How the Message ID and Correlation ID in an MQ Message Descriptor are handled in a Transmission Queue

IBM Techdoc: 7021177
http://www.ibm.com/support/docview.wss?rs=171&uid=swg27021177

Date last updated: 08-Dec-2011

Angel Rivera – rivera@us.ibm.com
IBM WebSphere MQ Support

+++ Objective +++

The objective of this techdoc is to show how the Message ID (MsgId) and the Correlation ID (CorrelId) in an MQ Message Descriptor (MQMD) are handled in a message that is sent from a Remote Queue Definition to a Transmission Queue and from there to the destination queue

This document uses a visualization of the original message shown as a smaller envelope inside a bigger envelope will help to better visualize the processing.

When a message is placed in a remote queue definition, the payload is placed in a small envelope that has MsgId=1 and CorrelId=0 in the MQ Message Descriptor. Then, when the message is forwarded to a transmission queue, a big envelope is made ready with MsgId=2 and CorrelId=0. Subsequently, when the small envelope is placed inside the big envelope, the MsgId=1 of the small envelope is assigned to the Correlation Id of the big envelope. In this case, the Correlation Id of the big envelope is set to CorrelId=1. This demonstrates how utilities that handle the big envelope know the Message ID of the payload in the small envelope.

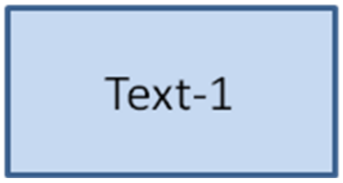
Related WebSphere Support Technical Exchange (WSTE) webcast:

A Day in the Life of a WebSphere MQ Transmission Queue
http://www-01.ibm.com/support/docview.wss?uid=swg27021403
Description: This webcast focuses on how the WebSphere® MQ message descriptor is handled in a message that is sent from a remote queue definition to a transmission queue. Additional topics related to the transmission queues are also covered, such as basic troubleshooting, clusters and triggering of channels.
Presented by: Angel Rivera
Level of Difficulty: Beginner   Date: 26 April 2011

++
+++ Chapter 1: Basic scenario of Put and Get of a message in a local queue
++

Let's start with the basic scenario of doing an MQ PUT of a message into a local queue and then doing an MQ GET of the same message from the local queue.

- The application that uses the MQ client prepares a text message.
This is represented as a piece of paper with the actual contents: **Text-1**



- The application invokes the MQ PUT command and provides the text message.
For example, the sample utility "amqsput" can be used to

$ amqsput Q1 QMGR
```
Sample AMQSPUT0 start
target queue is Q1
Text-1
Sample AMQSPUT0 end
```

- We can use the "amqsbcg" utility to display the message from the queue.

```
$ amqsbcg Q1 QMGR

=== begin of output of amqsbcg  ===
AMQSBCG0 - starts here
**********************
 MQOPEN - 'Q1'

MQGET of message number 1
****Message descriptor****
  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 1208
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120514D5F414E47454C49544F20B5493E4C20001F02'
  CorrelId : X'000000000000000000000000000000000000000000000000'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'QM1                                             '
  ** Identity Context
  UserIdentifier : 'rivera      '
  AccountingToken :
   X'16010515000000BA1E06D2603C3514885EDBC2EF030000000000000000000000B'
  ApplIdentityData : '                                 '
  ** Origin Context
  PutApplType    : '11'
  PutApplName    : 're MQ\java\jre\bin\javaw.exe'
  PutDate  : '20100714'    PutTime  : '23361704'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber   : '1'
  Offset         : '0'
  MsgFlags       : '0'
  OriginalLength : '-1'

****   Message      ****
 length - 6 bytes

 00000000:  5465 7874 2D31                         'Text-1          '

 No more messages
 MQCLOSE
=== end of output of amqsbcg  ===
```

Notice that the output displayed by amqsbcg shows 2 sections:

1: The Message Descriptor or MD (a good analogy is an envelope): it is a set of fields that are needed by MQ to properly handle the message. This set of fields (which do not include the actual message) is prepared by the MQ client code

2: The actual message or payload ("Text-1" in our example).

Two of the fields in the MD are of special interest for this techdoc and they are:
Message ID (MsgId)
Correlation ID (CorrelId)

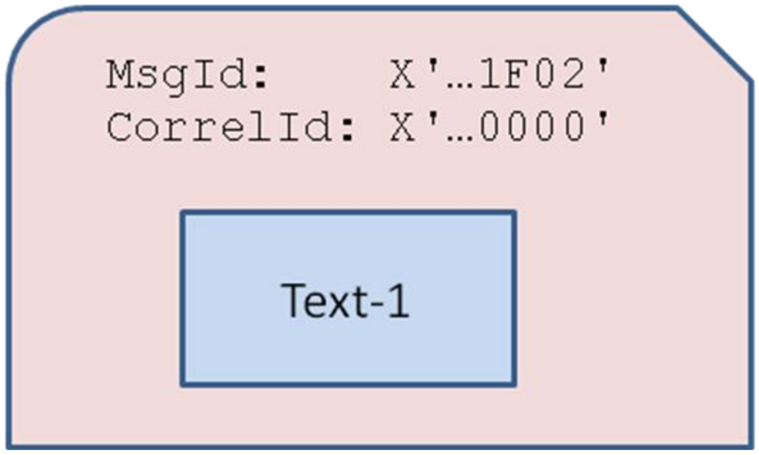
The MQ code generates an ID for each Message ID, such as:
MsgId :   X'414D5120514D5F414E47454C49544F20B5493E4C20001F02'

For short, let's identify this MsgId with the 4 last characters: 1F02
MsgId: X'...1F02'

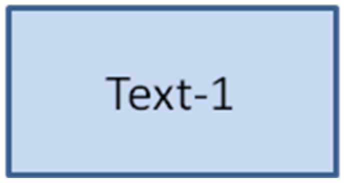
At this point, the Correlation ID is set to 0:
CorrelId : X'000000000000000000000000000000000000000000000000'

For short, let's identify this CORRELID with the 4 last characters: 0000
CorrelId: X'...0000'

The MD and the payload are represented as a traditional envelope used in paper mail, and the payload is the piece of paper that is stuffed inside the envelope.



When the receiving application performs an MQ GET, then the MQ code will receive the envelope (MD + Data) and will extract the data and discard the MQMD.
In this case, the actual message is:



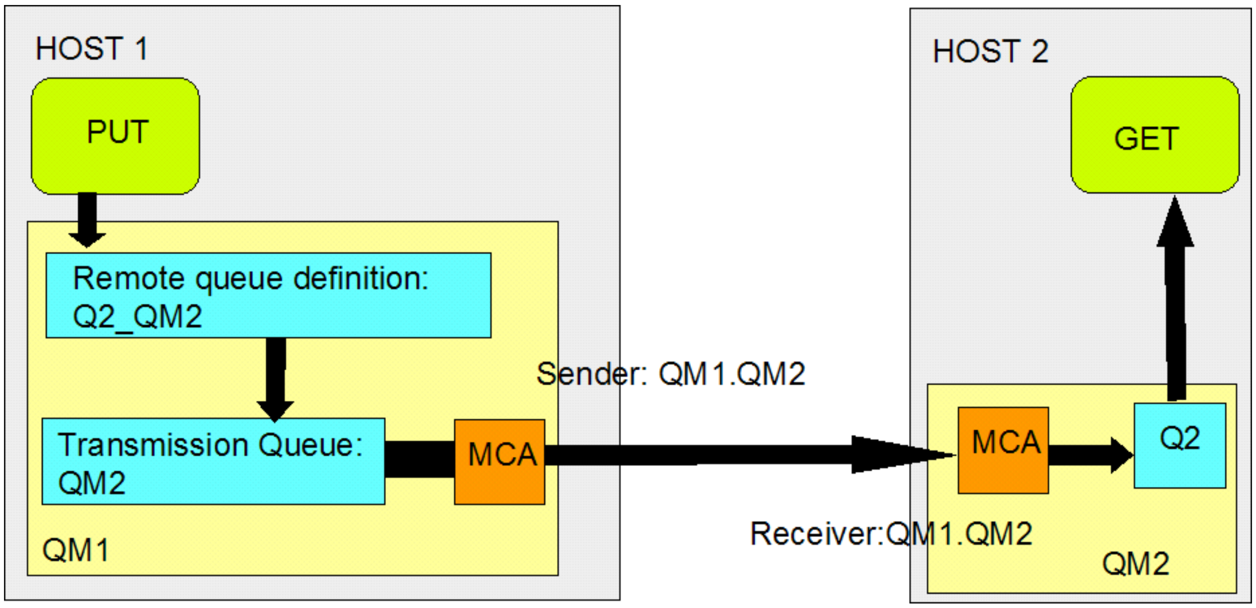
+++
+++ Chapter 2: Scenario of doing a Put into a Remote Queue Definition and message is placed in a Transmission Queue
+++

This scenario builds on top of the basic scenario and instead of doing an MQ PUT on a local queue, the PUT is done on a Remote Queue Definition which is connected to a Transmission Queue (XMITQ).

The topology for this example is as follows:

Queue Manager 1:	QM1
Remote Queue definition:	Q2_QM2
Transmission Queue:	QM2
Sender Channel:	QM1.QM2

Queue Manager 2:	QM2
Receiver Channel:	QM1.QM2
Local Queue:	Q2



Here is the processing:

Step 1: The application does the MQ PUT of the actual message "Text-2" into the Remote Queue Definition.

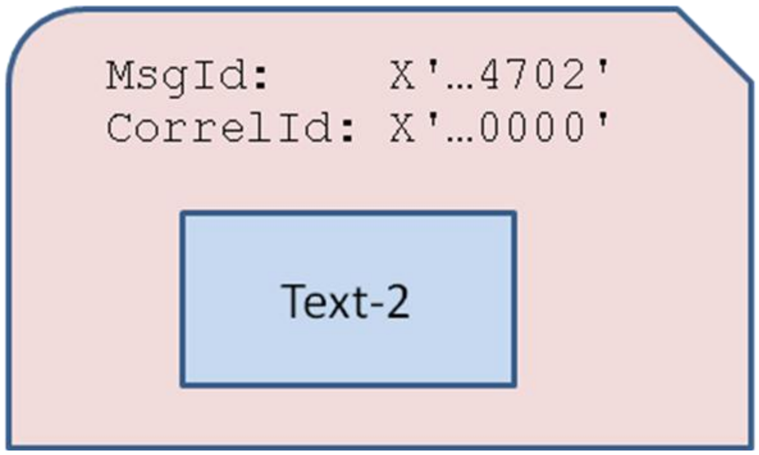
Technically speaking, the remote queue definition is NOT a local queue and it does NOT store messages. It can be seen as a transient local queue where there is some processing of the data and immediately it is forwarded to the Transmission Queue.

Thus, if we could "freeze" the processing at the Remote Queue Definition and pause enough to look at the data we will see that we have an envelope with data: the MQMD with the MsgId/CorrelId and the data of the actual message.

The MQ code generates an ID for the Message ID, such as:
  MsgId :   X'414D5120514D5F414E47454C49544F20B5493E4C20004702'
For short, let's identify this MsgId with the 4 last characters: 4702
  MsgId: X'...4702'

At this point, the Correlation ID is set to 0:
  CorrelId : X'000000000000000000000000000000000000000000000000'
For short, let's identify this CORRELID with the 4 last characters: 0000
  CorrelId: X'...0000'

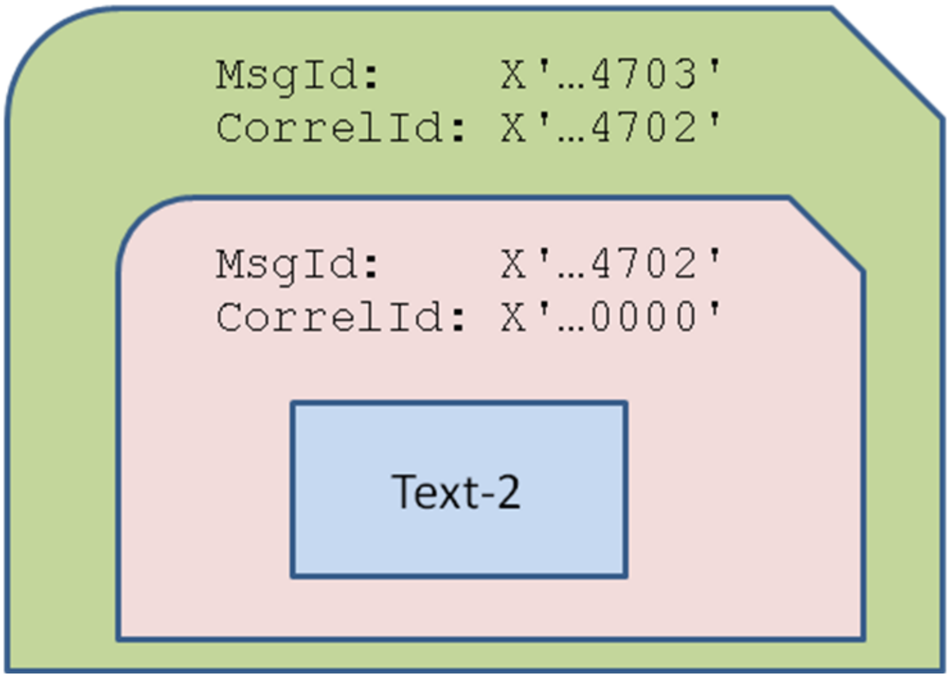
This is represented by:



Step 2: The message is forwarded from the Remote Queue Definition to the associated Transmission Queue.

This step is the one that is the source for confusion for some MQ customers.

When the message is forwarded from the Remote Queue Definition to the Transmission Queue, it is placed inside a bigger envelope and this is the one that is transmitted by the Message Channel Agent (MCA) of the Sender Channel of the local queue manager to the MCA of the Receiver Channel of the remote queue manager.

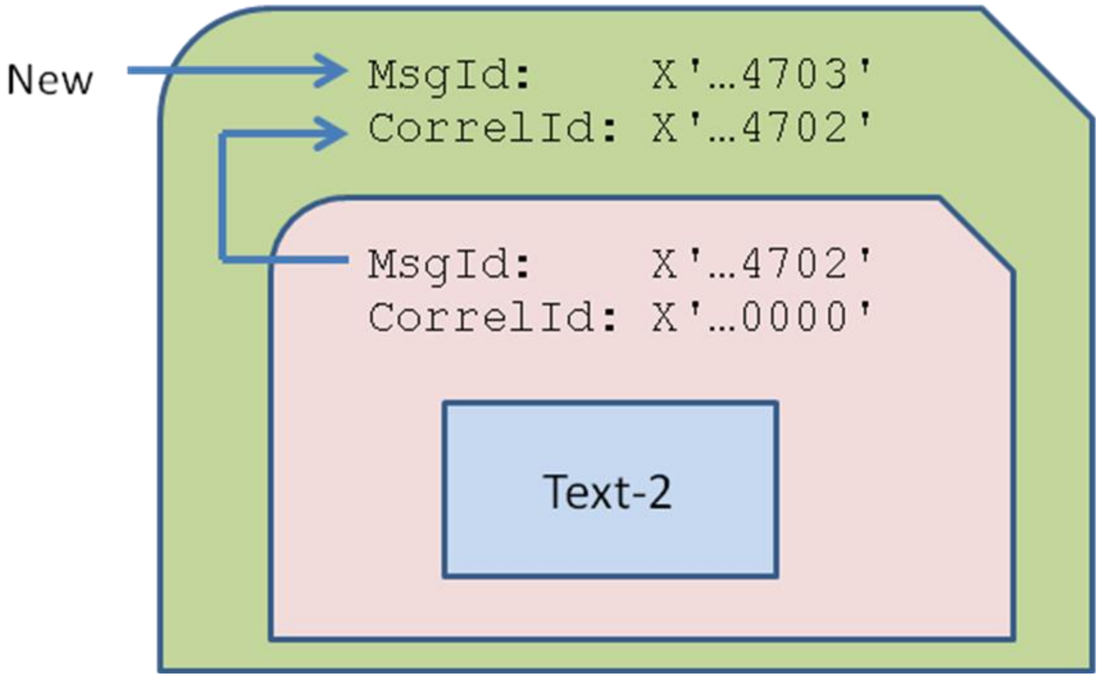


Notice that the bigger envelope has to have its own Message ID and it is different than other Message IDs already used. In this case, the MsgId for the bigger envelope is:
  MsgId: X'…4703'

Notice that the smaller envelope (original message), has its original MsgId:
  MsgId: X'…4702'

It is important for the MQ utilities that transmit the bigger envelope to know what is the message id of the original message (which is in the payload). For this reason, the CorrelId of the bigger envelope is used: the original MsgId (of the smaller envelope) is copied into the CorrelId of the bigger envelope):
  CorrelId: X'…4702'

The following picture shows the relationship:



The output of amqsbcg from the Transmission Queue shows the physical representation of the bigger envelope that has inside a smaller envelope.

Notice that the big envelope has a special format name:
  Format : 'MQXMIT '
This is also known as the Transmission Header.

$ amqsbcg QM2 QM1

```
=== begin of output of amqsbcg  ===
AMQSBCG0 - starts here
**********************
 MQOPEN - 'QM2'

 MQGET of message number 1
****Message descriptor****
  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 437
  Format : 'MQXMIT  '
  Priority : 0  Persistence : 0
  MsgId :    X'414D5120514D5F414E47454C49544F20B5493E4C20004703'
  CorrelId : X'414D5120514D5F414E47454C49544F20B5493E4C20004702'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'QM1                                             '
  ** Identity Context
  UserIdentifier : 'rivera      '
  AccountingToken :
   X'16010515000000BA1E06D2603C3514885EDBC2EF03000000000000000000000B'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType     : '7'
  PutApplName     : 'QM1                        '
  PutDate   : '20100714'    PutTime   : '23585193'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber    : '1'
  Offset          : '0'
  MsgFlags        : '0'
  OriginalLength : '-1'

****   Message      ****

 length - 434 bytes

00000000:  5851_4820 0100 0000 5132 2020 2020 2020 'XQH ....Q2      '
00000010:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000020:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000030:  2020 2020 2020 2020 514D 5F46 5231 2020 '        QM2   '
00000040:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000050:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000060:  2020 2020 2020 2020 4D44 2020 0100 0000 '        MD  ....'
00000070:  0000 0000 0800 0000 FFFF FFFF 0000 0000 '........    ....'
00000080:  2202 0000 B804 0000 4D51 5354 5220 2020 '".......MQSTR   '
==> These 2 lines show the original MsgId of the "smaller envelope"
==> 00000090:  0000 0000 0000 0000 414D 5120 514D 5F41 '........AMQ QM_A'
                                   ******************
==> 000000A0:  4E47 454C 4954 4F20 B549 3E4C 2000 4702 'NGELITO .I>L .G.'
               ***************************************
000000B0:  0000 0000 0000 0000 0000 0000 0000 0000 '................'
000000C0:  0000 0000 0000 0000 0000 0000 2020 2020 '............    '
000000D0:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
000000E0:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
000000F0:  2020 2020 2020 2020 2020 2020 514D 5F41 '            QM_A'
00000100:  4E47 454C 4954 4F20 2020 2020 2020 2020 'NGELITO         '
00000110:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000120:  2020 2020 2020 2020 2020 2020 7269 7665 '            rive'
00000130:  7261 2020 2020 2020 1601 0515 0000 00BA 'ra      ........'
00000140:  1E06 D260 3C35 1488 5EDB C2EF 0300 0000 '...`<5.ê^.......'
00000150:  0000 0000 0000 000B 2020 2020 2020 2020 '........        '
00000160:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000170:  2020 2020 2020 2020 0B00 0000 7265 204D '        ....re M'
00000180:  515C 6A61 7661 5C6A 7265 5C62 696E 5C6A 'Q\java\jre\bin\j'
00000190:  6176 6177 2E65 7865 3230 3130 3037 3134 'avaw.exe20100714'
==> These 2 lines show the actual payload:
==> 000001A0:  3233 3538 3531 3933 2020 2020 5465 7874  '23585193    Text'
                                            *********            ****
==> 000001B0:  2D32                                     '-2              '
               ****                                      **
```

```
No more messages
 MQCLOSE
 MQDISC
=== end of output of amqsbcg  ===
```

As you can see, the format of the message in the Transmission Queue is a bit confusing!

Lines 00000090 and 000000A0 show the MsgId of the "smaller envelope" and it is:
MsgId: X'…4702'

Lines 000001A0 and 000001B0 show the actual text message:
Text-2

The following tries to show the representation of the big envelope and small envelope.
Notice that the XHQ "eye catcher" in the first bytes of the payload indicate the beginning of the Transmission Header, which denotes it to be the "big" envelope.

BIGGER ENVELOPE (Notice the XQH "eye catcher")

```
****Message descriptor****
  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 437
  Format : 'MQXMIT  '
  Priority : 0  Persistence : 0
  MsgId :     X'414D5120514D5F414E47454C49544F20B5493E4C20004703'
  CorrelId : X'414D5120514D5F414E47454C49544F20B5493E4C20004702'
…
****   Message       ****
length - 434 bytes
00000000:  5851 4820 0100 0000 5132 2020 2020 2020 'XQH ....Q2      '
00000010:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000020:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000030:  2020 2020 2020 2020 514D 5F46 5231 2020 '        QM2   '
00000040:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
```

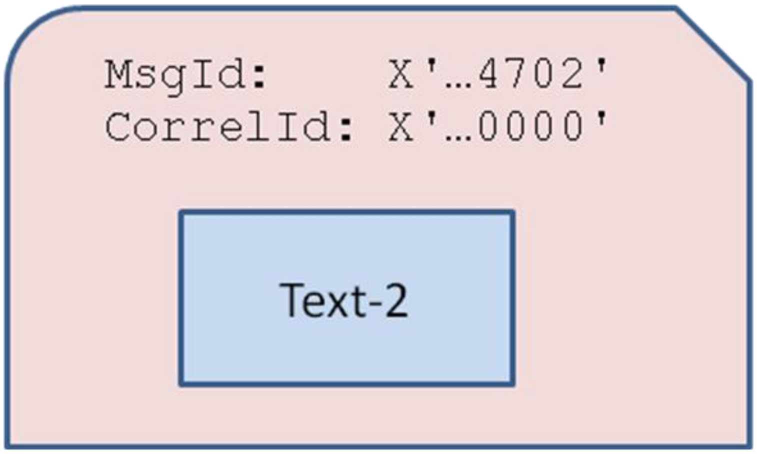
SMALLER ENVELOPE

```
00000050:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000060:  2020 2020 2020 2020 4D44 2020 0100 0000 '        MD  ....'
00000070:  0000 0000 0800 0000 FFFF FFFF 0000 0000 '........    ....'
00000080:  2202 0000 B804 0000 4D51 5354 5220 2020 '".......MQSTR   '
00000090:  0000 0000 0000 0000 414D 5120 514D 5F41 '........AMQ QM_A'
000000A0:  4E47 454C 4954 4F20 B549 3E4C 2000 4702 'NGELITO .I>L .G.'
000000B0:  0000 0000 0000 0000 0000 0000 0000 0000 '................'
000000C0:  0000 0000 0000 0000 0000 0000 2020 2020 '............    '
000000D0:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
000000E0:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
000000F0:  2020 2020 2020 2020 2020 2020 514D 5F41 '            QM_A'
00000100:  4E47 454C 4954 4F20 2020 2020 2020 2020 'NGELITO         '
00000110:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000120:  2020 2020 2020 2020 2020 2020 7269 7665 '            rive'
00000130:  7261 2020 2020 2020 1601 0515 0000 00BA 'ra      ........'
00000140:  1E06 D260 3C35 1488 5EDB C2EF 0300 0000 '...`<5.ê^.......'
00000150:  0000 0000 0000 000B 2020 2020 2020 2020 '........        '
00000160:  2020 2020 2020 2020 2020 2020 2020 2020 '                '
00000170:  2020 2020 2020 2020 0B00 0000 7265 204D '        ....re M'
00000180:  515C 6A61 7661 5C6A 7265 5C62 696E 5C6A 'Q\java\jre\bin\j'
00000190:  6176 6177 2E65 7865 3230 3130 3037 3134 'avaw.exe20100714'
000001A0:  3233 3538 3531 3933 2020 2020 5465 7874 '23585193    Text'
000001B0:  2D32                                    '-2              '
***   END of Message ***
```

Step 3: At the remote queue manager side, when the MCA of the Receiver channel completes the processing of receiving the bigger envelope, the MCA extracts the smaller envelope and places the smaller envelope into the local queue.

The empty bigger envelope is then discarded.

Thus, the local queue in the remote queue manager sees the original message:



The following is the output from amqsbcg for the receiving queue in the remote queue manager:

$ amqsbcg Q2 QM2

```
=== begin of output of amqsbcg  ===

AMQSBCG0 - starts here
**********************
 MQOPEN - 'Q2'

 MQGET of message number 1
****Message descriptor****

  StrucId  : 'MD  '  Version : 2
  Report   : 0  MsgType : 8
  Expiry   : -1  Feedback : 0
  Encoding : 546  CodedCharSetId : 1208
  Format : 'MQSTR   '
  Priority : 0  Persistence : 0
  MsgId : X'414D5120514D5F414E47454C49544F20B5493E4C20004702'
  CorrelId : X'000000000000000000000000000000000000000000000000'
  BackoutCount : 0
  ReplyToQ       : '                                                '
  ReplyToQMgr    : 'QM1                                             '
  ** Identity Context
  UserIdentifier : 'rivera      '
  AccountingToken :
   X'16010515000000BA1E06D2603C3514885EDBC2EF03000000000000000000000B'
  ApplIdentityData : '                                '
  ** Origin Context
  PutApplType    : '11'
  PutApplName    : 're MQ\java\jre\bin\javaw.exe'
  PutDate  : '20100714'    PutTime  : '23585193'
  ApplOriginData : '    '

  GroupId : X'000000000000000000000000000000000000000000000000'
  MsgSeqNumber   : '1'
  Offset         : '0'
  MsgFlags       : '0'
  OriginalLength : '-1'

****   Message      ****

 length - 6 bytes

00000000:  5465 7874 2D32                          'Text-2          '

 No more messages
 MQCLOSE
 MQDISC
=== end of output of amqsbcg  ===
```

+++ end +++