MQSeries for OS/390 - Log extract program

Version 2.2

September, 2002

Colin Paice
IBM Hursley

PAICE@UK.IBM.COM

Take Note!

Before using this report be sure to read the general information under "Notices".

Second Edition, August 2002

This edition applies to Version 2.2 of **MQSeries for OS/390 - Log extract program** and to all subsequent releases and modifications unless otherwise indicated in new editions.

Table of Contents

Notices

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 rites 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.

Trademarks and service marks

The following terms, used in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

- IBM
- MQSeries
- OS/390
- TSO
- REXX

Summary of Amendments

Date	Changes
30 June 2001	Initial release
August 2002	REXX code replaced with load modules
September 2002	Fixed 0C4 abend. Users need to reapply the updated usermod
September 2002	Fixed abend code 4, and remove spurious WTOs.

Preface

This SupportPac processes MQSeries for OS/390 V5.2 log data sets to give information about persistent messages processed by applications, and information about objects that have been defined or altered.

It does not currently support shared queue, though it is intended to provide this support at a future date.

Bibliography

MQSeries for OS/390 System Administration Guide Version 5 Release 2. IBM Corporation SC34-5652-00

Chapter 1. What is provided

Introduction

This SupportPac extends the log print utility, by providing two additional functions.

The first creates one or more datasets from the messages in the MQSeries log datasets. Typical uses of the data include

1. Reviewing which persistent messages were put or got to a queue and whether the request was committed. This allows messages to be replayed. A 'C' program is provided which puts messages to the same queue as they were originally put to. Note putting messages to system queues can cause problems such as duplicate messages.
2. Reviewing persistent messages that were put or got, but the request was backed out.
3. Displaying which applications backed out rather than committed
4. Identifying the volume of data through queues, to identify the high use queues.

The second function creates a file of changes to MQSeries objects from information recorded in the log data sets. Typical uses include identifying which applications set object attributes, and recreating object definitions for recovery purposes after a major failure.

Additional materials with this SupportPac

Included in the SupportPac are two partitioned data sets that contains the programs to extract the log records, a C program to replay messages, and supporting JCL.

There are two files called *mo12.loa* and *mo12.pds* contained in *mo12.zip*. These files need to be transferred to the destination TSO system as sequential binary file with a record format of FB 80.

Send mo12.loa to TSO with a name MO12.LOADSEQ.
Send mo12.pds to TSO with a name MO12.PDSSEQ.

Use one of the following methods to accomplish this:

- To send them via ftp ensure the BINARY option is set then use the following commands
  site fixrecfm 80 (optional)
  Put MO12.PDS MO12.PDSSEQ
- With Personal Communications, use the *Send Files to Host* option under the Transfer menu item to transmit to TSO
  PC File MO12.PDS
  Host File MO12.PDSSEQ
  Transfer Type PDS

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

Repeat with the MO12.LOA file.

On TSO, issue the command:*receive indsname(MO12.PDSSEQ)* to unload the sequential files into a TSO partitioned dataset. When prompted for a filename, reply: *dsn(MO12.PDS)*

This PDS contains the following members:

For message extraction and replay

CSQ1LOGP	JCL to run the modified CSQ1LOGP utility
CMQMSG2	A C header file which maps the record header
CPROG	Example C program source which reads a file and replays messages
REPLAY	JCL to run the example C program to replay messages
UMOD52	Usermod that needs to be applied to MQSeries 5.2 (the only supported release) using SMP/E to enable the Supportpac
COUNT	This contains a JCL example of how to count the number of bytes put to a queue

For extracting information about changes to objects

CSQ1LOGP	JCL to run the modified CSQ1LOGP utility
CMDS	JCL to create a file for input to CSQUTIL.

On TSO, issue the command: *receive indsname(MO12.LOADSEQ)* to unload the sequential files into a TSO partitioned dataset. When prompted for a filename, reply: *dsn(MO12.LOAD)*

This PDS contains the following member:

CCP1LRPL	This is an additional module needed by CSQ1LOGP for processing messages.

Chapter 2. Running the modified CSQ1LOGP

Before you can extract messages from the MQSeries log data sets you need to use SMP/E to apply the usermod in this support pac, to the MQSeries modules CSQ1LOGP and CSQ1LOUT. The usermod causes the module CCP1LRPL to be called to process records.

CSQ1LOGP can use active logs (if the queue manager is not running) or archive logs. You can select a subset of the data to be processed, for example an RBA range or updates to a particular page set.

See the *System Administration Guide* for instructions on how to use CSQ1LOGP. Sample JCL is given in member CCP1LOGP of the MO12.PDS.

The processing is as follows.

1. CSQ1LOGP reads the specified log data sets and passes the records to CCP1LRPL. For records that match the specified filters a table is built up of units of work and messages processed by the unit of work.
2. When a commit or a backout request is found then the information for that unit of work is written to a file. The file used depends on the request.
   a. For a commit requests then if the file CCPCMT is present then the data is written to this file. If the file CCPBOTH is present the data is also written to this file.
   b. For a backout request, if the file CCPBACK is present then the data is written to this file, if the file CCPBOTH is present then the data is also written to this file. There is a flag in the data to indicate if it was written for a commit or a backout request.
3. When changes to objects are detected then the information is written to the file CCPCMDS in a format similar to that provided by the SDEFS function of CSQUTIL.
4. When the last record has been processed, information about remaining units of work is written to the file CCPINFL.

The optional JCL statements needed for a CSQ1LOGP job are

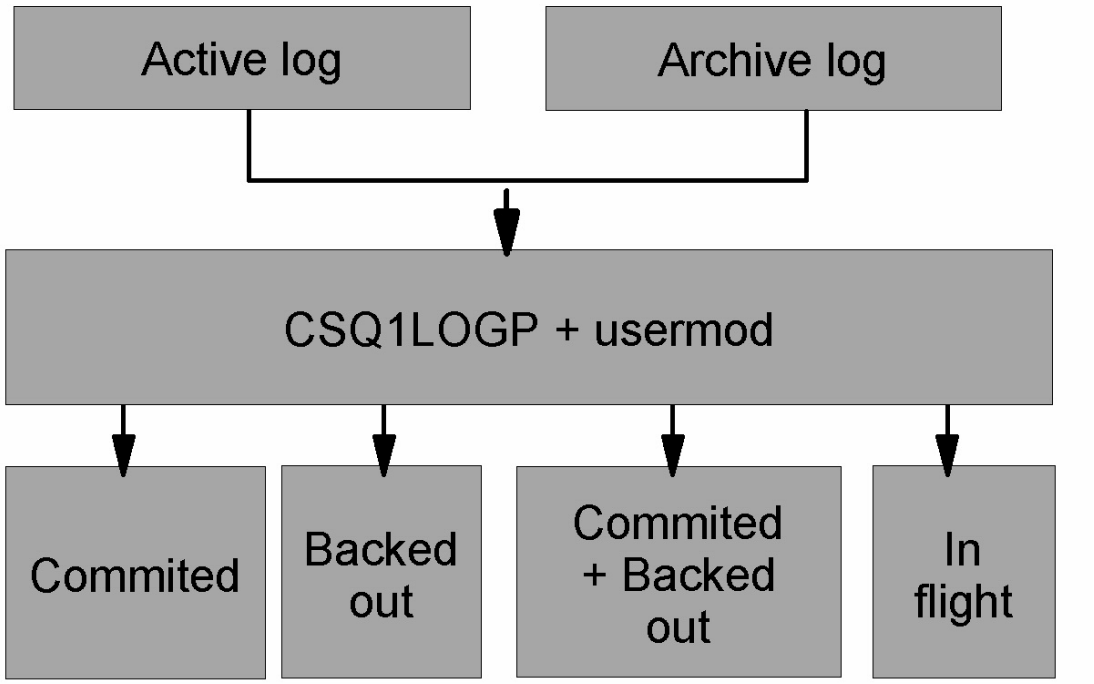
CCPCMT	This contains records for Units of Work which were committed
CCPBACK	This contains records for Units of Work which were backed out
CCPBOTH	This contains records for Units of Work which were committed or backed out. There is a field which defines whether the record was committed our backed out
CCPINFL	This contains records for Units of Work which were inflight at the end of the processing
CCPCMDS	This contains records for objects that were created, or altered, or were changed using MQSET

The files have the following attributes

1. Record format VB
2. Record length 32756
3. Block size 32760

If you do not want the output files then omit the DD definition.

The following figure shows the flow of data



Abends

If an internal error is detected then the CCP1LRPL will abend with User abend code of 0, and a reason code. If this happens, please report the abend and reason code to the author of this support pac.

Record layout for the data sets

The files contain information about persistent messages. Messages are identified by their queue name and an eight character key. Once a message has been got, the key can be reused by another message, so it is important to ensure that time sequence is maintained.
In the records are times. A time stamp can only be extracted from a Begin-UR record or from an MQPUT request. Thus if there is only a long running transaction which is getting messages, the times when the gets occurred will all be the time the transaction started (the Begin-UR record).
If there are many short units of work, or many messages being put, the time is reasonably accurate (milliseconds), otherwise the times will become less and less accurate.

The information in the files has the following layout: (Note there is a 4 byte Record Descriptor Word at the front of each record because the files are Variable Blocked format).

Name	Offset (dec)	Length (dec)	Description
Date	1	8	The date in format yyyy.ddd in EBCDIC
Time	10	12	The approximate time(time-now) in hh:mm:ss.tttmmm format in

			EBCDIC
Time delta	26	5	Approximate time difference in milliseconds from the start of the unit of work in EBCDIC
STCK	31	8	8 byte STCK format of the time-now
urid	39	6	RID of start unit of work.
	45	12	Reserved
Userid	57	8	Userid
UOWSTCK	65	8	STCK of start of Unit of work
	73	8	Reserved
Jobtype	81	8	One of BATCH, RRSBATCH, IMS, CICS, CHIN or nulls for an internal task
Jobname	89	8	
URLOC	97	3	Where did URID information come from - BUR for begin ur or 'CP ' for from checkpoint information
Data length	100	4	Length of the message data(if any)
Queue name	104	48	Name of queue, for get, put or expired messages
Message Key	152	8	Information to identify the message. This is present when the queue name is non blank
Verb	160	6	Text description of activity Alter - when objects are changed Define - when Objects are created Get Put Expire - the message expired Abort2 - the message was backed out Phase1 - the first phase of two phase commit Phase2 - the second phase of two phase commit, or the only phase of one phase commit.
Status	166	1	Status of unit of work • B Backed out • C Committed • I Inflight
Segment	167	4	For a put the message is stored in segments, this field is the message segment number starting from 1. For a get of a message this is 0. It is also 0 if the message has expired.
*	171	*	Message data part
BorA	171	1	‘B’ or ‘A’ for commands to indicate the Before definition or the After definition
Command	172	80	The command needed to change the object.

Replay messages

MO12.PDS contains a C program that can be used to replay the messages. This has been changed from the previous version to support the new record layout

This program reads an output file from CSQ1LOGP, such as CCPCMT, rebuilds the messages and issues MQPUT1 to put the message to the specified queue. It has an option to put all messages, or only those that have not been deleted.
The member CPROG contains the C program source and JCL to compile and run it. The program uses CMQMSG2, also in the PDS, which defines the record layout of the CSQ1LRPL dataset.

The JCL to run this program is in member CCPRUN in the PDS.

Count the number of bytes put to each queue

The following JCL will use DFSORT facilities to process the file from the rexx program to add up the number of bytes put to each queue. This is in member COUNT of the PDS.

```
//TOOLRUN  EXEC PGM=ICETOOL,REGION=1024K
//TOOLMSG  DD SYSOUT=*
//DFSMSG   DD SYSOUT=*
//TOOLIN   DD *
    SORT FROM(IN) TO(TEMP1) USING(CTL1)
    DISPLAY FROM(TEMP1) LIST(OUT1) ON(5,48,CH) ON(53,4,BI)
//CTL1CNTL DD *
* Select the records which were put
  INCLUDE COND=(164,3,CH,EQ,C'Put')
* Sort by queue name
  SORT FIELDS=(108,48,CH,A)
* Only copy the queue name and size of user data to output record
  OUTREC FIELDS=(1,4,108,48,104,4)
* Add up the number of bytes procesed
  SUM FIELDS=(104,4,FI)
//IN       DD DISP=SHR,DSN=PAICE.CMT
//TEMP1    DD DISP=(NEW,PASS),DSN=&TEMP1,SPACE=(CYL,(50,10))
//OUT1     DD SYSOUT=*
```

This produces output like:

BA1	3605616
BA10	3572328
BA2	3612624
BA3	3579336
BA4	3572328
BA5	3491736
BA6	3574080
BA7	3532032
BA8	3577584
BA9	3539040
SYSTEM.ADMIN.CHANNEL.EVENT	186120
SYSTEM.ADMIN.QMGR.EVENT	384
SYSTEM.CHANNEL.SYNCQ	46488312

Extracting changes to objects

This section describes how do display changes to objects, such as by define or alter commands, or by an MQSET verb.

Rerun changes to commands

The following JCL will extract the data from the commands file, and will create a file suitable for using with CSQUTIL to replay the changes. This is in the member CMDS in the PDS.

```
//PAICED   JOB 1,MSGCLASS=H MSGLEVEL=(0,0)
//* Delete the old dataset
//S1       EXEC PGM=IEFBR14
//C        DD DISP=(MOD,DELETE),DSN=PAICE.TEMP1,SPACE=(CYL,(50,10))
//S1  EXEC PGM=SORT
//SYSIN   DD *
 OPTION COPY
 OUTFIL FNAMES=(C),INCLUDE=(175,1,CH,EQ,C'A'),OUTREC=(176,80),CONVERT
//SORTIN   DD DISP=SHR,DSN=PAICE.CMDS
//C        DD DISP=(NEW,CATLG),DSN=PAICE.TEMP1,SPACE=(CYL,(50,10))
//SYSOUT   DD SYSOUT=*
```

Note the column 175 is the column in the file(171) with the **B**efore or **A**fter image indicator, + 4 for the Record Descriptor word. To select the records which match the before image use INCLUDE=(175,1,CH,EQ,C'B')

The output from this is a file that looks like

```
Alter -
 CHANNEL('TEST') -
 CHLTYPE(RCVR)   -
 DESCR('Test change')
```

End of Document