# MA1I: MQSeries CICS bridge playback

Version 1.0 **MA1I: MQSeries CICS bridge playback**

**User Guide**

Roy Saxton
IBM UK Laboratories Ltd.
Hursley Park

28th February, 2003

Document Number MA1I

| **Take Note!** |
| --- |
| Before using this User's Guide and the product it supports, be sure to read the general information under "Notices". |

**First Edition, February 2003**

This edition applies to Version 1.0 of MQSeries CICS bridge playback, and to all subsequent releases and modifications until otherwise indicated in new editions.

A form for reader's comments is provided at the back of this publication. If the form has been removed, address your comments to:

```
IBM United Kingdom Laboratories
Transaction Systems Marketing Support (MP207)
Hursley Park
Hursley
Hampshire, SO21 2JN, England
```

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

---

# Table of Contents

---

# Notices.

*(Ref #1.)*

**The following paragraph does not apply in any country where such provisions are inconsistent with local law.**

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

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 licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe any of the intellectual property rights may be used instead of the IBM product. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

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, New York 10594, USA.

The information contained in this document has not be submitted to any formal IBM test and is distributed AS IS. The use of the 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 has 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. *(Ref #2.)*

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

WebSphere
CICS
CICS/ESA
IBM
MVS
ESA
OS/390
z/OS

---

# About this User's Guide

This book tells you how to:

- Install MQSeries CICS bridge playback
- Convert messages recorded under CICS 3270 Bridge Passthrough (SupportPac CA1E) into a format suitable for replaying to the CICS 3270 bridge using playback
- Create records that can be replayed to the CICS DPL bridge using playback
- Replay prerecorded dialogs with the CICS bridge and analyze the results.

---

## Who this book is for

This book is intended for anyone who plans to use the MQSeries CICS bridge and wants to:

1. Verify that the MQSeries CICS bridge and the resources it uses are correctly installed and configured
2. Investigate how the bridge responds to exceptional conditions
3. Confirm that MQSeries CICS bridge security authorizes access to protected resources as required.

---

## Prerequisites

MQSeries CICS bridge playback requires the following, or later, products:

- MQSeries for OS/390 Version 2 Release 1
- CICS Transaction Server for OS/390 Version 1 Release 3
- OS/390 Version 2.5 and LE Version 2.5

To record CICS bridge dialogs using 3270 transactions, this SupportPac also requires:

- SupportPac CA1E, CICS 3270 Bridge Passthrough

---

## Useful Publications

For information about the MQSeries CICS bridge:

- *MQSeries for OS/390 Version 5 Release 2 System Administration Guide SC34-5652*

- *WebSphere MQ for z/OS Version 5 Release 3 System Setup Guide SC34-6052*

For information about the CICS 3270 bridge:

- *CICS External Interfaces Guide SC33-1944*

For information about CICS 3270 Bridge Passthrough:

- *CICS 3270 Bridge Passthrough User's Guide*

---

# Introduction

MQSeries CICS bridge playback allows you to replay prerecorded dialogs with the MQSeries CICS bridge. It can be used to assist in MQSeries CICS bridge installation verification testing, and it allows you to investigate how the bridge will respond to exceptional conditions. It can also be used to verify bridge security settings, for example before putting an application into production.

The SupportPac contains a program that can create recorded dialogs, MBRRCRD, and a program that can play the dialogs back, MBRPLAY. It also contains a small number of ready made dialogs that can be used immediately.

MBRRCRD can create dialogs for both the CICS 3270 bridge and the CICS DPL bridge. If you want to create dialogs for the CICS 3270 bridge, you need to install CICS SupportPac CA1E, CICS 3270 Bridge Passthrough. You can use Passthrough to capture message flows that are to be converted by MBRRCRD into the format required by MBRPLAY. If you want to create dialogs for the CICS DPL bridge, all you need as input to MBRRCRD is a file containing the name of a DPL program and the commarea data that is to be sent.

MBRPLAY sends inbound messages specified in the prerecorded dialogs to the bridge, and receives outbound messages from the bridge. The inbound and outbound message may be logged, and the outbound messages analyzed for exceptional conditions.

---

# Installation

This chapter tells you how to:

- Download the playback SupportPac from the Web
- Unpack the ZIP file
- Transfer the files to a TSO system
- Unload the files into partitioned datasets ready for use.

---

## Download

1. Go to URL:

   http://www.ibm.com/software/mqseries/txppacs/ma1i.html

2. Download ma1i.zip in binary format.

---

## Unpack the ZIP file

Use **INFOZIP's** UNZIP to unpack the **ma1i.zip** file.

This produces the following files:

| | |
|---|---|
| **license2.txt** | License file |
| **ma1i.dat** | An MVS partitioned dataset containing Data files |
| **ma1i.def** | An MVS partitioned dataset containing resource definitions |
| **ma1i.jcl** | An MVS partitioned dataset containing JCL |
| **ma1i.loa** | An MVS partitioned dataset containing executables |
| **ma1i.log** | Transient data destination |
| **ma1i.pdf** | This User's Guide in Adobe PDF format |

---

## Transfer to TSO

The library files need to be transferred to the destination TSO system as sequential binary files with a record format of FB 80. Use one of the following methods to accomplish this:

- Use the 'Send Files to Host' option under the Transfer menu item of Personal Communications, specifying:

  ```
  PC File              ma1i.dat etc
  Host File            ma1i.dataseq etc
  Transfer Type      loadlib
  ```

  The Transfer type of **loadlib** may need to be correctly setup. To do this, use the "Setup.Define Transfer Types" option under the Transfer menu item and create the **loadlib** type with the Ascii, CRLF and Append checkboxes all unselected, the **Fixed** radio button selected and the **LRECL** set to **80**

- Use the ftp commands below, ensuring that the BINARY option is set:

  ```
  site fixrecfm 80 (optional)
  ```

```
put ma1i.dat ma1i.dataseq

put ma1i.def ma1i.defsseq

put ma1i.jcl ma1i.jclseq

put ma1i.loa ma1i.loadseq

put ma1i.log ma1i.brlogseq
```

---

## Create libraries

On TSO, issue the following commands to unload these sequential files into TSO partitioned datasets:

- **receive indsname(ma1i.dataseq)**

  when prompted for a filename, reply

  `dsn(ma1i.data)`

- **receive indsname(ma1i.defsseq)**

  when prompted for a filename, reply

  `dsn(ma1i.defs)`

- **receive indsname(ma1i.jclseq)**

  when prompted for a filename, reply

  `dsn(ma1i.jcl)`

- **receive indsname(ma1i.loadseq)**

  when prompted for a filename, reply

  `dsn(ma1i.load)`

- **receive indsname(ma1i.brlogseq)**

  when prompted for a filename, reply

  `dsn(ma1i.brlog)`

Confirm that the datasets other than ma1i.brlog (which is not a PDS) have been unloaded correctly by browsing the PDS and ensuring the appropriate members exist as shown in the following table.

| DATASET | MA1I.DATA | MA1I.DEFS | MA1I.JCL | MA1I.LOAD |
|---|---|---|---|---|
| | BADVECP | MBRCIDEF | MBRCIDEF | DPLABEND |

| | | | | |
|---|---|---|---|---|
| | BADVECVR | MBRMQDEF | MBRMQDEF | DPLTARGT |
| | CEMTINQP | | MBRPLAY | MBRRCRD |
| MEMBERS | CEMTINQR | | MBRPLAYX | MBRPLAY |
| | DPLABNDP | | MBRRCRD | |
| | DPLABNDR | | | |
| | DPL04 | | | |
| | DPL04P | | | |
| | DPL04R | | | |
| | FILEA | | | |
| | FILEAP | | | |
| | FILEAR | | | |
| | RUNORDER | | | |

# Preparing to use playback

Before you can use MQSeries CICS bridge playback, the following preparation is required:

1. Run the MQ resource definition utility CSQUTIL, using member MBRMQDEF of library MA1I.DEFS as input. Sample JCL is provided in member MBRMQDEF of library MA1I.JCL to help you do this. Be sure to make the necessary substitutions in MA1I.JCL(MBRMQDEF) before submitting the job.

   The MQ resources that are defined are:

   **Queues**
   ```
   CICS.BRIDGE.REQUEST.LOCAL
   CICS.BRIDGE.REQUEST.IDENTIFY
   CICS.BRIDGE.REQUEST.VERIFY_UOW
   CICS.BRIDGE.REQUEST.VERIFY_ALL
   STORE.CLOCK.VALUE
   ```

   **Processes**
   ```
   CICS_BRIDGE_AUTH_LOCAL
   CICS_BRIDGE_AUTH_IDENTIFY
   CICS_BRIDGE_AUTH_VERIFY_UOW
   CICS_BRIDGE_AUTH_VERIFY_ALL
   ```

2. Ensure that the queue manager that you will use has a dead letter queue defined for it.

3. Run the CICS resource definition utility DFHCSDUP, using member MBRCIDEF of library MA1I.DEFS as input. Sample JCL is provided in member MBRCIDEF of library MA1I.JCL to help you do this. Be sure to make the necessary substitutions in MA1I.DEFS(MBRCIDEF) and MA1I.JCL(MBRCIDEF) before submitting the job.

   The CICS resources that are defined are:

   **Transactions**

   **Programs**
   ```
   DPLABEND
   DPLTARGT
   ```

   **TD Queue**
   ```
   BRLG
   ```

4. Add the group MA1I to your startup group list, or use CEDA to install the group dynamically.

5. Add library MA1I.LOAD to the CICS DFHRPL concatenation

6. Add the following DD statement to your CICS job and substitute the high level qualifier of the library where you installed SupportPac MA1I for `++MA1IQUAL++`:

   ```
   //BRLOG DSN=++MA1IQUAL++.BRLOG,DISP=SHR
   ```

   The substitution is likely to be of the form `<TSOID>.MA1I`, where `<TSOID>` is the id under which you received the XMIT format datasets.

7. Make sure that the CICS INITPARM parameter specifies an initiation queue, since MQSeries CICS bridge playback uses triggering to start bridge monitors with specific characteristics. The MQ CICS trigger monitor CKTI must also be running when playback is used.

8. If you want to follow the procedure described in "Getting started", compile and link the programs in the C language version of the FILEA sample application. You need to install group DFH$DFLA to make the FILEA resources available in your CICS region.

---

# Getting started

This chapter provides a basic practical introduction to the features of MQSeries CICS bridge playback:

- The playback program itself, MBRPLAY
- The playback recording program, MBRRCRD

---

## Using the playback program MBRPLAY

The playback program MBRPLAY may be run using either of the supplied JCL members MA1I.JCL (MBRPLAY) or MA1I.JCL(MBRPLAYX):

1. JCL member MBRPLAY invokes program MBRPLAY with a set of parameters that cause it to log all of the messages that flow in a dialog and, optionally, to analyze the results obtained in the log.

2. JCL member MBRPLAYX invokes program MBRPLAY with a set of parameters that enable it to run without logging the message flows.

### Invoking playback program MBRPLAY with logging

To invoke program MBRPLAY in a way that causes it to log the messages that flow in a dialog, you need to use JCL member MBRPLAY. Before submitting the job, it is necessary to make a number of

substitutions in the JCL according to the instructions in the comments. Many of the substitutions are for high level qualifiers of datasets and are self-explanatory. This JCL invokes program MBRPLAY with 3 startup parameters, however, and you need to make substitutions for these as follows:

**++P1++** Replace this with the name of the queue manager that is connected to the CICS region where the bridge is to run.

**++P2++** Replace this with the high level qualifier for the dataset where logging records will be written. The userid under which the MBRPLAY batch job runs must have authority to create datasets with the high level qualifier that you specify. Datasets for logging do not need to be preallocated.

**++P3++** This term needs to be substituted with either Y (for YES) or N (for NO). If you specify Y, the log records that are written will be analyzed after the run is complete. For now, specify Y.

Notice that DDNAME B3270RUN addresses a member called RUNORDER in library MQ1I.DATA. Open another ISPF session and edit member RUNORDER. If the SupportPac has just been installed, it will contain the single record FILEA. If this is not the case, change whatever is there to FILEA and save the file.

Make sure that the queue manager is available and connected to the CICS region where the bridge will run, and check that the MQ CICS trigger monitor CKTI is running. Submit the job, and program MBRPLAY should replay a dialog that was created from message flows obtained by running the FILEA application under CICS 3270 Bridge Passthrough. The necessary playback records and a runtime parameter file are in the libraries addressed by DDNAMEs B3270PLA and B3270PRM, respectively.

If the job finishes with a non-zero return code, examine the SYSOUT dataset in the output from the job. This should identify the cause of the problem and give sufficient diagnostic information for you to fix it.

If the job finishes with a return code of zero, open an ISPF session at option 3.4. Type in the high level qualifier that you specified as parameter 2 (`++P2++` substitution) in the EXEC PGM=MBRPLAY statement and press ENTER. You should see two datasets listed with identifiers similar to the following:

```
<++P2++>.D23SEP02.T095252.LOGDINAC
<++P2++>.D23SEP02.T095252.SUMDINAC
```

Where `<++P2++>` is the value you specified as parameter 2. Note that the names will not be identical to these. The last 3 qualifiers will show the date and time that your log file was created, and 'LOG' or 'SUM' followed by 5 characters.

If you browse the file whose last qualifier is of the form `**.LOG*`, you will see that the first screen appears similar to the following:

```
+-------------------------------------------------------------------------------+
|-------------------------------------------------------------------------------|
|+++ Mon Sep 23 09:52:52 2002       Test 0001    Test Instance    FILEA          |
|                                                                               |
|* This MQSeries CICS bridge playback instance was created from a message       |
|* set initially recorded under SupportPac CA1E, CICS Bridge Passthrough.       |
|*                                                                              |
|* The message set was obtained by running transactions DMNU and DADD           |
|* of CICS sample application FILEA under Passthrough.  The exchange of         |
|* messages should complete normally with no errors when replayed using         |
|* MQSeries CICS bridge playback.                                               |
|*                                                                              |
```

```
|MD  ............................MQCICS  ........CSQ VRS2         §ä..ç¼s. (é.+áï¬|
|MD  ............................4MQCICS  ........CSQ VRS2         §ä..ç¼s.CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2         §ä..ßÖ@. (é.+áï¬|
|MD  ............................4MQCICS  ........CSQ VRS2         §ä..ßÖ@.CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2         §ä..¬©Ⅱ .CSQ VRS2|
|MD  ............................4MQCICS  ........CSQ VRS2         §ä..ßÖ@.CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2         §ä..ÀⅡ I. (é.+áï¬|
|MD  ............................4MQCICS  ........CSQ VRS2         §ä..ÀⅡ I.CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2         §ä..Ñ¤¦.CSQ VRS2|
|MD  ............................4MQCICS  ........CSQ VRS2         §ä..ÀⅡ I.CSQ VRS2|
|                                                                                  |
|No message available on SYSTEM.DEAD.LETTER.QUEUE                                  |
|                                                                                  |
|                                                                                  |
|Processing completed normally for Test Instance FILEA                             |
|                                                                                  |
|Processing complete for all tests                                                 |
+----------------------------------------------------------------------------------+
```

This is a log of the messages that flowed when the dialog was replayed, showing all of the message data including the MD structures. The messages are alternately inbound to the bridge and outbound from the bridge. If you scroll to the right, you will see a lot more binary data belonging to each of the messages. You will be able to identify the CICS Information Header structure (eyecatcher CIH) and, for some of the messages, bridge vector structures that are characterized by the strings 1802I, 1802O and 1804O. If you are an experienced user of the CICS bridge, you will also be able to identify a representation of the application data structure in some of the messages.

You should look now at the contents of file MA1I.DATA(FILEAR). If you start an ISPF browse session with this file, you will see that the first screen looks like this:

```
+----------------------------------------------------------------------------------+
|* This MQSeries CICS bridge playback instance was created from a message         |
|* set initially recorded under SupportPac CA1E, CICS Bridge Passthrough.         |
|*                                                                                 |
|* The message set was obtained by running transactions DMNU and DADD             |
|* of CICS sample application FILEA under Passthrough.  The exchange of           |
|* messages should complete normally with no errors when replayed using           |
|* MQSeries CICS bridge playback.                                                  |
|*                                                                                 |
|....PMO ..........................................................................|
|....PMO ..........................................................................|
|....PMO ..........................................................................|
|....PMO ..........................................................................|
|....PMO ..........................................................................|
+----------------------------------------------------------------------------------+
```

This file contains records in a format required by MBRPLAY, details of which are given in "The structure of a MQSeries CICS 3270 bridge playback record". It was generated by program MBRRCRD from a set of messages recorded under CICS 3270 Bridge Passthrough. Each of the records in which you can see the string PMO represents an inbound message to the bridge. If you scroll far enough to the right, you will see an MQ MD structure and a CIH structure in each. Notice that the comments at the start of the file are reproduced in the log file described above.

The name of this data file, FILEAR, is also of importance. Note that it ends in 'R', which identifies it as a file that contains records in playback format.

Look now at the contents of file whose last qualifier is of the form `**.SUM*`. This is a file that was created during post-run analysis of the log file, because you specified a value of 'Y' for parameter 3 (`++P3++` substitution) in the JCL. The contents of the file are similar to the following:

```
+-------------------------------------------------------------------------------+
|-------------------------------------------------------------------------------|
|********** CICS Bridge Test Mon Sep 23 09:52:52 2002.  Summary Report **********|
|-------------------------------------------------------------------------------|
|                                                                               |
|Test number 0001   Test Instance    FILEA                                      |
|All expected diagnostic elements received                                      |
|No unexpected diagnostic elements received                                     |
|Test succeeded                                                                 |
+-------------------------------------------------------------------------------+
```

This summary report refers to "expected diagnostic elements". If you look now in member MA1I.DATA (FILEAP), you will see the following:

```
+-------------------------------------------------------------------------------+
|Parameter file for playback instance FILEA                                     |
|-----------------------------------------                                      |
|                                                                               |
|Runtime Parameters:                                                            |
|                                                                               |
|USERID              =                                                          |
|AUTHENTICATOR       =                                                          |
|GETWAIT_INTERVAL    =                                                          |
|REQUESTQ            = SYSTEM.CICS.BRIDGE.QUEUE                                 |
|REPLYQ              = SYSTEM.DEFAULT.LOCAL.QUEUE                               |
|DEAD_LETTER_QUEUE   = SYSTEM.DEAD.LETTER.QUEUE                                 |
|FACILITYLIKE        =                                                          |
|                                                                               |
|Diagnostic elements expected:                                                  |
|                                                                               |
|MQCIH.AbendCode     =                                                          |
|MQCIH.ErrorOffset   =                                                          |
|MQCIH.Format        =                                                          |
|MQCIH.ReturnCode    =                                                          |
|MQCIH.CompCode      =                                                          |
|MQCIH.Reason        =                                                          |
|MQDLH.Reason        =                                                          |
|Bridge-message      =                                                          |
|                                                                               |
+-------------------------------------------------------------------------------+
```

This file contains a set of name-value pairs. All of those following the description "Diagnostic elements Expected:" identify values that could occur as a result of sending messages to the CICS bridge. All the names take null values in this case, so no such diagnostic elements are expected when playback instance FILEA is run.

Notice, too, that this file contains a set of name-value pairs described as "Runtime Parameters". Most of these are self-explanatory, and they control the way in which the playback instance is run.

The final thing to notice is that the name of this member ends in 'P', which identifies it as a file that contains runtime parameters.

## An example that invokes CEMT INQUIRE TASK

Members CEMTINQR and CEMTINQP are provided in library MA1I.DATA to implement a CEMT INQUIRE TASK dialog with the CICS 3270 bridge using playback. If you wish to run it, you must specify CEMTINQ as the playback instance in the Running Order file and submit job MBRPLAY.

The log and summary files that are created should show that no errors were detected. If you browse the log file, you should recognize some of the information in the data sent in the outbound messages as relating to the tasks running in your CICS region.

## An example that invokes a DPL program

Members DPL04R and DPL04P are provided to implement a dialog using the CICS DPL bridge. The target DPL program is DPLTARGT, and it writes the commarea data that is included in the message to CICS temporary storage queue TSDPLTGT. Four messages are sent in the unit of work, so four records are written to the temporary storage queue. If you wish to run this playback instance, you must specify DPL04 in the Running Order file and submit job MBRPLAY. You can browse the temporary storage queue after MBRPLAY has finished using the CICS command CEBR TSDPLTGT to confirm that program DPLTARGT has run.

The log and summary files that are created should show that no errors were detected.

## An example that causes an abend under the CICS 3270 bridge

Members BADVECVR and BADVECVP are provided to implement a dialog that terminates in a CICS 3270 bridge task abend. File BADVECVR was created by copying FILEAR and editing the vector version field in the last inbound message. The field was changed from the character string '0000', which correctly identifies the vector version, to binary zeros, which is incorrect.

To see how exceptions are managed by playback, you should enter BADVECV in the Running Order file and submit job MBRPLAY. When the job has run, browse the log file that has been produced. In addition to the recorded inbound and outbound messages you will see the following information:

```
+-------------------------------------------------------------------------------+
|FAILURE ANALYSIS for Test Instance  BADVECV                                    |
|                                                                               |
|Analyzing message  from SYSTEM.DEFAULT.LOCAL.QUEUE                             |
|                                                                               |
|MQCIH.AbendCode    MBRP                                                        |
|MQCIH.ErrorOffset  00c1                                                        |
|                                                                               |
|Analysis of CIH complete                                                       |
|                                                                               |
|No message available on SYSTEM.DEAD.LETTER.QUEUE                               |
|                                                                               |
|                                                                               |
|Processing complete for all tests                                              |
+-------------------------------------------------------------------------------+
```

In this example, the last inbound message contained a bad vector version value and this resulted in abend MBRP. The error offset is shown as 00c1, which is the offset of the vector version field.

Look now at the summary file that has been created for this run. It should contain the following information:

```
+-------------------------------------------------------------------------------+
|-------------------------------------------------------------------------------|
|********** CICS Bridge Test Mon Sep 23 15:20:47 2002.  Summary Report **********|
|-------------------------------------------------------------------------------|
|                                                                               |
|Test number 0001   Test Instance  BADVECV                                      |
|All expected diagnostic elements received                                      |
|No unexpected diagnostic elements received                                     |
|Test succeeded                                                                 |
+-------------------------------------------------------------------------------+
```

Notice that although the test generated an abend, it is reported as having succeeded. This is explained by the contents of the runtime parameter file BADVECVP, which is as follows:

```
+-------------------------------------------------------------------------------+
|Parameter file for playback instance BADVECV                                   |
|-------------------------------------------                                    |
|                                                                               |
|Runtime Parameters:                                                            |
|                                                                               |
|USERID               =                                                         |
|AUTHENTICATOR        =                                                         |
|GETWAIT_INTERVAL     =                                                         |
|REQUESTQ             = SYSTEM.CICS.BRIDGE.QUEUE                                |
|REPLYQ               = SYSTEM.DEFAULT.LOCAL.QUEUE                              |
|DEAD_LETTER_QUEUE    = SYSTEM.DEAD.LETTER.QUEUE                                |
|FACILITYLIKE         =                                                         |
|                                                                               |
|Diagnostic elements expected:                                                  |
|                                                                               |
|MQCIH.AbendCode      = MBRP                                                    |
|MQCIH.ErrorOffset    = 00c1                                                    |
|MQCIH.Format         =                                                         |
|MQCIH.ReturnCode     =                                                         |
|MQCIH.CompCode       =                                                         |
|MQCIH.Reason         =                                                         |
|MQDLH.Reason         =                                                         |
|Bridge-message       =                                                         |
+-------------------------------------------------------------------------------+
```

The file specifies an abend code of MBRP and an error offset of 00c1 as the diagnostic elements that are expected. Since these match the observed results exactly, the summary file reports that the test has succeeded.

## An example that causes an abend under the CICS DPL bridge

Members DPLABNDR and DPLABNDP are provided to implement a dialog that terminates in a CICS DPL bridge task abend. File DPLABNDR was created by copying DPL04R and editing the file so that the target DPL program in the last inbound message is DPLABEND. Program DPLABEND issues EXEC CICS ABEND ABCODE('DPLA') and in so doing causes the 4 messages that were sent in the unit of work to be backed out.

To run this example, specify DPLABND in the Running Order file and submit job MBRPLAY. If you browse the log file that is produced, you will see the following introductory entries displaying comments from the playback file and the inbound and messages:

```
+-------------------------------------------------------------------------------+
|-------------------------------------------------------------------------      |
|+++ Mon Sep 23 16:49:21 2002       Test 0001    Test Instance  DPLABND          |
|                                                                               |
|* This MQSeries CICS bridge playback instance was created using program        |
|* MBRRCRD and input data file DPL04.                                           |
|*                                                                              |
|* DPL04 specifies a target DPL program named DPLTARGT in each of four          |
|* messages in a unit of work.  The commarea data sent in each inbound          |
|* message is written to temporary storage queue TSDPLTGT.                      |
|*                                                                              |
|* For playback instance DPLABND, however, the final inbound message has        |
|* been edited to specify target DPL program DPLABEND.  The latter program      |
|* issues EXEC CICS ABEND ABCODE('DPLA'), which causes the unit of work         |
|* to be backed out.  The expected diagnostic information for this error        |
|* situation is defined in the accompanying parameter file DPLABNDP.            |
|*                                                                              |
|MD  ............................MQCICS  ........CSQ VRS2        §ädâ$... (é.+áï¬|
|MD  ............................4MQCICS  ........CSQ VRS2        §ädâ$...CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2        §ädâ#.C.CSQ VRS2|
|MD  ............................4MQCICS  ........CSQ VRS2        §ädâ$...CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2        §ädâ@▯..CSQ VRS2|
|MD  ............................4MQCICS  ........CSQ VRS2        §ädâ$...CSQ VRS2|
|MD  ............................MQCICS  ........CSQ VRS2        §ädâ=rO.CSQ VRS2|
|MD  ............................4MQCICS  ........CSQ VRS2        §ädâ$...CSQ VRS2|
+-------------------------------------------------------------------------------+
```

If you now scroll forward, you will see the following error analysis for the test:

```
+-------------------------------------------------------------------------------+
|FAILURE ANALYSIS for Test Instance  DPLABND                                    |
|                                                                               |
|Analyzing message  from SYSTEM.DEFAULT.LOCAL.QUEUE                             |
|                                                                               |
|MQCIH.Format                 MQFMT_STRING                                      |
|Bridge-message               CSQC754E Bridge task abend DPLA in program DPLABEND|
|MQCIH.ReturnCode   0005      MQCRC_APPLICATION_ABEND                           |
|MQCIH.AbendCode    DPLA                                                        |
|                                                                               |
|Analysis of CIH complete                                                       |
|                                                                               |
|Analyzing messages from SYSTEM.DEAD.LETTER.QUEUE                               |
|                                                                               |
|Message 1 retrieved from SYSTEM.DEAD.LETTER.QUEUE                              |
|                                                                               |
|MD  ............................4MQDEAD  ........CSQ VRS2        §ädâ$... (é.+áï¬|
|                                                                               |
|Analyzing message                                                              |
|                                                                               |
|MQDLH.Reason       019d      MQFB_CICS_UOW_BACKED_OUT                          |
|                                                                               |
|Message 2 retrieved from SYSTEM.DEAD.LETTER.QUEUE                              |
|                                                                               |
|MD  ............................4MQDEAD  ........CSQ VRS2        §ädâ#.C.CSQ VRS2|
```

```
|                                                                                |
|Analyzing message                                                               |
|                                                                                |
|MQDLH.Reason       019d      MQFB_CICS_UOW_BACKED_OUT                           |
|                                                                                |
|Message 3 retrieved from SYSTEM.DEAD.LETTER.QUEUE                               |
|                                                                                |
|MD  ........................4MQDEAD  ........CSQ VRS2        §ädâ@▯ ..CSQ VRS2|
|                                                                                |
|Analyzing message                                                               |
|                                                                                |
|MQDLH.Reason       019d      MQFB_CICS_UOW_BACKED_OUT                           |
|                                                                                |
|Message 4 retrieved from SYSTEM.DEAD.LETTER.QUEUE                               |
|                                                                                |
|MD  ........................4MQDEAD  ........CSQ VRS2        §ädâ=rO.CSQ VRS2|
|                                                                                |
|Analyzing message                                                               |
|                                                                                |
|MQDLH.Reason       019b      MQFB_CICS_APPL_ABENDED                             |
|                                                                                |
|No more messages available on SYSTEM.DEAD.LETTER.QUEUE                          |
|                                                                                |
|                                                                                |
|Processing complete for all tests                                               |
+--------------------------------------------------------------------------------+
```

If you now browse the test summary file that is produced, you will see the entries similar to the following:

```
+--------------------------------------------------------------------------------+
|--------------------------------------------------------------------------------|
|********** CICS Bridge Test Mon Sep 23 16:49:21 2002.  Summary Report **********|
|--------------------------------------------------------------------------------|
|                                                                                |
|Test number 0001   Test Instance  DPLABND                                       |
|All expected diagnostic elements received                                       |
|No unexpected diagnostic elements received                                      |
|Test succeeded                                                                  |
|                                                                                |
+--------------------------------------------------------------------------------+
```

If you now browse file DPLABNDP, you will see that all the diagnostic elements that have been detected by MBRPLAY have been anticipated:

```
+--------------------------------------------------------------------------------+
|Parameter file for playback instance DPLABEND                                   |
|-------------------------------------------                                     |
|                                                                                |
|Runtime Parameters:                                                             |
|                                                                                |
|USERID              =                                                           |
|AUTHENTICATOR       =                                                           |
|GETWAIT_INTERVAL    =                                                           |
|REQUESTQ            = SYSTEM.CICS.BRIDGE.QUEUE                                  |
|REPLYQ              = SYSTEM.DEFAULT.LOCAL.QUEUE                                |
|DEAD_LETTER_QUEUE   = SYSTEM.DEAD.LETTER.QUEUE                                  |
|FACILITYLIKE        =                                                           |
```

```
|                                                                              |
|Diagnostic elements expected:                                                 |
|                                                                              |
|MQCIH.AbendCode      = DPLA                                                   |
|MQCIH.ErrorOffset    =                                                        |
|MQCIH.Format         = MQFMT_STRING                                           |
|MQCIH.ReturnCode     = 0005                                                   |
|MQCIH.CompCode       =                                                        |
|MQCIH.Reason         =                                                        |
|MQDLH.Reason         = 019d                                                   |
|MQDLH.Reason         = 019d                                                   |
|MQDLH.Reason         = 019d                                                   |
|MQDLH.Reason         = 019b                                                   |
|Bridge-message       = CSQC754E                                               |
+------------------------------------------------------------------------------+
```

Notice that a separate diagnostic element must be specified for every message that is sent to the dead letter queue.

### Invoking playback program MBRPLAY with no logging

To invoke program MBRPLAY in a way in which no messages are logged, you need to use JCL member MBRPLAYX. Before submitting the job, it is necessary to make a number of substitutions in the JCL according to the instructions in the comments. The substitutions for high level qualifiers of datasets are self-explanatory. This JCL invokes program MBRPLAY with 2 startup parameters, however, the second of which is always NOLOGGING. You need to make a substitution for the first parameter as follows:

**++P1++** Replace this with the name of the queue manager that is connected to the CICS region where the bridge is to run.

Invoking program MBRPLAY without logging the results is useful if you want to run high-volume tests. You should, however, run the tests with logging (use JCL member MBRPLAY) before running with NOLOGGING to ensure that the applications are working the way you intend.

---

## Using the playback recording program MBRRCRD

You can run the playback recording program MBRRCRD using the supplied JCL member MA1I.JCL (MBRRCRD). Before submitting the job, it is necessary to make a number of substitutions in the JCL according to the instructions in the comments. Many of the substitutions are for high level qualifiers of datasets and are self-explanatory. Program MBRRCRD requires a startup parameter specified in the JCL as `++P1++`, however, and this requires further explanation.

The values that the startup parameter can take are 3270, DPL or DPLNOCIH. The meanings of these values are as follows:

**3270** MBRRCRD is to create a playback file for the CICS 3270 bridge. It requires as input the set of inbound messages in the dialog, initially recorded using CICS 3270 Bridge Passthrough.

**DPL** MBRRCRD is to create a playback file for the CICS DPL bridge. It requires as input a file containing the name of the target DPL program and the commarea data that is to be sent.

**DPLNOCIH** MBRRCRD is to create a playback file for the CICS DPL bridge, but the inbound record is not to include a CIH structure. You cannot use this format if the dialog requires any of the fields in the CIH, for example MQCIH.UowControl or MQCIH.Authenticator. It follows that the format can only be used for a single message in a unit of work, and only limited security controls can be applied.

## Creating playback records for the CICS 3270 bridge

Set the parameter represented by `++P1++` to the value 3270 to create a playback file for the CICS 3270 bridge.

You must now substitute the name of the input data file for `++SOURCEDATA++`. Use the value `++MA1IQUAL++.DATA(FILEA)`, where `++MA1IQUAL++` is the high level qualifier of the libraries you installed for SupportPac MA1I. This member contains a set of messages recorded under CICS 3270 Bridge Passthrough. The messages represent an exchange that took place when transactions DMNU and DADD of the CICS sample application FILEA were invoked.

You must also substitute the name of the output data set where you want the playback records for the 3270 bridge to be created, identifed by substitution string `++PLAYBACKFILE++`. Specify a value of `++MA1IQUAL++.DATA(TST3270R)`, where `++MA1IQUAL++` is the high level qualifier of the libraries you installed for SupportPac MA1I.

After completing all of the substitutions, submit job MBRRCRD. When the job terminates, browse file TST3270R. The playback records in the file are identical to those in FILEAR.

You can confirm that TST3270R contains a valid set of playback records by copying member FILEAP as TST3270P, entering a value of TST3270 in the Running Order file, and submitting job MBRPLAY. The log file that is produced from TST3270R will not contain any comment records, because those were added manually to FILEAR, but the output should otherwise be the same as that from FILEAR.

## Creating playback records for the CICS DPL bridge

The CICS DPL bridge will accept inbound messages with or without a CIH structure. If the messages do not contain a CIH structure, then multiple messages cannot be included in a unit of work, and certain security controls cannot be applied.

### Creating DPL playback records that include a CIH structure

Set the parameter represented by `++P1++` to the value DPL to create a playback file for the CICS DPL bridge that includes a CIH in each inbound message.

Substitute a value of `++MA1IQUAL++.DATA(DPL04)` for the dataset addressed by DDNAME B3270DAT, where `++MA1IQUAL++` is the high level qualifier of the libraries you installed for SupportPac MA1I. This member contains the following records:

```
+-------------------------------------------------------------------------+
|                                                                         |
|* This data file is used to create a test in which four messages        |
|* are contained in the unit of work                                     |
|                                                                         |
```

```
|* Name of target DPL program:                                                |
|                                                                             |
|DPLPGM = DPLTARGT                                                            |
|                                                                             |
|* Data that will be passed in the commarea to the target program:            |
|                                                                             |
|First Record in a multi-record Unit of Work                                  |
|First of two Middle Records in a multi-record Unit of Work                   |
|Second of two Middle Records in a multi-record Unit of Work                  |
|Last Record in a multi-record Unit of Work                                   |
|                                                                             |
|#END OF FILE                                                                 |
+-----------------------------------------------------------------------------+
```

This file specifies that the DPL program to be linked for each inbound message is DPLTARGT. In addition, there are 4 separate records that are not comment lines. Each such record will form the commarea data in a single inbound message. There will, therefore, be 4 inbound messages in the set created for the playback file. The messages will all form part of a single unit of work.

Before submitting job MBRRCRD, you must also substitute the name of the output data set where you want the playback records for the DPL bridge to be created. Specify a value of `++MA1IQUAL++.DATA(TSTDPLR)`, where `++MA1IQUAL++` is the high level qualifier of the libraries you installed for SupportPac MA1I.

After completing all of the substitutions, submit job MBRRCRD. When the job terminates, browse file TSTDPLR. The playback records in the file are identical to those in DPL04R.

You can confirm that TSTDPLR contains a valid set of playback records by copying member DPL04P as TSTDPLP, entering a value of TSTDPL in the Running Order file, and submitting job MBRPLAY. The log file that is produced from TSTDPLR will not contain any comment records, because those were added manually to DPL04R, but the output should otherwise be the same as that from DPL04R.

**Creating DPL playback records that do not include a CIH structure**

To create a DPL playback file that does not include a CIH structure in each inbound message, set parameter `++P1++` on the EXEC PGM statement to DPLNOCIH.

Copy member DPL04 as NOCIH04 (for example). You may want to edit the 4 records that will become the commarea data so they make no reference to units of work, since the messages cannot all belong to the same unit of work if the messages do not contain a CIH structure.

Substitute a value of `++MA1IQUAL++.DATA(NOCIH04)` for the dataset addressed by DDNAME B3270DAT, where `++MA1IQUAL++` is the high level qualifier of the libraries you installed for SupportPac MA1I. You must also substitute a value of `++MA1IQUAL++.DATA(NOCIH04R)` for the dataset addressed by DDNAME B3270PLA.

When the substitutions are complete, submit job MBRRCRD. If you examine the output dataset NOCIH04R, you will see that none of the records contain the CIH structure (search on string CIH, which is the eyecatcher for the structure).

You can confirm that the CICS DPL bridge can be invoked successfully with playback file by submitting job MBRPLAY. Don't forget to edit the running order file RUNORDER, which needs to

contain a value of NOCIH04. Also, you should copy member DPL04P as NOCIH04P before submitting the job.

**Abending a bridge task when no CIH is in the inbound message**

It is instructional to cause an abend in the bridge task when no CIH is included in the inbound message causing the abend.

Edit file NOCIH04R, changing the last occurrence of DPLTARGT (which is in the 4th message in the set) to DPLABEND. Save the file, and submit job MBRPLAY once again. If you examine the log file that is produced for the run, you will see that this time only the last message is sent to the dead letter queue. This is because a message with no CIH structure is effectively the only message in a unit of work.

---

## Capturing message flows under CICS Bridge Passthrough

If you want to create dialog recordings to replay to the MQSeries CICS bridge for your own 3270 transactions, you must first capture message flows using CICS Bridge Passthrough (SupportPac CA1E).

The User's Guide for Passthrough gives details of how you should do this. In summary, you need to run your transaction under Passthrough transaction BRMQ, recording the messages in a transient data queue. SupportPac MA1I provides a dataset MA1I.BRLOG that is suitable as a transient data destination, and a CICS resource definition that associates this dataset with TD Queue BRLG in member MA1I.DEFS(MBRCIDEF).

You should record at least the inbound messages in this TD Queue. You can also record the outbound messages in the TD Queue, but they will be ignored by MBRRCRD when the playback recording is made.

When you have recorded the messages, you will need to set the TD Queue Closed before you can access the records within dataset MA1I.BRLOG. You can use MA1I.BRLOG directly for input into MBRRCRD, or you can copy the contents to another dataset first. Make sure that you copy to a dataset with the same DCB attributes as MA1I.BRLOG.

You can now create the playback records by specifying the file with the recorded messages (DDNAME B3270DAT) in member MA1I.JCL(MBRRCRD), and submitting the job. Don't forget that your target file for the playback records must have a name that ends in 'R'.

Before you can replay the records, you must also have a parameter file with a name identical to your playback file except for the last character, which must be 'P'.

---

# Security considerations

The MQSeries CICS bridge Monitor can be started with different security characteristics to control access to protected CICS resources. The level of authorization is determined by the `AUTH=<SECLEVEL>` parameter in the startup string for the monitor. The possible values that `<SECLEVEL>` can take are as follows:

- LOCAL
- IDENTIFY
- VERIFY_UOW
- VERIFY_ALL

See *MQSeries for OS/390 Version 5 Release 2 System Administration Guide SC34-5652* or *WebSphere MQ for z/OS Version 5 Release 3 System Setup Guide SC34-6052* for details of the protection that these levels of authorization provide.

If the CICS bridge request queues are triggered, startup parameters can be passed to the monitor from process definitions associated with the queues. SupportPac MA1I provides a set of request queues and processes that enable monitors to be triggered with the different levels of authorization shown in the following table. The monitors are started with different transaction identifiers to allow you to distinguish between them.

| Request queue | AUTH value | Transaction ID |
|---|---|---|
| CICS.BRIDGE.REQUEST.LOCAL | LOCAL | BSLO |
| CICS.BRIDGE.REQUEST.IDENTIFY | IDENTIFY | BSID |
| CICS.BRIDGE.REQUEST.VERIFY_UOW | VERIFY_UOW | BSVU |
| CICS.BRIDGE.REQUEST.VERIFY_ALL | VERIFY_ALL | BSVA |

In order to replay a dialog with the MQSeries CICS bridge using a specific level of security, you must specify the request queue associated with that level of security in the runtime parameter file. The parameter file also allows you to specify a userid and, if necessary, a passticket or password to be checked for the userid.

For example, if you wanted to replay a dialog with `AUTH=VERIFY_ALL`, you could specify something like the following in the runtime parameter file:

```
USERID           = RACHEL
AUTHENTICATOR    = MYPWD
GETWAIT_INTERVAL = 9000
REQUESTQ         = CICS.BRIDGE.REQUEST.VERIFY_UOW
REPLYQ           = LOCAL.QUEUE
FACILITYLIKE     = TC03
```

This would cause bridge monitor BSVU to be started with AUTH=VERIFY_UOW if necessary, if it were not already running. A value of RACHEL would be inserted into MQMD.UserIdentifier and a value of MYPWD would be inserted into MQCIH.Authenticator in every inbound message. Note however that only the values in the first message in a unit of work are checked for VERIFY_UOW. In this case, the authenticator MYPWD would be verified for user RACHEL, and the authority of RACHEL to access any protected resources invoked under the bridge would be checked.

**Note:** The value of AUTHENTICATOR is not encrypted in the runtime parameter file, the messages that flow, or the log file that is created. You must take adequate precautions to prevent unauthorized persons from reading these files or otherwise monitoring the messages.

---

# Reference Section

---

## Inputs and outputs for MBRPLAY

The inputs required by program MBRPLAY and the outputs that it produces are as follows:

- Inputs:
  - Startup parameters specifying *either:*
    - The queue manager
    - The high level qualifier of the log files
    - Whether post-run analysis is required (Y/N)

    *or*
    - The queue manager
    - NOLOGGING
  - A 'Running Order' file specifying the playback instance (or instances) to run, for example FILEA. This is addressed by DDNAME B3270RUN.
  - A member containing playback records. The name must be equal to the name specified in the Running Order file suffixed with 'R', for example FILEAR. The member must be in the library addressed by DDNAME B3270PLA.
  - A member containing runtime parameters. The name must be equal to the name specified in the Running Order file suffixed with 'P', for example FILEAP. The member must be in the library addressed by DDNAME B3270PRM.
- Outputs:
  - A log file containing inbound and outbound messages and an analysis of any error conditions that are detected. The filename is of the form **.LOG*.
  - If requested ('Y' as parameter 3 in the MBRPLAY job), a post-run analysis of the log file saying whether expected diagnostic elements were obtained. The filename is of the form **.SUM*.

The runtime progress of program BRMPLAY is reported in the SYSPRINT dataset of the job output.

---

## Inputs and outputs for MBRRCRD

The inputs required by program MBRRCRD and the outputs that it produces are as follows:

- Inputs:
  - A startup parameter from the set {3270|DPL|DPLNOCIH}. This defines whether the playback file is to be created for the CICS 3270 bridge or the CICS DPL bridge, and in the latter case whether a CIH structure is to be included in each inbound message.
  - For a playback file for the CICS 3270 bridge, a file containing inbound messages recorded under CICS 3270 bridge Passthrough. The file can also contain outbound messages, but

these will be ignored when the playback file is created.

- For a playback file for the CICS DPL bridge, a file containing the name of a target DPL program, and a separate record containing commarea data for every inbound record. If startup parameter 'DPL' is specified, all the messages will be included in a single unit of work. If startup parameter 'DPLNOCIH' is specified, each message will comprise a separate unit of work.

  The file is addressed by DDNAME B3270DAT.

- Outputs:
  - The name of the member that is to receive the playback file. This is addressed by DDNAME B3270PLA.
- DDNAME B3270WSI is reserved and should address DUMMY.

The runtime progress of program BRMPLAY is reported in the SYSPRINT dataset of the job output.

---

## The Running Order file

The Running Order file contains the name prefixes of playback file - parameter file pairs, one per line. The following entries in a running order file would, for instance, cause playback files FILEAR, DPL04R and CEMTINQR to be run one after the other:

```
FILEA
DPL04
CEMTINQ
```

Parameter files FILEAP, DPL04P and CEMTINQP, respectively, would be used to set the runtime environments.

---

## The runtime parameter file

The runtime parameter file and the file containing playback records must have a common name prefix. The parameter file is identified by the suffix 'P', and the playback file by the suffix 'R'. Thus, a parameter file - playback file pair might be called FILEAP and FILEAR, respectively.

Whereas the playback file is generated by program RCRD3270, the parameter file must be created manually. It contains a set of keyword - value pairs to establish the environment for the corresponding playback file, as follows:

```
USERID           =
AUTHENTICATOR    =
GETWAIT_INTERVAL = 9000
REQUESTQ         = CICS.BRIDGE.REQUEST.IDENTIFY
REPLYQ           = LOCAL.QUEUE
FACILITYLIKE     =
```

It is not necessary to specify any values in the parameter file (although the parameter file must exist). If you do not specify values, the following default values are used:

```
USERID           = <12 blanks>
AUTHENTICATOR    = <8 blanks>
GETWAIT_INTERVAL = 5000
REQUESTQ         = SYSTEM.CICS.BRIDGE.QUEUE
REPLYQ           = SYSTEM.DEFAULT.LOCAL.QUEUE
FACILITYLIKE     = CBRF
```

The value of USERID, if specified, will be set in field MQMD.UserIdentifier in every inbound message. The values of MQOO_SET_IDENTITY_CONTEXT and MQPMO_SET_IDENTITY_CONTEXT will be set automatically when USERID is specified.

AUTHENTICATOR specifies a password or passticket that will be set in feild MQCIH.Authenticator in every inbound message.

GETWAIT_INTERVAL specifies the time in milliseconds for which PLAYBACK will wait for reply messages to arrive at the REPLYQ.

REQUESTQ must always specify a real bridge request queue. Since the MQSeries CICS bridge monitor can be triggered with start data from the request queue, it is useful to specify different Request queues to establish different monitor environments. For example, you might wish to specify a Request queue CICS.BRIDGE.VERIFY.UOW that starts a monitor with AUTH = VERIFY_UOW. You could define a transaction BRVU, say, to CICS, with the same definition as CKBR (the default monitor transaction) to distinguish it from the latter when it is running.

Note that `AUTH=VERIFY_UOW` and `AUTH=VERIFY_ALL` require MQOO_SET_IDENTITY_CONTEXT to be set in the MQOO word, and MQPMO_SET_IDENTITY_CONTEXT to be set in the MQPMO structure. This can be done by editing the playback file, taking due account of its structure, with a hexadecimal editor. Otherwise, if you specify a USERID value in the parameter file, MQOO_SET_IDENTITY_CONTEXT and MQPMO_SET_IDENTITY_CONTEXT will be set automatically at runtime.

REPLYQ can either specify a real bridge reply queue, or the special value NULL if you want the value of MQMD.ReplyToQ in each message to be set to spaces.

FACILITYLIKE can either specify NULL or the name of an installed terminal to use as a model. When NULL is specified, the value of field MQCIH.FacilityLike is set to spaces.

The parameter file can also specify name - value pairs indicating what diagnostic elements are expected from the bridge, in the case where errors are caused deliberately in order to test error handling by the bridge. The name - value pairs are only used if 'Y' for post-run analysis of the log file is specified as a start parameter for PLAYBACK. Some examples are:

```
MQCIH.Format     = MQFMT_STRING
Bridge-message   = CSQC751E
MQCIH.ReturnCode = 0006
MQCIH.CompCode   = 0046
MQCIH.Reason     = 0065
MQDLH.Reason     = 0192
```

If the diagnostic elements actually received are not exactly the ones specified here, a record is written to the summary file saying that the test has failed.

There are no default values for diagnostic elements.

In addition to keyword - value pairs, the parameter file can also contain comments. Any line that does not start with a valid keyword in Column 1 is a comment line. Comment lines in the parameter file are not logged by MBRPLAY, unlike those in the playback file.

---

## Input file for creating CICS 3270 bridge playback records

Flows initially recorded using CICS Bridge Passthrough are required if 3270 playback records are being created. Details of how to record the flows are given in the documentation for SupportPac CA1E.

The records can be edited manually, if required, before being converted to playback records. Comments can be added by starting them with * in column 1. The comments are written unchanged to the log output by MBRPLAY. Be aware, however, that some fields in a playback record are updated dynamically by MBRPLAY at playback time. These include MQMD.MsgId, MQMD.CorrelId and MQCIH.Facility.

Program MBRPLAY filters just the inbound messages from the recorded flows as input for the playback records.

---

## Input file for creating CICS DPL bridge playback records

If DPL playback records are required, the records in the input file must be in the following form:

```
DPLPGM = <DPL program name>
<commarea data for first message>
<commarea data for second message>
...
<commarea data for last message>
```

One playback record is created for every commarea record. The same DPL program is used for each. Every such record that is created has the correct value inserted in MQCIH.UOWControl to make it part of the same unit of work. All the records in the set comprise a single unit of work.

---

## The structure of a MQSeries CICS 3270 bridge playback record

Every playback record contains an MQOO fullword, an MQPMO structure, an MQOD structure, an MQMD structure and an MQCIH structure in concatenation. Some records may also contain a BRMQ vector structure, an application data structure in short or long format (ADS OR ADSL), and an application data structure descriptor in short or long format (ADSD or ADSDL).

The possible message structures are as follows. Only the short forms of the ADS and ADSD are represented here, although either the short or long forms may be included.

- Record with MQCIH only:

```
 ----------------------------------
| MQOO | MQPMO | MQOD | MQMD | MQCIH |
 ----------------------------------
```

- Record with MQCIH and ADS but no ADSD:

```
 ------------------------------------------------------
| MQOO | MQPMO | MQOD | MQMD | MQCIH | BRMQ structure | ADS |
 ------------------------------------------------------
```

- Record with MQCIH, ADS and ADSD:

```
 -------------------------------------------------------------------
| MQOO | MQPMO | MQOD | MQMD | MQCIH | BRMQ structure | ADS | ADSD |
 -------------------------------------------------------------------
```

---

## The structure of a Websphere MQ CICS DPL bridge playback record

Every record contains an MQOO fullword, an MQPMO structure, an MQOD structure, and an MQMD structure. The data that follows the MQMD structure depends on whether the file was created using DPL or DPLNOCIH as an invocation parameter, and whether the input records contained commarea data.

The possible structures are as follows.

- Record created using option DPLNOCIH, no commarea specified:

```
 ---------------------------------
| MQOO | MQPMO | MQOD | MQMD | ProgName |
 ---------------------------------
```

  When no commarea data is included, the program name MUST be exactly 8 characters long. It cannot be a name that is padded to the right with spaces.

- Record created using option DPLNOCIH, commarea specified:

```
 -----------------------------------------------
| MQOO | MQPMO | MQOD | MQMD | ProgName | Commarea |
 -----------------------------------------------
```

- Record created using option DPL, no commarea specified:

```
 ------------------------------------------
| MQOO | MQPMO | MQOD | MQMD | MQCIH | ProgName |
 ------------------------------------------
```

- Record created using option DPL, commarea specified:

```
 ------------------------------------------------------------
| MQOO | MQPMO | MQOD | MQMD | MQCIH | ProgName | Commarea |
 ------------------------------------------------------------
```

---

# The logic of program MBRPLAY

Program MBRPLAY first reads the running order file and stores the names in an internal array. All subsequent processing is performed once for every name in the array. This establishes the outer loop of the program.

For each name in the array, MBRPLAY reads the parameter file to establish the runtime environment. It then reads the playback file sequentially. Any comment lines in the file are written unchanged to the log file. Lines that are not comments are playback records, and dynamic changes are made to the records as appropriate. For example MQCIH.Facility needs to be copied from the first outbound message to all succeeding inbound messages for 3270 transactions.

When any necessary changes have been made, the message is MQPUT to the Bridge request queue. The MQMD, MQCIH and any application data are also written to the log file.

After the inbound message has been sent to the request queue, the program issues MQGET with MQGMO_WAIT for the Reply queue. If no message is retrieved within GETWAIT_INTERVAL, it is assumed that an error has occurred and this is reported in the log. Further diagnosis is also carried out -- see 'Error Handling' below.

If a message is successfully retrieved from the Reply queue, key fields in the MQCIH are examined to see whether there has been an error response. If there has, the test instance is ended and further diagnosis is carried out --see 'Error Handling' below.

## Error handling by program MBRPLAY

An error is flagged if program MBRPLAY receives a specific error response from the Bridge, or if it does not receive an outbound that it was expecting.

Certain fields in the MQCIH are used by the Bridge to return diagnostic information in its outbound messages. These fields are all analyzed by MBRPLAY every time it receives an outbound message on the Reply queue. If any values are detected that show an error has occurred, the test instance is ended and a report written to the log file.

No matter whether a specific error was reported by the Bridge, or an expected outbound reply was not received, the application now attempts to GET messages with the correct value of MQMD.MsgId from the dead letter queue, the Reply queue (if at least one outbound message has already been retrieved from there), and the Request queue. If a message can be retrieved from the dead letter queue, the value of MQDLH.Reason is compared with the set of MQFB_* values to determine the cause of the failure. Any findings are written to the log file.

## Log file analysis by program MBRPLAY

Log file analysis is an option that is specified as a startup parameter for MBRPLAY. If the option is

selected, the log file is reopened when all test instances have completed and read record-by-record from the start. A summary report for the run is written to a separate summary file. Expected diagnostics that are included in the parameter file as a set of name-value pairs are compared with diagnostics that are actually found for each test instance, and if they do not match a record is written to the summary file that the test has failed.

Summary report files and log files are created with unique names to ensure that multiple test programs can run concurrently. The names are created using a combination of the date and time stamps returned by the standard C library function ctime, and a string of characters derived from an instantaneous store clock value. 