a non-shared-queue environment, an IMS system generation would not be required and the location of the workload would not alter. We would need only to cold-start IMS with the SHAREDQ parameter turned off and then start the MSC link between IME1 and IME2. This provides a very easy fallback and could easily be achieved within the 1 hour timeframe required by Flexcab management.

When we move to data sharing, the SYSID parameter of the TRANSACT macro will be removed, which allows the transaction to be processed on either IME1 or IME2.

3.5 Transaction Classes

IME1 and IME2 both have their own set of message classes that they use. There are only two classes that are common between the two IMS systems. One is used to process APPC transactions and the other for transactions that are called by application code to dump out trace information. The classes did not have to be changed because the workload will not alter. Transactions will continue to process on the IMS system that they did before shared-queue. When data sharing is introduced, the classes will become important because there will be some work that we will want to process in only a single IMS system. An example of this is the pseudo-serial transactions described in 3.3, “Serial Transactions” on page 16. In a shared-queue environment where data sharing is performed, we still want the pseudo-serial transactions to be input from any IMS system in the sharing group, but we want them to run in only one MPR within the shared-queue environment.

3.6 Size of the Messages

The size of the input and output messages within the Flexcab environment is of importance in determining the IMS buffer size, the size of the message queue structure and how the structure is allocated. Analysis of the message profile was performed using IMSPARS reports. For more information on how the message analysis was performed, refer to D.4.1, “Sizing the Message Queue Structure” on page 135.

3.7 APPC and OTMA

The Flexcab environment uses APPC but does not use OTMA. The APPC transactions that are processed in the Flexcab environment are explicit transactions. This means that the move to a shared-queue environment will have no impact on the way that they need to be defined.

Part 2. Preparation for Shared Queues

Chapter 4. Preparation for IMS Shared Queues

The Telstra systems programming environment (Aplex) was used to implement the initial IMS shared queues. This environment has two IMSs (IMA1CTL and IMA2CTL) running on two MVSs (A01 and A02). The two IMSs are in database sharing for the IMS IVP application using the IMS Resource Lock Manager (IRLM) as the lock manager (IMA0IRM on both MVSs).

This chapter covers only the preparation required to set up the common queue systems (CQS) environment for the IMS IVP application in the Telstra system programmer environment (Aplex). The same process was executed to migrate a shared queue to Telstra's regression testing environment (Cplex). This chapter does not cover the tasks required for setting up IMS database sharing.

These are the preparation steps:

- Naming standards
- Coupling facility definitions
- CQS definitions
- IMS parameters

4.1 Naming Standards

In this section, we discuss the naming standards for

- Shared-queues group name
- CQS address space
- Coupling facility structures
- MVS log stream offload data sets
- CQS data sets

The names used in Telstra are summarized in Figure 14 on page 32.

4.1.1 Shared Queue Group Name

The naming standard for the shared queue group name is:

```
IMx0
```

where x is the data sharing group name. For example, IMA0 is the shared queue group name for IMA1 and IMA2. In this environment, CQS has the same shared queue group name.

4.1.2 CQS Address Space

The name for the CQS address space follows the same naming conventions as for the IMS address spaces. Table 1 on page 22 lists the names of the address spaces in the IMS environment.

Table 1. Address Space Names in the Sysplex		
Task	**Name in MVS A01**	**Name in MVS A02**
Control Region	IMA1CTL	IMA2CTL
DLI	IMA1DLI	IMA2DLI
DBRC	IMA1DBR	IMA2DBR
IRLM	IMA0IRM	IMA0IRM
CQS	IMA1CQS	IMA2CQS

4.1.3 Coupling Facility Structures

For shared queues, IMS uses the coupling facility (CF) for full-function and Fast Path message queues, overflow queues, and logger structures. The naming standards for the IMS structures in the CF are shown in Table 2.

Table 2. Names for the IMS Structures in the Sysplex	
Structure	**Name in the IMS sysplex**
Full Function message	IMA0MSGQ
Fast Path message	IMA0EMHQ
full-function message overflow	IMA0MSGQOV
Fast Path message overflow	IMA0EMHQOV
full-function logger	IMA0MSGL
Fast Path logger	IMA0EMHL

4.1.4 MVS Log Stream Offload Data Sets

MVS log stream offload data sets are required for each of the MSGQ and EMHQ log streams when the structure reaches a set threshold. The naming standard for the MVS log stream offload data sets is:

```
IMA0.CQS.MSGL.LOGOFFLD
IMA0.CQS.EMHL.LOGOFFLD
```

where IMA0 is the shared-queue groupname.

4.1.5 CQS Data Sets

Two checkpoint data sets (MSGQ and EMHQ) are used by each CQS address space. The naming standards for the CQS checkpoint data sets are:

```
IMAn.CQS.MSGQ.CHKPT
IMAn.CQS.EMHQ.CHKPT
```

where n is the instances of IMS (for example, IMA1 and IMA2). There is also a pair of structure recovery data sets (SRDS) for each MSGQ and EMHQ structure. The naming standards for the SRDS datasets are :

```
IMA0.CQS.MSGQ.SRDS1    IMA0.CQS.MSGQ.SRDS2
IMA0.CQS.EMHQ.SRDS1    IMA0.CQS.EMHQ.SRDS2
```

where IMA0 is the shared-queue group name.

4.2 Coupling Facility

There are a number of things that need to be defined to the coupling facility:

- CFRM policy
- CQS structures
- MVS log stream

4.2.1 Coupling Facility Resource Manager Policy

Define the CFRM policy, as shown in Figure 5.

```
//STEP1    EXEC PGM=IXCMIAPU
//STEPLIB  DD   DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DATA TYPE(CFRM) REPORT(YES)
 DEFINE POLICY NAME(CFPOLA2) REPLACE(YES)
 CF NAME(VC1CF01)
 TYPE(9674)
 MFG(IBM)
      PLANT(02)
 SEQUENCE(41826)
      PARTITION(1)
      DUMPSPACE(2000)
```

Figure 5. Define the Coupling Facility Policy

4.2.2 IMS CQS Structures

The CQS message structures are defined in the CFRM policy, as shown in Figure 6. Message structure overhead for IMS Version 6 is about 600 KB using the formula in *OS/390 PR/SM Planning Guide*, GA22-7236. Thus, the minimum structure size definable is 768 KB. More details of structure sizing are given in Appendix D, “Sizing Coupling Facility Structures” on page 129. Since only one coupling facility was available, there is only one entry in the PREFLIST. More than one CF is desirable in production to ameliorate the effects of a coupling facility or link failure.

```
DATA TYPE(CFRM) REPORT(YES)
STRUCTURE  NAME(IMAOMSGQ)          FF message
     SIZE(768)
     INITSIZE(768)
    PREFLIST(VC1LA1)
 STRUCTURE  NAME(IMAOMSGQOV)       FF message overflow
     SIZE(768)
     INITSIZE(768)
    PREFLIST(VC1LA1)
```

Figure 6 (Part 1 of 2). IMS CQS Structures

```
 STRUCTURE  NAME(IMAOEMHQ)         FP message
     SIZE(768)
     INITSIZE(768)
    PREFLIST(VC1LA1)
STRUCTURE  NAME(IMAOEMHQOV)        FP message overflow
     SIZE(768)
     INITSIZE(768)
    PREFLIST(VC1LA1)
STRUCTURE  NAME(IMAOMSGL)          FF logger
         SIZE(512)
         INITSIZE(512)
   PREFLIST(VC1LA1)
STRUCTURE  NAME(IMAOEMHL)          FP logger
        SIZE(512)
        INITSIZE(512)
      PREFLIST(VC1LA1)
```

Figure 6 (Part 2 of 2). IMS CQS Structures

4.2.3 MVS Log Stream

A log stream is a collection of data in a structure in a CF, on a DASD, or a combination of the two. The logger structures and the log streams are defined in the LOGR policy. There is a full-function (FF) logger structure and a Fast Path (FP) logger structure. A sample is shown in Figure 7.

```
//STEP1    EXEC PGM=IXCMIAPU
//STEPLIB  DD   DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 DATA TYPE(LOGR) REPORT(YES)
    DEFINE STRUCTURE NAME(IMAOMSGL)            FF logger
           LOGSNUM(1)
           AVGBUFSIZE(4096)
           MAXBUFSIZE(65272)
    DEFINE STRUCTURE NAME(IMAOEMHL)            FP logger
           LOGSNUM(1)
           AVGBUFSIZE(4096)
           MAXBUFSIZE(65272)
    DEFINE LOGSTREAM NAME(CQS.MSGL.LOGOFFLD)   FF logstream
           STG_DUPLEX(YES)
           LS_SIZE(125000)
           LOWOFFLOAD(0) HIGHOFFLOAD(50)
           STRUCTNAME(IMAOMSGL) HLQ(IMA0)
    DEFINE LOGSTREAM NAME(CQS.EMHL.LOGOFFLD)   FP logstream
           STG_DUPLEX(YES)
           LS_SIZE(64)
           LOWOFFLOAD(0) HIGHOFFLOAD(50)
           STRUCTNAME(IMAOEMHL) HLQ(IMA0)
```

Figure 7. Defining the Logger Structures

4.3 Common Queue Server

A number of things must be set up for the CQS:

- CQS subsystem names
- Program properties table
- Dispatching priority
- RACF-started task
- CQS system check point data sets
- Structure recovery data sets
- CQS address space
- Local structure definitions
- Global structure definitions
- CQS initialization parameters
- Base primitive environment (BPE) parameters
- Authorizing connections to CQS

4.3.1 CQS Subsystem Names

Define the CQS subsystem names to MVS in member IEFSSNxx (IEFSSNA0) of SYS2.PARMLIB

```
SUBSYS SUBNAME(CMA1)          /* IMS CQS for IMA1  */
SUBSYS SUBNAME(CMA2)          /* IMS CQS for IMA2  */
```

4.3.2 Program Properties Table

Define CQS to MVS Program Properties Table (PPT) in member SCHEDxx (SCHED01) of SYS2.PARMLIB

```
PPT PGMNAME(CQSINIT0) CANCEL KEY(7) NOPREF        /* IMS V6 CQS          */
 NOSWAP NOPRIV DSI PASS SYST AFF(NONE)
```

4.3.3 Dispatching Priority

Telstra's Aplex environment is running with Work Load Manager (WLM) in goal mode. The CQS address space name is defined to WLM with Workload of SYSTEM and Service Class of SYSSTC. This is the same for IRLM and IMS Control Region address space.

4.3.4 Started Task Control RACF

Define the CQS started task to the resource access control facility (RACF). For example

```
AU  IMA1CQS DFLTGRP(STCPROCS) OWNER(STCPROCS) PASSWORD(xxxxxx)
ALU  IMA1CQS NAME( CQS STARTED TASK ) RESUME
ALU  IMA1CQS DATA( STC ** CQS STARTED TASK  ID )
CO  IMA1CQS GROUP(QPIMS)  OWNER (QPIMS)
```

4.3.5 CQS System Checkpoint Data Sets

Allocate MSGQ and EMHQ checkpoint data sets for each CQS. The CQS checkpoint data sets contain information used to restart CQS. Figure 8 shows a sample job to delete and define the VSAM ESDS checkpoint data sets for IMA1CQS.

```
//ALLOC EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DELETE (IMA1.CQS.MSGQ.CHKPT) CLUSTER
  IF MAXCC = 8 THEN SET MAXCC = 0
  DELETE (IMA1.CQS.EMHQ.CHKPT) CLUSTER
  IF MAXCC = 8 THEN SET MAXCC = 0
   DEFINE CLUSTER                          -
        (NAME(IMA1.CQS.MSGQ.CHKPT)         -
          TRK(1 1)                         -
          NONINDEXED                       -
          SHAREOPTIONS(2,3)                -
          RECSZ(505,505)                   -
          REUSE                            -
          CISZ(512)                        -
          VOLUMES(SOOOO6))
  DEFINE CLUSTER                           -
       (NAME(IMA1.CQS.EMHQ.CHKPT)          -
        TRK(1 1)                           -
        NONINDEXED                         -
        SHAREOPTIONS(2,3)                  -
        RECSZ(505,505)                     -
        REUSE                              -
        CISZ(512)                          -
        VOLUMES(SOOOO6))
```

Figure 8. Allocating the Checkpoint Data Sets

4.3.6 Structure Recovery Data Sets

Allocate a pair of structure recovery data sets (SRDS1/2) for each MSGQ and EMHQ structure. During structure checkpoints, all recoverable data objects in the structure are written to SRDS. These data sets must be large enough to hold the contents of the primary and overflow message queue structures. Figure 9 on page 27 shows a sample job to delete and define the SRDSs (VSAM entry sequence data sets or ESDSs).

```
//ALLOC EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DELETE IMA0.CQS.MSGQ.SRDS1
  SET MAXCC = 0
  DELETE IMA0.CQS.MSGQ.SRDS2
  SET MAXCC = 0
  DELETE IMA0.CQS.EMHQ.SRDS1
  SET MAXCC = 0
  DELETE IMA0.CQS.EMHQ.SRDS2
  SET MAXCC = 0
  DEFINE CLUSTER                          -
        (NAME(IMA0.CQS.MSGQ.SRDS1)        -
     TRK(100 10)                          -
     NONINDEXED                           -
     SHAREOPTIONS(2,3)                    -
     RECSZ(32761,32761)                   -
     REUSE                                -
     CISZ(32768)                          -
     VOLUMES(SO0006))
DEFINE CLUSTER                            -
        (NAME(IMA0.CQS.MSGQ.SRDS2)        -
     TRK(100 10)                          -
     NONINDEXED                           -
     SHAREOPTIONS(2,3)                    -
     RECSZ(32761,32761)                   -
     REUSE                                -
     CISZ(32768)                          -
     VOLUMES(SO0006))
DEFINE CLUSTER                            -
        (NAME(IMA0.CQS.EMHQ.SRDS1)        -
     TRK(100 10)                          -
     NONINDEXED                           -
     SHAREOPTIONS(2,3)                    -
     RECSZ(32761,32761)                   -
     REUSE                                -
     CISZ(32768)                          -
     VOLUMES(SO0006))
DEFINE CLUSTER                            -
      (NAME(IMA0.CQS.EMHQ.SRDS2)          -
     TRK(100 10)                          -
     NONINDEXED                           -
     SHAREOPTIONS(2,3)                    -
     RECSZ(32761,32761)                   -
     REUSE                                -
     CISZ(32768)                          -
     VOLUMES(SO0006))
```

Figure 9. Allocating the Structure Recovery Data Sets

4.3.7 Setting Up the CQS Address Space

Set up CQS started task job control language (JCL) in members IMA1CQS and IMA2CQS of SYS1.IMS.A00.PROCLIB. Figure 10 on page 28 shows a sample JCL for IMA1CQS.

```
//*********************************************************************
//*           'IMA1CQS' IS THE CQS SERVING IMA1 AND CONNECTED TO IMA2CQS
//*
//*            CQSINIT POINTS TO MEMBER CQSIPMA1 IN PARMLIB CONTAINING
//*            CQS INITIALIZATION PARAMETERS
//*            BPECFG POINTS TO MEMBER BPECFMA1 IN PARMLIB CONTAINING
//*            BPE STARTUP PARAMETERS
//*
//*********************************************************************
//IMA1CQS  PROC RGN=2000K,SOUT=A,
//             BPECFG=BPECFMA1,CQSINIT=MA1
//CQSPROC  EXEC PGM=CQSINIT0,REGION=&RGN,
//             PARM='BPECFG=&BPECFG,CQSINIT=&CQSINIT'
//STEPLIB    DD DSN=IMA1.RESLIB,DISP=SHR
//PROCLIB   DD DSN=IMA1.PARMLIB,DISP=SHR
//SYSPRINT  DD SYSOUT=&SOUT
//SYSUDUMP  DD SYSOUT=&SOUT
```

Figure 10. JCL to Run the CQS

In the sample JCL, &BPECFG is the eight-character name of the BPE configuration PROCLIB member, and &CQSINIT is the three-character suffix for the CQS initialization PROCLIB member CQSIPxxx.

4.3.8 Local Structure Definitions

Set up local structure definitions in member CQSSLxxx (for example CQSSLMA1) in IMS PROCLIB (for example IMA1.PARMLIB). This member identifies the CQS checkpoint data sets for the MSGQ and EMHQ structures as shown in Figure 11.

```
STRUCTURE (STRNAME=IMAOMSGQ,CHKPTDSN=IMA1.CQS.MSGQ.CHKPT,SYSCHKPT=5000)
STRUCTURE (STRNAME=IMAOEMHQ,CHKPTDSN=IMA1.CQS.EMHQ.CHKPT,SYSCHKPT=5000)
```

Figure 11. Local Structure Definitions

4.3.9 Global Structure Definitions

Set up global structure definitions in member CQSSGxxx (for example CQSLMA0) in IMS PROCLIB (see Figure 12). This member contains information relating to the message structures. The exact copy of this member must be used by all CQSs in the same shared-queue group.

```
    STRUCTURE (STRNAME=IMAOMSGQ,OVFLWSTR=IMAOMSGQOV,
    SRDSDSN1=IMAO.CQS.MSGQ.SRDS1,
    SRDSDSN2=IMAO.CQS.MSGQ.SRDS2,
    LOGNAME=CQS.MSGL.LOGOFFLD,
    STRMIN=0,
    OBJAVGSZ=500,
    OVFLWMAX=50)
```

Figure 12 (Part 1 of 2). IMS CQS Structure Definitions

```
    STRUCTURE (STRNAME=IMAOEMHQ,OVFLWSTR=IMAOEMHQOV,
    SRDSDSN1=IMAO.CQS.EMHQ.SRDS1,
    SRDSDSN2=IMAO.CQS.EMHQ.SRDS2,
    LOGNAME=CQS.EMHL.LOGOFFLD,
    STRMIN=0,
    OBJAVGSZ=500,
    OVFLWMAX=50)
```

Figure 12 (Part 2 of 2). IMS CQS Structure Definitions

4.3.10 CQS Initialization Parameters

Set up the CQS initial parameters in member CQSIPxxx (for example, CQSIPMA1) in IMS PROCLIB. This member specifies the parameters used to initialize the CQS address space:

ARMRST=N Specifies whether MVS automatic restart manager is to be used to restart CQS after an abnormal termination (abend).

SSN=CMA1 One- to four-character subsystem name for CQS address space

CQSGROUP=IMA0 One- to five-character identifier. CQS concatenates this to characters CQS to create the CQS XCF group name

STRDEFL=MA1 Three-character suffix for CQS local structure definition proclib member CQSSLxxx

STRDEFG=MA0 three-character suffix for CQS global structure definition proclib member CQSSGxxx.

4.3.11 Base Primitive Environment Parameters

Set up the BPE parameters in the member pointed to by CQS startup parameter BPECFG (for example, BPECFMA1). This member specifies various internal tracing options and CQS exit routines, as shown in Figure 13.

```
********************************************************************
* CONFIGURATION FILE FOR BPE WITH CQS                              *
********************************************************************
LANG=ENU                    /* LANGUAGE FOR MESSAGES (ENU = U.S. ENGLISH) */
# DEFINITIONS FOR BPE SYSTEM TRACES
TRCLEV=(AWE,HIGH,BPE)       /* AWE SERVER TRACE                          */
TRCLEV=(CBS,MEDIUM,BPE)     /* CONTROL BLK SRVCS TRACE                   */
TRCLEV=(DISP,HIGH,BPE,PAGES=12) /* DISPATCHER TRACE WITH 12 PAGES (48K BYTES) */
TRCLEV=(LATC,LOW,BPE)       /* LATCH TRACE                               */
TRCLEV=(SSRV,HIGH,BPE)      /* GEN SYS SERVICES TRACE                    */
TRCLEV=(STG,LOW,BPE)        /* STORAGE TRACE                             */
TRCLEV=(USRX,MEDIUM,BPE)    /* USER EXIT TRACE                           */
```

Figure 13 (Part 1 of 2). BPE Configuration File for Use with CQS

```
# DEFINITIONS FOR CQS TRACES
TRCLEV=(CQS,HIGH,CQS)        /* CQS GENERAL TRACE                    */
TRCLEV=(STR,MEDIUM,CQS)      /* CQS STRUCTURE TRACE                  */
TRCLEV=(INTF,HIGH,CQS)       /* CQS INTERFACE TRACE                  */
# USER EXIT LIST PROCLIB MEMBER SPECIFICATION
/* EXITMBR=(CQSEXITO,CQS)    SPECIFY PROCLIB DATASET              */
                             /* MEMBER CQSEXITO AS CQS'S             */
                             /* USER EXIT LIST MEMBER                */
```

Figure 13 (Part 2 of 2). BPE Configuration File for Use with CQS

4.3.12 Authorizing Connections to CQS

If security checking on the CF is required, then RACF profiles need to be set up. The profile name must be CQSSTR, concatenated with the CQS structure name defined in CQSSGxxx and CQSSLxxx members. An example is

```
RDEFINE FACILITY CQSSTR.IMAOMSGQ UACC(NONE)
 PERMIT CQSSTR.IMAOMSGQ CLASS(FACILITY) ID(xxxx) ACCESS(UPDATE)
SETROPTS CLASSACT(FACILITY)
```

In Telstra′s Aplex (system programmer environment), security was not required so no profiles were set up.

4.4 IMS Parameters

In this section, we describe the IMS shared-queue and CQS parameters, the IMS data communications parameters, and the IMS control region parameters.

4.4.1 IMS Shared Queue and CQS Parameters

Set up parameters DFSSQxxx (for example, DFSSQMA1) related to the shared queues and CQS address space in IMS proclib:

CQS=IMA1CQS CQS procedure name to be started during IMS initialization or with an MVS start command

CQSSSN=CMA1 CQS subsystem name to which local IMS must connect to access the shared queue

EMHQ=IMA0EMHQ Primary EMH shared-queue structure name

MSGQ=IMA0MSGQ Primary full function shared-queue structure name

SQGROUP=IMA0 IMS XCF group name, DFSxxxxx, that IMS connects at startup to enable queue sharing

4.4.2 IMS Data Communication Parameters

Set up IMS data communication (DC) parameters DFSDCxxx (for example, DFSDCMA1) in IMS proclib to allow cloning of IMS systems in the shared-queue group without duplicating any definitions. This is not used at Telstra, because the secondary master is assigned to a spool line and the primary master is automatically reassigned at IMS startup to an automation terminal. VTAM generic resources are not used in this environment. The parameters are these:

PMTO node name for primary master terminal operator (MTO)

PMTO1-8 Lterm names 1 through 8 for primary MTO

PMTOG	Lterm generic name for PMTO. Same for all IMSs in VTAM generic
SMTO	node name for secondary MTO
SMTO1-8	Lterm names for 1 through 8 secondary MTO
SMTOG	generic lterm name for SMTO. Same for all IMSs in the same shared IMS environment
TRUNC	Default for scratch pad area (SPA) truncation option

4.4.3 IMS Control Region Execution Parameters

Set up IMS control region execution parameters DFSPBxxx (for example DFSPBMA1) in IMS proclib, as shown in Figure 14 on page 32. The parameters are these:

SHAREDQ=MA1	To activate shared queues
QBUF	Initial number of queue buffers, default to value in MSGQUEUE macro
QBUFSZ	Buffer size in message queue pool
QBUFMAX	Maximum number of queue buffers that IMS can expand to
SHMSGSZ	Size of short message record in bytes, default to data set control block (DCB) of SHMSG data set
LGMSGSZ	Size of long message record in bytes, default to DCB of LGMSG data set
QBUFHITH	High threshold percentage for message queue buffer to dynamically expand
QBUFLWTH	Low threshold percentage for message queue buffer to dynamically compress
QBUFPCTX	Amount by which message queue buffer expands. It is a percentage of the initial allocated queue buffer, not the size of buffer the buffer pool at time of threshold

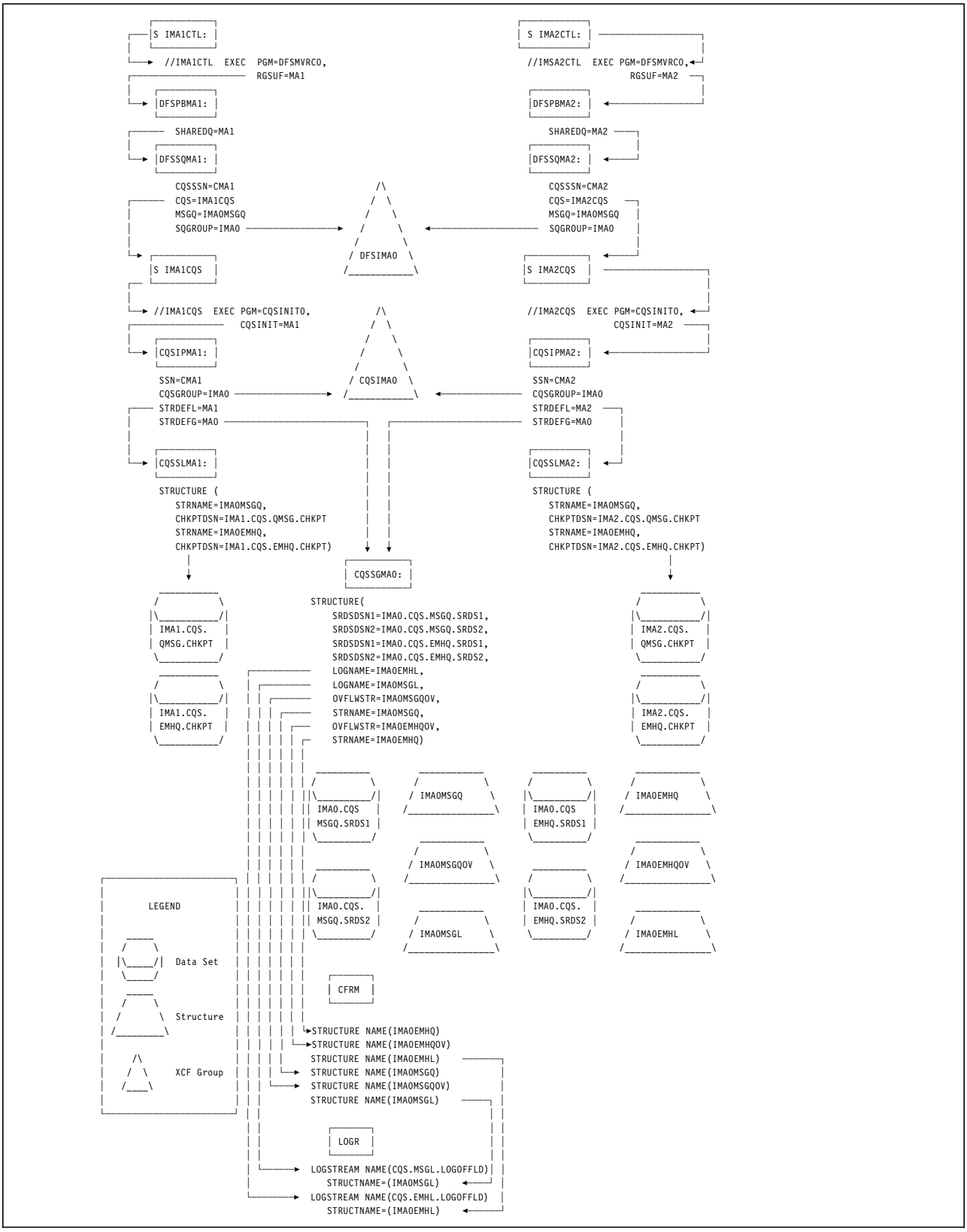


Figure 14. Shared Queue Objects and Proclib Member Names at Telstra

Part 3. Operations and Automation

Chapter 5. IMS Shared Queue Operations

IMS Shared Queues introduces a number of new components and operational changes. Additional operational considerations required in an IMS shared-queue environment include the management of IMS and CQS address spaces, IMS shared-queue structures in the coupling facility, and monitoring and support of the shared-queue environment.

Some of the operational and automation changes observed at Telstra are documented in this chapter, including some messages to be intercepted by automation so that corrective actions can be performed in a timely manner. The operational considerations are these:

1. New components
2. IMS changes
3. New messages
4. Shared-queue commands

5.1 New Components

New components in the IMS shared-queue environment consist of these:

- Common queue server (CQS)
- Coupling facility (CF)
- CQS checkpoints data sets
- Structure recovery data sets (SRDS)
- MVS log stream

5.1.1 Common Queue Server

CQS is the interface for IMS to communicate with the shared queue. It manages the messages in the shared-queue list structures and notifies IMS when work exists for it. The interface takes CQS system checkpoints for CQS recovery and structure checkpoints for structure recovery. CQS runs in a separate address space. One CQS address space is required for each IMS. New operational and automation procedures are required to start up and shut down CQS.

5.1.1.1 Normal Start of CQS

CQS is normally started up by IMS automatically. It can be manually started by issuing the MVS START command:

```
S cqsjobname (for example, S IMA1CQS)
```

When CQS has successfully started, it takes a system checkpoint for each structure:

```
CQSO03OI SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ
CQSO03OI SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
CQSO02OI CQS READY CMA2CQS
```

When IMS has a established connection to it, CQS takes another set of system checkpoints:

```
CQSO003OI SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ
CQSO003OI SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
```

5.1.1.2 Restart of CQS

When CQS is restarted after an abend, it reads its checkpoint data sets to obtain the log token of the last CQS system checkpoint. If this log token is in the structure, CQS restarts warm.

If CQS cannot find the CQS checkpoint log token in the structure, it issues these messages:

```
*394 CQSO032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOMSGQ
*395 CQSO032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOEMHQ
```

Enter a log token (for example, /394, 00000000002AB5E3) to restart CQS warm, or reply cold to cold-start CQS.

Note: The comma after the WTOR number is required.

If CQS is unable to find the log token representing the last CQS system checkpoint in the checkpoint data sets but the log token is available in the structure and the restart information is available in the log stream, CQS then issues these messages:

```
*097 CQSO031A CONFIRM CQS RESTART FOR STRUCTURE IMAOEMHQ  , FROM CHECKPOINT LOGTOKEN 00000000002AB5E3
*098 CQSO031A CONFIRM CQS RESTART FOR STRUCTURE IMAOMSGQ  , FROM CHECKPOINT LOGTOKEN 0000000001408E84
```

Enter CONFIRM to restart CQS

Automation should be set up to ensure that the WTORs are replied to promptly, to avoid delay in CQS startup. Delay in turn delays the IMS startup as IMS does not start until CQS has completed its startup.

5.1.1.3 Cold Start of CQS

If no valid log token is available in the checkpoint data sets and no valid token and restart information is in the structure, the first CQS to start performs a cold start automatically and subsequent CQSs issue these commands:

```
*844 CQSO032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOEMHQ
*845 CQSO032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOMSGQ
```

In this situation, enter COLD, to cold-start CQS.

If, during startup, CQS detects the need for a structure recovery, initiates a structure rebuild. If the structure rebuild fails for any reason, CQS issues this message:

```
CQSO034A CANNOT REBUILD STRUCTURE IMAOMSGQ  FROM LOG, ENTER COLD, CONTINUE OR CANCEL
```

To cold-start CQS, enter COLD. In this situation, data objects in the structure are lost because the structure recovery fails.

Automation should be set up to ensure that the write to operator with reply (WTORs) messages are replied to promptly, to avoid delay in CQS startup.

5.1.1.4 Normal Shutdown of CQS

CQS can be shut down by IMS during IMS normal shutdown or if no IMSs are connected to it, CQS can be shut down by issuing MVS command P cqsjobname (for example, P IMA1CQS). This command is rejected if IMS is running. If CQS is shut down while IMS is active, IMS no longer has access to the shared queues

and any work in progress on that IMS remains in the lock queue until CQS is restarted.

If CQS shuts down normally, it takes a CQS system checkpoint. This checkpoint information is used during normal CQS restart to restore the environment. The log token displayed can be used for CQS restarts, in case CQS checkpoint data sets and coupling facility structures fail:

```
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0MSGQ, LOGTOKEN 00000000017A7390
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0EMHQ, LOGTOKEN 00000000003BD373
CQS0021I CQS SHUTDOWN COMPLETE CMA1CQS
```

CQS purges log records older than two structure checkpoints. Ensure that no CQS system is inactive for that length of time or CQS cannot be warm-started successfully. To prevent this, we recommend that CQS should always be restarted immediately.

5.1.1.5 Abnormal Termination of CQS

If CQS abends, IMS does not automatically abend but stays active. CQS can be restarted by issuing the MVS START command, `S cqsjobname` (for example `S IMA1CQS`). When CQS is restarted, IMS initiates connection to CQS and attempts to resynchronize with CQS.

CQS can be restarted automatically if CQS has been defined to automatic restart manager (ARM). If ARM is not implemented, automation should be put in place to detect CQS failures and restart CQS immediately.

5.1.2 Coupling Facility

IMS shared queues uses the coupling facility (CF) to store its messages in the MSGQ and EMHQ list structures. The EMHQ list structure is required only if the system has been generated as a Fast Path system. The IMS shared-queue list structures are persistent and remain allocated even if all CQSs have disconnected.

5.1.2.1 Structure-Full Condition

When the message structure becomes full and overflow processing is not possible, the following message is issued:

```
*CQS0205E STRUCTURE IMA0MSGQ          IS FULL CMA1CQS
```

This message is issued the first time CQS reaches the structure-full condition, and again when CQS goes out of overflow mode, back to overflow mode, and then reaches the structure-full condition. Automation can trap this message and alert operators to the structure-full condition.

When the structure is full, users entering IMS transactions receive the following message on the input terminals:

```
DFS070  17:24:14  UNABLE TO ROUTE MESSAGE    for non-EMH terminals
DFS2194 17:25:14  SHARED EMHQ NOT AVAILABLE  for EMH terminals
```

5.1.2.2 Structure Rebuild

CQS takes structure checkpoints that write information about the structure to the SRDS to be used for structure recovery. It also uses the MVS logger to record information about the shared-queue structures.

Structure rebuild is required to recover messages in the shared-queue structures. During a rebuild, CQS reads messages from the SRDS and copies

them to the structure. It then reads the logs to recover the structure to the current point in time.

A structure rebuild can be initiated by operators using:

```
SETXCF START,REBUILD,STRNAME=structurename
```

Rebuild can also be initiated by CQS, if it detects an empty structure and a valid SRDS or if the rebuild threshold for loss of connectivity is reached. Figure 15 shows sample messages from a structure rebuild.

```
CQS0240I CQS CMA1CQS  STARTED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ
CQS0241I CQS CMA1CQS  COMPLETED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE REBUILD CMA1CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0020I CQS READY CMA1CQS
CQS0220I CQS CMA1CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQS0200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA1CQS
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA1CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0221I CQS CMA1CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
```

Figure 15. Messages Issued during a Structure Rebuild

5.1.2.3 Structure Failure

If the shared-queue structures are lost, CQS detects the need for a structure recovery. It then attempts to perform a structure recovery from the SRDS and the MVS log stream. However, if the log stream and the SRDS are unavailable, CQS is unable to rebuild the structure and issues this message:

```
*917 CQS0034A CANNOT REBUILD STRUCTURE IMAOMSGQ FROM LOG, ENTER  COLD, CONTINUE OR CANCEL
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN 0000000000469A33
```

In this situation, a cold start of CQS is required. This can be done by automation:

```
R 917,COLD
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN 000000000034B869
CQS0020I CQS READY CMA2CQS
```

If SRDS is available for the recovery but the MVS log stream is not available, CQS issues these messages:

```
CQS0242E CQS CMA2CQS  FAILED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ    RC=0000005E
CQS0244E STRUCTURE RECOVERY REQUIRED AFTER RECOVERY FAILURE FOR STRUCTURE IMA
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE REBUILD CMA2CQS
BPE0006I CQS  STRD TCB ABEND U0014-00000390, THD=STRD
```

In this situation the SRDS needs to be deleted and redefined to force a cold start of CQS.

5.1.2.4 Structure Overflow

When the usage of the MSGQ and EMHQ structures reaches the percentage defined in OVFLWMAX parameter of the global CQS proclib member (CQSSGxxx), CQS moves the queue with the largest used data elements to the overflow structure, until the usage of the data element is less than (OVFLWMAX - 20%). When this threshold is reached, the messages shown in Figure 16 on page 39 are issued in CQS address space.

```
CQS0260I CQS CMA2CQS  STARTED OVERFLOW THRESHOLD PROCESSING FOR STRUCTURE IMAOMSGQ
CQS0200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 1 CMA1CQS
CQS0261I CQS CMA2CQS  COMPLETED OVERFLOW THRESHOLD PHASE 1 FOR STRUCTURE IMAOMSGQ
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 1 CMA1CQS
CQS0200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 2 CMA1CQS
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 2 CMA1CQS
CQS0262I CQS CMA2CQS  COMPLETED OVERFLOW THRESHOLD PHASE 2 FOR STRUCTURE IMAOMSGQ
CQS0220I CQS CMA2CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQS0200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA1CQS
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA1CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0221I CQS CMA2CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
```

Figure 16. Sample Messages Issued When Overflow Threshold Is Reached

To determine which queues are in the overflow structure, use the /DISPLAY OVERFLOWQ command.

5.1.2.5 Resizing CQS Coupling Facility Structures

Over time, the structures in the coupling facility may need to be resized to accommodate growth in the system. To resize the structures,

- The CFRM policy needs to be updated.
- The new CFRM policy is activated using the command

  SETXCF START,POLICY,TYPE=CFRM,POLNAME=policyname
- Initiate rebuild using the SETXCF REBUILD command.

The resizing change remains pending until the rebuild of the structures occurs. The structures are quiesced during certain stages of the rebuild and this quiescing may impact all the IMSs in the shared-queue group accessing that structure.

5.1.3 CQS Checkpoint Data Sets

Each CQS has two VSAM ESDS data sets for use as checkpoint data sets, one for the MSGQ and the other for the EMHQ (if Fast Path is defined). These are dynamically allocated at CQS initialization, and contain status information taken during CQS system checkpoint. They are required by CQS for restarts.

5.1.3.1 Print CQS Checkpoint Data Sets

IDCAMS can be used to print the contents of the CQS checkpoint datasets, as shown in Figure 17.

```
//JS010    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//FILE1    DD DISP=SHR,DSN=IMA1.CQS.MSGQ.CHKPT
//SYSIN    DD *
PRINT INFILE(FILE1)
```

Figure 17. JCL to Print a CQS Checkpoint Data Set

5.1.4 Structure Recovery Data Sets

Structure recovery data sets (SRDSs) contain information for recovering structures. A pair of SRDS is required for each IMS message queue list structure. During structure checkpoint, the SRDS are dynamically allocated and all recoverable data objects from the structure are copied to the SRDS. Thus, the SRDSs must be large enough to hold the data objects from the primary and overflow message queue structures.

During the structure checkpoint process, CQS quiesces all activities against the structure. As no work for the structure can be performed while this is happening, structure checkpoints should be taken during non-peak time to minimize impact to the users. Telstra intends to take one structure checkpoint during the night. If DASD space is an issue, two structure checkpoints can be taken to clean up the log stream.

5.1.4.1 Redefine SRDS

After SRDSs are deleted and redefined, a structure checkpoint should be taken to allow structure recovery when required. If CQS detects that a structure contains client data, the data is not available on the SRDSs, and the CQS log stream is not valid for structure recovery, then CQS issues the following message during CQS start up:

```
CQS0009W STRUCTURE IMAOMSGQ         REQUIRES STRUCTURE CHECKPOINT FOR RECOVERY
```

Automation should trap message CQS0009W and take a structure checkpoint.

If a structure checkpoint is not taken in response to CQS0009W and the structure fails, CQS cannot rebuild the structure and issue CQS0034A. Data objects in the structures are lost when the structure is unable to recover.

As SRDSs can be deleted and defined while CQS is up and CQS does not detect that the SRDS were redefined until the next restart or during the next structure checkpoint, CQS0009W message may not issued for a long while after the SRDS are deleted and redefined. The process to delete and define SRDSs should therefore include taking a structure checkpoint immediately or at the next non-peak time (regardless of whether the log stream is available or not), as a safety precaution to allow future structure recovery if required.

5.1.4.2 SRDS Size

The SRDSs must be large enough to hold all the data objects from the primary and overflow structures. If the SRDS is too small, the following message is issued during structure checkpoint:

```
IEC070I 203-204,IMA2CQS,IMA2CQS,SRDSD101,1295,S00006,
IEC070I IMA0.CQS.MSGQ.SRDS1,IMA0.CQS.MSGQ.SRDS1.DATA,CAT.USER03.A00
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA2CQS
CQS0054E WRITE FAILED FOR SRDSDSN1, RC=00000008/2908001C CMA2CQS
CQS0054E DSN=IMA0.CQS.MSGQ.SRDS1                          CMA2CQS
CQS0222E CQS CMA2CQS  FAILED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
```

CQS continues processing but the structure checkpoint process is abandoned. An operational process should be put in place to increase the size of the SRDSs when the structure sizes are increased. Automation can be used to alert the operator to message CQS0222E, so that the SRDS can be resized to accommodate the structures.

5.1.4.3 Printing the Structure Recovery Data Sets

IDCAMS can be used to print the contents of the SRDS, as shown in Figure 18.

```
//JSO1O    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//FILE1    DD DISP=SHR,DSN=IMAO.CQS.MSGQ.SRDS1
//SYSIN    DD *
   PRINT INFILE(FILE1)
```

Figure 18. JCL to Print the Structure Recovery Data Set

5.2 IMS Changes

In a shared-queue environment, additional IMS operational and automation processes are required during IMS startup and shutdown.

Note: AOI transactions scheduled by the AOI exit (DFSAOUE0) should be defined as serial to guarantee that the transactions run locally.

Operational considerations include these:

- Normal IMS start
- IMS cold start
- IMS emergency restart
- IMS normal shutdown

5.2.1 Normal IMS Start

During IMS normal start, IMS starts up CQS address space if it is not already up. IMS issues this message:

```
*DFSO226A CTL REGION WAITING FOR CQS (IMA2CQS ), RESPONSE TO CONNECT REQUEST
```

If CQS is initializing, no action is required; CQS will complete the connect processing. If CQS issues a WTOR as part of the CQS initialization message, respond to the WTOR to complete CQS initialization. After CQS initialization, IMS identifies itself to the CQS as part of its startup.

5.2.2 IMS Cold Start

If IMS is shut down cleanly before the cold start, it has no effect on the shared queues. If IMS failed to shut down cleanly and was then cold started, all in-doubt messages are moved from the lock queue to the cold queue,where they stay (in the cold queue in the coupling facility) until the structure is deleted or, the IMS/ESA Message Requeuer software can be used to move the messages from the cold queue to the ready queue.

5.2.3 IMS Emergency Restart

CQS need not necessarily terminate when IMS abends. During emergency restart IMS connects to CQS and attempts to resynchronize with CQS to resolve any in-flight work in the lock queue.

5.2.4 Normal IMS Shutdown

In a shared-queue environment, IMS is normally shut-down using the command /CHECKPOINT FREEZE.

/CHECKPOINT DUMPQ can be used, but it is treated as /CHECKPOINT FREEZE in the shared-queue environment. If CQS is not active while IMS is processing the /CHECKPOINT FREEZE, the shut down hangs and this message is issued:

```
DFS1937I  IMS SHUTDOWN WAITING FOR CQS   IMA1CQS
```

A display on IMS shutdown status is shown in Figure 19.

```
R,297 /DISPLAY SHUTDOWN STATUS

DFS000I     TERMINAL  USER     STATUS   IMA1
DFS000I        1-   1          INPUT  IN PROCESS   IMA1
DFS000I     TERMINAL  USER     STATUS   IMA1
DFS000I       NO OUTPUTTING LINES   IMA1
DFS000I         MSG-IN    1     MSG-OUT   0   IMA1
DFS000I       MASTER ACTIVE   IMA1
DFS000I     IMSLU=XANET.IMA1APPC     #APPC-CONV=         0  PURGING    IMA1
DFS000I     OTMA PHASE=0    IMA1
DFS000I     CQS JOBNAME = IMA1CQS  NOT AVAILABLE     IMA1
DFS000I     *98229/133332*   IMA1
*928 DFS996I *IMS READY*  IMA1
```

Figure 19. Displaying IMS Shutdown Status

If CQS is unavailable when the shutdown command is issued, the command is rejected:

```
DFS1975 COMMAND REJECTED AS CQS IS NOT AVAILABLE
```

In both these cases, CQS must be restarted before IMS can be shut down successfully. Automation should be set up to alert the operator to the DFS1975 and DFS1937I messages and start up the CQS pointed to by the CQS JOBNAME field so that IMS can be shut down without delay.

During normal IMS shutdown, CQS also shuts down. To prevent CQS from shutting down when IMS is shut down, use the NOCQSSHUT keyword on the IMS shutdown command:

```
/CHECKPOINT FREEZE NOCQSSHUT
```

Existing operational and automation procedures to shut down IMS can be changed to use NOCQSSHUT if CQS address space is not to be shut down when IMS shuts down.

5.3 New Messages

IMS shared queues introduces a number of new messages. These messages are prefixed with CQS and DFS. Following are some new shared-queue messages suggested for automation. For a complete list of new and changed messages, refer to the section “New, Changed and Deleted Messages and Abends” of the *IMS/ESA Messages and Codes*, GC26-8739.

These are the new messages suggested for automation:

DFS226A	If CQS is initializing, then no action. Respond to CQS WTOR if any.

DFS3909A	Enter RETRY when CQS ready or ABORT to terminate IMS.
CQS0008W	Structure is volatile. Consider taking a structure checkpoint.
CQS0009W	Structure requires structure checkpoint for recovery. Consider taking a structure checkpoint.
CQS0031A	Respond to CQS WTOR to restart for structure from checkpoint token.
CQS0032A	Respond to CQS WTOR for CQS restart for structure.
CQS0033A	Enter checkpoint token for CQS restart for structure.
CQS0034A	Respond to CQS WTOR for structure rebuild.
CQS0205E	Structure full. Consider removing data objects from structure
CQS0222E	Structure checkpoint failed. Analyze the return code of this message and take corrective action.
CQS0352E	Log write error. Analyze the reason and take corrective action.

5.4 Shared-Queue Commands

After the shared queue environment is implemented, the shared-queue components needs to be supported and monitored.

5.4.1 IMS Commands

Below are some useful IMS commands available in the shared-queue environment. For a complete list and details of commands, refer to *IMS/ESA Operations Guide*, SC26-8741.

To shut down IMS without shutting down CQS, enter:

```
/CHECKPOINT FREEZE NOCQSSHUT
```

To display the status of the CQS, enter:

```
/DISPLAY CQS

JOBNAME   VERS#  STATUS
IMA1CQS     1.1  CONNECTED
*98225/170643*
```

To display the status of the IMS list structures for all shared queues, enter

```
/DISPLAY STRUCTURE ALL

STRUCTURE NAME    TYPE  STATUS
IMAOMSGQ          MSGQ  CONNECTED, AVAILABLE
IMAOEMHQ          EMHQ  CONNECTED, AVAILABLE
*98225/170615*
```

To display the status of the IMS overflow structures, enter:

```
/DISPLAY OVERFLOWQ STRUCTURE ALL

STRUC-RSCTYPE    OFLSTRUC-RSCNAME LUNAME-TMEMBER    TPNAME-TPIPE
IMAOMSGQ         IMAOMSGQOV
TRANSACTION      PART
IMAOEMHQ         IS NOT IN OVERFLOW MODE        LINE 001 PTERM 001.
*98210/140601* LINE 001 PTERM 001.
```

To display the queue count for a specific queue type for queues older than the message age specified, enter:

```
/DISPLAY QCNT TRANSACTION ] LTERM ] BALG ] APPC ] OTMA ] REMOTE MSGAGE n
```

For example, for transactions, enter:

```
/DISPLAY TRANSACTION tranname QCNT

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       1             1    98231/131718 98231/131718
*98231/163536*
```

To display the queue count for a specific transaction name in the structure, enter:

```
/DISPLAY TRANSACTION tranname QCNT

TRAN          GBLQCT
IVTNV              1
IVTNO              2
*98243/112714*
```

To dequeue transactions from the ready or lterm queue, first stop the transactions and nodes. They must be stopped for transactions to dequeue. Enter this:

```
/DEQUEUE TRANSACTION ] LTERM ] MSNAME  xxxxxxx  PURGE ] PURGE1 ] FIRST
```

To activate the queue manager and shared-queue tracing, enter:

```
/TRACE SET ON ] OFF  TABLE QMGR  SQTT
```

To display status of the queue manager and shared-queues trace table, enter:

```
/DISPLAY TRACE TABLE QMGR ] SQTT
```

5.4.2 CQS Commands

Below are new commands to support CQS. The details of these commands are in *IMS/ESA Operations Guide*, SC26-8741.

To initiate a CQS structure checkpoint of all structures or of a specified structure, enter:

```
/CQCHKPT SHAREDQ STRUCTURE ALL ] structurename
DFS1972I CQCHKPT SHAREDQ COMMAND COMPLETE FOR STRUCTURE=IMAOMSGQ
DFS1972I CQCHKPT SHAREDQ COMMAND COMPLETE FOR STRUCTURE=IMAOEMHQ
```

To initiate a CQS systems checkpoint for all structures or for a specific structure, enter:

```
/CQCHKPT SYSTEM STRUCTURE ALL ] structurename

DFS1972I CQCHKPT SYSTEM  COMMAND COMPLETE FOR STRUCTURE=IMAOMSGQ
DFS1972I CQCHKPT SYSTEM  COMMAND COMPLETE FOR STRUCTURE=IMAOEMHQ
```

If CQS shutdown sharedq is set to ON, a structure checkpoint is taken during CQS shutdown. If more than one CQS is shut down, then more than one structure checkpoint is taken. Taking more than one structure checkpoint delays CQS and IMS shutdown time and hangs other IMSs that are still active in the shared-queue environment. To address this problem, we recommend using this sequence:

```
/CQSET SHUTDOWN SHAREDQ  ON ] OFF STRUCTURE ALL ] structurename

CQSO220I CQS CMA1CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN ...
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQSO221I CQS CMA1CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQSO021I CQS SHUTDOWN COMPLETE
```

To display information about the coupling facility list structure holding IMS messages, enter:

```
/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ             5619      856     5617     1705 0001/0001
IMAOMSGQOV            N/A      N/A      N/A      N/A    N/A
IMAOEMHQ              126        4      127        4 0001/0001
IMAOEMHQOV            N/A      N/A      N/A      N/A    N/A
*98231/165104*
```

All new /CQx commands are supported from AOI.

5.4.3 Cross-System Coupling Facility Commands

Below are some XCF commands required to support the IMS shared-queue environment. For a complete list and details of XCF commands, refer to *OS/390 MVS System Commands*, GC28-1781.

To display the IMS message structure, enter the commands shown in Figure 20 on page 46.

```
D XCF,STR,STRNAME=IMAOMSGQ

IXC360I  07.56.34  DISPLAY XCF 474
STRNAME: IMAOMSGQ
STATUS: ALLOCATED
     POLICY SIZE   :768 K
     POLICY INITSIZE: 768 K
     REBUILD PERCENT: N/A
     PREFERENCE LIST: VC1LA1
     EXCLUSION LIST IS EMPTY

ACTIVE STRUCTURE
----------------
     ALLOCATION TIME: 08/14/1998 16:29:23
     CFNAME        :VC1LA1
     COUPLING FACILITY: 009674.IBM.02.000000041826
                    PARTITION: 1   CPCID: 00
     ACTUAL SIZE   :768 K
     STORAGE INCREMENT SIZE: 256 K
     VERSION       :B0E70786 D3BAD709
     XCF GRPNAME   :IXCLO017
     DISPOSITION   :KEEP
     ACCESS TIME   :NOLIMIT
     MAX CONNECTIONS: 8
     # CONNECTIONS :2

CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
---------------- -- -------- -------- -------- ---- -------
CMA1CQS          01 00010087 A01      IMA1CQS  002B ACTIVE
CMA2CQS          02 00020077 A02      IMA2CQS  0167 ACTIVE
```

Figure 20. XCF Commands to Display IMS Message Queue Structure

To change the actual size of a structure (within the range of INITSIZE and SIZE defined in the CFRM policy), enter

SETXCF START,ALTER,STRNAME=structurename,SIZE=nnnn

To terminate a failed connection when the structure must be deleted, enter:

SETXCF FORCE,CONNECTION,STRNAME=structurename,CONNAME=ALL

For example, use this:

```
SETXCF FORCE,CONNECTION,STRNAME=IMAOMSGQ,CONNAME=CMA2CQS

IXC354I THE SETXCF FORCE REQUEST FOR CONNECTION 808
CMA2CQS IN STRUCTURE IMAOMSGQ WAS COMPLETED:
CONNECTION WAS DELETED
```

To delete the structure, enter:

SETXCF FORCE,STRUCTURE,STRNAME=structurename

For example, use this:

```
SETXCF FORCE,STR,STRNAME=IMAOMSGQ

IXC353I THE SETXCF FORCE REQUEST FOR STRUCTURE 838
IMAOMSGQ WAS COMPLETED:
STRUCTURE WAS DELETED
```

To initiate structure rebuild or structure copy, use

```
SETXCF  START,REBUILD,STRNAME=structurename
```

To activate a CFRM policy, use

```
SETXCF  START,POLICY,TYPE=CFRM,POLNAME=policyname
```

5.4.4 IMS Commands Not Effective with Shared Queues

The following commands are no longer effective in an IMS shared-queue environment:

- /DISPLAY Q
- /DISPLAY Q TRAN
- /NRESTART BUILDQ FORMAT SM] LM] QC
- /CHECKPOINT SNAPQ] DUMPQ] PURGE
- /ERESTART BUILDQ

5.4.5 IMS Online Change

IMS does not coordinate online changes across the sysplex. A procedure to use is this:

1. /MODIFY PREPARE on each IMS.
2. /DISPLAY MODIFY on each IMS and resolve any work in progress.
3. /MODIFY COMMIT on each IMS.
4. /MODIFY ABORT on each IMS where online change is not successful.
5. Undo the online change with another online change if any command fails after an online change has been successful.

This complex process is easier with cloned configurations and shared ACBLIB, MATRIX, MODBLKS, and FORMATx across the sysplex. MODSTAT IDs must be the same across the sysplex.

Chapter 6. Monitors and Reporting in the Shared-Queue Environment

Telstra uses several monitor and reporting products to focus on performance issues. Some of them are used to investigate and locate problems, and others are used as measurement tools when running application tests.

6.1 IMS Monitoring Tools

Five IMS monitoring tools are used at Telstra:

- IMS performance analysis and reporting system (IMSPARS)
- Omegamon/IMS
- Epilog/IMS
- SAS PDB
- IMS utilities

6.1.1 IMS Performance Analysis and Reporting System

The IBM product IMSPARS is used for historical log analysis for the purpose of performance measurement.

IMSPARS is a performance analysis and tuning aid for IMS database and data communication systems. It performs statistical analysis by processing the IMS log records. It provides several optional graphical and tabular reports of the analysis dealing with

- Transaction transit times
- IMS resource usage
- IMS resource availability

IMSPARS produces a variety of optional reports for all working levels within the IMS business. The reports range from management exception reports, which show whether or not critical operands are within specified limits, to detailed reports about, for example, transaction transit time reports by logical terminal, by transaction code, by message class, or by time of day. Such reports can identify transactions involved in poor response and may indicate the cause of the problem. Certain problems can be tracked back to their sources.

Telstra uses IMSPARS to provide historical performance data to assist in diagnosis and resolution of performance-related problems.

6.1.2 Omegamon/IMS

Omegamon/IMS from Candle Corp, is a real-time performance monitor providing basic real-time data about the IMS environment. It has a CUA interface and provides automatic alerts when predefined service levels are not met or system problems occur.

Telstra uses Omegamon/IMS to alert operators to existing or potential problems using the Omegamon/IMS exception thresholds. Omegamon/IMS real-time

commands are used to further identify and resolve IMS performance-related problems.

6.1.3 Epilog/IMS

Omegamon Epilog from Candle Corp, has three major subcomponents:

- The Epilog collector
- The Epilog reporter
- The maintenance utilities

The Epilog collector collects several different types of performance-related information from IMS log records and selected IMS control blocks. Information collected includes transaction response time, resource utilization, and degradation data. The Epilog collector usually is started automatically when Omegamon address space is started. It writes data to the Epilog data store at the end of each collection interval, which is typically the standard RMF interval of 15 minutes.

Telstra uses Epilog to provide historical performance data to assist in performance-related problem diagnosis and resolution.

6.1.4 SAS Performance Data Base

The most frequently used product is SAS PDB. It is mainly used by application development to produce performance-related reports for the testing of new or changed applications. It is also used by system support staff during problem diagnosis and resolution.

SAS is a product that uses the IMS log data sets as input. Each day the results of the analysis of the SLDS is written to the performance data base (PDB). SAS provides an easy-to-use interface to create individual reports based on the contents of the PDB. There are many in-house developed SAS and assembler programs providing reports the end user needs.

6.1.5 IMS-Related Programs

The programs DFSILTA0 and DFSISTS0 are tools used occasionally to process IMS logs.

DFSILTA0 reads IMS online logs to collect information about individual IMS transactions. It collects information on the transaction code, the message processing program (MPP) processing that transaction, the assigned dependent region, the priority, and the transaction class. It calculates the time:

- When the transaction is received.
- When the get unique call is issued.
- When the message processing program terminates.
- When the transaction was placed on the output queue.
- When the output message starts to the terminal.

The utility calculates total response time, time on the input and output queues, and processing time. It can be used to provide a new look tailored to the user′s specification.

DFSISTS0 reads online logs as they are, or tailored by DFSILTA0 to produce statistics on IMS transactions.

Although there are no message queues in a shared-queues-environment, the log records read by these utilities will still be in place. Their meaning is slightly changed because the message queues are replaced by the shared-queue residing in the coupling facility.

The utilities cannot process CQS logs, but they still provide the same information as in a non-shared-queues environment

6.1.6 Planning for Shared Queues

Today all of the above-mentioned monitors and tools are available for IMS Version 6. They can process the IMS Version 6 log record format and are in production on all systems where IMS Version 6 is in production mode. However, none of the products is focused on the special considerations for the shared-queues environment. For example, they cannot read the logs produced by CQS.

One of the planning considerations is to find one or more tools that can be used by all the persons involved in running a shared-queues environment.

6.2 MVS Tools

These two tools are used to monitor the system at the MVS level:

- CA-ASTEX
- Strobe

6.2.1 CA-ASTEX

CA-ASTEX is used to monitor real-time and historical DASD performance.

ASTEX (an acronym for Automated Storage Expert) is a product that analyzes the performance of the cache and DASD levels of the storage hierarchy, finds where problems exist, then maximizes performance by recommending solutions to I/O response time problems and cache utilization problems. In addition, ASTEX can automatically optimize the utilization of cache and DASD resources.

6.2.2 STROBE

STROBE hooks into a target address space and monitors all activity. This tool is mainly used by application developers.

The STROBE performance measurement system has two major components: the monitoring facility, which collects performance data during execution of the measured program or subsystem; and the reporting facility, which analyzes, reduces, and reports measurement data. In addition, the ISPF dialog provides convenient, full-screen control of both facilities. The STROBE monitoring facility collects performance data during execution of a target program or subsystem by deploying a measurement task within the address space of the program or subsystem.

6.2.3 Monitors for the Shared-Queues Environment

At present, there is only one monitor for the shared-queues environment.

6.2.3.1 IMS/ESA Performance Analyzer

IMS/ESA Performance Analyzer (IMSPA) is the follow-on product for IMSPARS. For the duration of the implementation of IMS shared queues at Telstra, we used a pre-release version of IMS/ESA Performance Analyzer Version 1 Release 2. This release became generally available after the residency completed.

The log reports produced by IMSPA are used rather intensively by Telstra system programmers. As each IMS in a shared-queues environment writes the same log records it would write in a non-shared-queues environment, we could expect to get the same familiar reports to work with, except for the Resource Usage Report for Message Queue Buffers (which shows zero for all values).

IMSPA supports shared-queues insofar as it can read log data sets from more than one IMS. If an IMSPA report is run with the input of the system log data sets (SLDSs) of different IMSs, the Resource Usage and Availability Report is produced separately for each IMS system in the sysplex, indicating the IMS ID in the heading.

To produce Transaction Transit Time reports, the log data sets of the involved IMSs are merged before calculating the transit times, and one report is created for each sysplex. This has to be done, since a message placed on a shared-queue can be processed by any IMS system that has access to the shared-queue. There are three steps:

1. Accepting the input message from the terminal and putting it on the shared-queue
2. Scheduling an application, reading the message from the shared-queue, processing it, and putting an output message on the shared-queue
3. Reading the output message from the shared-queue and sending it to the terminal.

Each of these transaction processing steps in a shared-queues environment can be processed by a different participating IMS subsystem. Transaction Transit Time reports take this fact into account and show additional columns containing CQS times, originating IMS ID and processing IMS ID.

See Appendix C, “IMS and CQS Log Records” on page 117 for a description of the log records created during the processing of a transaction.

Expectations: IMSPA leaves it as a user responsibility to provide all the log data sets necessary to produce a comprehensive Transaction Transit Time report for a shared-queues environment. The lack of input SLDS validation, and failure to include all necessary SLDSs, could lead to errors in the reporting. For the next release of IMSPA, it is planned to get SLDS information from the Recon, based on the time period for which the report is requested. IMSPA should then allocate the correct SLDSs automatically.

Although IMSPARS is running out of service and the base GPARS has restrictions in its Year 2000 capability, Telstra has not yet decided to replace the two of them with IMSPA.

6.2.3.2 Requirements

There is limited information about the shared queues provided by standard IMS and CQS commands. When going into production with shared-queues, it is vital to have more information. For example, there is a need to know:

- The length of queue on the shared-queue structure. This would involve providing exception handling that would be triggered when the queue length exceeds defined limits.
- Display of structure details. Information of interest includes structure sizes, number of data elements, and list-element controls allocated and used. Handling exceptions should also be triggered when a defined percentage of use of the data elements or list-element controls is exceeded. This is essential, as running out of either forces IMS into a degraded mode. A high water mark for the data elements and list-element controls would also be very helpful.
- Display of general shared-queues information. The type of information required is the time when the last CQS system checkpoint was taken, the last structure checkpoint was taken, and where the overflow threshold is set.
- Display which destinations (transactions and output messages) are on the CQS cold queue and the CQS lock queue.
- Display whether or not there are queues on the overflow structure and, if so, the names of those queues.
- Display which queues a particular IMS has registered interest in. This is obtained from the event monitor control block (EMC) entries in the message queue structure.

A tool such as IMS/ESA Message Requeuer Version 3 enables much of this information to be obtained. Additionally, it will retrieve messages from the cold queue and enable the installation to re-insert them into the shared queues.

Part 4. Testing the Shared-Queues Environment

In these chapters we describe the functional tests we ran in a shared-queue environment using the standard IMS IVP configuration.

These IVP tests were performed to validate the environment before starting all the procedures planned to bring this environment to the preproduction and production environments at Telstra.

We have investigated different behaviors from either a CQS or an IMS point of view in attempting to reproduce normal and abnormal IMS and CQS operation.

These basic tests have been run on the shared-queue system programmer environment put in place at Telstra. We did not attempt to develop a complete scenario to be able to reproduce all the different customer environments. Because of the customer′s normal configuration, we didn′t run any tests with APPC, OTMA, or Fast Path transactions. Nor did we test some types of coupling facility failures, since only one coupling facility was available and other system programmers were using that facility during our tests.

Clearly, there are different and more extensive test cases and benchmarks that any shop should run before moving to shared-queues. However, running the tests described in the following chapters is considered a very good starting point to prove the basic product functionality. These tests will be very helpful for any customer to better understand the product, and its related operations, along with CQS/IMS actions.

We used standard IVP transactions IVTNO, IVTNV, IVTCV, PART, and DSPALLI. During some of the tests, we changed IVTNV to a response-mode transaction and ran PART as a serial transaction. IVTNO was modified to run as a local transaction on one IMS and remote on the other to provide as many variations of test cases as possible.

We have divided the test descriptions into:

1. Chapter 7, Setup and Startup Tests
2. Chapter 8, IMS and CQS Normal and Abnormal End
3. Chapter 9, IMS Cold Start
4. Chapter 10, Changing Shared-Queue Structure Characteristics
5. Chapter 11, Miscellaneous Shared-Queue Tests

Figure 21 on page 56 and Figure 14 on page 32 shows the environment in which the tests were run.

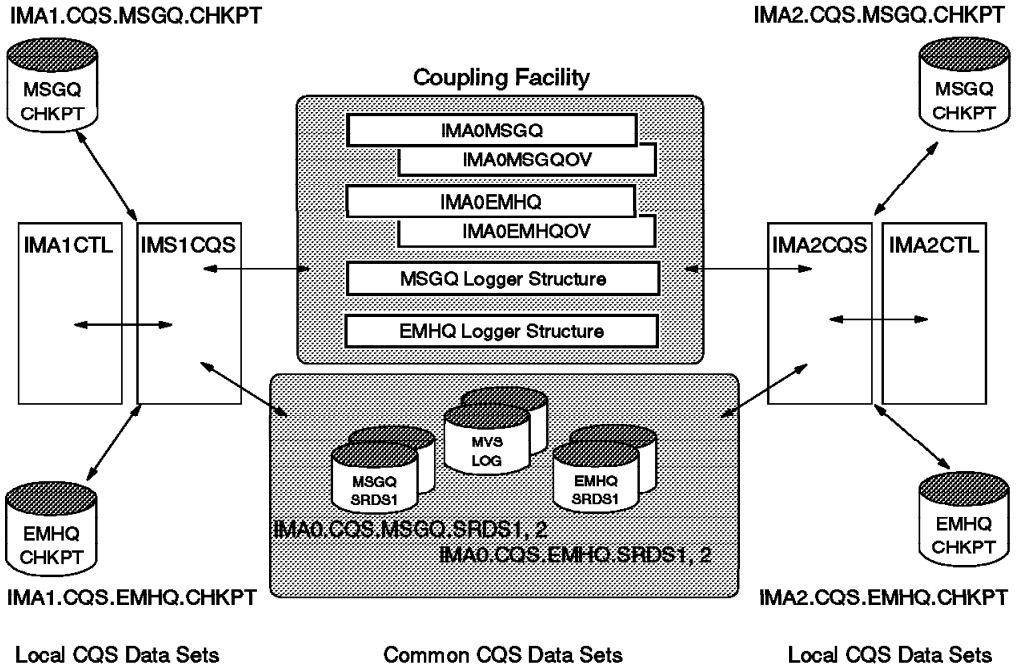


Figure 21. Environment for Shared Queue Tests at Telstra

Chapter 7. Setup and Startup Tests

In this chapter, we describe the setup of the shared queues environment, and the startup of an IMS in this environment. The minimum sizes of the shared-queues structures and the structure recovery data sets are also discussed.

7.1 Shared Queues Startup

The CQS address space is automatically started during IMS startup if it is not already active. If the CQS address space is required to be active before IMS is started, the MVS START command can be used. During IMS initialization, the following message is issued when IMS is waiting for CQS initialization:

```
DFSO226A CTL REGION WAITING FOR CQS ...
```

CQS always tries to warm start from the last system checkpoint that was saved in its checkpoint data set. During startup, the CQS checkpoint data set is interrogated to find a valid log token from which to restart. If a valid checkpoint data set is found (containing a valid log token) then a SYSTEM CQS warm start is performed. If the checkpoint data set is not valid (for example, it has been deleted and redefined) then the related coupling facility structure is analyzed to look for the restarting log token. If a log token is found in the structure, then the following message is issued asking to confirm the token (chosen from the coupling facility structure):

```
CQSO031A CONFIRM CQS RESTART FOR STRUCTURE strname...
```

You must either reply with a log token or restart Cold. If the coupling facility structure does not contain a log token or the log token eventually chosen from the structure is no longer available from the MVS log stream, then the following message is issued asking for a valid log token or Cold start:

```
CQSO032A ENTER CHECKPOINT LOGTOKEN ...
```

During startup, all the CQS log records following the chosen checkpoint token are processed in order to restore the CQS environment for committed data and back out uncommitted data.

Considering that the log records older than two structure checkpoints are deleted from the MVS log stream when a CQS checkpoint is taken, a CQS cold start is required if a CQS has been inactive since the penultimate checkpoint.

After a CQS cold start and resynchronization between IMS and CQS, IMS is responsible for deciding if inflight messages have to be moved to the CQS private cold queue.

During CQS startup, a connection to the defined structures is made. The first CQS that connects to the structures initializes them as well, along with the initialization of the structure recovery and the checkpoint data sets.

The messages shown in Figure 22 on page 58 are from the first CQS to start when the structures are allocated for the first time (no structure is allocated) or when a complete CQS cold start is performed (after deleting the checkpoint data set, SRDS, and coupling facility structure).

```
IEF403I IMA1CQS - STARTED - TIME=13.16.16
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ WAS SUCCESSFUL.
JOBNAME:IMA1CQS ASID:023F CONNECTOR NAME:CMA1CQS
CFNAME:VC1LA1
IXL015I STRUCTURE ALLOCATION INFORMATION FOR
STRUCTURE IMAOMSGQ, CONNECTOR NAME CMA1CQS
 CFNAME     ALLOCATION STATUS/FAILURE REASON
 --------   --------------------------------
 VC1LA1     STRUCTURE ALLOCATED
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQ WAS SUCCESSFUL.
JOBNAME:IMA1CQS ASID:023F CONNECTOR NAME:CMA1CQS
CFNAME:VC1LA1
IXL015I STRUCTURE ALLOCATION INFORMATION FOR
STRUCTURE IMAOEMHQ, CONNECTOR NAME CMA1CQS
 CFNAME     ALLOCATION STATUS/FAILURE REASON
 --------   --------------------------------
 VC1LA1     STRUCTURE ALLOCATED
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQOV WAS SUCCESSFUL.
JOBNAME:IMA1CQS ASID:023F CONNECTOR NAME:CMA1CQS
CFNAME:VC1LA1
IXL015I STRUCTURE ALLOCATION INFORMATION FOR
STRUCTURE IMAOMSGQOV, CONNECTOR NAME CMA1CQS
 CFNAME     ALLOCATION STATUS/FAILURE REASON
 --------   --------------------------------
 VC1LA1     STRUCTURE ALLOCATED
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQOV WAS SUCCESSFUL.
JOBNAME:IMA1CQS ASID:023F CONNECTOR NAME:CMA1CQS
CFNAME:VC1LA1
IXL015I STRUCTURE ALLOCATION INFORMATION FOR
STRUCTURE IMAOEMHQOV, CONNECTOR NAME CMA1CQS
 CFNAME     ALLOCATION STATUS/FAILURE REASON
 --------   --------------------------------
 VC1LA1     STRUCTURE ALLOCATED
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0020I CQS READY CMA1CQS
```

Figure 22. Messages Issued at Startup from the First CQS

The messages issued by the second CQS are shown in Figure 23.

```
IEF403I IMA2CQS - STARTED - TIME=13.27.49
 IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ WAS SUCCESSFUL.
 JOBNAME:IMA2CQS ASID:01B6 CONNECTOR NAME:CMA2CQS
 CFNAME:VC1LA1
 IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQ WAS SUCCESSFUL.
 JOBNAME:IMA2CQS ASID:01B6 CONNECTOR NAME:CMA2CQS
 CFNAME:VC1LA1
 IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQOV WAS SUCCESSFUL.
 JOBNAME:IMA2CQS ASID:01B6 CONNECTOR NAME:CMA2CQS
 CFNAME:VC1LA1
 IXL015I STRUCTURE ALLOCATION INFORMATION FOR
 STRUCTURE IMAOEMHQOV, CONNECTOR NAME CMA2CQS
  CFNAME     ALLOCATION STATUS/FAILURE REASON
  --------   --------------------------------
  VC1LA1     STRUCTURE ALLOCATED
```

Figure 23 (Part 1 of 2). Messages Issued at Startup from the Subsequent CQSs

```
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQOV WAS SUCCESSFUL.
 JOBNAME:IMA2CQS ASID:01B6 CONNECTOR NAME:CMA2CQS
 CFNAME:VC1LA1
IXL015I STRUCTURE ALLOCATION INFORMATION FOR
 STRUCTURE IMAOMSGQOV, CONNECTOR NAME CMA2CQS
  CFNAME     ALLOCATION STATUS/FAILURE REASON
  --------   --------------------------------
  VC1LA1     STRUCTURE ALLOCATED
*119 CQS0032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOEMHQ
*120 CQS0032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOMSGQ
```

Figure 23 (Part 2 of 2). Messages Issued at Startup from the Subsequent CQSs

It is required that the reply cold (for the very first CQS start) or a log token (obtained from the last CQS0030I) be given. A log token reply must be of the form

```
xx,logtoken
```

where the comma is required. The reply is required because CQS2 has found the structures that have been allocated by the first CQS to start (CQS1) but it has not been able to recognize a valid log token in the structure.

CQS will take a system checkpoint (for both full-function and Fast Path structures) after the completion of the startup phase (see **a** / **b** in Figure 24) and a system checkpoint as soon as IMS and CQS synchronization takes place (IMS will be waiting for CQS to startup), as shown by **c** / **d** In Figure 24 the resulting messages are shown:

```
R 119,COLD

 a   CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN 000000

R 120,COLD

 b   CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN 000000
     CQS0020I CQS READY CMA2CQS
 c   CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN 000000
 d   CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
```

Figure 24. Messages Issued by CQS at Completion of Startup

The IXL014I messages indicate that CQS has successfully connected to both the primary and overflow structures, but after successful connection at initialization, CQS disconnects from the overflow structures. The MVS command

```
D XCF,STR,STRNAME=IMAOMSGQOV
```

shows the structure as STATUS: NOT ALLOCATED. The status is changed to ALLOCATED when the CQS overflow process begins.

7.2 Shared-Queue Structure — Minimum Size

Specifying the structure size is an activity that takes into account several different factors about the IMS environment such as transaction rate, message and queue buffer sizes, customer requirements, and so on. For more detail on how we worked out our structure sizes, please see Appendix D, “Sizing Coupling Facility Structures” on page 129.

The minimum MSGQ structure size that allows CQS to start is 768 KB, specified in the CFRM definitions. A sample structure is shown in Figure 25 on page 60.

```
D XCF STR,STRNAME,IMAOMSGQ

STRNAME:IMAOMSGQ
   STATUS:ALLOCATED
   POLICY SIZE    :768 K
   POLICY INITSIZE:768 K
   ACTUAL SIZE    :768 K
   STORAGE INCREMENT SIZE:256 K
   DISPOSITION    :KEEP
   ACCESS TIME    :NOLIMIT
   MAX CONNECTIONS:8
   # CONNECTIONS  :2
   CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
   ---------------- -- -------- -------- -------- ---- ------
   CMA1CQS          01 00010029 A01      IMA1CQS  0295 ACTIVE
   CMA2CQS          02 00020013 A02      IMA2CQS  015A ACTIVE

D XCF,STR,STRNAME=IMAOEMHO

STRNAME:IMAOEMHQ
   STATUS:ALLOCATED
   POLICY SIZE    :768 K
   POLICY INITSIZE:768 K
   ACTUAL SIZE    :768 K
   STORAGE INCREMENT SIZE:256 K
   DISPOSITION    :KEEP
   ACCESS TIME    :NOLIMIT
MAX CONNECTIONS:8
    # CONNECTIONS  :2
    CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
    ---------------- -- -------- -------- -------- ---- --------
    CMA1CQS          01 00010021 A01      IMA1CQS  0295 ACTIVE
    CMA2CQS          02 00020013 A02      IMA2CQS  015A ACTIVE
```

Figure 25. Message Queue Structures with a Minimum Size

Figure 26 shows the structure characteristics with the definitions above. It also shows the number of list entries and data elements allocated.

```
R 769,/CQQUERY STATISTICS STRUCTURE ALL

DFS000I  STRUCTURE NAME  LEALLOC  LEINUSE ELMALLOC ELMINUSE    LE/EL
DFS000I  IMAOMSGQ            126        3      127         3  0001/0001
DFS000I  IMAOMSGQOV           N/A      N/A      N/A       N/A     N/A
DFS000I  IMAOEMHQ             126        4      127         4  0001/0001
DFS000I  IMAOEMHQOV           N/A      N/A      N/A       N/A     N/A
DFS000I  *98211/123301*   IMA1
```

Figure 26. Display of Characteristics for CQS Structures

We tried to use SIZE=256K and SIZE=512K for full-function and Fast Path shared-queue structures, but the connection failed with the IXL messages and return codes shown in Figure 27 on page 61.

```
IXL013I  IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ FAILED
CQS0014E STRUCTURE IMAOMSGQ     INIT FAILURE; FUNC=CONN
JOBNAME:IMA1CQS ASID:024B CONNECTOR NAME:CMA1CQS
IXLCONN RETURN CODE:0000000C, REASON CODE:02010C08
```

Figure 27. CQS Messages Issued on Startup when the Structures are too Small

IXLCONN return and reason codes are detailed in *OS/390 MVS Programming: Sysplex Services Guide*, GC28-1771. After the IXL messages, CQS abended with the following message:

```
BPE0006I  U0014 - 00000200 INVALID STRUCTURE SIZE
```

The minimum we used for the logger structure was 512 KB; using a value less than this produced the following message:

```
IXG231I  IXGCONN REQUEST=CONNECT TO LOG STREAM SYSPLEX.IMS.MSGQ ...
RETURN CODE:00000008 REASON CODE:00000853
```

IXGCONN return and reason codes are explained in *OS/390 MVS Programming: Assembler Services Reference*, GC28-1910. After the IXG message, CQS abended with the following BPE message:

```
BPE0006I  U0014 - 000000A0
```

7.3 Minimum SRDS Size

CQS uses two structure recovery data sets (SRDS) for each structure pair (SRDS1 and SRDS2 for each message structure). They are used to save coupling facility structure contents when a structure checkpoint is taken. Structure checkpoint requests alternate between the two structure recovery data sets.

SRDSs are dynamically allocated when a structure checkpoint begins as well as during any structure recovery activity. They will be deallocated when the activities end; CQS therefore holds the SRDS only while a structure checkpoint or a structure recovery is in process.

For SRDS sizing, we tried to take a structure checkpoint with an SRDS that was not large enough to contain the whole coupling facility structure. Figure 28 shows the IDCAMS statements we used to make the full-function SRDS very small (one track).

```
  DEFINE CLUSTER                        -
       (NAME(IMAO.CQS.MSGQ.SRDS1)       -
       TRK(1)                           -
       NONINDEXED                       -
       SHAREOPTIONS(2,3)                -
       RECSZ(32761,32761)               -
       REUSE                            -
       CISZ(32768)                      -
       VOLUMES(SO0006))
```

Figure 28 (Part 1 of 2). Defining SRDS with IDCAMS

```
DEFINE CLUSTER                        -
     (NAME(IMA0.CQS.MSGQ.SRDS2)       -
     TRK(1)                           -
     NONINDEXED                       -
     SHAREOPTIONS(2,3)                -
     RECSZ(32761,32761)               -
     REUSE                            -
     CISZ(32768)                      -
     VOLUMES(S00006))
```

Figure 28 (Part 2 of 2). Defining SRDS with IDCAMS

Then we tried to initiate a checkpoint with the following command:

```
/CQCHKPT SHAREDQ STRUCTURE ima0messageq
```

The following message was received:

```
DFS1974I CQCHKPT SHAREDQ REQUEST REJECTED FOR STRUCTURE=IMAOMSGQ, REASON CODE = 5
```

This message means there was a CQS internal error. The messages from the CQS address space are shown in Figure 29.

```
CQSO220I CQS CMA2CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA2CQS
IEC070I 203-204,IMA2CQS,IMA2CQS,SRDSD101,1295,S00006,
IEC070I IMA0.CQS.MSGQ.SRDS1,IMA0.CQS.MSGQ.SRDS1.DATA,CAT.USER03.A00
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA2CQS
CQSO054E WRITE FAILED FOR SRDSDSN1, RC=00000008/2908001C CMA2CQS
CQSO054E DSN=IMA0.CQS.MSGQ.SRDS1                          CMA2CQS
CQSO222E CQS CMA2CQS  FAILED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
```

Figure 29. Messages Issued by CQS When a Structure is Not Large Enough

Message IEC070I is issued when there is no more space in the structure.

CQS and IMS both continued processing, but the structure checkpoint was not taken, thereby jeopardizing any later structure recovery if the need should arise.

Thus, the SRDS must be large enough to contain the primary structure and the overflow structure (if it has been defined) to avoid the exposure caused by failed structure checkpoints.

If the SRDS is redefined (delete/define) while CQS is active and no structure checkpoint is taken, then the next time that CQS restarts, the following message is received:

```
CQSO009W STRUCTURE IMAOMSGQ  REQUIRES STRUCTURE CHECKPOINT FOR RECOVERY
```

and a structure checkpoint must be taken so that structure recovery can be performed if the need arises.

Chapter 8. IMS and CQS Normal and Abnormal End

Shared Queues is a so-called *pulling* sharing architecture rather than the *pushing* architecture exhibited by MSC or CICSplex. When a new transaction arrives on an empty MSGQ, multiple IMSs that are capable of processing the message try to acquire the work (they have registered an interest in the transaction code).

IMS performs all the scheduling functions except passing control to the dependent region. The IMS that acquires the message will lock it to prevent the message from being retrieved by more than one IMS and then give control to the dependent region that issues the GU/GN sequence to the input/output program control block (IOPCB).

This means that only one IMS gets the message and the others end up with an unsuccessful scheduling process (pseudo scheduling). This technique may appear wasteful, but it is designed to work better during periods of high transaction rates. With shared queues, a transaction can be processed on any IMS in the sysplex that has declared interest in the transaction code and has the required resources available (MPR and PSB started). Exceptions are serial transactions and those generated by APPC and OTMA.

The transaction is processed directly on the front-end IMS if:

- The destination is a local scheduler message block or SMB (local transaction).
- A dependent region of the required serving class is waiting for work
- The input transaction fits into a single-queue buffer LRECL.
- The transaction is not defined as SERIAL.

This avoids several undesirable acts:

- Giving the message-queue structure access to:
  - Putting the message in the ready queue
  - Reading the message
  - Moving the message to the lock queue (it uses the copy in the IMS buffer).
- Notifying all the IMSs in the IMSplex of a queue going from empty to not empty
- Avoids pseudo-scheduling of other IMS regions
- Reduces the IMS logging caused by the scheduling process.

Such messages are placed on the lock queue instead of the transaction-ready queue.

The use of shared queues introduces some considerations regarding the transaction scheduling process:

- The LIMIT COUNT and LIMIT PRIORITY are ignored. Transactions are always scheduled on a NORMAL priority basis.

- PARLIM=0 is always set for any PARLIM specification. Any input transaction can cause a new region to be scheduled up to the value of MAXRGN.

Within the following scenarios, we analyzed how the messages flow through the IMS sysplex and the impact on them when IMS abends.

In this chapter, we describe the following tests:

- 8.1, Input IMS Abended and Messages Processed by Other IMS
- 8.2, CQS Address Space Cancelled
- 8.3, CQS Cold Start
- 8.4, Transaction Processing after a CQS Warm Start

8.1 Input IMS Abended and Messages Processed by Other IMS

In this section, we describe the behavior of IMS in the following circumstances:

- 8.1.1, IMS Abends with Nonconversational Transactions in the Shared Queue
- 8.1.2, IMS Abends with a Transaction Inflight
- 8.1.3, IMS Abends with Conversational Transactions in the Shared Queue
- 8.1.4, IMS Abends with a Response-Mode Transaction on the Shared Queue

8.1.1 IMS Abends with Nonconversational Transactions in the Shared Queue

In this section, we enter a transaction from each of IMA1 and IMA2, and then cold start IMA1 while both transactions are queued. A region is started on IMA2, and both transactions are processed. IMA1 and IMA2 have been generated using a cloned IMS system definition; that is, the same databases and terminals.

To better show how the messages flow within the IMS sysplex we describe the activity done on IMA1/CQS1 on the left side of the page and the activity done on IMA2/CQS2 on the right side.

IMA1, IMA1CQS	IMA2, IMA2CQS
No region is available to process transactions	No region is available to process transactions
Log on to a static terminal A1IP0005	Log on to a static terminal A1IP0006
/FORMAT IVTNO (a nonconversational transaction)	
Display last one	
	/FORMAT IVTNO
	Display last two
	Display of the global queue count shows two transactions on the full-function shared queue and all belong to the same class

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME  QCNT-TOTAL  QCNT-AGED  TSTMP-OLD    TSTMP-NEW
IVTNO              2          2  98216/105317 98216/105317
*98216/105928*
```

IMA1 was recycled:

/F IMA1CTL,STOP

/S IMA1CTL (/ERE,AUTO=Y)

An MPR for the serving class was started

/START REGION IMA2MPH1

We saw the output for the transaction entered from this terminal after hitting PA1 (display last two parameters)

Log off terminal A1IP0006

Log on to static terminal A1IP0005 (previously logged on IMA1)

We saw the output for the transaction entered from A1IP0005 (last one) when we entered PA1 (the transactions were processed on this IMS because of start of MPR)

/DISPLAY QCNT TRAN MSGAGE 0 shows 0 transactions on the shared-queue

```
QUEUENAME    QCNT-TOTAL  QCNT-AGED  TSTMP-OLD  TSTMP-NEW
*98216/111858*
```

This test showed that as soon as a message region for the queued transactions becomes available in any surviving IMS, that IMS will process the transactions that are on the shared-queue. The output for the transactions is queued to the terminal that entered the transaction. This output can be displayed by logging on to the *same* terminal on any surviving IMS.

8.1.2 IMS Abends with a Transaction Inflight

In this test, a transaction is entered in IMA1, and is being processed on IMA2 when IMA1 abends. The terminal logs on to IMA2, and the output is delivered to the terminal through IMA2.

IMA1 / IMA1CQS	IMA2 / IMA2CQS
Log on to static terminal A1IP0001	
No region is available to process transactions	No region is available to process transactions
/FORMAT IVTNO (a nonconversational transaction)	
tadd (sunny italy 11 22)	
	From system console pending reply, we entered:
	/DISPLAY QCNT TRAN MSGAGE 0

```
QUEUENAME  QCNT-TOTAL  QCNT-AGED  TSTMP-OLD    TSTMP-NEW
IVTNV      1           1          98217/102909 98217/102909
*98217/103103*
```

/START REGION IMA2MPH1

Transactions processed on this IMS and the WTOR is sent on this system. We didn't reply to the WTOR, leaving the transaction in flight 066 INSERT IS DONE, REPLY

/DISPLAY QCNT TRAN MSGAGE 0 showed no message on the shared-queue

```
QUEUENAME    QCNT-TOTAL  QCNT-AGED  TSTMP-OLD  TSTMP-NEW
*98217/103513*
```

IMA1 was cancelled using MVS modify command

```
F IMA1CTL,STOP
```

Log on to static terminal A1IP0001 previously logged on IMA1

/DISPLAY QCNT TRAN MSGAGE 0

/DISPLAY QCNT LTERM MSGAGE 0

Showed no message on the shared-queue

```
QUEUENAME   QCNT-TOTAL  QCNT-AGED TSTMP-OLD  TSTMP-NEW
*98217/104150*
```

We replied to WTOR 66 to complete the transaction:

After replying to WTOR the A1IP0001 lterm received the output for the IVTNO TADD transaction (sunny italy 11 22)

/DISPLAY QCNT TRAN MSGAGE 0

showed no transactions on the shared-queue

```
QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98216/111858*
```

We did an emergency restart of IMA1 and logged on to terminal A1IP0001 but received no output. It had already been received by the same terminal (A1IP0001), which had logged on to the other IMS while IMA1 was down.

This showed that the output is queued to the terminal A1IP0001 and will be displayed on the first IMS that logs on to LTERM A1IP0001.

8.1.3 IMS Abends with Conversational Transactions in the Shared Queue

In this test, we see what happens when a conversational transaction is interrupted during the conversation, and the terminal logs on to another IMS:

IMA1 / IMA1CQS	IMA2 / IMA2CQS
No region is available to process transactions	No region is available to process transactions
Log on to a static terminal A1IP0005	
/FORMAT IVTNO (a conversational transaction)	
Display last one	
A1IP0005 showed as conversation in progress	

From system console, pending reply, we entered:

/DISPLAY QCNT TRAN MSGAGE 0

It showed one transaction on the full-function shared-queue.

```
QUEUENAME    QCNT-TOTAL  QCNT-AGED  TSTMP-OLD    TSTMP-NEW
IVTNO            1           1      98216/123144 98216/123144
*98216/123514*
```

IMA1 was modified down using the MVS command

/F IMA1CTL,STOP

Log on to A1IP0005 (previously logged on IMA1)

/START REGION IMA2MPH1 on IMA2

We saw the output for the transaction entered from the other IMS after hitting PA1 (display last one) but the terminal was not in conversation mode.

/DISPLAY QCNT TRAN MSGAGE 0

It showed 0 transactions on the shared-queue.

```
QUEUENAME   QCNT-TOTAL  QCNT-AGED  TSTMP-OLD  TSTMP-NEW
*98216/124818*
```

IMA1 was restarted

/S IMA1CTL (/ERE,AUTO=Y)

Log on to A1IP0005

After any command, the terminal was displayed as CONVERSATION IN PROGRESS status and /EXIT has been required to clear that status.

8.1.4 IMS Abends with a Response-Mode Transaction on the Shared Queue

In these tests, transaction IVTNV was modified to be a response-mode transaction.

IMA1 / IMA1CQS

No region is available to process transactions

Log on to a static terminal A1IP0005

/FORMAT IVTNV (a response-mode transaction)

Display last one

Terminal in response mode

IMA2 / IMA2CQS

No region is available to process transactions

From system console pending reply, we entered:

/DISPLAY QCNT TRAN MSGAGE 0

It showed one transaction on the full-function shared-queue for IVTNV

```
QUEUENAME  QCNT-TOTAL  QCNT-AGED  TSTMP-OLD    TSTMP-NEW
IVTNO          1           1      98216/132011 98216/132011
*98216/132433*
```

IMS was abended via MVS command

```
/F IMA1CTL,STOP
```

/START REGION IMA2MPH1 on IMA2

IVTNV was processed on this IMS

/DISPLAY QCNT LTERM A1IP0005 MSGAGE 0

It shows one message for the terminal

```
QUEUENAME    QCNT-TOTAL  QCNT-AGED  TSTMP-OLD     TSTMP-NEW
A1IP0005         1           1      98216/134012  98216/134012
*98216/134223*
```

Log on to A1IP0005

We saw the output for the transaction entered from the other IMS after hitting PA1 (display last one parameter)

/DISPLAY QCNT TRAN MSGAGE 0

It showed no transactions on the shared-queue.

/DISPLAY QCNT LTERM A1IP0005 MSGAGE 0

It showed no message queued for the terminal.

/S IMA1CTL

Log on to A1IP0005

A1P0005 was able to do normal operations

From the above scenarios, we saw that no affinity is held within the IMS sysplex environment: we can log on to another IMS even if a significant status exists. It is possible to see the output for the same static terminal name if logon is performed on another IMS. The difference seen in Scenario 8.1.3, “IMS Abends with Conversational Transactions in the Shared Queue” on page 66 is that a terminal that entered a conversational transaction keeps its status (the related CCB stays on the entering subsystem) across an IMS restart.

It is a user responsibility to take into account the affinity and associated significant status. VTAM generic resources could help in solving this problem because the program will consider affinity during the logon process.

8.2 CQS Address Space Cancelled

In this section, we describe the following tests:

- 8.2.1, CQS Cancelled with Transactions on the Shared Queue
- 8.2.2, CQS Cancelled with Transactions in SQ and Checkpoint Data Sets Deleted
- 8.2.3, CQS Abend with Transactions on SQ, Checkpoint Data Sets and SRDS Deleted

8.2.1 CQS Cancelled with Transactions on the Shared Queue

CQS is a general purpose queueing facility that handles shared-queues in a sysplex environment on behalf of a client. IMS is the client in the shared-queues environment. CQS runs in its own address space and is started automatically during IMS startup. Each CQS serves a single IMS.

CQS maintains the shared queues in the coupling facility through the use of list structures. Up to 32 CQSs can be connected to each of the structures.

In the following tests we tried to prove CQS and CQS/IMS functionality. We did some test cases related to abending CQS and then restarting it, either warm or cold, together with IMS emergency restart. The tests we ran used only full-function messages and therefore relate to full-function structure activity, since Telstra does not use EMH.

Three transactions were entered from IMA1 and two from IMA2 when no serving region was active. The following results were seen:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME      QCNT-TOTAL   QCNT-AGED    TSTMP-OLD       TSTMP-NEW
IVTCV               1            1        98218/080122   98218/080122
IVTNO               1            1        98218/080132   98218/080132
IVTNV               1            1        98218/080158   98218/080158
PART                2            2        98218/080235   98218/080243
*98218/080315*
```

Both CQSs address spaces were cancelled using the MVS cancel command:

```
C IMA1CQS
C IMA2CQS
```

and the CQSs issued the following messages:

```
CQSO105I INTF CLEANUP SUCCESSFUL:CLIENT=IMA1
CQSO105I INTF CLEANUP SUCCESSFUL:CLIENT=IMA2
```

After the CQSs went down, it was possible to enter IMS commands such as

- /DISPLAY TRANSACTION xxx
- /DISPLAY A
- /DISPLAY LTERM xxx

and receive the related output. It was not possible to process a shared-queue command, as the following message and response show:

```
/DISPLAY QCNT TRAN MSGAGE 0

  DFS1975 08:34:19 COMMAND REJECTED AS CQS IS NOT AVAILABLE
```

We tried to enter new transactions, and received the following message for each transaction:

```
DFSO70 08:36:44 UNABLE TO ROUTE MESSAGE
```

At this time only one CQS, CQS2, was restarted. The messages issued are shown below:

```
IEF403I IMA2CQS - STARTED - TIME=11.07.34
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ WAS SUCCESSFUL.
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQ WAS SUCCESSFUL.
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQOV WAS SUCCESSFUL.
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQOV WAS SUCCESSFUL.
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0020I CQS READY CMA2CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN ...
```

During the CQS restart phase, IMA2 resynchronized with CQS2:

```
DFS0226A CTL REGION WAITING FOR CQS (IMA2CQS ), RESPONSE TO CONNECT REQUEST -
```

The messages were still on the shared-queue, as shown below:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTCV                   1              1        98218/080122 98218/080122
IVTNO                   1              1        98218/080132 98218/080132
IVTNV                   1              1        98218/080158 98218/080158
PART                    2              2        98218/080235 98218/080243
*98218/080606*
```

An MPR was started on IMA2CTL:

```
/START REGION IMA2MPH1
```

All the queued transactions were processed; in a shared-queues environment any eligible IMS with an available serving region can process the transactions. The related output was delivered, but only to the terminals connected to IMA2/CQS2; the terminals connected to IMA1 must wait for CQS1 to come up before they can receive output.

A display on the shared-queue showed output in progress for the terminal connected to IMA1:

```
/DISPLAY QCNT LTERM MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
A1IP0001                1              1        98218/082626 98218/082626
A1IP0002                1              1        98218/082652 98218/082652
A1IP0003                1              1        98218/082713 98218/082713
*98218/083312*
```

CQS1 was restarted, as shown below:

```
/S IMA1CQS

IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQ WAS SUCCESSFUL.
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ WAS SUCCESSFUL.
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOEMHQOV WAS SUCCESSFUL.
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQOV WAS SUCCESSFUL.
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ    , LOGTOKEN ...
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ    , LOGTOKEN ...
CQSO020I CQS READY CMA1CQS
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ    , LOGTOKEN ...
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ    , LOGTOKEN ...
```

During the CQS restart phase, IMA1 resynchronized with CQS1:

```
DFSO226A CTL REGION WAITING FOR CQS (IMA1CQS ), RESPONSE TO CONNECT REQUEST -
```

The output for the transactions entered from IMA1 (ima1ctl/ima1CQS) were able to reach their final destinations. The results of these commands:

/DISPLAY QCNT TRAN MSGAGE 0
/DISPLAY QCNT LTERM MSGAGE 0

both showed

```
  QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
  *98218/084131*
```

8.2.2 CQS Cancelled with Transactions in SQ and Checkpoint Data Sets Deleted

We investigated what happens when we have transactions on the shared queue and the CQS address space is restarted after deleting the CQS checkpoint data set.

Some transactions (response-mode, non-response-mode, conversational, and MSC) were entered on each IMS. The queues contained the transactions shown below:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTCV                   1              1       98218/085911 98218/085911
IVTNO                   1              1       98218/085931 98218/085931
IVTNV                   1              1       98218/085945 98218/085945
PART                    2              2       98218/090101 98218/090112
*98218/090205*
```

Both CQSs were abended with the MVS cancel command:

```
/C IMA1CQS

CQSO105I INTF CLEANUP SUCCESSFUL:CLIENT=IMA1
CQSO105I INTF CLEANUP SUCCESSFUL:CLIENT=IMA2
```

After the CQSs went down, new logons were rejected but it was possible to enter IMS commands such as

- /DISPLAY TRANSACTION xx

- /DISPLAY A
- /DISPLAY LTERM xxx

Shared queue commands were not available:

```
/DISPLAY QCNT TRAN MSGAGE 0

DFS1975 09:04:19 COMMAND REJECTED AS CQS IS NOT AVAILABLE
```

Trying to enter new transactions we got:

```
DFS070 09:06:44 UNABLE TO ROUTE MESSAGE
```

All the checkpoint data sets were deleted for the CQS address spaces. During IMA1CQS and IMA2CQS restart, message CQS0031A was received and we confirmed the use of the log token found by CQS in the shared-queue structures. Confirmation was indicated by these messages:

```
/S IMA1CQS

*133 CQS0031A CONFIRM CQS RESTART FOR STRUCTURE IMAOMSGQ    CMA1CQS
*134 CQS0031A CONFIRM CQS RESTART FOR STRUCTURE IMAOEMHQ    CMA1CQS
R 133,CONFIRM
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ
R 134,CONFIRM
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
CQS0020I CQS READY CMA1CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
```

IMS and CQS resynchronized through DFS0226A.

After IMA1CQS was started, it was possible to inquire on the shared-queue from IMA1, as follows:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD      TSTMP-NEW
IVTCV                   1              1         98218/085911  98218/085911
IVTNO                   1              1         98218/085931  98218/085931
IVTNV                   1              1         98218/085945  98218/085945
PART                    2              2         98218/090101  98218/090112
*98218/091205*
```

The same procedure was used to restart CQS2:

```
/S IMA2CQS

*138 CQSO031A CONFIRM CQS RESTART FOR STRUCTURE IMAOMSGQ CMA2CQS
*139 CQSO031A CONFIRM CQS RESTART FOR STRUCTURE IMAOEMHQ CMA2CQS
R 138,CONFIRM
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ
R 139,CONFIRM
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
CQSO020I CQS READY CMA2CQS
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ
```

IMS and CQS resynchronized through DFS0226A.

An MPR was started on IMA2 and it processed all the messages on the shared-queue. Transaction IVTNO was the MSC transaction defined to process as local on IMA2. In fact the output to the previously entered transactions reached the related terminals and no more replies were present on the shared-queue. These commands

```
/DISPLAY QCNT TRAN MSGAGE 0
/DISPLAY QCNT LTERM MSGAGE 0
```

both showed there were no transactions or output messages on the shared-queues.

```
QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98218/092451*
```

8.2.3 CQS Abend with Transactions on SQ, Checkpoint Data Sets and SRDS Deleted

During this test, we followed the same path as for 8.2.2, “CQS Cancelled with Transactions in SQ and Checkpoint Data Sets Deleted” on page 71 with the difference that we deleted all the checkpoint data sets and the SRDSs. We got exactly the same results. Deleting the SRDSs does not impact normal operation unless it is required to use them during recovery.

8.3 CQS Cold Start

A series of tests was performed after cold-starting CQS:

- 8.3.1, CQS Cold Start with Transactions in Flight (IMS Running)
- 8.3.2, CQS Cold Start with IMS Emergency Restart and Transactions In-flight
- 8.3.3, Test Begun after Replying to WTOR (Output in Flight)

8.3.1 CQS Cold Start with Transactions in Flight (IMS Running)

A transaction was entered from IMA1CTL/IMA1CQS without any MPR in the IMS sysplex. The transaction was:

```
/FORMAT IVTNV
TADD (valerio oldest 11 22)
```

This was done with the intention of having a WTOR pending and, therefore, the transaction waiting in the queue before a commit:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD     TSTMP-NEW
IVTNV                   1              1         98218/125526 98218/125526
*98218/125620*
```

Following that, IMA1CQS was cancelled with the C IMA1CQS command.

The following command was entered on IMA1CTL, and produced the message shown:

```
/CQCHKPT

DFS1973I CQCHKPT SHAREDQ REQUEST REJECTED FOR REASON CODE=   2
```

Code 2 means no CQS address space is available.

From IMA2CTL, more than two structure checkpoints were taken, causing the CQS log records older than the last two structure checkpoints to be deleted.

The following sequence of messages was received for each structure checkpoint:

- From the IMS side:

```
/CQCHKPT SHAREDQ STRUCTURE ALL

DFS058I 14:14:37 CQCHKPT COMMAND IN PROGRESS   IMA2
DFS1972I CQCHKPT SHAREDQ COMMAND COMPLETE FOR STRUCTURE=IMAOMSGQ
DFS1972I CQCHKPT SHAREDQ COMMAND COMPLETE FOR STRUCTURE=IMAOEMHQ
```

- From the CQS side:

```
CQS0220I CQS CMA2CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOEMHQ
CQS0220I CQS CMA2CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQS0200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA2CQS
CQS0200I STRUCTURE IMAOEMHQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA2CQS
CQS0201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA2CQS
CQS0201I STRUCTURE IMAOEMHQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA2CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOEMHQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQS0221I CQS CMA2CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOEMHQ
CQS0221I CQS CMA2CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
```

IMA1CQS was then restarted and a reply cold was given to CQS0032A. The cold reply was returned because the CQS log records were deleted from the MVS log stream by the two structure checkpoints.

Transaction IVTNV was still in the shared-queue:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD       TSTMP-NEW
IVTNV                   1              1         98218/125526   98218/125526
*98218/125620*
```

At this time an MPR (IMA2MPH1) was started on IMA2. The WTOR was issued and IVTNV transaction was left in flight:

```
601 INSERT IS DONE, REPLY
```

IMA2CQS was brought down and two structure checkpoints were taken from IMA1

```
/CQCHKPT SHAREDQ STRUCTURE IMAOMSGQ
```

IMA2CQS was restarted cold and the same sequence of messages as IMA1CQS was received. The reply to 601 WTOR was given and we saw the transaction IVTNV complete successfully. The response sent to the input terminal was

```
valerio oldest 11 22
```

The test was repeated doing all the activity previously described but on only one member of the IMS sysplex: IMA1CTL and IMA1CQS. The steps were these:

1. Message entered from IMA1 - IVTNV on the shared-queue.
2. IMA1CQS cancelled.
3. Two structure checkpoints taken from IMA2.
4. IMA1CQS restarted cold; transaction still in the shared-queue.
5. Started an MPR on IMA1 and WTOR received: transaction in flight.
6. IMA1CQS cancelled.
7. Again, two structure checkpoints taken from IMA2.
8. IMA1CQS restarted cold.
9. Reply to WTOR.

The same results were obtained with the same sequence of messages: transaction IVTNV completed successfully and the output was received by input terminal.

The same test was performed using an MSC transaction (IVTNO), and even in this case the transaction completed successfully after running in the expected local system.

8.3.2 CQS Cold Start with IMS Emergency Restart and Transactions In-flight

The same sequences as in 8.3.1, “CQS Cold Start with Transactions in Flight (IMS Running)” on page 73 was repeated but with an emergency restart of IMS. We had transactions entered from IMA1:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNV                   1              1        98219/113107 98219/113107
*98219/113157*
```

We started an MPR on IMA1 and IVTNV was left in flight without responding to the WTOR.

At this time, we cancelled CQS1 and two structure checkpoints were forced by IMA2 through

```
/CQCHKPT SHAREDQ STRUCTURE ALL.
```

After that, we modified down IMA1

```
F IMA1CTL,STOP
```

We lost the WTOR and then did an emergency restart of IMA1. CQS1 was restarted by the IMA1 startup procedure and gave us these messages:

```
*029 CQSO032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOMSGQ
*030 CQSO032A ENTER CHECKPOINT LOGTOKEN FOR CQS RESTART FOR STRUCTURE IMAOEMHQ
 R 29,COLD
 R 30,COLD
```

At the completion of the restart phase for either CQS1 or for IMA1, the IVTNV was held in the shared-queue structure:

```
/DISPLAY QCNT TRAN MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNV                   1              1        98219/113107 98219/113107
*98219/115234*
```

An MPR was started, again on IMA1. A reply was given to the new WTOR and the transaction response was sent back to the original input terminal.

The test was repeated using an MSC transaction, which gave the same result.

8.3.3 Test Begun after Replying to WTOR (Output in Flight)

We entered a response mode transaction IVTNV (with the data “TDAA qaqa wsws 11 22”) from IMA1, where we started an MPR and left the WTOR pending. Then, we carried out these steps:

1. We cancelled CQS1.
2. Two structure checkpoints were processed on IMA2 with the following command:

   ```
   /CQCHKPT SHAREDQ STRUCTURE IMAOMSGQ
   ```
3. We replied to WTOR (while the input terminal was still in response mode)
4. CQS1 was restarted cold, as two structure checkpoints had been taken from IMA1.
5. The input terminal remained in response mode.

We saw the IVTNV transaction completed (the display of “qaqa wsws 11 22” was successful from the other IMS) but the related output never reached the final destination until an IMS checkpoint (/CHECKPOINT command) was taken on IMA1. We had to /RSTART NODE to clear the response mode status.

8.4 Transaction Processing after a CQS Warm Start

We entered a response mode transaction IVTNV (TADD have fun 11 22) from IMA1, where we started an MPR and left the WTOR pending:

```
INSERT IS DONE, REPLY
```

CQS1 was then cancelled

We replied to the WTOR (input terminal still in response mode) and the transaction completed. We were able to display the just-added entry (“have fun 11 22″”) from the other IMS (IMA2).

CQS1 was restarted (warm start)

The input terminal received the output (entry added) and went out of the response mode.

Chapter 9. IMS Cold Start

In this chapter, we describe a series of tests to see what happens when an IMS subsystem is cold-started, and how the related messages are handled. We tried to see what happens when there is an inflight transaction and IMSs are cold started. We tried four different scenarios:

- 9.1, Processing Messages Queued across an IMS Cold Start
- 9.2, Cold Start during Processing of a Transaction
- 9.3, Cold Start during Processing of a Response-Mode Transaction
- 9.4, Cold Start during Processing of a Conversational Transaction

9.1 Processing Messages Queued across an IMS Cold Start

In this test, we queued messages from IMA1 and IMA2 onto the shared queue. We then cold-started each of the IMSs, and processed the messages stored on the shared queue. The subsequent steps were these:

1. One PART transaction was entered from IMA1, and an IVTNO transaction was entered from IMA2. IVTNO is an MSC transaction, defined as remote on IMA2 and local on IMA1. The following display shows two transactions queued:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD  TSTMP-NEW
IVTNO                   1              1    98212/152306   98212/152331
PART                    1              1    98212/152417   98212/152447
*98212/152515*
```

2. IMA1 and IMA2 were shut down normally and then cold started with the command /CHECKPOINT NRESTART CHKPT 0.
3. The queued transactions were still on the shared-queue, as shown in the following display:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                   1              1        98212/152306 98212/152331
PART                    1              1        98212/152417 98212/152447
*98212/152515*
```

4. An MPR serving the these transaction classes was started on IMA1 and all the transactions were correctly processed. The IMS cold start had no effect on any of the messages queued across the cold start, as shown:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98212/153143*
```

9.2 Cold Start during Processing of a Transaction

In this test, a transaction was entered on IMA1, was processed on IMA2, and IMA1 was cold started. Activity performed on IMA1 is shown on the left and the activity on IMA2 is shown on the right side of the page.

IMA1	IMA2
Log on to static terminal A1IP0008.	Log on to static terminal A1IP0009.
No region is available to process transactions.	No region is available to process transactions.
/FORMAT IVTNO (a response-mode transaction)	
TADD luigi luigi 1234 5678	
	/DISPLAY QCNT TRANSACTION MSGAGE 0
	This shows one message on the shared-queue for IVTNV.

```
QUEUENAME    QCNT-TOTAL    QCNT-AGED  TSTMP-OLD     TSTMP-NEW
IVTNV            1             1      98216/161312 98216/161312
*98216/161323*
```

	/START REGION IMA2MPH1
	As a result of running the IVTNV transaction on IMA2, we got the WTOR on this system:
	772 INSERT IS DONE, REPLY

IMA1 was modified down via MVS command:

```
F  IMA1CTL,STOP
```

	WTOR 772 was still present.

IMA1 was then cold started:

```
S IMA1CTL,AUTO=N
/ERESTART COLDSYS.
```

	WTOR 772 still present
	We replied to WTOR 772, and IVTNV completed.
	/DISPLAY QCNT LTERM MSGAGE 0
	showed one message for input terminal A1IP0008

```
QUEUENAME       QCNT-TOTAL      QCNT-AGED     TSTMP-OLD     TSTMP-NEW
A1IP0008            1               1         98216/165506 98216/165506
*98216/165536*
```

Logging on to A1IP0008, we received the response	
(luigi luigi 1234 5678)	

9.3 Cold Start during Processing of a Response-Mode Transaction

In this test, we entered a response-mode transaction on IMA1, processed this transaction on IMA2, and cold-started IMA2 while the transaction was in flight.

IMA1

Log on to static terminal A1IP0009.

No region is available to process transactions.

```
/FORMAT IVTNO (a response-mode transaction)
TADD  cetor cetor 1234 5678
```

IMA2

Log on to static terminal A1IP0010.

No region is available to process transactions.

```
/DISPLAY QCNT TRANSACTION MSGAGE 0
```

showed one message on the shared-queue for IVTNV.

```
QUEUENAME     QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNV             1              1        98216/171312 98216/171312
*98216/161723*
```

```
/START REGION IMA2MPH1
```

As a result of running the IVTNV transaction on IMA2, we got the WTOR:

@797 INSERT IS DONE, REPLY

On this system, the following displays showed there were no messages on the shared-queue.

```
/DISPLAY QCNT TRANSACTION MSGAGE 0
/DISPLAY QCNT LTERM  MSGAGE 0
```

produced

```
QUEUENAME      QCNT-TOTAL   TSTMP-OLD    TSTMP-NEW
*98216/161941*
```

IMA2 was brought down with the MVS command:

```
F IMA2CTL,STOP
```

As a result, the 797 WTOR was lost.

Terminal A1IP0009 is still in response mode.

IMA2 is cold started

```
S IMA1CTL,AUTO=N
/ERESTART COLDSYS.
```

Log on to terminal A1IP0010

```
/DISPLAY QCNT TRANSACTION MSGAGE 0
/DISPLAY QCNT LTERM  MSGAGE 0
```

showed no message on the shared-queue.

```
QUEUENAME      QCNT-TOTAL   QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98216/163320*
```

Terminal A1IP0009 is still in response mode. To clear this situation, the following command was entered:

```
/RSTART NODE A1IP0009
```

9.4 Cold Start during Processing of a Conversational Transaction

In this test, a conversational transaction was entered on IMA1, was processed by IMA2, and IMA2 was cold started while the transaction was in flight.

IMA1	IMA2
Log on to static terminal A1IP0001	Log on to static terminal A1IP0004
No region is available to process transactions.	No region is available to process transactions.

```
/FORMAT IVTCV (a conversational transaction)
TADD  gigi gigi 1111 2222
```

Terminal A1IP0001 was left in the middle of a conversation.

/DISPLAY QCNT TRANSACTION MSGAGE 0

showed one message on the shared-queue for IVTNV

```
QUEUENAME    QCNT-TOTAL   QCNT-AGED   TSTMP-OLD    TSTMP-NEW
IVTCV            1            1       98217/091913 98217/091913
*98217/092011*
```

/START REGION IMA2MPH1

As a result of running the IVTCV transaction on IMA2,
we got the WTOR on this system:

@038 INSERT IS DONE, REPLY

/DISPLAY QCNT TRANSACTION MSGAGE 0

and

/DISPLAY QCNT LTERM MSGAGE 0

showed no message on the shared-queue.

```
QUEUENAME      QCNT-TOTAL     QCNT-AGED   TSTMP-OLD  TSTMP-NEW
*98217/092800*
```

IMA2 was brought down with the MVS command F IMA2CTL,STOP

As a result, the 038 WTOR was lost.

Terminal A1IP0001 is still in conversation mode.

IMA2 is cold started with the commands

```
S IMA1CTL,AUTO=N
/ERESTART COLDSYS.
```

Log on to terminal A1IP0010.

/DISPLAY QCNT TRANSACTION MSGAGE 0

and

```
                                        /DISPLAY QCNT LTERM MSGAGE 0
                                        showed no message on the shared-queue
QUEUENAME       QCNT-TOTAL   QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98217/094126*
```

Terminal A1IP0001 is still in conversation mode. To clear this situation, the /EXIT command was entered.

9.5 Summary

As a result of the tests described in 9.3, “Cold Start during Processing of a Response-Mode Transaction” on page 81 and 9.4, “Cold Start during Processing of a Conversational Transaction” on page 82, we see that it is no longer possible to display transactions that were in flight on the cold-started IMS. These transactions have been moved to the CQS private cold queue.

The cold-started IMS will write a x′6740′ log record during the CQS and IMS resynchronization process, to flag the messages that have been moved to the CQS cold queue.

It is possible to see transactions on the CQS cold queue after taking a dump of the structure with the command:

```
Dump comm=(dump information)
nnSTRLIST=(STRNAME=messageqstrname,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER))
```

The resulting dump can be formatted using the command IPCS STRDATA ALLDATA, and analyzing all the List 1 entries. For further details on reading structure dumps, please see Appendix B, “Analyzing the Message Queue Structure” on page 115.

To requeue the transaction from the cold-start queue, you can use the IMS/ESA Message Requeuer V3 or an equivalent product.

For the terminals entering a response-mode or a conversational transaction, some action must be taken on the connected IMS to clear them. The related control blocks are on IMA1 and they retain their status until the terminal is reset.

Chapter 10. Changing Shared-Queue Structure Characteristics

In this chapter, we describe

- 10.1, Structure Resizing
- 10.2, Structure Overflow Tests
- 10.3, Structure Rebuild Tests

10.1 Structure Resizing

The size of a structure can be dynamically altered in the Parallel Sysplex environment. The CFRM policy is updated by the IXCMIAPU utility with the new SIZE parameter and then activated with the following command:

```
SETXCF START,POL,POLNAME=xxx,TYPE=CFRM
```

The structure is put in change-pending status to make the new SIZE parameter operational. While the structure is allocated, a rebuild of the structure itself has to be done:

```
SETXCF START,REBUILD,STRNAME=yyyyyyy
```

Structure activity is quiesced during some steps of the rebuilding process. This will affect any IMS in the sysplex that tries to access the structure.

If the CFRM policy includes an INITSIZE that differs from the SIZE parameter, the ALTER command can be used to increase the structure size:

```
SETXCF START,ALTER,STRNAME=yyyyyy,SIZE=zzzz
```

In the shared-queues environment, the first CQS that connects to a structure allocates the INITSIZE if there is enough space in the coupling facility.

The primary structure size can be dynamically changed by the CQS itself during CQS activity. CQS dynamically alters the size of the structure from INITSIZE to SIZE when the overflow threshold in CQSSGxxx (OVFLWMAX) is reached. The dynamic alteration by CQS is valid for the primary structure only.

In our environment, the structure sizes were:

IMA0MSGQ	IMA0MSGQOV
SIZE : 1536 KB	SIZE : 1536 KB
INITSIZE : 1024 KB	INITSIZE : 1024 KB
OVFLWMAX = 50%	

Transactions were submitted to IMS and queued to the structures without any serving MPR available.

When the primary structure IMA0MSGQ reached the overflow threshold, CQS increased the structure from INITSIZE to SIZE. OVFLWMAX=50% is the value we specified in CQSSGxxx; 70% is the default if no value is assigned.

Using the dynamic structure-size altering capability allows CQS to avoid the need for overflow processing.

The messages issued are these:

```
CQSO260I CQS CMA1CQS  STARTED OVERFLOW THRESHOLD PROCESSING FOR STRUCTURE IMAOMSGQ
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 1
CQSO265I STRUCTURE ALTER REQUEST STARTED FOR STRUCTURE IMAOMSGQ
CQSO266I STRUCTURE ALTER REQUEST COMPLETED FOR STRUCTURE IMAOMSGQ
CQSO264I CQS CMA1CQS  TERMINATED OVERFLOW THRESHOLD PROCESSING, ALTER SUCCESSFUL ...
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 1
```

The same set of messages was obtained by operator command to change the size without waiting to reach the overflow threshold (OVFLWMAX).

```
SETXCF START,ALTER,STRNAME=IMAOMSGQ,SIZE=1536
```

At this point, more transactions were entered to force messages into the overflow structure, and then we added even more to fill up the overflow structure as well. When this happened, the overflow structure did not increase from its INITSIZE to SIZE. Instead, it became full, and the following message was issued:

```
*CQSO205E STRUCTURE IMAOMSGQOV       IS FULL CMA2CQS
```

A display of the overflow structure is shown in Figure 30.

```
D XCF,STR,STRNAME=IMAOMSGQOV

IXC360I  16.50.06  DISPLAY XCF 727
STRNAME:IMAOMSGQOV
STATUS:ALLOCATED
     POLICY SIZE    :1536 K
     POLICY INITSIZE:1024 K
     REBUILD PERCENT:N/A
     PREFERENCE LIST:VC1LA1
     EXCLUSION LIST IS EMPTY

ACTIVE STRUCTURE
----------------
     ALLOCATION TIME:07/29/1998 14:05:26
     CFNAME         :VC1LA1
     COUPLING FACILITY:009674.IBM.02.000000041826
                       PARTITION:1   CPCID:00
     ACTUAL SIZE    :1024 K
STORAGE INCREMENT SIZE:256 K
     VERSION        :BOD2C982 9CFAD504
     XCF GRPNAME    :IXCLO018
     DISPOSITION    :KEEP
     ACCESS TIME    :NOLIMIT
     MAX CONNECTIONS:8
     # CONNECTIONS  :2
```

Figure 30. Display of a Structure That is Full

This works as documented; the manual states that the overflow structure does not dynamically alter from its INITSIZE. It can, however, be altered manually as shown in Figure 31 on page 87.

```
SETXCF START,ALTER,STRNAME=IMAOMSGQOV,SIZE=1536

IXC530I SETXCF START ALTER REQUEST FOR STRUCTURE IMAOMSGQOV ACCEPTED.
IXC533I SETXCF REQUEST TO ALTER STRUCTURE IMAOMSGQOV
COMPLETED.  TARGET ATTAINED.
CURRENT SIZE:      1536 K  TARGET:      1536 K
IXC534I SETXCF REQUEST TO ALTER STRUCTURE IMAOMSGQOV
COMPLETED.  TARGET ATTAINED.
CURRENT SIZE:              1536 K  TARGET:      1536 K
CURRENT ENTRY COUNT:       1235    TARGET:      1235
CURRENT ELEMENT COUNT:     1231    TARGET:      1231
CURRENT EMC COUNT:         3317    TARGET:      3317

D XCF,STR,STRNAME=IMAOMSGQOV

STRNAME:IMAOMSGQOV
 STATUS:ALLOCATED
POLICY SIZE    :1536 K
     POLICY INITSIZE:1024 K
     REBUILD PERCENT:N/A
     PREFERENCE LIST:VC1LA1
     EXCLUSION LIST IS EMPTY

ACTIVE STRUCTURE
----------------
     ALLOCATION TIME:07/29/1998 07:21:52
     CFNAME         :VC1LA1
     COUPLING FACILITY:009674.IBM.02.000000041826
                       PARTITION:1   CPCID:00
     ACTUAL SIZE    :1536 K
STORAGE INCREMENT SIZE:256 K
     VERSION        :B0D26F4E 99A36903
     DISPOSITION    :KEEP
     ACCESS TIME    :NOLIMIT
     MAX CONNECTIONS:8
 # CONNECTIONS  :0
```

Figure 31. Manually Altering the Size of a Structure

In this instance, the number of connections is zero because IMS and CQS were down when the command to alter the structure size was issued. The ALTER command can also be issued while IMS and CQS are up and running and the structure is allocated.

The customer chooses whether to use an INITSIZE equal to the SIZE parameter or not. The decision should take into account factors such as coupling facility storage availability and automation procedures when the ALTER process begins.

10.2 Structure Overflow Tests

The overflow process occurs whenever the primary structure reaches the value of OVFLWMAX, specified in CQSSGxxx (global parameters). The process consists of two phases: one to select the queue using the most data elements (all activity in the structure is quiesced), and one move the selected queues to the overflow structure. The selected queues remain quiesced until they are moved to overflow; other queues are reactivated.

A structure checkpoint is taken after any overflow occurs. The SRDS size should be large enough to ensure that the primary plus the overflow structure can be written.

The shared-queue structures are designed to avoid the condition of message-queue full seen in a non-shared-queue environment (U758 abend). When the overflow structure becomes full, or when it is not defined and the overflow threshold (OVFLWMAX in CQSSGxxx) is reached, CQS will reject new messages for the queues attempting to use overflow.

The overflow structure is not mandatory for a shared-queue environment. If it is not requested (OVFLWSTR not defined in the STRUCTURE keyword of CQSSGxxx), then it can be added only at a complete CQS cold start.

When the overflow processing is initiated, a maximum of 512 queue names (transaction codes and lterm names) could be selected in an attempt to move a minimum of 20% of the primary structure to the overflow structure. There could be cases where a queue name represents most of the primary structure occupancy: that should be considered in sizing the overflow structure since a queue cannot be split between the primary and overflow structure. When a queue is selected for a move, the whole queue is moved to the overflow structure.

With our deliberately chosen full-function structure size definition of 1536 KB, it was not difficult to reach the point where overflow processing was invoked and objects from the primary structure were moved to the overflow structure.

The CQS design intention is to avoid overflow processing as much as possible. That means making the primary structure large enough to handle normal operation. One consideration to bear in mind is that the overflow processing is triggered when the data element portion of the structure becomes full. Running short of list entries does not start the overflow process; instead it causes the structure to be considered full.

For more information regarding structure size, please see Appendix D, “Sizing Coupling Facility Structures” on page 129.

To test the use of the overflow structure, several IMS sample transactions (PARTS) were entered and queued to the structure. Since the SIZE and INITSIZE parameters on the CFRM policy were the same, CQS went directly to overflow processing without trying to alter the primary structure size dynamically. This is shown in Figure 32

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
PART                  481            481       98210/133421 98210/135138
*98210/135225*
```

Figure 32 (Part 1 of 2). Display of Structures using Overflow

```
/DISPLAY OVERFLOWQ STRUCTURE ALL

STRUC-RSCTYPE    OFLSTRUC-RSCNAME LUNAME-TMEMBER    TPNAME-TPIPE
    IMAOMSGQ         IS NOT IN OVERFLOW MODE
    IMAOEMHQ         IS NOT IN OVERFLOW MODE
    *98210/135246*

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE    LE/EL
     IMAOMSGQ       1235      539     1231       575  0001/0001
     IMAOMSGQOV      N/A      N/A      N/A       N/A     N/A
     IMAOEMHQ       1235        5     1231         4  0001/0001
     IMAOEMHQOV      N/A      N/A      N/A       N/A     N/A
*98210/135304*.
```

Figure 32 (Part 2 of 2). Display of Structures using Overflow

More PART transactions were entered until a display of the overflow structure shows:

```
/DISPLAY OVERFLOWQ STRUCTURE ALL

STRUC-RSCTYPE    OFLSTRUC-RSCNAME LUNAME-TMEMBER    TPNAME-TPIPE
     IMAOMSGQ         IMAOMSGQOV
     TRANSACTION      PART
     IMAOEMHQ         IS NOT IN OVERFLOW MODE
     *98210/140601*
```

The messages observed in the CQS address space when structure overflow processing occurred are shown in Figure 33:

```
CQSO260I CQS CMA2CQS  STARTED OVERFLOW THRESHOLD PROCESSING FOR STRUCTURE IMAOMSGQ
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 1 CMA1CQS
IXLO14I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQOV WAS SUCCESSFUL.
CQSO261I CQS CMA2CQS  COMPLETED OVERFLOW THRESHOLD PHASE 1 FOR STRUCTURE IMAOMSGQ
JOBNAME:IMA1CQS ASID:0222 CONNECTOR NAME:CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 1 CMA1CQS
CFNAME:VC1LA1
IXLO15I STRUCTURE ALLOCATION INFORMATION FOR
STRUCTURE IMAOMSGQOV, CONNECTOR NAME CMA1CQS
CFNAME     ALLOCATION STATUS/FAILURE REASON
--------   --------------------------------
VC1LA1     STRUCTURE ALLOCATED
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 2 CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 2 CMA1CQS
CQSO262I CQS CMA2CQS  COMPLETED OVERFLOW THRESHOLD PHASE 2 FOR STRUCTURE IMAOMSGQ
CQSO220I CQS CMA2CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA1CQS
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQSO221I CQS CMA2CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
```

Figure 33. Messages from CQS During Overflow Processing

As a consequence of triggering overflow processing, a structure checkpoint is taken and therefore a system checkpoint is taken.

Different IMS IVP transactions (IVTNO) were then entered to test a second overflow. The resulting sequence of messages is the same as before, as shown in Figure 34 on page 90.

```
CQSO260I CQS CMA2CQS  STARTED OVERFLOW THRESHOLD PROCESSING FOR STRUCTURE IMAOMSGQ
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 1 CMA1CQS
CQSO261I CQS CMA2CQS  COMPLETED OVERFLOW THRESHOLD PHASE 1 FOR STRUCTURE IMAOMSGQ
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 1 CMA1CQS
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR OVERFLOW THRESHOLD PHASE 2 CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER OVERFLOW THRESHOLD PHASE 2 CMA1CQS
CQSO262I CQS CMA2CQS  COMPLETED OVERFLOW THRESHOLD PHASE 2 FOR STRUCTURE IMAOMSGQ
CQSO220I CQS CMA2CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA1CQS
CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
CQSO221I CQS CMA2CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ

/DISPLAY OVERFLOWQ STRUCTURE ALL

STRUC-RSCTYPE    OFLSTRUC-RSCNAME LUNAME-TMEMBER    TPNAME-TPIPE
IMAOMSGQ         IMAOMSGQOV
TRANSACTION      IVTNO
TRANSACTION      PART
IMAOEMHQ         IS NOT IN OVERFLOW MODE
*98210/141907*
```

Figure 34. CQS Messages Issued during Overflow Processing

CQS keeps the selected queues in overflow mode until all the objects (transactions and output messages) for those queues are deleted from the overflow structure. CQS scans the overflow structure every 15 minutes to see if the structure is empty or not. These messages are issued to the CQS address space:

```
CQSO200I STRUCTURE IMAOMSGQ      QUIESCED FOR OVERFLOW SCAN START CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ      RESUMED AFTER OVERFLOW SCAN START CMA1CQS
CQSO200I STRUCTURE IMAOMSGQ      QUIESCED FOR OVERFLOW SCAN END CMA1CQS
CQSO201I STRUCTURE IMAOMSGQ      RESUMED AFTER OVERFLOW SCAN END CMA1CQS
```

In order for a queue name (transaction or lterm) to be taken out of overflow, that queue must be empty at the time the overflow scan runs. The checking occurs every 15 minutes until all the messages in the overflow structure are dequeued. Even if there are no messages in the overflow structure, CQS waits for the SCAN process to occur before allowing CQS to revert to using only the primary structure. If another message for one of the overflow structure queues arrives during the 15 minutes it is immediately put in the overflow structure, thereby forcing CQS to keep using overflow mode.

An MPR was started to process the PART transactions.

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO               519           519        98210/140730 98210/141424
PART                502           502        98210/133504 98210/140601
*98210/143057*
```

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                 519            519       98210/140730 98210/141424
*98210/143305*
```

When PART transactions were all processed, display of the overflow structure shows IVTNO as the only transaction type in the overflow structure.

```
/DISPLAY OVERFLOWQ STRUCTURE ALL

STRUC-RSCTYPE    OFLSTRUC-RSCNAME LUNAME-TMEMBER    TPNAME-TPIPE
IMAOMSGQ         IMAOMSGQOV
TRANSACTION      IVTNO
IMAOEMHQ         IS NOT IN OVERFLOW MODE
*98210/143704*
```

As part of this test, we also noticed that whenever we issued an IMS command, the use of list entries per data element in the structure increased. See Figure 35.

```
/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ            1235      539     1231      575  0001/0001
IMAOMSGQOV           N/A      N/A      N/A      N/A     N/A
IMAOEMHQ            1235        5     1231        4  0001/0001
IMAOEMHQOV           N/A      N/A      N/A      N/A     N/A
*98210/135304*.

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ            1971      540     1968      577  0001/0001
IMAOMSGQOV           N/A      N/A      N/A      N/A     N/A
IMAOEMHQ            1235        5     1231        4  0001/0001
IMAOEMHQOV           N/A      N/A      N/A      N/A     N/A
*98210/135643*
```

Figure 35. Increased Use of LE/LD with an IMS Command

This situation was investigated and it was found that copies of the command responses were destined for the secondary master but that the secondary master was assigned to a spool line that was down at the time of test. As a result, the command responses for secondary master, which could not be dequeued, were placed in the lock queue. This factor should be taken into consideration when a shared-queues environment is operating and messages for a destination cannot be delivered.

10.3 Structure Rebuild Tests

We performed some tests that required a structure rebuild:

- 10.3.1, Structure Rebuild after Structure Delete (no Connection)
- 10.3.2, Structure Rebuild after Structure Delete (Connection to the Structure)

10.3.1 Structure Rebuild after Structure Delete (no Connection)

The full-function and Fast-Path structures used in a shared-queues environment are persistent structures from a Parallel Sysplex viewpoint; they remain allocated even if all the CQSs disconnect and all CQSs and IMSs cold start. The structures are used to keep the data contents of IMS objects (messages), as well as some other application information.

The contents of the shared-queue structures must be consistent across normal and abnormal termination and across any hardware or software failure. The structures can be manually or automatically recovered through the following actions:

- If at least one CQS is running, the recovery can be initiated by the following command:

  ```
  SETXCF START,REBUILD,STRNAME=strname,LOCATION=NORMAL/OTHER
  ```

- CQS initiates a structure rebuild during start-up if it detects an empty structure and finds an SRDS containing a valid structure checkpoint
- MVS initiates a structure rebuild if the rebuild threshold for loss of connectivity is reached. This specified is by the REBUILD parameter in the CFRM policy.

When a structure recovery is required, CQS uses the SRDS as a recovery starting point (comparable to database image copy) and then reads all the log records since the structure checkpoint was taken. The time needed to recover the structure depends on the number of log records to be applied. Taking frequent structure checkpoints makes structure recovery faster; however, no CQS can access the structure during a structure checkpoint. A structure checkpoint and a valid SRDS must be available for a structure recovery to be successful.

A structure checkpoint is taken:

- When the MVS log becomes full or approaches full condition
- After successful structure rebuild
- After successful overflow process
- When the command /CQCHKPT SHAREDQ STRUCTURE xxx is issued
- During CQS normal termination if the command /CQSET SHUTDOWN SHAREDQ was entered.

We performed a test to see what happens during structure recovery. The test started when there were no connections to the structure. CQS and IMS were down before deleting the structure.

Both CQS and IMS were shut down after leaving twenty messages on the full-function shared queue, as shown in the following display:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME        QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO               20             20        98213/071253 98213/071316
*98213/071351*
```

The full-function shared-queue structure, IMA0MSGQ, was deleted to force a structure rebuild when CQS was started. The following command

```
/CHECKPOINT FREEZE
```

shut down both IMSs, which brought down CQSs as well:

```
SETXCF FORCE,STR,STRNAME=IMAOMSGQ
```

deleting the full-function shared-queue structure. Once IMA1 started, CQS1 rebuilt the full-function structure as shown in Figure 36.

```
   CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE REBUILD CMA1CQS
1  CQSO240I CQS CMA1CQS  STARTED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ
1  CQSO241I CQS CMA1CQS  COMPLETED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ
   CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE REBUILD CMA1CQS
2  CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
   CQSO020I CQS READY CMA1CQS
3  CQSO220I CQS CMA1CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
   CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA1CQS
   CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA1CQS
4  CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
   CQSO221I CQS CMA1CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMA1
5  CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ, LOGTOKEN ...
```

Figure 36. Messages Issued during a CQS Structure Rebuild

In the sequence of events shown in Figure 36, we saw:

1 A structure recovery started and completed.

2 A CQS system checkpoint was taken as a result of CQS start-up.

3 A structure checkpoint was issued after the completion of the structure recovery.

4 A CQS system checkpoint was taken for each IMS startup and each IMS/CQS synchronization.

5 A CQS system checkpoint was taken after the completion of the structure checkpoint.

At the completion of the structure recovery, the 20 IVTNO transactions were successfully recovered:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                20             20        98213/071253 98213/071316
*98213/072214*
```

If CQS2 is started before the structure rebuild by CQS1 is completed, the following messages are received:

```
IXL013I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ FAILED.
JOBNAME:IMA2CQS ASID:0159 CONNECTOR NAME:CMA2CQS
IXLCONN RETURN CODE:0000000C,  REASON CODE:02010C09
IXL014I IXLCONN REQUEST FOR STRUCTURE IMAOMSGQ WAS SUCCESSFUL.
IXLCONN RETURN CODE:0000000C,  REASON CODE:02010C09
```

Return code 0000000C with reason code xxxx0C09 indicates that a rebuild is in progress.

CQS2 fails in SRDS dynamic allocation after about 2 minutes because the SRDS is in use during structure recovery by CQS1, and abends as follows:

```
CQSO050E DYN ALLOC   FAILED FOR SRDSDSN1, RC=00000004/02100000 CMA2CQS
DUMPID=001 REQUESTED BY JOB (IMA2CQS )
CQSO050E DSN=IMAO.CQS.MSGQ.SRDS1                          CMA2CQS
DUMP TITLE=IMA2CQS  - CQS  STRD TCB ABEND U0014-00000360, MODULE=CQSIST30, THD=STRD
CQSO001E CQS INITIALIZATION ERROR IN CQSIST30, CQSDYNAO RC=0000000C SRDSDSN1
BPE0006I CQS  STRD TCB ABEND U0014-00000360, THD=STRD
```

There appears to be a 2 minute timer such that if CQS could not allocate the SRDSs, it will abend with U0014.

When the CQS abended, IMS abends with U0071 because it was unable to identify itself to the CQS.

```
*DFSO226A CTL REGION WAITING FOR CQS (IMA2CQS ), RESPONSE TO CONNECT REQUEST -
DFS1936E UNABLE TO IDENTIFY THE IMS CONTROL REGION TO CQS AS A CLIENT -  FUNCTION=CQSCONN ,
          RETURN CODE=X'00000010', REASON CODE=X'00000430' IMA2
DFS629I IMS CTL TCB ABEND - IMS 0071          IMA2
```

10.3.2 Structure Rebuild after Structure Delete (Connection to the Structure)

A second scenario was covered, forcing the structure delete starting from a failed-persistent state.

Seven messages were placed on the full-function shared-queue without any MPR available on IMA1 and IMA2:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                   7              7        98213/075151 98213/075211
*98213/075249*
```

Both CQSs were brought down with the MVS CANCEL command, and the related connections to the IMA0MSGQ structure showed as failed persistent:

```
D XCF,STR,STRNAME=IMAOMSGQ

CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
---------------- -- -------- -------- -------- ---- ----------------
CMA1CQS          01 00010011 A01      IMA1CQS  0283 FAILED-PERSISTENT
CMA2CQS          02 00020005 A02      IMA2CQS  0169 FAILED-PERSISTENT
```

The connections to IMA0MSGQ were deleted using the following command:

```
SETXCF FORCE,CONNECTION,STRNAME=IMAOMSGQ,CONNAME=ALL
```

The full-function shared-queue structure IMA0MSGQ was then deleted, using the command:

```
SETXCF FORCE,STR,STRNAME=IMAOMSGQ
```

After the full-function shared-queue structure was deleted, CQS1 was restarted. The messages shown in Figure 37 were issued to the CQS address space when the structure was rebuilt at CQS1 restart.

```
   CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE REBUILD CMA1CQS
 1 CQSO240I CQS CMA1CQS  STARTED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ
   CQSO241I CQS CMA1CQS  COMPLETED STRUCTURE RECOVERY FOR STRUCTURE IMAOMSGQ
   CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE REBUILD CMA1CQS
 2 CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ,  LOGTOKEN ...
   CQSO020I CQS READY CMA1CQS
 3 CQSO220I CQS CMA1CQS  STARTED STRUCTURE CHECKPOINT FOR STRUCTURE  IMAOMSGQ
   CQSO200I STRUCTURE IMAOMSGQ         QUIESCED FOR STRUCTURE CHECKPOINT CMA1CQS
   CQSO201I STRUCTURE IMAOMSGQ         RESUMED AFTER STRUCTURE CHECKPOINT CMA1CQS
 4 CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ,  LOGTOKEN ...
 5 CQSO030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMAOMSGQ,  LOGTOKEN ...
   CQSO221I CQS CMA1CQS  COMPLETED STRUCTURE CHECKPOINT FOR STRUCTURE IMAOMSGQ
```

Figure 37. Messages Issued by CQS During A Structure Rebuild

See page 93 for an explanation of 1 through 5.

The rebuild process was for the full-function shared-queue structure only and the normal process was related to EMH shared-queue structure. CQS2 was restarted and it went through the normal starting process because the rebuild was done by CQS1.

A display of the transactions queued still showed seven messages on the full-function shared queue, all of which were processed as soon as an MPR was started, regardless of which IMS the MPR was started on.

Chapter 11. Miscellaneous Shared-Queue Tests

These are a set of miscellaneous tests of interest to Telstra that also may be pertinent to your environment:

- 11.1, MSC Link Test
- 11.2, Serial Transaction Test
- 11.3, EMH Structure Testing
- 11.4, On the Full-Function Shared Queue Initial Program Load of an MVS with Messages
- 11.5, Fall Back to Local Queues

11.1 MSC Link Test

Replacement of the MSC traffic with the use of shared-queues is the main reason why Telstra wants to implement shared-queues.

The messages in a non-shared-queue environment whose destination is other IMSs through the MSC link are put in the shared-queue structure for all the IMSs belonging to the shared-queue group in the sysplex. They are available for processing, according to the MSC definition, by the targeted IMS. This means that if we keep the same MSC transaction definitions used in the non-shared-queue environment and move to shared queues, the transactions involved are still processed according to the MSC definitions only — that is, local transactions in the front-end IMS and remote transactions in the remote IMSs. The transactions are processed according to the SYSIDs.

The MSC links can still be defined between the systems in the IMS sysplex for back-up purposes but cannot be started in a shared-queues environment. MSC link communication takes place only if the IMSs are not in the same shared-queue group. Messages received from outside the shared-queue group are placed on the shared-queue so that they are available to be processed by any IMS in the group.

We ran the following tests:

- 11.1.1, MSC Transactions in the Shared-Queue Environment
- 11.1.2, MSC Transactions from a Shared-Queue Environment to a non-Shared-Queue Back End

11.1.1 MSC Transactions in the Shared-Queue Environment

The IVTNO transaction was modified to be remote in IMA2 and local in IMA1:

```
/DISPLAY TRANSACTION IVTNO  (from IMA2)

TRAN     CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM    RC
IVTNO    RMT     0     0 65535     2  1  1  1     0     0  NONE     0
PSBNAME:DFSIVP1
*98212/163028*

/DISPLAY TRANSACTION IVTNO  (from IMA1)

TRAN     CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM    RC
IVTNO     17     0     0 65535     2  1  1  1     0     0  NONE     0
PSBNAME:DFSIVP1
*98212/163035*
```

We entered the IVTNO transaction from IMA2 without any MPRs in IMA1 or IMA2 so that the transaction is queued to the MSGQ structure:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       1             1    98212/163023 98212/163023
*98212/163051*
```

We started up an MPR for IVTNO on IMA2, as follows:

```
/START REGION IMA2MPH1

DFS058I 16:30:59 START COMMAND IN PROGRESS

/DISPLAY REGION A

 REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    1 IMA2MPH1  TP                       WAITING           17
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
      DBTRGN    DBT   NONE
      IMA2DBR   DBRC
      IMA2DLI   DLS
*98212/163122*
```

Even though an MPR is available on IMA2 to run IVTNO, the transaction does not run because it was defined as remote. It remains queued on the MSGQ structure:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       1             1    98212/163023 98212/163023
*98212/163134*
```

Start up an MPR for IVTNO on IMA1, as shown below:

```
/START REGION IMA1MPH1

DFSO58I 16:32:12 START COMMAND IN PROGRESS

/DISPLAY REGION A

REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    1 IMA1MPH1  TPE   IVTNO     DFSIVP1  WAIT-SYNCPOINT    17
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
      DBTRGN    DBT   NONE
      IMA1DBR   DBRC
      IMA1DLI   DLS
      *98212/163217*
```

When the MPR was started on IMA1, where IVTNO is defined as a local transaction, the transaction was processed. The output from the transaction is queued to the lterm that entered the transaction on IMA2.

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98212/163231*
```

11.1.2 MSC Transactions from a Shared-Queue Environment to a non-Shared-Queue Back End

IVTNO was defined as remote on IMA1 and IMA2. An MSC link was defined from IMA1 and IMA2 to IMSH, which is outside the shared-queue environment. This is shown diagrammatically in Figure 38.

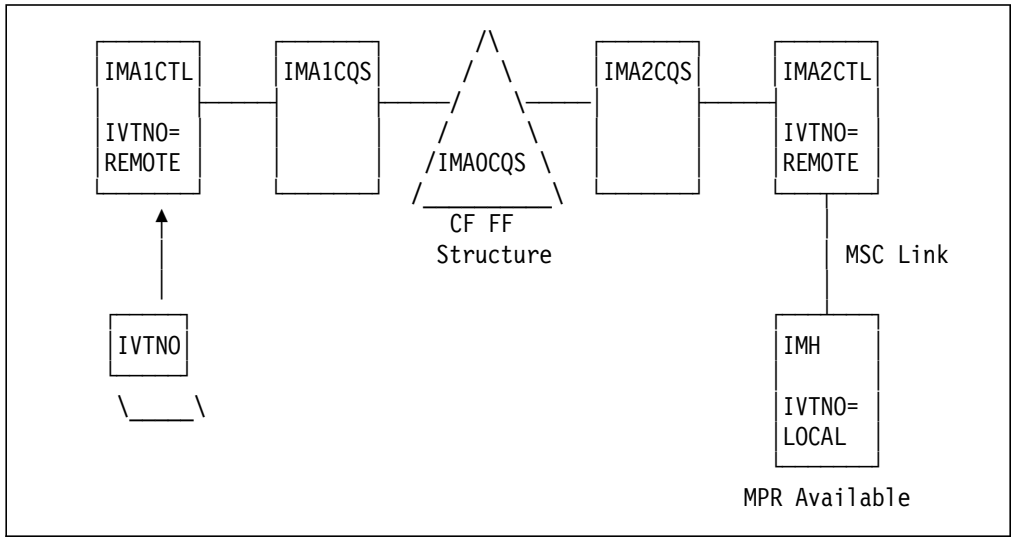


Figure 38. MSC Test Scenario 2

From IMA1 and IMA2, the display is as follows:

```
/DISPLAY TRANSACTION IVTNO

TRAN     CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM    RC
IVTNO    RMT     0     0 65535     2  1  1  1     0     0  NONE     0
PSBNAME:DFSIVP1
*98215/090318*

/DISPLAY ASMT MSPLINK ALL

LINK PLINK    TYPE  ADDR     MAXSESS  NODE
   1 IMSH     VTAM  00000000    255   IMSH
*98215/090324*
```

IVTNO was defined as local in IMSH and the MSC link was defined in IMSH to IMA2 only.

The links on IMSH are as follows:

```
/DISPLAY LINK ALL

LINK PARTNER   RECD  ENQCT  DEQCT    QCT   SENT
  1   SS       0     0      0        0     0      PSTOPPED IDLE COLD FORCE
  *98215/090150*

/DISPLAY ASMT MSPLINK ALL

LINK PLINK    TYPE  ADDR     MAXSESS  NODE
   1 IMA2     VTAM  00000000   255    IMA2
   *98215/090159*

/DISPLAY TRANSACTION IVTNO

TRAN     CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM    RC
IVTNO     17     0     0 65535     2  1  1  1     0     0  NONE     0
PSBNAME:DFSIVP1
*98215/090408*
```

There were no MPRs on any IMSs (IMA1,IMA2 nor IMSH) and the MSC link from IMA2 to IMSH was stopped. IVTNO was entered from IMA1. The transaction was queued in the MSGQ structure, as shown:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       1             1    98215/090547 98215/090547
*98215/090616*
```

The link between IMA2 and IMSH was started and the transaction was dequeued from the MSGQ structure to the IMSH local queue:

```
DFS2160I   9:08:10 LINK 001 STARTED BY PARTNER SS NODE IMA2    .
DFS2168I  9:08:10 CONNECTION ESTABLISHED ON LINK 0001     .

/DISPLAY TRANSACTION IVTNO

TRAN     CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM    RC
IVTNO     17     1     1 65535     2  1  1  1     0     0  NONE     0
PSBNAME:DFSIVP1
*98215/090831*
```

When an MPR was started on IMSH, the queued IVTNO was processed, and the reply was sent back to the initiating terminal on IMA1:

```
/START REGION IMSHMPH1

DFS058I 09:09:12 START COMMAND IN PROGRESS

/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98215/090607*
```

11.2 Serial Transaction Test

In a shared-queues environment, IMS registers interest in serial transactions using the queue name for the transaction code along with the IMSID of the IMS itself.

In our test, IVTNO was defined as serial, so registration is for IVTNO•••IMA1, where •=blank. This means that only the local IMS can process the serially defined transaction. Serial transactions use the CQS transaction serial queue for the first queue buffer and the transaction staging queue for any remaining queue buffers.

In a shared-queue environment, if a serial transaction abends, it is USTOPped and placed back in the transaction serial queue as the first message (to maintain the transaction sequence). To enable the USTOPped serial transaction to run again requires the transaction code to be started or the command /DEQUEUE SUSPEND to be issued. IMS then reregisters interest in the serial transaction.

The serial transactions are processed in first-in, first-out order, the same way they are processed in a non-shared-queues environment. If there is a need to process the transactions in the arrival order across all the IMSs (defined here as a pseudo-serial transaction) then specifying SERIAL=YES in all the IMSs is not sufficient; serialization occurs within only one IMS. Other solutions should be considered:

- Define the involved transaction on only one IMS; the transaction then can only be entered and processed on that IMS.
- Define the involved transaction as local on one IMS and remote on the others.
- Define the involved transaction on all the systems but assign it to a class that has only one region available across the IMSs.

The following are the tests we performed on serial transactions. IVTNO was initially defined as parallel and four transactions were queued to the shared-queue structure **a**.

Both IMSs were then shut down and the IVTNO transaction was modified via an IMS system generation to run as serial (SERIAL=YES).

The IMSs in the shared-queues environment were restarted without any MPRs.

Two IVTNO transactions were entered on each of the IMSs (IMA1 and IMA2).

The following is the result of this processing:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       4             4    98211/161524 98211/161542  a
IVTNO   IMA2                2             2    98212/105733 98212/105738  b
IVTNO   IMA1                2             2    98212/105630 98212/105719  c
*98212/105743*
```

As shown, there were four nonserial IVTNO transactions (queued on 98211/161524), two IVTNO entered on IMA1, **b** and two entered on IMA2 **c**.

An MPR was started on IMA2. The two IVTNO transactions entered on that IMS were successfully processed, as shown by the following output:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       4             4    98211/161524 98211/161542
IVTNO   IMA1                2             2    98212/105630 98212/105719
*98212/105725*
```

An MPR was then started on IMA1 and the following is the output:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                       4             4    98211/161524 98211/161542
*98212/110251*
```

It was not possible to process the four nonserial IVTNO transactions queued on 98211 between 161524 and 161542 **a**.

The command /STOP TRANSACTION IVTNO followed by the command /DEQUEUE TRANSACTION IVTNO PURGE did not remove the transactions from the queue. This is correct because after the system generation, IVTNO became serial and any activity for a serial transaction is performed against the CQS transaction serial queue and not on the transaction-ready queue.

The only way to process these transactions is to perform another IMS system generation and define the IVTNO transaction back to nonserial (SERIAL=NO). When this is implemented and an MPR is started on any IMS in this

shared-queues group, the four IVTNO (nonserial) transactions queued were processed and the command /DISPLAY QCNT TRANSACTION MSGAGE 0 showed no messages in the shared-queue.

The previous scenario is a description of a working-as-designed situation where we tried to change from nonserial to serial a transaction that had a global queue count greater then zero.

We reran the tests with only serial transactions on the shared-queue:

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO   IMA2               1             1    98212/110444 98212/110444
IVTNO   IMA1               1             1    98212/110458 98212/110458
*98212/111819*
```

These transactions were correctly forced to process on the IMS where the transactions were entered:

- IVTNO entered from IMA1 processed on IMA1
- IVTNO entered from IMA2 processed on IMA2

11.3 EMH Structure Testing

This test was cursory as Telstra does not use any EMH transactions, but the Fast-Path control macro (FPCTRL) is present in their generation deck for DEDB support. The test was conducted to show the use of EMH structure. The IMS EMH IVP transaction (IVTFD) was used for this test.

The test began with starting the IFP region in IMA1. Initial statistics are shown below.

```
/DISPLAY Q BALGRP

BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
*NO QUEUES*
*98212/081921*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98212/081930*
```

```
/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ              126       12      127       12 0001/0001
IMAOMSGQOV            N/A      N/A      N/A      N/A    N/A
IMAOEMHQ              126        4      127        4 0001/0001
IMAOEMHQOV            N/A      N/A      N/A      N/A    N/A
*98212/081939*
```

The transaction IVTFD TADD was entered, without any reply to the WTOR, as shown below:

```
/DISPLAY Q BALGRP

BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
DFSIVP4      1            0          1          1
*98212/082315*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98212/082348*

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ              126       12      127       12 0001/0001
IMAOMSGQOV            N/A      N/A      N/A      N/A    N/A
IMAOEMHQ              126        4      127        4 0001/0001
IMAOEMHQOV            N/A      N/A      N/A      N/A    N/A
*98212/082406*
```

Log on to IMA2 and enter one IVTFD display transaction. The displays are shown below:

```
/DISPLAY Q BALGRP

 BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
*NO QUEUES*
*98212/082601*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME     QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
DFSIVP4               1             1    98212/082510 98212/082510
*98212/082614*
```

```
/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ              126       12      127       12 0001/0001
IMAOMSGQOV            N/A      N/A      N/A      N/A    N/A
IMAOEMHQ              126        5      127        5 0001/0001
IMAOEMHQOV            N/A      N/A      N/A      N/A    N/A
*98212/082623*
```

The display showed that the IVTFD transaction entered from IMA2 is queued to the EMH structure because there is no local IFP region and the remote IFP region is not available.

A further five IVTFD transactions were entered from IMA1 where the IFP region is running but not available (it is still processing the initial TADD transaction.) The resulting display is shown below:

```
/DISPLAY Q BALGRP

BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
DFSIVP4        1          5          6          1
*98212/083049*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
DFSIVP4           1              1        98212/082510 98212/08     2510
*98212/083104*

/CQQUERY STATISTICS STRUCTURE ALL

  STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
  IMAOMSGQ              126       12      127       12 0001/0001
  IMAOMSGQOV            N/A      N/A      N/A      N/A    N/A
  IMAOEMHQ              126        5      127        5 0001/0001
  IMAOEMHQOV            N/A      N/A      N/A      N/A    N/A
  *98212/083119*
```

The five transactions entered from IMA1 do not show up in the structure display. This is because no more than five transactions are queued to this balancing group.

One more IVTFD transaction was then entered from IMA1.

The number of IFP regions servicing this balancing group divided by 4 (that is 1/4) is less than the number of transactions queued (6), so there are more transactions to be processed than can sensibly be processed locally. They are passed to CQS and put on the EMHQ structure, as shown below:

```
/DISPLAY Q BALGRP

BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
DFSIVP4      1            5          6          1
*98212/083242*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME     QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
DFSIVP4                2             2    98212/082510 98212/083230
*98212/083253*

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
     IMAOMSGQ         126       12      127       12 0001/0001
     IMAOMSGQOV       N/A      N/A      N/A      N/A    N/A
     IMAOEMHQ         126        6      127        6 0001/0001
     IMAOEMHQOV       N/A      N/A      N/A      N/A    N/A
     *98212/083303*
```

An IFP was then started on IMA2:

```
/START REGION IMA2FPI1
DFS058I 08:35:45 START COMMAND IN PROGRESS
```

All transactions in the structure were processed, as shown below:

```
/DISPLAY Q BALGRP

BALGRP    NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
DFSIVP4         1            0          3          3
*98212/083703*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
     *98212/083728*

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ             126       12      127       12 0001/0001
IMAOMSGQOV           N/A      N/A      N/A      N/A    N/A
IMAOEMHQ             126        4      127        4 0001/0001
IMAOEMHQOV           N/A      N/A      N/A      N/A    N/A
*98212/083738*
```

The five transactions entered from IMA1 on the local BALGRP were not processed because the IFP on IMA1 was still processing the IVTFD TADD transaction; it was waiting for the WTOR.

```
/DISPLAY Q BALGRP

BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
DFSIVP4        1            5          6          1
*98212/083622*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
*98212/083635*

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
IMAOMSGQ              126       12      127       12 0001/0001
IMAOMSGQOV            N/A      N/A      N/A      N/A    N/A
IMAOEMHQ              126        4      127        4 0001/0001
IMAOEMHQOV            N/A      N/A      N/A      N/A    N/A
*98212/083646*
```

Once replied to the TADD WTOR, all the transactions on the IMA1 local balancing group were processed:

```
/DISPLAY Q BALGRP

BALGRP   NO.RGNS       MSG CT  ENQ COUNT  DEQ COUNT
DFSIVP4        1            0          6          6
*98212/084247*

/DISPLAY QCNT BALGRP MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
     *98212/084329*

/CQQUERY STATISTICS STRUCTURE ALL

STRUCTURE NAME    LEALLOC  LEINUSE ELMALLOC ELMINUSE   LE/EL
     IMAOMSGQ         126       12      127       12 0001/0001
     IMAOMSGQOV       N/A      N/A      N/A      N/A    N/A
     IMAOEMHQ         126        4      127        4 0001/0001
     IMAOEMHQOV       N/A      N/A      N/A      N/A    N/A
     *98212/084349*
```

11.4 On the Full-Function Shared Queue Initial Program Load of an MVS with Messages

During our test cases we tried to determine CQS and IMS behavior after a whole system outage. For this reason we did three IPL tests:

- The first test was done after cancelling the address spaces (CQS1 and IMA1) belonging to the MVS that was about to receive the IPL.
- The second test was run after cancelling only CQS1, with IMA1, still up at IPL time.
- The third test was executed by an IPL of the MVS while leaving both CQS and IMS up.

We report only the CQS messages; IMS was emergency restarted as usual. Ten messages were queued to the full-function shared queue (IMA0MSGQ). The CQS message is

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                      10            10     98216/072521 98216/072533
*98216/072612*
```

Both CQS1 and IMA1 were MVS cancelled and an IPL performed.

These messages were issued by CQS1 upon restart:

```
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0EMHQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0MSGQ, LOGTOKEN ...
CQS0020I CQS READY CMA1CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0MSGQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0EMHQ, LOGTOKEN ...
```

Operation was as expected: a system checkpoint was taken at CQS startup and IMS synchronization was performed. The full-function shared-queue still contains the ten messages that were previously queued.

The test was repeated, each time with ten messages on the full-function shared queue (IMA0MSGQ). CQS1 was cancelled and a repeat IPL of MVS was performed while IMA1 was still up and running.

This is what we saw when CQS1 restarted:

```
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0EMHQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0MSGQ, LOGTOKEN ...
CQS0020I CQS READY CMA1CQS
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0MSGQ, LOGTOKEN ...
CQS0030I SYSTEM CHECKPOINT COMPLETE, STRUCTURE IMA0EMHQ, LOGTOKEN ...
```

We performed a repeat IPL of the system with both CQS1 and IMA1 up and running and we got the same CQS results.

Operation was as expected: a system checkpoint was taken at CQS startup and IMS synchronization was performed with the full-function shared-queue still containing the ten messages.

```
/DISPLAY QCNT TRANSACTION MSGAGE 0

QUEUENAME          QCNT-TOTAL     QCNT-AGED    TSTMP-OLD    TSTMP-NEW
IVTNO                      10            10     98216/072521 98216/072533
*98216/092113*
```

11.5 Fall Back to Local Queues

To fall back from a shared-queues environment to a non-shared-queues environment requires an IMS cold start. Consequently, it is not possible to read the messages from the shared-queue. We strongly recommend that IMS be brought down normally before going back to non-shared-queue operation; otherwise, the messages on the queue are left unprocessed. Even in normal situations, an attempt to restart IMS and then shut it down should be made to allow a clean fallback. Restart the failed IMS with an emergency restart, shut it down normally, and then cold-start it without shared-queues.

We must emphasize that an IMS cold start of an abnormally terminated IMS could also cause database integrity problems if the correct cold start procedure is not followed. A batch backout of inflight units of work is required.

We tested the fall-back situation from a shared-queues environment to a non-shared-queues environment. Using our sysplex environment, we brought down IMA1 normally and we modified IMA2 down in order to simulate a normal and an emergency fallback, together.

The first test we did was to move from the nonshared to the shared queue without performing an IMS cold start. The following abend occurred:

```
DFS3910I IMS IN A SHARED QUEUE ENVIRONMENT IS UNABLE TO RESYNC WITH CQS
DFS629I IMS RST TCB ABEND - IMS 0231 ...SUBCODE 3
```

In our fallback test we proceeded as follows:

1. IMA1 was closed normally, using /CHECKPOINT FREEZE.
2. IMA1CQS was closed using the MVS purge command.
3. Member DFSPBMA1 was updated with “SHAREDQ=,.” If no value is specified for the SHAREDQ parameter, local queues are used instead of shared queues.
4. IMA1 was then restarted cold and all the queues formatted:

   ```
   S IMA1CTL,AUTO=N
   /NRESTART CHKPT 0 FORMAT ALL
   DFS994I COLD START COMPLETED.  IMA1
   ```

5. After receiving message DFS994I, we were able to log on to IMA1, and successfully enter IMS commands and transactions. These completed as expected.
6. We tried a CQS command and we got

   ```
   /DISPLAY QCNT TRANSACTION MSGAGE 0

   DFS1976 10:10:35 KEYWORD INVALID - SHARED QUEUES NOT ENABLED
   ```

7. On the IMA2 side, we modified it down using MVS command:

   ```
   F IMA2CTL,STOP
   ```

8. IMA2CQS was intentionally left running.
9. DFSPBMA2 was updated also with SHAREDQ=,
10. DBRC commands NOTIFY PRILOG, CHANGE.SUBSYS, DELETE.SUBSYS were used to allow the desired cold start in our system programming environment for these particular tests. The correct procedure to follow for a production environment (batch backout of inflight work units) was noted earlier:

```
NOTIFY.PRILOG OLDS(DFSOLP01) STARTIME(982151100376) -
RUNTIME(982151145000) SSID(IMA1) LASTREC(0)

CHANGE.SUBSYS SSID(IMA2) STARTRCV
CHANGE.SUBSYS SSID(IMA2) ENDRECOV
DELETE.SUBSYS SSID(IMA2)
```

11. IMA2CTL was restarted cold

    ```
    S IMA2CTL,AUTO=N
    /NRESTART CHKPT 0 FORMAT ALL
    DFS994I COLD START COMPLETED.  IMA2
    /START DC
    ```

    After that we were able to log on to IMA2 and do IMS business as usual.

After falling back to a non-shared-queues environment, installation managers could choose whether to delete the shared-queue structures, depending on their own needs and intentions to restart use of the shared-queues environment after a short time. The messages already left in the shared-queue will still be there, available to be processed as soon as a shared-queues environment is setup again. They can be extracted using IMS Message Requeuer Version 3 and requeued to the local message queues in a non-shared-queues environment.

Appendix A. Shared Queue Planning Worksheets

Figure 39, Figure 40 on page 112, and Figure 41 on page 113 were used in the planning for shared queues.

The installation is at liberty to choose its own variable names throughout. These worksheets show a simplified naming convention where underscores are replaced by suitable values to produce unique names. See Figure 1 on page 10 for the generic names set used at Telstra.

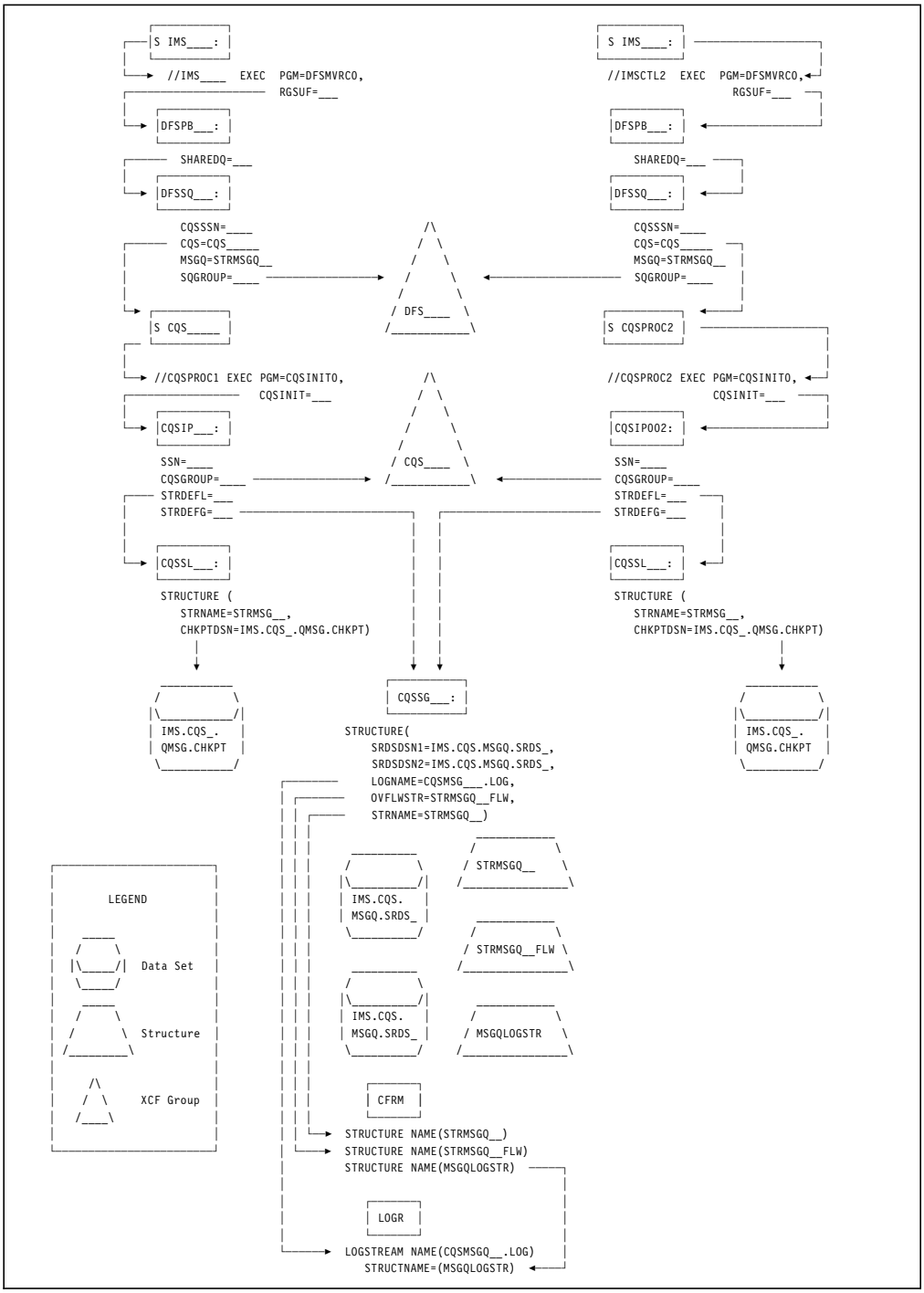


Figure 39. Shared Queue Objects Naming Worksheet - Full Function Only

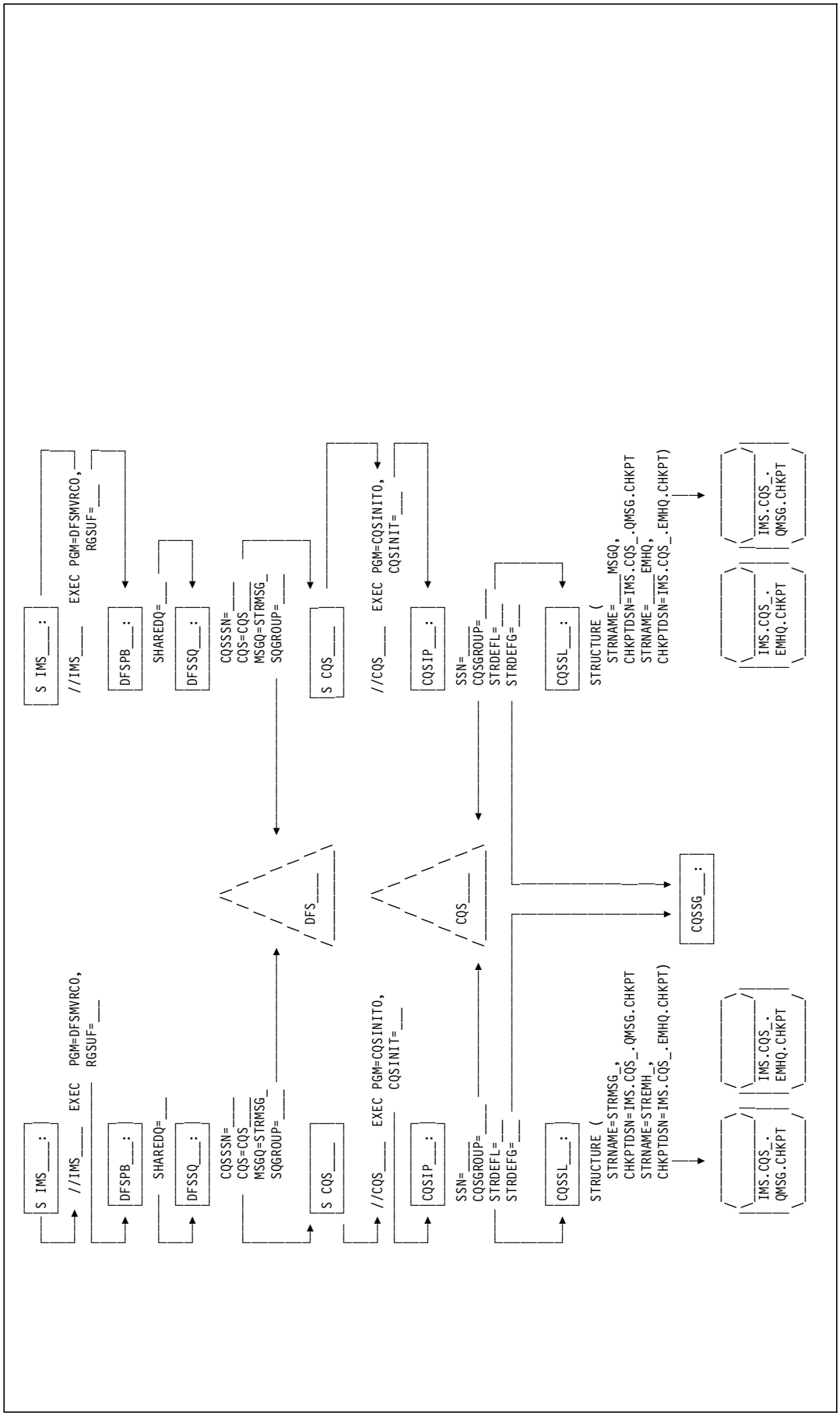


Figure 40. Fast Path and Full Function Shared Queue Objects Naming Worksheet - Top Half

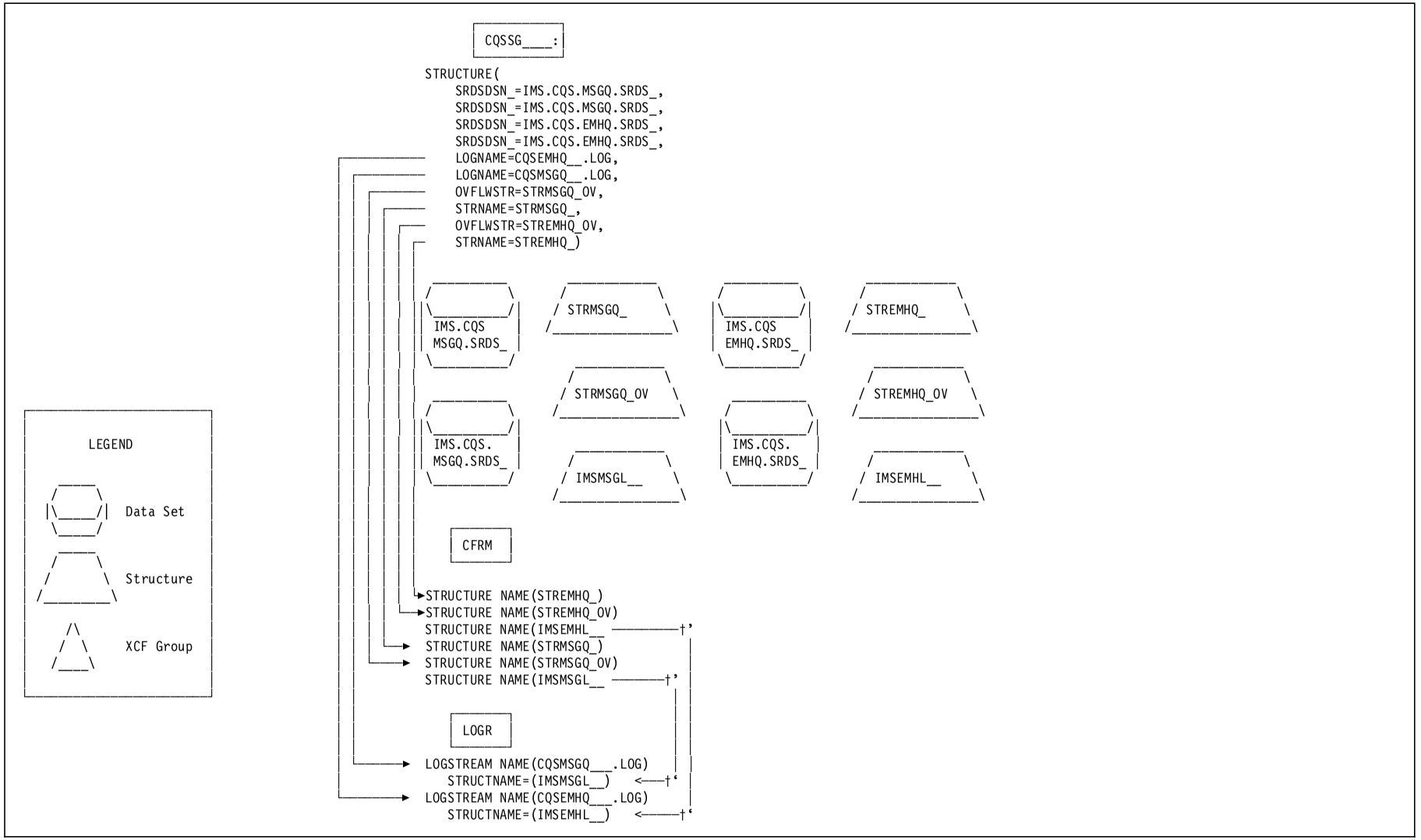


Figure 41. Fast Path and Full Function Shared Queue Objects Naming Worksheet - Bottom Half

Appendix B. Analyzing the Message Queue Structure

A shared-message queue structure is divided into 11 CQS private queues and 11 CQS client queues. IMS commands allow you to determine the contents of the client queues, but there are no IMS commands to determine the contents of the private queues.

The IMS/ESA Message Requeuer Version 3 provides several functions for managing the shared-queues:

- BROWSE tells you how many messages are on the queues, including the cold queue. BROWSE can read (copy) messages to a data set
- QUERY tells you whether you have old messages on the queue and the number of messages
- RECOVER deletes or requeues messages from the cold queue
- UNLOAD removes messages from the queue

IMS/ESA Message Requeuer was not available in Telstra during the residency.

There are two private queues for which users may need to know the contents: the cold queue (COLDQ) and the lock queue (LOCKQ). The most effective way to determine what is on the private queues is to take a dump of the structure and then use IPCS to analyze the dump. The commands to perform the dump are as follows:

```
DUMP COMM=(Dump Title)
nn,STRLIST=(STRNAME=structure name,(LISTNUM=ALL,ADJ=CAPTURE,EDATA=UNSER))
```

In the Interactive Problem Control System (IPCS), issue the STRDATA ALLDATA command to display the contents of the structure.

An IMS shared-queue structure comprises 192 list entries (0 - 191) of which the first 71 are CQS private queues. The cold queue uses list entry 1 and lock queue uses list entries 27 through 37.

Figure 42 shows a list-entry summary of one of the lock queue entries.

```
List Number.......................... 28
    List Number Status................... Complete

       List Controls:

          List Element Count Limit....... 40856
          List Element Count............. 2
          List State Transition Count.... 3
          List Cursor Direction.......... Head-to-tail
          List Cursor.................... 00000000 00000000 00000000
```

Figure 42. Display of a List Entry in a Dump

List number 28 is within the lock queue range, showing that it is a lock queue entry.

The List Element Count of 2 indicates that there are two messages in the list (lock queue).

A few pages below the list-entry summary, the text of the two messages will be displayed. A subset of the display for Message 1 is shown in Figure 43 on page 116.

```
Entry Position................... 1

     List Entry Controls:

        Data Elements in Entry...... 1
        List Number................. 28
        List Entry ID............... 00000000 00001FF6 00000003

    +0000  01C9E5E3  D5E54040  40404040  40404040  ] .IVTNV           ]
    +0010  C9D4C1F1  40404040  B0FFFAA7  A2EC0E06  ] IMA1    ...xs... ]
    +0020  C9D4C1F1  40404040  B0FFFAA7  A2EC0E06  ] IMA1    ...xs... ]
    +0030  00000000  00000000  00000000  00000000  ] ................ ]
```

Figure 43. Display of a List Entry in a Dump

The entry position (1) indicates that this is the first message within list entry 28. The dump shows that the message (IVTNV) was entered on, and is being processed by, IMA1.

The list entry summary indicated that two messages within list entry 28. The second message is displayed following Message 1.

The total message count on the lock queue is determined by summing the list element counts for all the list entries in the range 27 through 37.

A nonzero list element count for list entry 1 indicates that there are messages on the cold queue for this message queue structure. In-flight messages (those being processed by an IMS) will be put on the cold queue when the IMS system is cold-started after it terminated abnormally.

Appendix C. IMS and CQS Log Records

It is sometimes necessary to understand the sequence of IMS log records produced by an executing system. The patterns shown here are typical and can be used as the basis for understanding the relationships between the log records created during message handling and scheduling.

C.1 IMS Log Records

The log records usually of interest in a transaction processing environment are these:

Record Type (hex)	Record Meaning
01	IMS Message (Input) as it appears on the IMS Message Queue. Indicates an insert was issued to put an incoming message or message switch on the message queue. More precisely, the record indicates that an input queue record was either filled with message segments or received the last message segment.
02	Command logging. Marks completion of processing for an IMS command.
03	IMS Message (Output) as it appears on the IMS Message Queue. Indicates an insert was issued to put an outgoing message or program switch on the message queue for a terminal or program. More precisely, the record indicates that an output queue record was either filled with message segments or received the last message segment.
06	IMS Accounting Record. Indicates the start and stop time of IMS batch programs, the IMS system, and the closing of the log.
07	Application Accounting. Indicates MPP or BMP execution completed. This record contains accounting information for MPPs and BMPs. This record is also created for region start, region stop, and backout complete.
08	Processing Program Scheduling. Marks the start of MPP or BMP execution.
09	Overflow sequential access method (OSAM) Sequential Buffering Statistics. Contains details of application and database usage of sequential buffering.
0A	CPI-C Driven Program Start/Terminate. Indicates the start or termination of a CPI Communications driven program.
10	Security Violation. Indicates security was violated and identifies the violating terminal or user.
11	Start of Conversation. Indicates a conversation was scheduled, and all resources were reserved.
20	Database Open. Identifies the database being opened and the associated program specification block (PSB), and gives the date and time the database was opened.

21 Database Close. Identifies the database being closed, and gives the date and time the database was closed.

24 Database Error. Records each occurrence of an I/O error and identifies the database and program (VSAM only).

30 Message Prefix Change Record. Contains format name and Multiple Systems Coupling (MSC) flags.

31 Message Queue Get Unique. Marks a get unique (GU) call to the message queue to pass an incoming message to the application program or to start an outgoing message to its destination terminal.

32 DL/I Message Queue Input Rejected. Indicates the removal of an input message used by a program that abended.

33 Message Queue Records Freed. Indicates a set of message queue records (DRRNs) was freed.

34 Cancellation of a Message. Indicates a previously logged message was cancelled.

35 Enqueue or Reenqueue of a Message for a Permanent Destination. Indicates an incoming or outgoing message was placed on the message queue.

36 Dequeue Save or Delete of a Message for a Permanent Destination. Indicates a record was removed from the message queue. For outgoing messages, this log record marks confirmation from the terminal that the entire message was received without error.

37 QBLKs at SYNC Points. One record is written for each destination processed during queue manager processing at a sync point. The destination can be either a scheduler message block (SMB) or communication name table (CNT).

38 Transaction Reschedule. Indicates an unprocessed input message was returned to the input SMB after an application abended.

39 Output Queues to Be Released. One record is written for each output queue to be freed during cleanup processing of a RELEASE call.

40 Checkpoint Records. Defines a series of records containing checkpoint information.

41 Batch Checkpoint Records. A batch region or a BMP issued a CHKP call.

45 Internal Resource Statistics. Contains statistics of pools, buffers, scheduling, storage, and latches.

50 Database Update. Records characteristics of each database update call including time, type of call (processing option), program name, database name, and database organization. The type 50 record with subcode 50 (*after* image) is written after changing data in the buffer pool but before actually writing to disk. The type 50 record with subcode 51 (*before* image) is written before the old information is overlaid. The type 50 record with subcode 52 is written before an ISRT call for a new root for a key sequenced data set (KSDS).

56 External Subsystem Event. Identifies ESAF connection and error events.

59 Fast-Path log records. Different subtypes for EMH, data entry database (DEDB) and main storage base (MSDB):

- 5901 and 5903 are the input and output messages.
- 5911 and 5916 are the input and output enqueue records on the EMHQ structure.
- 5937 and 5938 are the commit and abort records.
- 5936 is the output dequeue message record.

67 Trace Records. Two types of trace records are of interest:

- 6701 and 6703, which are communication trace records used for the Resource Availability report
- 67FD and 67FF, which are SNAP trace records used for the Management Exception report

70 Online Change. Indicates an online change took place.

72 User CREATE, DELETE, or MODIFY. Used to log the addition of an LTERM dynamically to the IMS system.

C.2 Log Record Patterns in a Local Queue Environment

The log record patterns in this subsection show the relationship between log records in some typical situations, and indicate how log analysis of such patterns can reveal the following four components of response time:

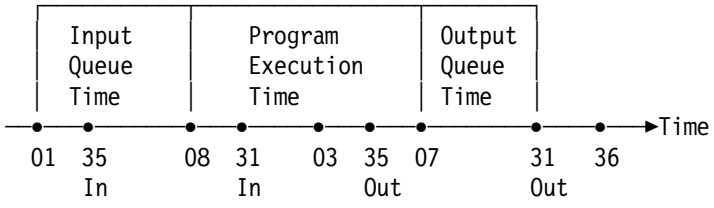
- Input queue time
- Processing time
- Output queue time
- Program switch time (when appropriate)

The numbers at the bottom of each log record pattern are the log record numbers.

For a discussion of log record patterns when shared queues are employed in an IMS Version 6 sysplex environment, see C.3.1, “Log Record Patterns in a Shared-Queue Environment” on page 122.

C.2.1 Single-Segment Input with a Single-Segment Response

For a simple transaction that accepts a single one-segment input message and returns a single one-segment response to the originating terminal, the record pattern is recorded on the log:



The sequence of events recorded on the log in this situation is as follows:

01 An insert was issued to put an incoming message on the message queue.

35 An incoming message was placed on the message queue.

08 An MPP or BMP started.

31 A GU was issued to the message queue to pass an incoming message to an application program.

03 An insert was issued to put an outgoing message on the message queue for a terminal.

35 An outgoing message was placed on the message queue.

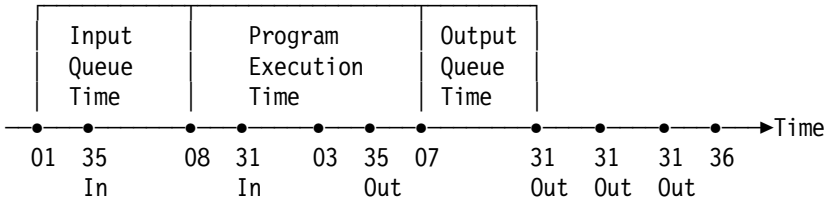
07 The MPP or BMP ended.

31 A GU was issued to the message queue to send an outgoing message to a destination terminal.

36 A record was removed from the message queue.

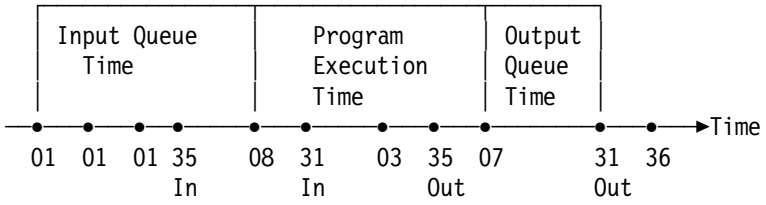
C.2.2 Single-Segment Input with Operator Logical Paging on Output

A transaction that accepts a single one-segment input message and responds with operator logical paging to the originating terminal may produce a log record pattern like this:



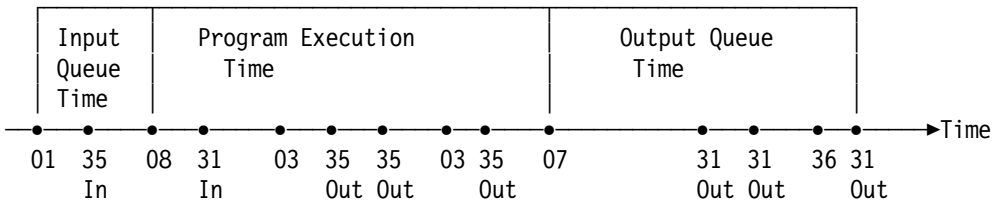
C.2.3 Multisegment Input with a Single-Segment Response

A transaction that accepts a single three-segment message and returns a single response produces the following log record pattern:



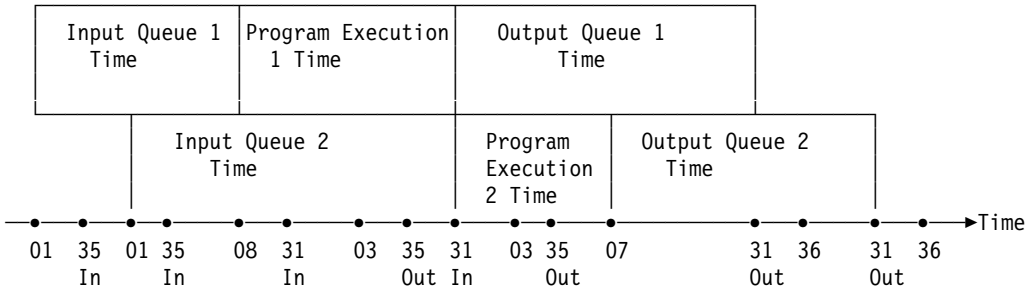
C.2.4 Single-Segment Input with Multiple Single-Segment Responses

This transaction accepts a single one-segment input message and responds to three different terminals so that only the last response is to the originating terminal. It may produce a log record pattern like this:



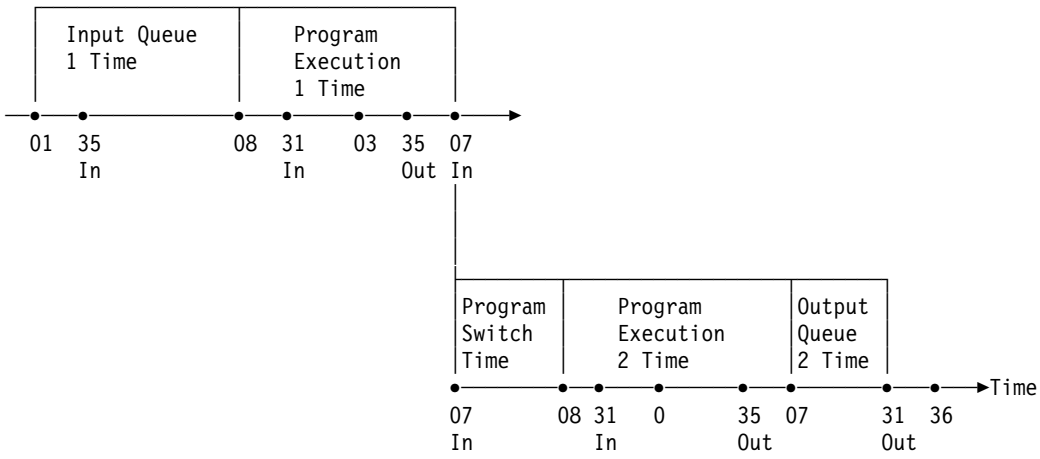
C.2.5 Multiple Single-Segment Inputs with a Single-Segment Response

A complex log record pattern is produced when a program processes two one-segment input messages that produce single responses to their corresponding originating terminal. Since the program execution time for the first transaction extends until receipt of the second transaction, this time may be longer than the actual processing time. Any time spent waiting for the next input after processing has completed is recorded in the 31 log record that gets the next input message. This time (subqueue 6 time) is subtracted from Program Execution 1 time before reporting. The resulting pattern is as follows:



C.2.6 Single-Segment Input with a Program Switch

In this scenario, a program switch occurs. Program 1 accepts a single one-segment message, and sends a program switch message which causes Program 2 (secondary transaction) to be scheduled. Program 2, in turn, responds to the originating terminal. The following log record pattern applies:



C.3 Shared Queues in an IMS Sysplex

With IMS Version 6, all of the IMS subsystems in a sysplex can share a common set of queues for input, output, and Fast Path messages. A message placed on a shared queue can be processed by any IMS subsystem that has access to the shared queue and is capable of processing the message. The common queue server (CQS) is the facility that manages the shared queues.

In general, IMS handles messages in the following manner:

1. IMS subsystems register interest in those queues for which they are able to process messages (work).
2. When an IMS subsystem receives a message and places it on the shared queue, all IMS subsystems that have registered interest in that queue are notified.
3. One IMS subsystem retrieves the message and processes it.
4. The IMS subsystem that processes the message places a response on the queue.
5. The IMS subsystem that submitted the original message is notified that the response message was placed on the queue.
6. The IMS subsystem that submitted the original message sends the response message to the originating terminal.

With IMS Versions 4, 5, and 6 (nonshared-queue), each IMS subsystem has its own queues for both input and output messages, and has its own expedited message handler for Fast Path messages. The IMS subsystem that receives a message processes it, unless that IMS is set up to send the message to another IMS subsystem using MSC, Message Requeuer, or some other means.

C.3.1 Log Record Patterns in a Shared-Queue Environment

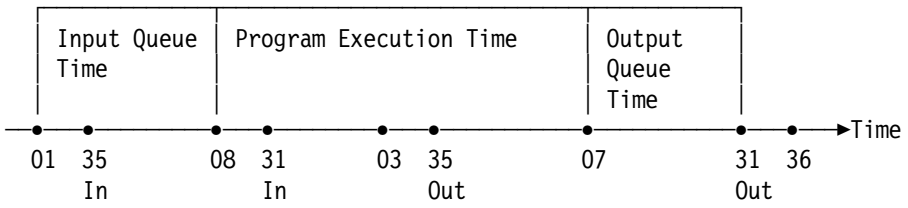
The log record flow of transactions that use shared queues for their messages is similar to the log record flow of standard transactions, except that the log records are not always on the same log file. With shared queue transactions, the log records for transit time reports may be on several IMS subsystem logs.

IMSPA Version 1.2 can be used to report transit times. It merges log records in time sequence from all IMS subsystems participating in the sysplex. This enables IMSPA to build a complete transaction-transit picture.

C.3.1.1 Single-Segment Input with a Single-Segment Response

Consider a simple transaction that has one input message, schedules a program and issues some DL/I database calls, then issues one output message back to the terminal.

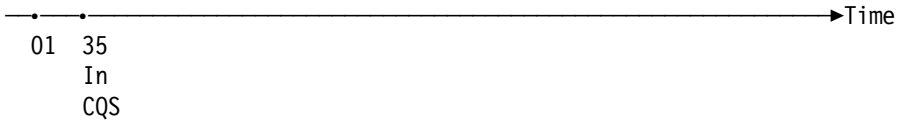
For an IMS system using the local message queue only, the log record sequence is:



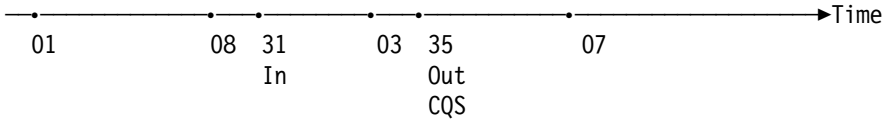
In a sysplex of three IMS subsystems using the shared queue for messages, this simple transaction could be processed by any number of IMSs, although most often only two would be involved (the FE and the BE systems). If the origin (statically-defined) terminal was logged on to more than one IMS (a rare event), a different IMS could select the reply and send it. Usually, only one IMS should have registered interest in an lterm queue.

The following scenario shows processing by all three subsystems:

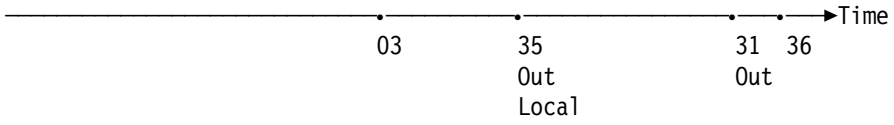
1. IMS1 accepts the input message from the terminal and puts it onto the shared queue.



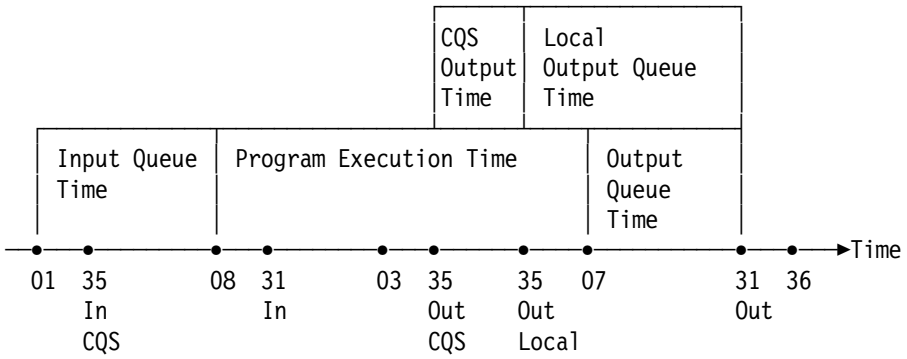
2. IMS2 reads the input message from the shared queue, schedules the application, and issues an output message that is put onto the shared queue.



3. IMS3 then reads the output message from the shared queue and sends the response back to the terminal.



4. After the log input from the three IMS subsystems is merged, the log record flow is:



For more complex transactions with message switches, the rules for shared-queue processing remain the same. Any IMS message can be processed by any of the eligible IMS subsystems in the sysplex.

C.4 CQS Log Records

CQS log records are written to the MVS log stream during processing. An MVS log structure and an MVS log stream have to be defined in the CFRM and LOGR policies using the MVS IXCMIAPU utility. The Telstra definitions are shown in Figure 7 on page 24.

The CQS log records are handled by the MVS logger with MVS buffers, coupling facility structures, and a log data set. When the HIGHOFFLOAD parameter is

reached, the data are offloaded to the log data set without quiescing the logging activity.

Offload also occurs for:

- Recovery of a log stream
- Structure rebuild for the coupling facility log stream
- When the last connector to a log stream disconnects

During the life of a transaction (one input message and one output message), CQS writes at least four CQS log records on the MVS logger. Other statistics and checkpoint log records are written during CQS normal operation. Some log records are batched to reduce the amount of log recorded and improve performance; they are written together in a single logger request.

CQS log records are used by CQS during restart and recovery situations and are deleted when no longer needed for recovery - (considered to be after two structure checkpoints). The physical delete of log data marked for deletion is driven by the offload process.

We have printed CQS log records using the standard DFSERA10 (IMS File Select and Formatting Print utility) using exit CQSERA30. IGSEXIT is the MVS System Logger default log stream subsystem exit routine: it copies the CQS log records that will be formatted by CQSERA30.

SYSUT1 points to the CQS log stream name that is specified in the start-up global definition member (CQSSGMA0).

```
//*************************************************************
//*      FUNCTION : PRINT MVS LOGGER WRITTEN BY
//*                 CQS FOR SHARED QUEUE INFORMATION
//*************************************************************
//JS010    EXEC PGM=DFSERA10
//STEPLIB  DD  DSN=IMA1.RESLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DSN=SYSPLEX.IMS.MSGQ,SUBSYS=(LOGR,IXGSEXIT),
//         DCB=(BLKSIZE=32760)
//SYSIN    DD  *
CONTROL  CNTL H=EOF
OPTION   PRINT EXITR=CQSERA30
END
//
```

Figure 44. Sample JCL to Print CQS Log Records

where STEPLIB points to IMS.RESLIB containing DFSERA10 module, and SYSUT1 points to the log stream defined in CQSSGxxx (LOGNAME=).

For the SUBSYS parameter,

```
SUBSYS=(LOGR,exit_routine_name,'SUBSYS-options1','SUBSYS-options2')
 where:
 SUBSYS-options1:
 FROM={({yyyy/ddd,hh:mm:ss})  ] OLDEST}
 TO={({yyyy/ddd,hh:mm:ss})  ] YOUNGEST}
 ,DURATION=(nnnn,HOURS)
```

```
,GMT]LOCAL
SUBSYS-options2:defined by the log stream owner
```

If an exit routine name is not specified on the SUBSYS=(LOGR,exit_routine_name,...) statement, IXGSEXIT will be used as the default log stream subsystem exit routine. For more information related to the IXGSEXIT refer to *OS/390 MVS Programming: Assembler Services Reference*, GC28-1910.

The log records written by CQS are:

00003 CQSCONN REQUEST	Connect to structure
00004 CQSDISC REQUEST	Disconnect from structure
00007 CQSPUT REQUEST	Put message on IMS queue
00008 CQSREAD REQUEST	Read message, move to lock queue
0000B CQSMOVE REQUEST	Move message to another queue
0000D CQSDEL REQUEST	Delete message from structure
00010 CQSSHUT REQUEST	CQS shut down
00032 SYSTEM CHECKPOINT	CQS system checkpoint
00040 INITIALIZE STRUCTURE	CQS structure initialization
00042 STRUCTURE CHECKPOINT	CQS structure checkpoint
00043 STRUCTURE REBUILD	CQS structure rebuild
00044 STRUCTURE OVERFLOW	CQS structure overflow processing
00060 STATISTICS	CQS statistics

Each printed log record shows the data and time (GMT) when the record was logged, the record type, and a dump of the record itself. CQS log record DSECTs can be obtained by assembling macro CQSLGREC with parameter RECORD TYPE=ALL for a more detailed description. More information is given in *IMS/ESA Common Queue Server Guide and Reference*, LY37-3730.

IMS has changed some of its log records in order to support shared-queues:

- A time stamp is now added to all log records (8 bytes)
- A unit-of-work identifier (UOW) is added to QMGR log records (a 34 byte field with 2 bytes for segment length and 32 for the CQS UOW).
- New subtypes have been added to x'67D0' log record:
- New X'3F' log records (release of UOWE in full function)
- New X'5911' log record (successful insert of an input message on an EMHQ structure)
- New X'5916' log record (successful insert of an output message on an EMHQ structure)
- New X'6740' log record (UOW moved to the CQS private cold queue).

IMS log records can be printed with the standard DFSERA10 utility using exit DFSERA30.

C.4.1 MVS Log Stream

The MVS log stream is a collection of data. There are two types of log streams:

- The CF log streams
- The DASD-only log streams

Telstra uses the coupling facility log stream. The CF log stream uses two levels of storage: the CF for interim storage and DASD log for more permanent storage. When the CF space for the log stream fills to a threshold, the data is offloaded to the DASD log data sets.

C.4.1.1 Print CQS Logs

DFSERA10 can be used to print CQS logs from the MVS system log, as shown in Figure 44 on page 124

```
//JSO1O    EXEC PGM=DFSERA1O
//STEPLIB  DD  DSN=IMA1.RESLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DSN=CQS.MSGL.LOGOFFLD,
//         DCB=(BLKSIZE=32760),
//         SUBSYS=(LOGR,IXGSEXIT)
//SYSIN    DD  *
 CONTROL  CNTL H=EOF
 OPTION   PRINT EXITR=CQSERA3O
 END
```

Figure 45. JCL to Print CQS Logs from the MVS System Log

C.4.1.2 MVS Log Offload Data Set

The offload data set is a VSAM linear DASD data set dynamically allocated by the system logger when required. Offloading is triggered when:

- The high offload threshold for the CF for a log stream is reached.
- Recovery for a log stream is complete, and the system logger flushes all log data to DASD.
- Structure rebuild occurs for the CF log stream.
- The last connector to the log stream disconnects.

When the MVS logging structure is filled to its threshold, the MVS logger offloads the oldest log data to the MVS log data set, while continuing to write to the logger structure. DFDSS can be used to dump the MVS log data set, as shown in Figure 46.

```
//STP1  EXEC PGM=ADRDSSU,REGION=4096K,TIME=14OO
//SYSPRINT DD   SYSOUT=*
//IN    DD   UNIT=339O,VOL=SER=DO6341,DISP=SHR
  PRINT INDD(IN) DS(IMAO.CQS.MSGL.LOGOFFLD.AOOOOOOO.DATA) TOL(ENQF)
```

Figure 46. Sample JCL to Dump the MVS Log Data Set

C.4.1.3 Delete Log Offload Data Sets

The log offload data sets are automatically deleted by the logger when they no longer hold valid data. To manually delete log offload data sets, the log stream definitions from the LOGR policy must be deleted. The system logger will subsequently delete all the log data sets associated with the log stream.

After log data is marked for deletion, the system logger does not delete the log data or log data sets until an offload occurs.

Do not use other methods for deleting the log offload data sets (for example, do not use the TSO/E DELETE command) because the system logger cannot perform any cleanup and the deleted data sets still count toward the data set directory limit.

Appendix D. Sizing Coupling Facility Structures

This appendix describes the contents of the list structures used by the IMS shared queues. Included are

- Components of a list structure
- Defining coupling facility structures
- Sizing coupling facility structures
- Message queue data set record allocation

D.1 Components of a List Structure

Coupling facility list structures are managed by the coupling facility control code (CFCC). This section is based on CFCC Version 4. CFCC details change frequently; you must obtain up-to-date information from the *OS/390 PR/SM Planning Guide*, GA22-7236.

IMS shared-queues are stored on coupling facility list structures. IMS uses two types of list structures: the full-function (MSGQ) structure and the Fast Path (EMHQ) structure. The MSGQ structure is always required, and the EMHQ structure is required only if the system has been generated as a Fast Path system. (IMS is a Fast Path system if the IMS system definition includes the FPCTRL macro.)

If the IMS system definition includes the FPCTRL macro, then the EMHQ structure is always required. There are cases where the IMS system definition has the FPCTRL macro but EMH is not being used to process transactions, such as with Fast Path databases being used. The EMHQ structure is still required because Fast Path transactions could be implemented through an online change if the system has been defined as Fast Path capable. If the system is defined as Fast Path capable, and you do not process EMH transactions, your EMHQ structure can be defined with the minimum size of 768 KB (see D.4, “Sizing Coupling Facility Structures” on page 135).

Shared-queue structures are persistent. They remain allocated even if all CQSs have disconnected, or all IMSs and their associated CQSs cold start.

List structures can contain four significant components:

- List headers
- List entries
- Lock table
- Event monitor controls (EMCs)

The lock table and the EMCs are optional components of an MVS list structure. CQS uses all four list structure components in the shared-queues environment.

Figure 47 on page 130 shows a full-function message queue list structure with two messages. Message 1 is stored as one data object, comprising a list entry control element and one data element. Message 2 is stored as three data objects (one in the ready queue and two in the staging queue), each comprising one list entry control element, and one to two data elements.

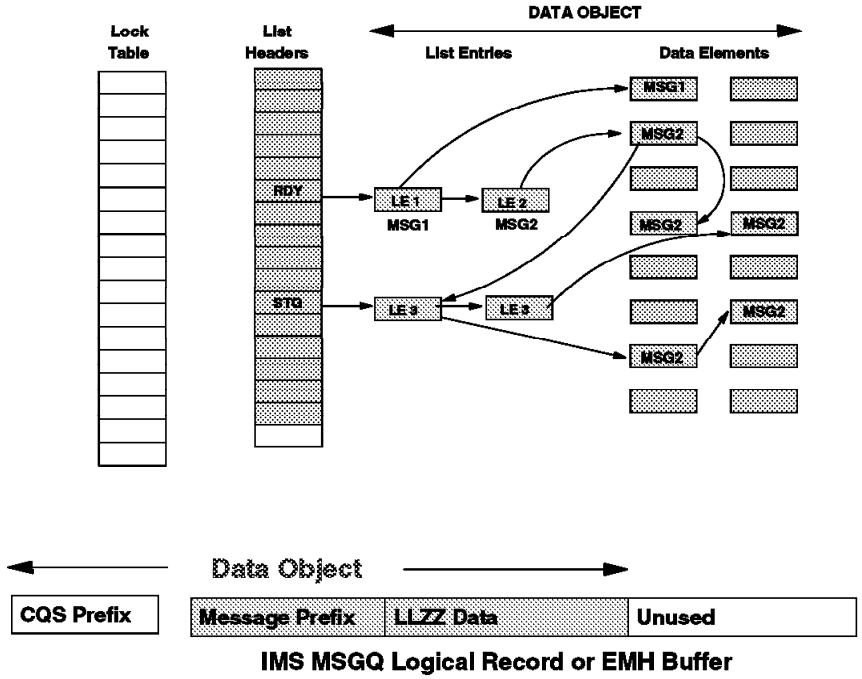


Figure 47. Components of a List Structure

D.1.1 List Headers

List headers are anchor points for CQS or IMS queues. A list header points to the first list entry in the queue. When a structure is allocated, 192 list headers are created: 71 for CQS private queues, and 121 for the IMS queues. For an MSGQ structure, 99 of the IMS list headers are used, and the remaining 22 are reserved for future use. For an EMHQ structure, 22 of the IMS list headers are used and the remaining 99 are reserved for future use.

D.1.2 List Entries

List entries are chained from list headers and are used to store messages. Each list entry contains one data object. A data object is a message queue logical record, either SHMSG or LGMSG, and may be a complete message. If a message is larger than a single IMS message queue logical record, the message consists of more than one data object. Each list entry is divided into two sections, a control area and one or more data elements. The control area contains the queue name, which is the transaction code in which IMS registers interest if it is capable of processing that type of message. The control area is also the anchor point for one or more 512-byte data elements in which the message is stored. Each object is divided into 512-byte sections, with each section stored in a data element. The data elements are chained together in order to maintain the sequence of the message pieces.

Figure 47 illustrates how the list elements and data elements are related. In this figure, assume the size of the IMS message queue logical records is 1 KB (requiring two data elements per list entry). The first message has a single list entry with one 512-byte data element containing the message. The second message is 2.5 KB long and five 512-byte data elements are required to store the message. Since the IMS message queue logical records are 1 KB in length, the 2.5 KB message would be seen as three data objects and therefore requires

three list entries. A fuller description of message queue logical record allocation is provided in D.2, “Short/Long Message Queue Data Set Record Allocation” on page 132.

When a message requires more than one IMS message queue object, the second and subsequent objects are chained from the staging queue using a unique queue name for the message based on the IMS ID and a STCK time. The first object is chained from the ready queue and points to the list entry in the staging queue.

IMS does not register interest in the unique queue name assigned to the staging queue entries and therefore is not notified of data objects placed on the staging queue. The object containing the first segment of the message is placed on the ready queue after the second through nth objects (if any) have been placed on the staging queue.

All list entries with the same queue name form a sublist. IMS requests the CQS to *put* a message on the end of the sublist if a new message arrives, or to *get* a message from the front of the sublist when it wants to process a message. In Figure 48, seven list entries are anchored off two list headers. There are three TRANX transactions. The grouping of TRANXs is referred to as a *sublist*. (The EMC area is described in D.1.4, “Event Monitor Controls” on page 132).

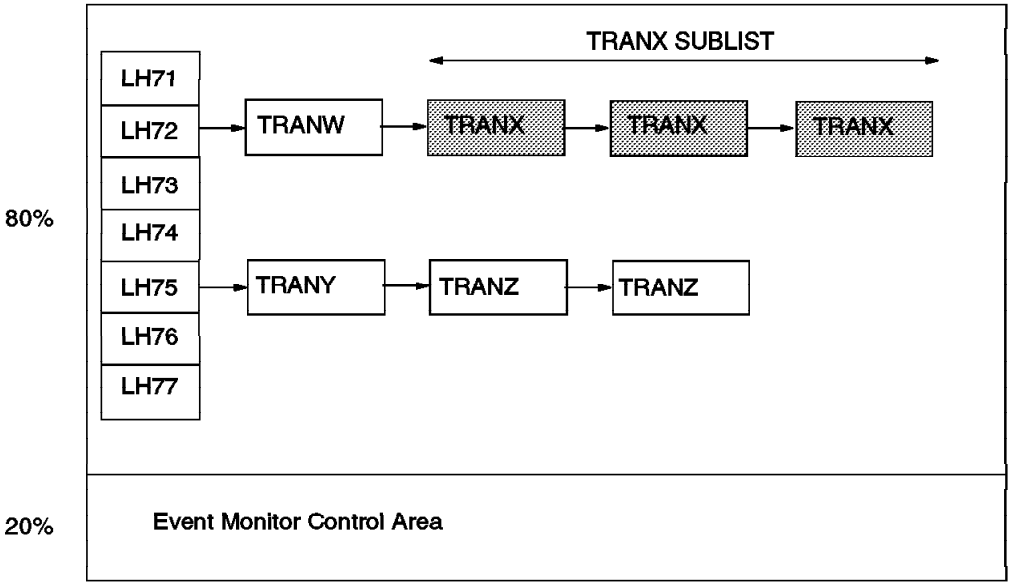


Figure 48. Transaction Sublists in the Shared Queue

D.1.3 Lock Table

The CQS uses the lock table to serialize access to the CQS control queue. The control queue is currently the only queue that is locked.

D.1.4 Event Monitor Controls

An EMC entry is a control block in the CF list structure that is used to represent a CQS that has a registered interest in a queue. Each EMC entry occupies 64 bytes of storage.

An EMC control block is created for each CQS that has an interest in a particular queue. If you have three CQSs that register an interest in queue TRANA, three EMCs are created. Take care when sizing the shared-queue structures to ensure enough space to store the number of EMCs required.

CQS reserves 20% of the total structure size for EMCs. If CFCC Level 4 or greater is being used, the area dynamically increases in 5% increments up to a maximum of 50%, if required.

Available data elements are used to create additional space for EMCs. If the EMC area is increased, the amount of space available for list entries is reduced.

If the EMC area runs out of space, any attempt by an IMS system to register an interest in a queue fails.

D.2 Short/Long Message Queue Data Set Record Allocation

The traditional IMS message-queue comprises three data sets, and a Qpool, that are managed as a unit. The Qpool buffers hold in-storage copies of the relevant message-queue physical records from QBLKS, SHMSG and LGMSG data sets. All record allocations in the Qpool reflect the data sets even if the buffers are never actually written to DASD.

QCBs represent output destinations for messages on the message-queue: lterms and MSC links are typically represented by QCBs. Messages destined for delivery outside this IMS are queued off the appropriate QCB representing the destination. QCBs are not relevant in shared-queue structure sizing calculations.

Inbound messages and program switches destined for transaction codes within this IMS are queued from the appropriate Scheduler Message Block (SMB), representing the transaction code, with the message data being logically stored on the message-queue data sets.

When the queue manager is asked to store part or all of a message, it allocates a logical record from either the short-message queue data set or the long-message queue data set. Which data set is chosen depends on the knowledge IMS has about the segment length and the conditions at the time of the request. When the segment is created by an inboard function, such as an application performing a message-queue ISRT call, then the queue manager knows the length of the segment since it is provided as the first 2 bytes of the inserted segment. If the segment is being created by IMS DC editing a message entered through the VTAM network, then the final length of the segment and message are not known until editing is completed. To optimize message-queue data set record allocation, IMS DC maintains a moving-average message size for each input terminal and uses that value to select the short-message LRECL or long-message LRECL to use for this message.

Once a long-message queue data set logical record has been selected, all further data for this message will be put in long-message queue data set records.

Multiple segments can be stored in a single logical record and can span across logical records.

All logical records used include the full message prefix (except for additional logical records used for single segment MFS spanning). The size of the prefix depends on the IMS release and the functions available within this IMS. See Table 17 in *IMS/ESA Installation Volume 2: System Definition and Tailoring*, GC26-8737 for a list of prefix sizes by IMS release. The text surrounding the table discusses maximum segment sizes.

For IMS conversations, the SPA is stored as the first segment of the message on the message-queue when destined for a transaction program or MSC. It is kept in the CQS lock queue and in the Qpool during iterations of the conversation. IMS Version 6, both shared and nonshared-queue, always keeps the SPA with the reply while the conversation is in process. This may affect Qpool specifications for installations with many conversations established, whether they are active or held.

The net effect of these factors is that it is often not possible to predict whether the short-message or the long-message will be selected by the queue manager for some messages.

When shared-queues are specified, each message-queue data set logical record becomes an object when it is passed to the CQS. Each object is located through its list element and occupies as many 512 byte data elements, as necessary, for the data that is to be stored. Unused bytes in the logical record are not passed to CQS.

D.3 Defining Coupling Facility Structures

All structures to be used in a Parallel Sysplex must be defined in the CFRM policy. In a shared-queue environment, you have to define structures for:

- Message queue structure pair (primary and overflow)
- EMHQ structure pair (primary and overflow)
- MVS logger structure for the full-function message queues
- MVS logger structure for the Fast Path message queues

IMS needs access to at least one CF structure for the full-function message queue. If the IMS definition includes Fast Path, a Fast Path EMH structure must be defined. Overflow structures for both of the structures can and should be defined, in case the primary structure fills up.

D.3.1 Defining a CFRM Policy

To enable shared queues, you must define a CFRM policy. The CFRM policy holds information about the structures used for the MSGQ, EMHQ, and system logger.

Use the couple data set format utility (IXCL1DSU) to format the couple data set, and the MVS administration utility (IXCMIAPU) to define the CFRM policy into the

couple data set. The couple data set must be accessible to all members of the sysplex.

The active CFRM policy definition may be updated with the new structures for the CQS, but the policy must be reactivated to implement any changes.

The parameters used to define a CF list structure are given below. See *OS/390 Setting Up a Sysplex*, GC28-1779 for the JCL used to run the MVS administration utility. The parameters are these:

NAME Name of the coupling facility structure to be created

SIZE Maximum size of the structure to be allocated, in units of 1 KB (1024 bytes)

INITSIZE Represents the initial amount of space allocated for the structure, in units of 1 KB (1024 bytes). This parameter is dependent on the release level of the CF structure.

Carefully choose the initial size of the structure. It may be preferable to avoid expansion of the structure while your IMS system is active, either by avoiding the INITSIZE parameter or by setting it to the same value as the SIZE parameter. A structure should have a fixed size, and be allocated up front, first to ensure that the capacity exists on the coupling facility, and second to eliminate unnecessary overhead in extending the coupling facility structure.

The INITSIZE parameter value should be chosen in conjunction with the STRMIN value, specified in the CQSSGxxx PROCLIB member. STRMIN does not override the INITSIZE parameter. The structure is allocated as defined in the CFRM policy if space on the coupling facility allows. STRMIN is used to ensure that a minimum structure size is allocated by MVS. Make sure that a structure of this minimum size is usable in your environment.

REBUILDPERCENT Determines the level of loss of connectivity at which structure rebuild takes place

When the REBUILDPERCENT parameter is set to 25, an attempt to rebuild the structure on a candidate alternative coupling facility is made when 25% of the connectivity to the structure has been lost. This percentage is based on a weighting factor. (WEIGHT parameter and CONNFAIL parameter in SFM policy). Refer to *OS/390 Setting Up a Sysplex*, GC28-1779, for a detailed explanation of this parameter.

PREFLIST Determines the preference list of coupling facilities. The structure being defined may be allocated on any of the coupling facilities defined in the preference list.

EXCLLIST Determines the exclusion list for the structure. If possible, the structure being defined is not allocated on a coupling facility that already contains one of the structures in the exclusion list.

D.4 Sizing Coupling Facility Structures

The structures examined are message queue structure, EMHQ structure, and the log structure.

D.4.1 Sizing the Message Queue Structure

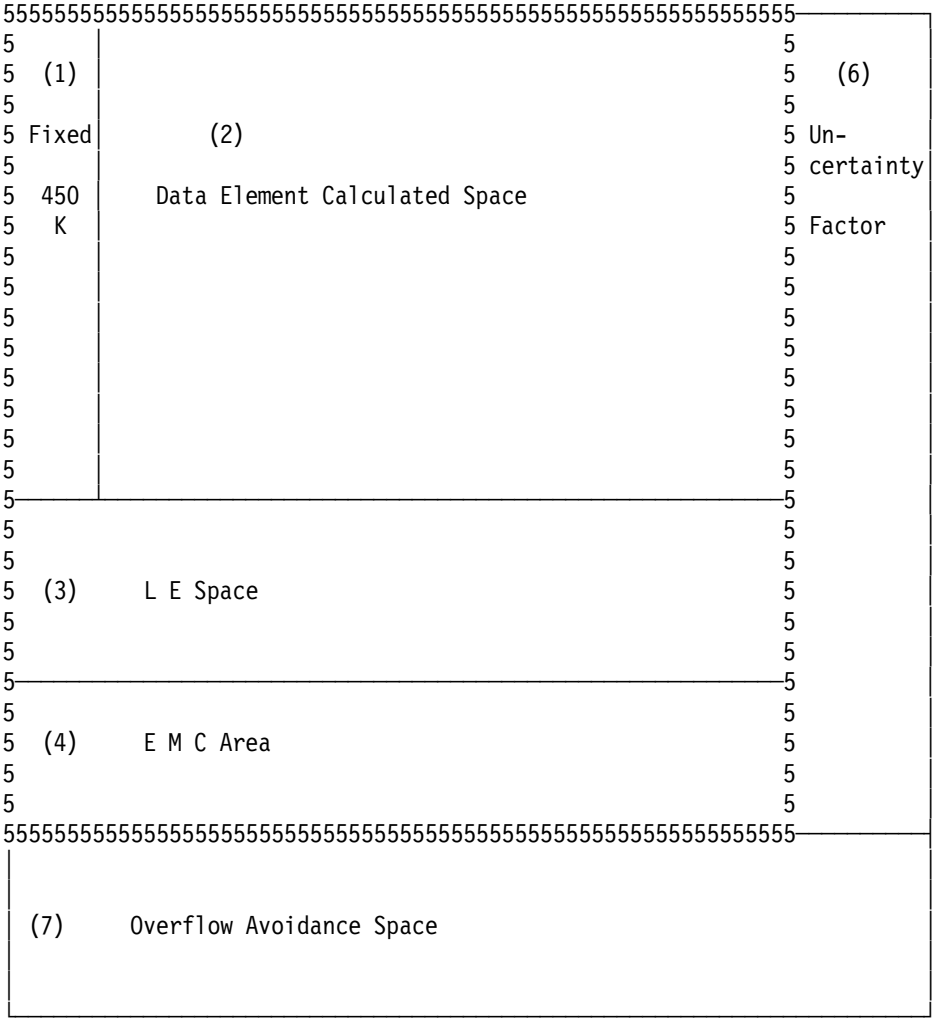


Figure 49. MSGQ Structure Allocation

Figure 49 suggests the components of a message queue list structure. List structure allocation does not allow the sizing of these components separately. Component size is calculated by MVS based on the information it gets from the first CQS to connect to the structure. IMS system programmers are likely to be aware of what has to fit into the different components, so they must calculate the complete size of the structure, working backward to make sure each component is large enough. The considerations to take into account when sizing the structure are discussed in the following. The numbers in Figure 49 refer to the numbers itemized in the text below.

1. *Fixed 450 KB*

   Every IMS Version 6 list structure has a portion that has a fixed size of 450 KB. This part of the list structure includes the list headers and the lock table.

This explains why we could not start our test IMS with the minimum structure size of 256 KB (see 7.2, “Shared-Queue Structure — Minimum Size” on page 59.) IMS could only be started after all size parameters for the IMS list structures — including the overflow structures — were increased to 768 KB. 512 KB was not enough, because 20% of this was reserved for the EMC area, leaving too little for the 450 KB fixed portion.

2. *Data element calculated space*

   The space you want to allocate for the storage of data elements is calculated as the amount that would be occupied by the data content of the transactions and messages in the coupling facility list structure. A good estimate for the space to allocate for data elements would be the amount of data you have to store today for message queues in a non-shared-queues environment. To map the space used in the message queue data sets to the way messages are stored in a CF list structure, you must understand the message queue in a non-shared-queues environment. The allocation of short-message and long-message queue data set is described in D.2, “Short/Long Message Queue Data Set Record Allocation” on page 132.

   The occupancy of the message queue data sets in a non-shared-queues environment can give you a good estimate of the amount of data used to store the messages. In Telstra, we used the IMSPARS Internal Resource Usage Report to find the long-message and short-message high water marks (capacity).

```
START  8/11/1998  11.32.48.34              I M S    L O G   T A P E   R E P O R T S (V1M9) END  8/11/1998 11.46.54.10 PAGE   1
                                                 IMS INTERNAL RESOURCE USAGE

MESSAGE QUEUE POOL STATISTICS                                                 INTERVAL :      14.06 (HHHH.MM.SS)
                                                                 COUNT /TRANSACT   /SECOND

          HIGHEST QBLKS RECORD EVER USED                             174
          HIGHEST SHMSG RECORD EVER USED                              93
          HIGHEST LGMSG RECORD EVER USED                             801
          LOCATE CALLS FROM QMGR                               1,396,539      6.95  1,651.22
          RECORD RELEASE CALLS FROM QMGR                         547,041      2.72    646.80
                •     •       •                                    •          •       •
                •     •       •                                    •          •       •
                •     •       •                                    •          •       •
```

Figure 50. IMSPARS Message Queue Pool Statistics Report

Figure 50 gives you an example of the IMSPARS Internal Resource Usage Report. If IMSPARS or IMSPA is not available, the high-water marks of the short- and long-message queue data set can be derived from the IMS system checkpoint log record 4502. The fields STQDDLO and STQLMR in the beginning of the log record show the high water marks for the short- and long-message queue. These numbers report the highest number of short-message and long-message queue logical records used since the last cold start. These numbers can also be used to calculate the average object size, if no other report is available.

Of course, it is up to the IMS system programmer to evaluate the base of the calculation. The programmer should know if any problem situations were included in the measurement period (that is, since the last cold start). If so, the high water marks could be abnormally high.

We strongly recommend that these numbers be obtained for an IMS Version 6 System. Otherwise, you must take into account that message prefixes change size from version to version and thus influence the stored length of the messages.

In IMS Version 6, a conversational transaction′s SPA is stored as the first message segment of both the input and output message, so that in a shared-queues environment any IMS that has registered interest in the transaction can process any step of the conversation. If the calculation is based on a version other than Version 6 report, and if you have many active conversations in process and the SPA is large, the calculation could change completely, as the SPA and the message remain in the message queue until the conversation is terminated.

If the high water marks are used as the planning base, the number of data elements required can be estimated as:

```
DE = SHW * Round Up ( SLRECL / 512) +
     LHW * ROUND UP ( LLRECL / 512)
```

DE Number of data elements required

SHW Short-message high water mark

LHW Long-message high water mark

SLRECL Short message logical record length

LLRECL Long message logical record length

Notes:

a. With this calculation you are on the safe side, as message queue records are not always filled and the calculation base is the high-water mark of message queue logical records.

b. The data elements required to store the SPA of conversational processing is included if the calculation is based on a Version 6 IMS System.

c. Long messages, which occupy more then one message queue logical record, are seen as multiple distinct records in the message queue data set. Each logical record will be regarded as a separate data object to be stored in the message queue structure.

3. *LE Space*

The number of list-element controls available is significant in that exhausting the list-element controls means the queue structure is full even if plenty of data elements are still available.

The number of list-element controls needed to keep these short and long messages in the structure would be the number of all records (SHW+LHW), as this is the number of data objects. But the number of allocated list-element controls is calculated internally as a certain ratio of data elements to list-element controls. This ratio is derived from the object average size. In Telstra, we used the IMSPARS Message Queue Utilization Report, which shows how many messages of specified ranges of length were stored in the long- and short-message queue data sets over a given time interval. Calculating the object average size as the average of long- and short-message record length should be accurate enough:

```
OBJAVGSZ = (SHW * SLRECL + LHW * LLRECL ) / (SHW + LHW)
```

The DE to LE ratio is internally calculated as

```
RATIO = Integer part of ( OBJAVGSZ + 24 ) / 512
```

For example, if the result of that calculation is 2.8, then RATIO would be set to 2

and the number of list-element controls to plan for in the structure should be

LE = DE / RATIO.

Notes:

a. This ratio means that space will be reserved in the allocated list re structure for the overhead and for the EMC area, with the rest containing data elements and list-element controls in the ratio RATIO/1.

b. With the above calculated value for OBJAVGSZ, the calculated number of list-element controls to be allocated is at least SHW + LHW.

4. *EMC Area*

As described in D.1.4, “Event Monitor Controls” on page 132, 20% of the total structure is to be reserved for EMCs. Multiply the currently calculated space you need by 1.25 to allow for 20% EMC area.

As the EMC area will be increased automatically by the coupling facility, if necessary, the calculation for the message queue structure could be invalidated. In addition, as any attempt by an IMS to register interest fails if the EMC area is full, care should be taken to ensure the EMC area is taken into account when sizing the CF structure.

Every transaction code included in the system generation of any of the IMSs participating in the shared-queues group requires an EMC, unless the transaction is stopped. That means if transaction TRANA is generated in IMS1 and in IMS2, two EMCs are required, as two interest registrations for this transaction have to be stored.

Every terminal active in one of the IMSs requires one EMC. This is for logged-on static terminals, and dynamic terminals where a user has signed on. To calculate the number of EMCs, add the overall number of transactions included in the system generation of any IMS participating in the shared-queues group to the number of static terminals and the number of expected active dynamic terminals.

Space for the EMC area is allocated in frames. A frame contains 64 pages and each EMC frame contains from 1 to 3317 EMCs:

EMCA = (Round Up (#EMC / 3317)) * 64 * 4096.

#EMC Number of EMCs required

EMCA Space required for the EMC area

5. *Minimum Size of the Structure*

When you define the message queue structure, there are no parameters for the allocation of the EMC area, nor for the number of data elements and list-element controls. Instead, there is only a SIZE parameter when creating the structure and an OBJAVGSZ parameter used by the first CQS connecting to the structure. In the previous steps we calculated the OBJAVGSZ parameter, now we have to calculate the SIZE parameter for the structure definition in the CFRM policy.

There are guidelines calculating the size of a structure in the coupling facility in Appendix B of the *OS/390 PR/SM Planning Guide*, GA22-7236. Several formulas are described there, and you need the following information to correctly size structures used by the CQS with IMS Version 6:

MDLES 120

LELX 1

LC	192
LTEC	256
LTEX	0
R_le	1
R_data	the integer part of (OBJAVGSZ+24)/512. For example, if the result of that calculation is 2.8, then R_data would be set to 2. OBJAVGSZ is defined in the CQSSGxxx proclib member

Notes:

a. These parameters are release-dependent and may change in future releases of IMS.

b. Control space is contained within the structure. The larger the structure, the larger the control space.

c. A structure dump shows you the maximum EMCs (maximum queue registrations).

Filling in the IMS-specific values into the formulas found in Appendix B "Coupling Facility Storage Allocation Formulas" in *OS/390 PR/SM Planning Guide*, GA22-7236, leads to the following formula:

```
SIZE = the larger of
       ( 450 KB + 200 * LE + 512 * DE ) * 1.25
       and
       5 * EMCA
```

where

LE	is the number of list entry controls
DE	is the number of list data elements
SIZE	is the space required for the list structure
EMCA	is the space required for the EMC area

Note: This calculation is not meant to be exact. All numbers in the formula are rounded, and the result again has to be rounded to a multiple of 1 KB, according to the SIZE parameter in the structure definition in the CFRM policy.

6. *Uncertainty Factor*

What we have calculated with the above formula is the minimum size required to move the nonshared system to the shared-queues environment. The structure should initially be defined to be larger than this, until experience of the system has been gained.

Having defined a structure, you can use the /CQQUERY STATISTICS command to show the maximum number of list-element controls and maximum number of data elements for the structure. When the IMSs in the shared-queues environment start storing messages in the shared-queue, you can also see the number of list-element controls and data elements currently in use. You cannot see the maximum number being used; MVS does not provide this information.

As all recommendations for sizing the CF structure say: “make it big !” — we allowed for a large uncertainty factor of 2. That means we allocated double the space estimated or calculated.

7. *Overflow Avoidance Space*

   If you decide to allocate an overflow structure in the coupling facility, and set the OVFLWMAX parameter in the global CQS parmlib member CQSSGxxx to, for example, 70, you would perhaps increase the overall size of the structure to avoid overflow processing during normal operation:

   ```
   SIZE· = SIZE * 100 / OVFLWMAX.
   ```

   The parameters OVFLWMAX and OVFLWSTR in the global CQS PARMLIB member cause messages to be moved into the overflow structure if the OVFLWMAX percent of the data elements is used. CQS always moves complete message queues to the overflow structure, starting with the largest queue. Overflow processing is performed until the usage of data elements is less than (OVFLWMAX - 20%). As the structure is quiesced during overflow processing, the structure should be large enough to avoid overflow processing during normal operation.

   Notes:

   a. Overflow processing is based on the usage of data elements. If all list-element controls of a structure are in use and the structure is full, no overflow processing is carried out. Therefore, it is better to make the OBJAVGSZ too small rather than too large.

   b. If you have a reasonable amount of conversational transactions in your message mix, then remember that the SPA and the output message stay in the structure until the conversation ends. This permanent occupancy of data elements affects overflow processing, but these messages are not eligible to be moved to the overflow structure. If they account for 50% of the structure size, then when we reach 70% full, every client queue with messages on it would go to overflow in order to reduce the number of data elements in use in the primary structure to 50%.

      If you increase the structure size by (100 / OVFLWMAX), you can avoid overflow processing during normal processing.

   c. The OVFLWMAX parameter is also used to increase the size of the primary structure, if the structure is allocated with the INITSIZE parameter. The structure is then automatically increased to SIZE.

   d. If both the increase of the structure size and the moving of message queues into the overflow structure can be used in one shared-queues environment, then the increase of structure size from INITSIZE to SIZE takes effect first and, if the structure fills up again, overflow processing begins. In both cases, the structure is quiesced for the duration of the process, but the moving of message queues to the overflow structure may take a longer time, as it iterates until the usage of data elements is reduced by 20%.

   e. The size of the overflow structure is not increased automatically by CQS. If the structure is defined with the INITSIZE parameter, it can be changed within the INITSIZE and SIZE limits by operator command.

   f. The messages issued, when CQS goes into structure resizing or overflow processing, could serve as a trigger to an automated process that detects the messages and alerts someone to find out what is causing the higher allocation of structure space.

D.4.1.1 Summary of the Calculation

1. Find out your short- and long-message high water mark and short- and long-message logical record length:

   SHW, LHW, SLRECL, LLRECL

2. Calculate the number of data elements:

   ```
   DE = SHW * Round Up ( SLRECL / 512) +
        LHW * ROUND UP ( LLRECL / 512)
   ```

3. Calculate the object average size:

   OBJAVGSZ = (SHW * SLRECL + LHW * LLRECL) / (SHW + LHW)

4. Calculate the ratio of data elements to list-element controls:

   RATIO = Integer part of (OBJAVGSZ + 24) / 512

5. Calculate the number of list-element controls:

   LE = DE / RATIO

6. Calculate space required by EMC area:

   EMCA = (Round Up (#EMC / 3317)) * 64 * 4096.

7. Calculate the minimum space required:

   ```
   SPACE = the larger of
           (( 450 KB + 200 * DE / RATIO + 512 * DE ) * 1.25 )
           and
           ( 5 * EMCA )
   ```

8. Apply uncertainty factor.

9. Perhaps increase by overflow-avoidance factor:

   SIZE· = SIZE * 100 / OVFLWMAX.

D.4.1.2 Alternative Calculation of Object Average Size

The above calculation of the required number of data elements and the average object size should be accurate enough to be used as input for the sizing formula of a message queue structure.

At Telstra we did a much more sophisticated calculation by using the IMSPARS/IMSPA Message Queue Utilization Report, which shows how many messages with a specified length range were stored in the long- and short-message queue data sets over a given time interval. This report is produced by evaluating the 01 and 03 IMS log records. It is helpful to select the range of message length in the IMSPARS/IMSPA reports in 512-byte steps to match data element size.

Start 20Aug1998 21.07.45.34 IMS/ESA Performance Analyzer End 21Aug1998 07.58.04.69 Page 1
Message Queue Utilization-IME1

Msg Length Interval	Msg Avg Length	Input Transaction Count	ShMsg	LgMsg	-Message Switch-- Count	ShMsg	LgMsg	-Program Switch-- Count	ShMsg	LgMsg	-Output Message-- Count	ShMsg	LgMsg	------Totals----- Count	ShMsg	LgMsg	Pct	Acc Pct
00000-00511	415	18398	15612	2786	-	-	-	53707	52712	995	103K	41549	61508	175K	109K	65289	63	63
00512-01023	584	49952	43716	6236	905	1798	6	-	-	-	1727	1961	666	52584	47475	6908	19	82
01024-01535	1380	893	-	893	5	6	5	1	-	1	6508	6	6508	7407	12	7407	3	85
01536-02047	1733	2097	-	2097	4	-	4	1091	-	1091	9638	4	9638	12830	4	12830	5	89
02048-02559	2163	-	-	-	4	2	4	-	-	-	29036	-	29036	29040	2	29040	10	100
02560-03071	2801	-	-	-	-	-	-	-	-	-	3	4	3	3	4	3	0	100
03072-03583	3471	-	-	-	9	6	15	-	-	-	-	-	-	9	6	15	0	100
03584-04095	3890	-	-	-	6	-	12	-	-	-	6	4	10	12	4	22	0	100
04096-04607	4335	-	-	-	20	4	40	-	-	-	9	16	16	29	20	56	0	100
04608-05119	4877	-	-	-	-	-	-	-	-	-	7	8	14	7	8	14	0	100
05120-05631	5323	-	-	-	-	-	-	-	-	-	15	22	30	15	22	30	0	100
NONREC MSGS	N/A	-	-	-	-	-	-	4	26	-	-	-	-	4	26	-	0	100
Total	718	71340	59328	12012	953	1816	86	54803	52738	2087	150K	43574	107K	277K	157K	121K	-/-	-/-

Est. Short Message Queue Dataset Record Size is 583
Est. Long Message Queue Dataset Record Size is 3498

Figure 51. Message Queue Utilization Report from IMSPARS or IMSPA

Figure 51 shows an example of the IMSPARS/IMSPA Message Queue Utilization Report, which includes input and output messages as well as message switches. The message length report takes into account that the messages may have different message prefixes. Message prefix length is included in the reported sizes.

With this information, the number of data elements can be calculated as the sum of the number of data elements required for the messages in a certain size range. The calculation has to be done separately for the long- and the short-message queues. This is because a message of a given length could end up in the short- short-message queue, or the long- message queue, or both, depending on how IMS allocates message-queue logical records for each message segment and not only on the length of the message. See D.2, “Short/Long Message Queue Data Set Record Allocation” on page 132 for a discussion of logical record allocation. The number of messages of a given size range is the product of the short-message high water mark and the percentage of messages of this size. This percentage is the number of messages in the range divided by the total number of messages in the report sample:

```
DE = ( LHW * LGRMSG / LALLMSG ) * Round Up ( GRSIZE / 512 )
  + ( SHW * SGRMSG / SALLMSG ) * Round Up ( GRSIZE / 512 )
```

Process the line above for as many ranges of message sizes as required and summarize:

DE Number of data elements to be allocated

LHW Long-message high-water mark, highest long-message record number ever used

SHW Short-message high-water mark, highest short-message record number ever used

LGRMSG Number of messages in the long-message group that belong to the size group. A size group is related to the report and means, for example, all messages with a length of 0 to 511 bytes. In the report we have one line per size group.

SGRMSG Number of messages in the short-message group that belong to the size group

LALLMSG Number of all messages in long-message queue

SALLMSG Number of all messages in short-message queue

GRSIZE Upper limit of message size in that group.

The IMSPARS Message Queue Utilization Report is also used to calculate the average size of all messages, as follows:

```
OBJAVGSZ = (SUM (( LGRMSG + SGRMSG ) * GRSIZE )) / ( LALLMSG + SALLMSG )
```

At Telstra, the calculations were done using a spreadsheet program, so we were able to adjust the calculation easily for test and production environments. The spreadsheet was also helpful in understanding the effect of changing one of the parameters.

D.4.2 Sizing the EMH Structure

Although EMH is not used by Telstra, we had to define a list structure for the EMH message queue because Telstra does use DEDBs. The FPCTRL macro is therefore included in the Stage 1 system definition. We allocated an EMH structure and an EMH overflow structure with the minimum size of 768 KB.

The same formula applies for the sizing of the EMH structure as for the sizing of the message queue structure. The object average size and the number of data elements are calculated differently.

EMH average message sizes are given in the Fast Path Log Tape Analysis Utility (DBFULTA0) report output *"Overall Summary of Transit Times by Transaction Code for IFP Regions"* as illustrated in Figure 52 and can be used for the data elements estimation. All EMH messages must be selected for the reporting to provide complete information and exception reports should not be used. EMH adds about 200 bytes as a message prefix before passing the message to CQS and the sizing estimation must allow for them.

```
OVERALL SUMMARY OF TRANSIT TIMES BY TRANSACTION CODE FOR IFP REGIONS
                  ------------ TRANSIT TIMES IN MILLI-SECONDS -----------  INPUT  MSG   OUTPUT  MSG
TRANS   -NO.OF-   ----TOTAL---   --INPUT Q --  --PROCESS --   --OUTPUT Q--  LENG  (CH)   LENG  (CH)-
CODE    -TRANS-   -AVG-  -MAX-   -AVG-  -MAX-  -AVG-  -MAX-   -AVG-  -MAX-  -AVG   -MAX  -AVG   -MAX
_______ _______   _____  _____   _____  _____  _____  _____   _____  _____  ____   ____  ____   ____

TPCCO     11655      96   414       12    260     54   376       29   165     24     42   255    584
TPCCP      4605     108   857        9    186     68   850       30   162     30     54   235    592
TPCCN      1829     328  1805        8    117    288  1716       31   154    115    206   541    966
TPCCS       169    3391  6788        8    117   3354  6728       28   121     12     22    25     46
```

Figure 52. Fast Path Log Tape Analysis Report

The maximum number of messages that can exist on the structure is the sum of all terminals on all shared-queue IMSs that are capable of entering EMH messages. Only one input or output message needs to be included in the estimation since the terminals are in response mode once a message has been accepted by EMH.

LOCAL-ONLY messages are never placed on the shared-queue structure - they can be ignored for sizing purposes. LOCAL-FIRST messages will not be passed to CQS when they are selected for processing on the local IMS but allowance must be made in structure sizing for the times when they are passed to CQS.

A simplistic way to size the data elements is to use the global count of terminals multiplied by the EMH buffer size.

D.5 Sizing of the Log Structure

CQS uses the MVS system logger to keep track of its activity in case a structure recovery or a CQS restart is needed. The system logger merges a stream of log records written by multiple CQS in a shared-queues group into a single log stream and writes it to the MVS logging structure in the coupling facility. The CQS log records are described in C.4, “CQS Log Records” on page 123.

We cannot advise you on how to size the log structure used by the MVS logger for writing the CQS log streams, but we can offer some thoughts about the amount of additional log data produced by the CQS.

For each message written to the shared queue, three CQS log records are written to the CQS log stream, one for each of the following events:

1. PUT message on shared-queue
2. READ message to process it
3. DELETE message from shared-queue

The DELETE log records are batched, which means they are written to the log stream when several of them are collected, or at least when a structure checkpoint is taken.

Additional log records are written when a message is moved from one queue to the other, or when a system or structure checkpoint is taken. The additional log records and the batched DELETE log record are not examined further, as the number of these records is relatively small compared with the number of log records derived from transaction processing.

To simplify the discussion, we assume that every transaction generates only one output message, which means that we must expect at least four CQS log records for each transaction. The CQS log records are in addition to the existing IMS log. The IMS log contains all the log records known from a non-shared-queues environment.

Following the advice to take structure checkpoints for the message queue structure at low activity time, Telstra decided to take two structure checkpoints per night. The structure checkpoints alternate between the two SRDS data sets. One SRDS contains only one structure checkpoint. CQS log records older than the last two structure checkpoints are no longer usable and are therefore deleted.

The CQS log records are written to a coupling-facility log structure. A log structure is a special form of a list structure. If the log structure fills up to the HIGHOFFLOAD parameter, it is automatically off-loaded to a linear dataset, which is allocated by the MVS logger when needed. In case of a structure rebuild, CQS repopulates the structure from the most recent SRDS and then applies all log records written after this checkpoint. If the last SRDS is not available, CQS uses the previous SRDS and applies all log records written since that structure checkpoint. It would shorten the time required for a structure rebuild to have all log records needed for recovery stored in the log structure, but this is generally impossible in a production environment.

Besides the discussion about the LE/DE ratio and the overhead, you need space to store the pure data. In Telstra today we have a transaction volume of 7 million transactions a day. The average object size for CQS log records is 4096 bytes. At least four CQS log records are written for each transaction (simplified). The space required for all the data elements to store the CQS log records for one day is 7 million * 4 * 4096 bytes, which is 112 GB. There is not enough space on the coupling facility to keep this amount of data in the log structure.

Because the offloading of the log structure does not impact CQS or its Client IMS, the size of the log structure can be defined at any suitable size. There is a lot of information on how to design a log structure in *OS/390 Setting Up a Sysplex*, GC28-1779 (see Chapter 9, “Planning for System Logger Applications”).

There is a minimum size you must allocate, however. There is a fixed portion of 300 KB needed for every CQS log structure if you follow the recommendation of

the OS/390 manual to set the AVGBUFSIZE to 4096 and the MAXBUFSIZE to 65272 using the formula for sizing a list structure in *OS/390 PR/SM Planning Guide*, GA22-7236. Our tests confirmed this value as we could not start CQS with a log structure of 256 KB but we succeeded with a size of 512 KB.

For performance reasons all users of the MVS system logger should make sure that the following APARs are applied to the MVS system:

- OW28207
- OW29849
- OW31383

Appendix E. System Maintenance Procedures

In this appendix, we describe the current IMS system maintenance procedures, the IMS installation philosophy, and the shared-queues maintenance implementation.

E.1 Current Procedures

The current Telstra SMP/E maintenance environment consists of one set of target and distribution zones that reside on the Systems Maintenance partition. System generations for all IMSs are performed on this LPAR and then shipped to the target LPARs for implementation. For development IMSs, the system generation cycle consists of weekly IMS system generations. For production systems, this a three-week cycle.

For large maintenance upgrades such as PUT or ESO upgrades or the application of complex or high-risk PTFs, a new SMP/E environment is cloned from the current environment and the maintenance is then applied to the new environment. Initial testing is performed in the System Programmers′ test IMS using the IBM-supplied IVPs. An implementation schedule is then developed to introduce the new environment through all development and production systems. It typically takes three to four months for the new environment to be introduced into the final production system.

For smaller maintenance items or one-off fixes, the current SMP/E environment is used. The maintenance is applied after the IMS production system generations are complete. This provides a three-week window to test these fixes before the next production system generations are required. The new maintenance is then introduced into the development systems via the weekly system generation cycle and tested before the next IMS production system generation cycle is scheduled to be performed. The process is illustrated in Figure 53 on page 148.

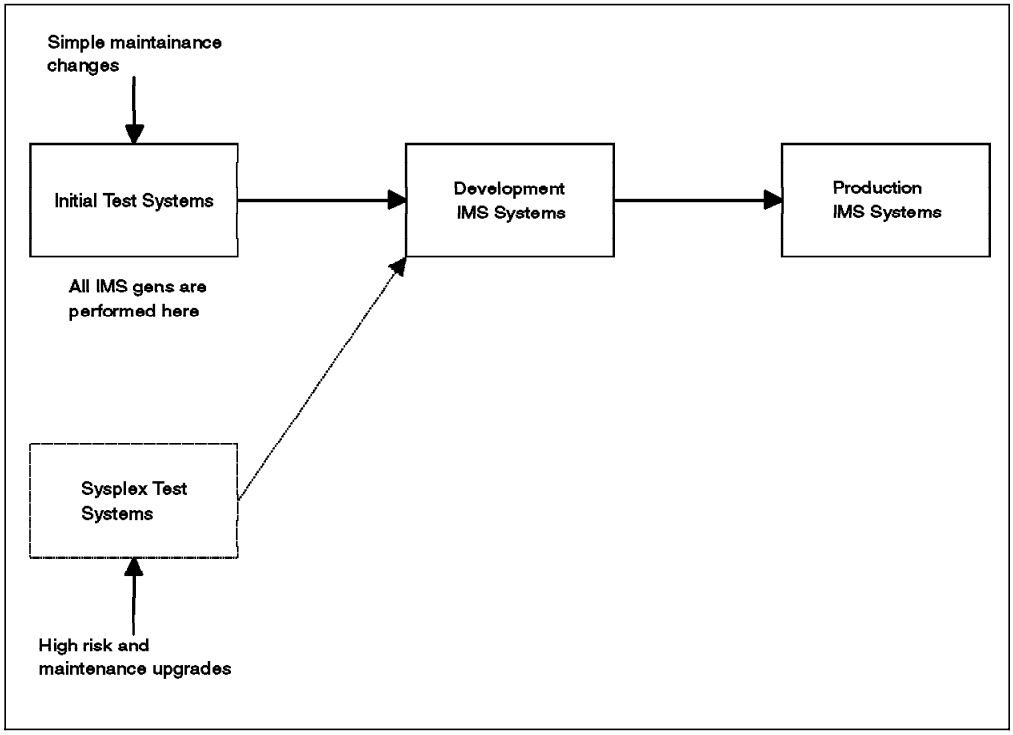


Figure 53. System Maintenance Procedures

Once the maintenance has been tested and there is a high level of confidence that it is reliable, it is SMP/E accepted. This ensures that subsequent maintenance can be applied and restored easily without impacting any prior maintenance.

The IMS system programmers are responsible for deciding the maintenance level of the IMS SMP/E environment. This group also meets with other groups to resolve non-IMS software corequisite requirements such as DB2, IRLM, and DFSMS when necessary.

E.2 IMS Version 6 Installation Philosophy

IMS Version 6 was initially implemented into the IMS System Programmers systems to enable full upgrade and testing procedures to be formulated and documented. Before implementing into the first development systems, the latest maintenance available is applied and IVP testing is reverified against the new maintenance level.

IMS Version 6 is then implemented in Development systems without exploitation of any new features provided by Version 6. This is done to ensure that only minimal change would be experienced by current applications during the Version 6 roll-out. The exploitation of new features provided by IMS Version 6, such as shared message queues, comes later once the initial implementation of Version 6 is stable.

It was decided to have IMS Version 6 at the highest maintenance level possible at the time of implementation into the first development systems. It is expected to be some months before the first production systems are upgraded, by which time any problems should have been discovered in the development regions and

appropriate corrective action taken. This would provide a relatively high maintenance level in production.

E.3 Positioning for Shared-Queue Implementation

It was thought that implementation and testing of the shared message queues feature would probably require additional maintenance to be applied at short notice as required. To facilitate this, it was decided to clone a new SMP/E environment from the existing IMS Version 6 environment (Figure 54). This would provide both stability for the non-shared-queues environment during roll-out and flexibility during implementation and testing in the shared-queues environment.

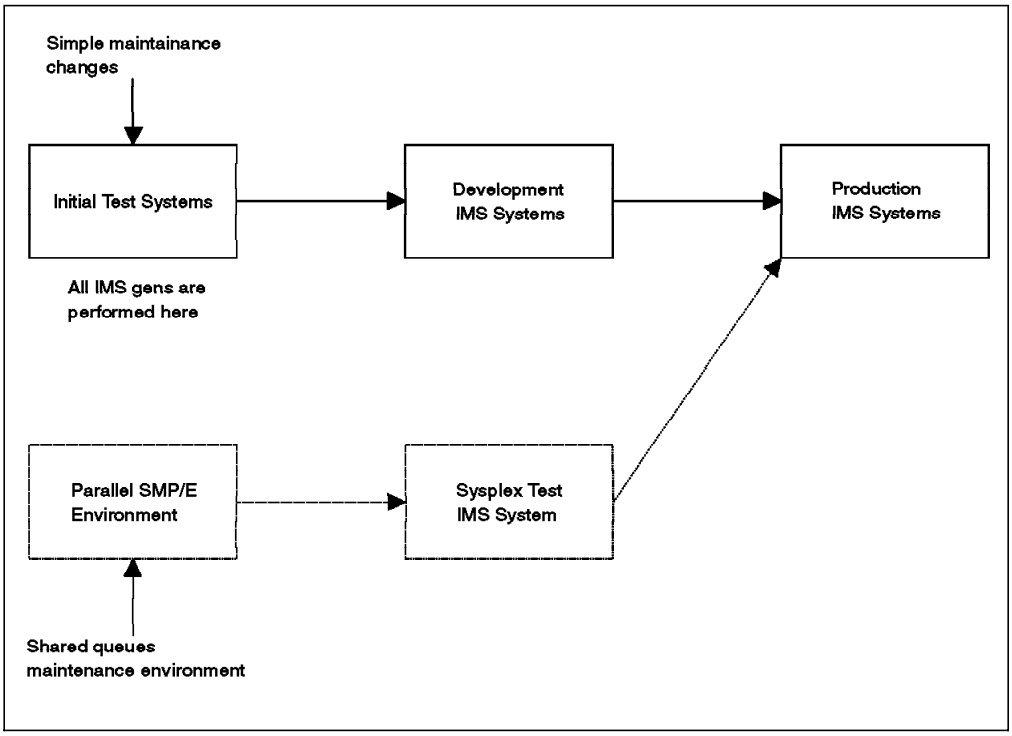


Figure 54. System Maintenance Procedures in a Shared-Queues Environment

Because of the existing high maintenance level of IMS Version 6, very few additional shared-queue related PTFs were identified as being available but not yet applied. A review of these PTFs deemed them unlikely to affect the shared-queues implementation, hence they were not applied.

We planned that the new shared-queues SMP/E environment would later form the basis of the next maintenance upgrade for the IMS Version 6 systems as it could potentially contain several maintenance upgrades by the time the shared-queues feature had been implemented and stabilized in the development regions.

Appendix F. Telstra IMS Shared Queues Project Plan

The Telstra IMS Shared Queues project plan is presented in this appendix. This plan is based on the plan created for the ABSA plan described in block level data sharing There are some assumptions about the environment that have to be understood before reviewing the plan:

- We are moving from one IMS Control Region environment to a Shared-Queues Environment by joining disparate systems connected by MSC. Although transaction definitions are cloned, they do not execute on both IMSs. Databases are not shared. Cloned IMSs are a future option.
- All the activities that are listed here have been executed against development system resources. For example, the initial setup and testing was performed on the IMSs of the IMS system programmers using the IVP sample transactions. Testing for Telstra-specific functionality was performed on the application development stress-test systems.
- Most of the steps listed here would have to be repeated in order to create the production shared-queue environment. However, there are a few steps like teleprocessing network simulator (TPNS) testing that would not be attempted on the final production environment.

1. IMS Shared-Queues Project Plan: Planning Segment

    a. Understand the Current Environment. The tasks are these:

        - Document the core business functions performed by the computing services group in your organization.
        - Map out the current hardware configuration and system software levels in use.
        - List the external links and types of links that connect IMS to external subsystems.
        - List the type of workload that is currently present in the major applications.
        - Prepare a current view of the enterprise computing environment with a view to processors, major application systems, and databases.
        - Identify the types of users that require the most resources.
        - Map out the service-level commitments for systems in your organization to be migrated.
        - Document the on-line and batch schedules for systems in your organization to be migrated.
        - Identify the performance and tuning tools and processes available currently.
        - Document the user modifications currently on your systems.
        - Create a list of the vendor products associated with the current environment.
        - List the databases and applications that currently present affinity status (for example, serial transactions and pseudoserial transactions).

- Document the user exits and user modifications that have hard-coded affinities to particular systems.
- List any IMS log processing programs that review only one system's logs per run.
- Determine if you have the necessary skills in-house to perform the migration of the IMS and MVS components into a shared-queues environment.
- Review and document your change and problem control processes.

b. Draft an overall architectural and component image of the sysplex environment. The tasks are these:

- List the business reasons associated with migrating to a sysplex environment.
- List the technical reasons associated with migrating to a sysplex environment.
- Map the selection of the hardware elements in the sysplex that the IMS environment will use.
- Define the software package elements and levels in the sysplex that the IMS environment will use.
- Map how the network will be connected to the sysplex for use with the IMS environment.
- Identify and define the final MVS/LPAR sysplex configuration:
  - System and LPAR names
  - MVS and SYSNAME, and so on
  - JES2 MAS/NJE configuration
  - MVS mastercatalog and user catalog configuration
  - The GRS/MIM reserves and global reserves expected in the target environment
- Define the migration strategy for the use of processors and LPARs from current to final configurations including interim phase.
- Identify how the work will be directed to processors and LPARs for batch, TSO, IMS and CICS.
- Define the migration strategy for IMS from the current environment to the final, including interim configurations for systems programming test, application development, integration, and production.
- Determine the dependencies that an IMS migration to a shared-queue mode might be subject to: VTAM GR, fully functional base environments including MVS/JES2, TSO/ISPF, program products.
- Identify any complicating factors that might exist at the end of this project for the following workloads: IMS, TSO, CICS, batch. Examples could be the current lack of process for the following situations in environments where cloned applications are running in multiple systems or LPARs:
  - Scheduled shutdowns of a processor or LPAR for hardware or software maintenance

- Unscheduled outages of MVS or its subsystems
- Subsystem maintenance rippled across systems.

List the scheduling and restart factors for applications when the above events occur.

- List the IMS resources that can be cloned (transactions, data sets, and so on).
- Identify the IMS resources that will have affinities to one specific IMS system within the sysplex.
- Identify remote system connections to IMS in the new sysplex configuration.
- Develop a preliminary view of the target environment showing processor, LPAR, coupling facility, major applications, target databases, and so on.
- Identify the security processes for the IMS environment in the sysplex.

c. Plan for running the sysplex in degraded mode. The tasks are these:

- Define what degraded mode means based on the above sysplex architecture design.
- Identify what core business functions must continue to operate when the sysplex is in degraded mode.
- Map how the network would interface with a sysplex running in degraded mode.

d. Develop the sysplex project plan. The tasks are these:

- Obtain necessary executive and financial approvals.
- Ensure that there is an effective process to communicate project status to executive management.
- Assign a project coordinator.
- Identify general Customer and Vendor staff and their responsibilities.
- Select the project management tools to use.
- Define the education required.
- Estimate changes associated with hardware, software, and personnel resource requirements for the planning, preparation, implementation, and ongoing phases of the project.
- Ensure that the necessary resources (weekend migration and testing time and system access) are available and are booked in advance.
- Develop a risk assessment plan and review with all involved functional personnel and management levels as required.
- Develop this project plan, assigning ownership and start and expected completion dates to items.

2. IMS Shared-Queues Project Plan: Preparation Phase

a. Develop naming conventions. The naming conventions must include subsystems, data sets, IMS region job names, CF structures, started task names for IMS regions, CQS procedure and structure names, MVS

logger names, MVS system names (names will be selected later in the process).

b. Items to consider:

- Some name changes will mean application JCL changes.
- Some names will be tied to the MVS system name.
- The staging of name changes in all environments.

c. Document how databases and applications will be accessed and used in the sysplex environment.

- Partitioned or cloned databases
- Partitioned or cloned applications
- Which resources— such as applications, databases, data sets, and terminal network elements— cannot be shared.
- Which applications or databases have to modified to be able to join the sharing environment.

d. Naming Standards. Change naming standards on the systems to sysplex naming standards so that shared and nonshared resources can be identified. (The names used in this project plan are obviously unique to the Telstra environment.)

- Define alias for high-level qualifier IMA1.*
- Define user catalog for IMA1.**
- Define RACF for IMA1.
- Create SYSM.IMA1.RESLIB library.
- Copy generated resource library into SYSM.IMA1.RESLI.B

e. Data Set Sharing. Determine and document what IMS system data sets will be shared or unique. (This list maps out the choices that Telstra made within the scope of what data sets are under user control for shared or unique status.)

- Naming standards for IMS**
  - Shared data sets IMA1.**
  - Nonshared data sets
- Shared data sets - prefixed by IMSV**:
  - DBDLIB
  - FORMAT/A/B
  - PGMLOAD
  - PGMLOAD
  - LOADERS
  - PROCLIB
  - RECON1/2/3
  - REFERRAL
  - TFORMAT
  - USOURCE
  - USOURCEN
  - MATRIX/A/B (prefixed by SYSM. and then IMSV.)
  - MODBLKS/A/B (prefixed by SYSM. and then IMSV.)
  - URESLIB prefixed by SYSM. then IMSV.
  - UPARMLIB

- Nonshared data sets prefixed by IMA1.*, IMA2.* and so on.
  - POLDS01/2/3/4/5
  - SOLDS01/2/3/4/5
  - WADS1/2
  - IMSMON
  - DFSTRA01/2
  - QBLKS
  - SHMSG
  - LGMSG
  - MODSTAT
  - BACKUP.
  - RDS
  - TCFSLIB
  - SYS01/2/3/4/5/6/7/8/9/10/11/12
  - RESLIB

    This will be prefixed by SYSM. and then IMA1. or IMA2, and so on:
  - JOBS
  - JCLLIB
  - ACBLIB/A/B
  - PSBLIB

f. Implement the Data Set Name Changes. The tasks are these:

- Rename IMS system data sets.
- Adjust JCL for system DBA and application jobs.
- Modify naming conventions when necessary for auto operation execs.
- Modify overrides in DFSPBxxx members.
- Change required data set names for audit control and accounting.
- Change data sets in the following procedures in IMSV.PROCLIB:
  - REGBMPNP
  - REGFPP
  - REGFFP
  - REGDLIP
  - REGBMPP
  - INTRDP
- Change all procs using Resource library (run search on IMSV.PROCLIB).
- Change members in library IMA1.JCLLIB:
  - ARCHJCL
  - CAJCL
- Change TCO library and member IV01.TCFSLIB(DFSTCF).
- Ensure that all shared data sets are on shared DASD.
- Build GDG base for files
  - IV01.MTOLOG.MONTH.*
  - DCMON

g. Prepare for Region Name Changes. The tasks are these:

- Change the following members in IV01.JOBS
  - VREG
  - VFF100A
  - ...
  - VFF110A
- Delete old dynamic allocations from SYSM.IMSV.URESLIB
  - DFSWADS1/2/3
  - DFSOLP01 - 05
  - DFSOLS01 - 05
  - DFSTRA01/02
- Run dynamic allocation to SYSM.IMA1.RESLIB
- Automation changes: Change region names, now 8 characters
- Operational changes:
  - Communicate with Operations on new region names
  - Update scheduler with new naming standards
- IMS Department Procedure Changes - Generation Procedures:
  - Copy new RESLIB
  - Setup dynamic allocation with every generation.
  - Communicate changes to all departments.
- DBA Department Changes: Add &IMSID parm into procedures.

h. Started Task Name Changes. The tasks to be planned are these:
- Change the following names:
  - IMSV51 to V01IMS51
  - DBRCV to V01DBRC
  - DLISASV to V01DLIS
- Add the following to the Started Task Table:
  - V01DBRC
  - V01IMS51
  - V01DLIS
- Setup procedures in PROCLIB for:
  - V01DBRC
  - V01IMS51
  - V01DLIS

i. Implement procedure changes for V01DBRC, V01IMS51, V01DLIS. The tasks are these:
- Change in IMSV.PROCLIB(DFSPBxxx):
  - DBRCNM= DBRCV to V01DBRC
  - DLINM = DLISASV to V01DLIS
- Start V01IMS51.IMSV.
- Communicate changes to operations group.
- Communicate changes to DBA/IMS departments.

j. Review and update JCL where required. This includes general support JCL, such as DBRC skeletal JCL, statistics JOB JCL, and application JCL.

k. Formulate the manner in which security will be implemented within the sysplex environment.

l. Review all the user modifications and exits that will be running in the sysplex environment for compatibility.

m. Design a process in which remote systems are connected and interface into the sysplex.

n. Produce the necessary plan to connect the IMS network into the sysplex environment.

o. Design the environment for the coupling facility. The tasks are these:

- Determine CQS Structure size.
- Determine MVS logger structure characteristics.
- Determine where the CQS and MVS Logger structures will be placed.
- Set up procedures to change CF structure sizes and placements.
- Develop a set of procedures for the recovery of CF structures.

p. Examine current operational procedures and update as necessary. The tasks are these:

- Design MVS, IMS, and CQS startup and shutdown steps.
- Identify the process to recover should systems and subsystems fail.
- Describe how message queues are managed or recovered currently.
- Determine the manner in which jobs will be scheduled in the sysplex environment - and examine the use of unique initiator classes for each system.

q. Examine and modify installation support procedures such as these:

- IMS systems generation in sysplex environments
- Maintenance of multiple queue-sharing IMS′s in a sysplex.
- Security maintenance for the sysplex resources
- On-line change utilities
- Management of the logs from multiple data-sharing systems.
- Database reorganization and recovery in sysplex environments (Review naming standards for CA and Image copy data set.)
- Application changes and maintenance in a cloned and noncloned environment

r. Develop plan to enable data sharing:

- Schedule time for a workshop with all interested parties present.
- Review IMS definition parameters associated with shared-queues.
- Understand changes to JCL, PARM imformation and procedures changes
- Investigate the most effective test bed for application and systems software in the sysplex environment.
- Examine scope of work to update operating procedures.
- Understand the requirements for automation changes.

s. Plan IMS startup JCL changes if any

t. End-of-Day JCL changes if any

u. Review potential performance issues in the sysplex environment associated with above conversion of database structures.

v. Determine how to implement processing in degraded modes. (This step was not part of the Telstra project plan specifically but is included for completeness.) The tasks are these:

- Finalize the choice of the core business functions that must be available regardless of the loss of processing capacity.
- Develop procedures associated with the loss of various sysplex components to restore functionally for the above-defined core business functions

w. For the stress testing phase, examine the TPNS requirements.

x. Examine existing disaster recovery plans and rework them to fit into the sysplex environment.

y. Examine automated operation programs and procedures and modify them as much as possible during the Preparation Segment of the project based on the systems, operational changes, and application changes that have occurred during this period.

z. Ensure that all necessary diagnostic procedures and facilities are in place.

aa. Plan Education and Documentation.

- Provide documentation of pending changes to Applications staff, Operations staff and end-users. Present all parties involved with the details of the projected changes and what effect they will have.

3. IMS Shared-Queues Project Plan: Implementation

a. Schedule daily half-hour morning meetings with all affected staff at this point within the project.

b. Create a clone of the IMS system. The tasks are these:

- Develop the series of unique IMS system data sets for the clone environment.
- Sysgen the clone environment.

Not required at Telstra since they are disparate systems

- Create procedures and execution time parameters for clone system.
- Develop the necessary support procedures and DBRC skeletal JCL for cloned or unique data sets and functions.

c. Implement Multiple Systems Coupling between IV01 and IV02. The tasks are these:

- Identify list of transactions that use facilities from other vendors, which might place them in affinity status
- Define MSC Links for IMA1 (MSLINK, MSNAME, MSPLINK macros).
- Define MSC links for IMA2 (MSLINK, MSNAME, MSPLINK macros).
- Setup generation procedures.

d. Establish Operating Procedures for MSC. The tasks are these:

- Automate /RSTART LINK command (TCO).

- Link termination /PSTOP LINK.
- Test commands to be used in MSC environment. See *IMS/ESA Administration Guide: System*, SC26-8730.
- Test message recovery when IMS goes down and is then restarted.
- Test events when a transaction is stopped. Establish automation procedures to ensure excessive queuing does not develop and so on.
- Test application program abends: sending of error messages to other systems.

e. Establish Monitoring Procedures for Multiple System Environment. The tasks are these:

- Use IMSPA reports to monitor the two systems.
- For SLR monitoring, set up procedures to merge logs.

f. Test Sysplex system to ensure functionality of all components. The tasks are these:

- Test /STA DC.
- Test all IMS commands.
- Run log transaction analysis utility (DFSILTA0).
- Run Fast-Path log analysis (DBFULTA0).
- Run statistical analysis utility DFSISTS0.
- Test off-line dump formatter.
- ERE testing.
- Test user modifications.
- Test MFS.
- Test interactive dump formatter.
- Test RACF signon.
- Test load list.
- Test IMS Monitor.
- Test OLDS recovery.
- Test SLDS recovery.
- Test archives.
- MTO spool.
- APPC testing.
- Test DB2.
- Test change accumulation.
- Exercise database updates.
- Test all DBRC commands.
- Test DFSDDLT0 to all database types.
- On-line image copy.
- Concurrent image copy.
- DEDB reorganization.
- High Speed DEDB reorganization (DBFUHDR0)
- DBTOOLs DEDB Unload or Reload
- Test PSBGEN.
- Test DBD generation, all types.
- Test ACB generation.
- Test on-line change.
- Dynamic allocation.
- Test DEDB initialization.
- Test DEDB SDEP scan.
- Test Nodemon
- Test automation.

- Concurrent copy (Use Global commands to DBR databases enure proper coordination.)
- Hardware compression.
- HSSP
- Test all global commands.
- Test other IBM products:
  - DB2
  - SLR
  - IMS Performance Analysis and Reporting utility (IMSPARS)
- Check transaction macro for MAXRGN= and SERIAL=YES specifications to ensure same effect in data-sharing if necessary
- Check application macro for SCHDTYP=SERIAL as above
- Coordinate the /MODIFY commands for controlling on-line change:
  - Stop affected database in all online systems using the database before initiating a sequence of /MODIFY commands.
  - If ACBLIB, FORMAT, or MODBLKS are shared, make sure all systems use the same active library.
  - Ensure that DFSUOCU0 updates only inactive libraries
- Establish procedures to protect databases from other system access during database reorganizations. (Use /DBR GLOBAL, or DBRC CHANGE.DB NOAUTH, or MTO procedure to stop allocations that would be active against the database.)
- Lockmax option: implement to reduce chance of runaway programs.
- Establish procedures to protect databases from update access by other subsystems while an online database image copy is in process (DFSUICP0).
- Establish procedures for change accumulation, using the merge function of DFSUCUM0.
- Establish procedures for database forward recovery. System logs for both systems need to be merged first, using the CA utility DFSUCUM0. Then the database recovery utility can be run.
- INIT STATUSGROUPB to cater for BA, BB status codes
- PROCOPTs in PSBs (update when only read is needed).
- Test OEM products

g. Establish Monitoring Procedures for Sysplex Environment. The tasks are these:

- Automate use of following MVS commands so information is displayed on the log at regular intervals:

```
DISPLAY XCF,STRUCTURE
DISPLAY XCF,STRUCTURE,STRNAME=
```

4. IMS Shared-Queues Project Plan: Performance

a. General monitoring

b. Testing of Failure Scenarios. The tasks are these:

- Test on-line transaction and BMP abends.
- Test IMS failure, such as IV01:
- Test cold start after IMS subsystem abend. Make sure that batch backouts are done for each inflight PSB

- Test MVS failure:
  - Note the effect on the other IMS U3303 abends for retained locks.
  - Use /ERE for the failed IMS.
- Test CF failure (two CFs required to test automatic rebuild):
  - Identify structure effects.
    - Rebuild attempted automatically.
    - When rebuild completes,
    - IMS reconnects to structure.

c. RACF, Test /MODIFY PREPARE RACF

d. TPNS-driven stress test with production level system images:
   - Determine requirements.
   - Identify transactions.
   - Build scripts.
   - Build tables for variable data.
   - Build TEST.
   - Do simulation.
   - Examine simulation.
   - Produce report with findings.
   - TPNS Test:
     - Take down IMSP IP02 (P01IMS51, P02IMS51).
     - Implement generation.
     - Start P01IRLM.
     - Start V01IMS51 and check if generated item is in
     - /SWI OLDS.
     - Start TPNS test with 20 terminals.
     - Start TPNS with 50 terminals.
     - Issue a /CHECKPOINT.
     - Run an IMS monitor.
     - Use /DIS POOL ALL regularly during the interval
     - At end of interval /SWI OLDS.
     - Run a DBFUALT0 Fast Path Log Analysis.

e. Test performance of production system in shared-queue mode and tune where necessary.

Appendix G. Special Notices

This publication is intended to help IMS systems programmers, database and system designers, and applications designers and programmers plan and implement IMS/ESA shared queues based on a case study of the experience of Telstra Australia. The information in this publication is not intended as the specification of any programming interfaces that are provided by IMS/ESA, CQS, or OS/390. See the PUBLICATIONS section of the IBM Programming Announcement for IMS/ESA and OS/390 for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

The following document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain 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.

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

BookManager
CICS
CICS/ESA
CUA
DB2
DFSMS
DFSMS/MVS
DFSMSdfp
DFSMSdss
IBM
IMS
IMS/ESA
MQ
MQSeries
MVS/ESA
OS/390
Parallel Sysplex
PR/SM
RACF
RMF
SP
System/390
VTAM

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Epilog/IMS and Omegamon/IMS are trademarks of Candle Corp.

SAS is a trademark of the SAS Institute.

IMSPARS is a trademark of IBM Corp.

Astex is a trademark of Computer Associates.

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

Appendix H. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

H.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 167.

- *IMS/ESA Version 6 Shared Queues*, SG24-5088
- *IMS/ESA Version 6 Guide*, SG24-2228
- *IMS/ESA Data Sharing in a Parallel Sysplex*, SG24-4303
- *IMS/ESA Sysplex Data Sharing: An Implementation Case Study*, SG24-4831
- *IMS/ESA Multiple Systems Coupling in a Parallel Sysplex*, SG24-4750
- *OS/390 MVS Parallel Sysplex Capacity Planning*, SG24-4680
- *Parallel Sysplex Coupling Facility Online Monitor: Installation and Users Guide*, SG24-5153

H.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

H.3 Other Publications

These publications are also relevant as further information sources:

- *IMS/ESA Common Queue Server Guide and Reference*, LY37-3730 (Available only to licensed customers)
- *IMS/ESA Customization Guide*, SC26-8732
- *IMS/ESA Installation Volume 2: System Definition and Tailoring*, GC26-8737
- *IMS/ESA Operator's Reference*, SC26-8742
- *IMS/ESA Messages and Codes*, GC26-8739
- *IMS/ESA Operations Guide*, SC26-8741
- *IMS/ESA Performance Analyzer User Guide*, SC26-9088
- *OS/390 MVS Programming: Assembler Services Reference*, GC28-1910
- *OS/390 Setting Up a Sysplex*, GC28-1779

- *OS/390 MVS Programming: Assembler Services Guide*, GC28-1762
- *OS/390 MVS Programming: Sysplex Services Guide*, GC28-1771
- *OS/390 MVS System Commands*, GC28-1781
- *OS/390 PR/SM Planning Guide*, GA22-7236
- *DFSMS/MVS DFSMSdss Storage Administration Reference*, SC26-4929

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
  ```

  For a list of product area specialists in the ITSO: type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Web Site on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.htm). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1)001-408-256-5422 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.htm). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Glossary

back-end IMS. Any IMS, other than the front-end IMS, that is in the same shared-queues group as the front-end IMS.

base primitive environment (BPE). A system-service layer component of IMS that provides a common set of system services.

checkpoint data set. A local data set that contains CQS system checkpoint information with respect to a group of shared queues.

client. A subsystem that uses the CQS. For example, IMS is a client of the CQS.

cold queue. A CQS private queue type that contains indoubt data objects for a client or a CQS that was cold-started.

common queue server (CQS). A server that receives, maintains, and distributes data objects from a shared queue on a coupling facility list structure for its clients.

coupling facility (CF). A special logical partition that provides high-speed caching, list processing, and locking functions in a sysplex.

CQS restart. Process by which the CQS starts up; either a cold start or a warm start. During a CQS warm start, the CQS environment is restored to the state it was in when CQS terminated. During a CQS cold start, the CQS environment is not restored to a previous state; it is reinitialized.

data object. A piece of client data that is placed on a shared queue through a CQS request.

front-end IMS. The IMS that the terminal was connected to when the initiating conversational transaction was entered.

list structure. A coupling facility data structure that consists of an array of lists and an optional lock table.

log token. A token that identifies a particular log record in the MVS log stream that is used to locate that log record.

master CQS. The CQS that coordinates a sysplexwide task. The other CQSs sharing in the task are participants. If the master CQS fails for any reason, another CQS takes over the role of master and either continues or aborts the task.

overflow structure. An MVS coupling facility list structure that contains shared queues when the primary structure reaches an installation-specified overflow threshold. The overflow structure is optional.

primary structure. An MVS coupling facility list structure that contains shared queues.

queue. A collection of data objects with the same name.

queue name. A 16-byte name used to identify the queue that contains a data object on the shared queues. Multiple data objects may be identified by the same name.

queue type. A logical grouping of queues, either by CQS (private queue types) or the client (client queue types). An example of a CQS private queue type is the lock queue type, in which the CQS groups all locked data objects. An example of a CQS client queue type is the IMS transaction ready queue, in which IMS groups all transaction queues that are ready to be processed.

shared queue. A coupling facility structure, managed by the CQS, that contains data objects that are available to all CQS clients in the sysplex.

structure recovery data set (SRDS). Shared data sets that contain structure checkpoint information for shared queues on a structure pair. There are two SRDSs per structure pair.

sysplex. A set of MVS systems that communicate with each other through hardware components and software services.

unit of work (UOW). To CQS, a client-defined grouping of data objects.

List of Abbreviations

ADDR	address
AOI	Automated Operator Interface
APPC	advanced program-to-program communication
ARM	MVS/ESA Automatic Restart Manager
ASID	address space identifier
ASTEX	Automated Storage Expert
BMP	batch message processing, region
BPE	base primitive environment
BPE	base primitive environment
BTS	batch terminal simulator
CA	change accumulation
CCB	conversational control block
CF	coupling facility
CFRM	Coupling Facility Resource Manager
CHKPT	checkpoint
CNT	communication name table
CNTL	control
CQS	Common Queue Server
CSECT	control section
CUA	common user access
DASD	direct access storage device
DB	database
DB/DC	database/data communications
DBA	database administrator
DBRC	database recovery control
DBT	database tools (software aids for IMS DB)
DCB	data set control block
DD	data set definition
DEDB	data entry database
DEQ	dequeue
DFDSS	data facility data set services (IBM software product)
DL/I	data language 1
DLI	data language interface
DMB	data management block
DRRN	device relative record number
DSECT	dummy control section
EMC	event monotor control block
EMH	expedited message handling
ENQ	enquiry
ERE	emergency restart
ESDS	entry sequenced data set (VSAM)
ESO	extended support offering
EXEC	execute/execution
FE	front end
FF	full function
FP	Fast Path
GDG	generation data group
GEN	generate/generator/generation
GRS	global resource serialization
GU	get unique
HW	hardware
I/O	input/output
IC	image copy
ID	identification/identifier
IDCAMS	the program name for access method services
IFP	IMS Past Path
IGSA	IBM Global Services Australia
IMS	information management system
IMS/ESA	Information Management System/Enterprise Systems Architecture
IMSPA	IMS performance analysis
IMSPARS	IMS performance analysis and reporting system
INIT	initialize/initial/initiate
IO-PCB	input/output program communication block
IPCS	interactive problem control system
IPL	initial program load
IRLM	IMS resource lock manager
ISC	intersystem communication

ISPF interactive system productivity facility

ISRT insert

ITSO International Technical Support Organization

IVP installation verification procedure/program

JCL job control language

KSDS key sequenced data set

LE list entry

LPAR logically partitioned mode

LTERM logical terminal

MPP message processing program

MPR message processing region

MRQ IMS Message Requeuer

MSC multiple systems coupling

MSDB main storage data base

MSG message

MTO master terminal operator

MVS multiple virtual storage

MVS/ESA multiple virtual storage/enterprise systems architecture

NRE normal restart

OLDS on-line log data sets

OSAM overflow sequential access method

OTMA Open Transaction Manager Access

PA performance analyzer

PDB performance database

PMG Postmaster-General's Department (Australia)

PPT program properties table

PR/SM processor resource/systems manager

PROCLIB procedure library

PSB program specification block

PTF program temporary fix

PUT put command

Q queue

QCB queue control block

QCT queue control table

QMGR queue manager

RACF resource access control facility

RECON recovery control (data set)

RMF resource measurement facility

RMT remote

RNL resource name lists

SAS Statistical Analysis System (vendor product, SAS Institute Inc., Cary, NC, USA)

SFM sysplex failure management

SLDS system log data set

SLR service level reporter (IBM program product)

SMB scheduler message block

SMF system management facility

SMP/E system modification program/extended

SPA scratchpad area

SQ shared queue

SQE shared-queues environment

SQG shared-queues group

SRDS structure restart data sets

STC started task control

SUBSYS subsystem

SVC supervisor call instruction

SYS system

SYSPLEX systems complex

TM transaction manager

TPNS teleprocessing network simulator

TRAN transaction

TSO/E time sharing option extensions

UOW unit of work

VSAM virtual storage access method

VTAM virtual telecommunications access method

WFI waiting for input

WLR workload router

WTOR write to operator with reply

XCF cross-system coupling facility

Index

D

E

F

G

H

I

L

M

N

O

P

Q

U

V

W

X

ITSO Redbook Evaluation

IMS/ESA Shared Queues: A Planning Guide
SG24-5257-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@vnet.ibm.com

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

SG24-5257-00
Printed in the U.S.A.