BEA Tuxedo® 10.0

MQ Adapter

Version 10.0
Document Released: September 28, 2007

Contents

4. Running the Tuxedo MQ Adapter

CHAPTER 1

Understanding the Tuxedo MQ Adapter

This chapter contains the following topics:

Accessing WebSphere MQ With and Without the Tuxedo MQ Adapter

WebSphere MQ is an XA compliant resource manager, and it is possible to access WebSphere MQ directly from Tuxedo server programs that issue MQ APIs such as MQGET and MQPUT1. Such server programs can be placed in a group associated with the WebSphere MQ Resource Manager, and no adapter program is required to access such servers.

Some customers have existing WebSphere MQ programs that they want to access from BEA Tuxedo without having to modify their MQ programs or to write any MQ code on the Tuxedo side of their applications. For these customers, BEA has developed the MQ Adapter for Tuxedo 10.0. The BEA MQ Adapter for Tuxedo supports both transactional and non-transactional access to WebSphere MQ applications and data.

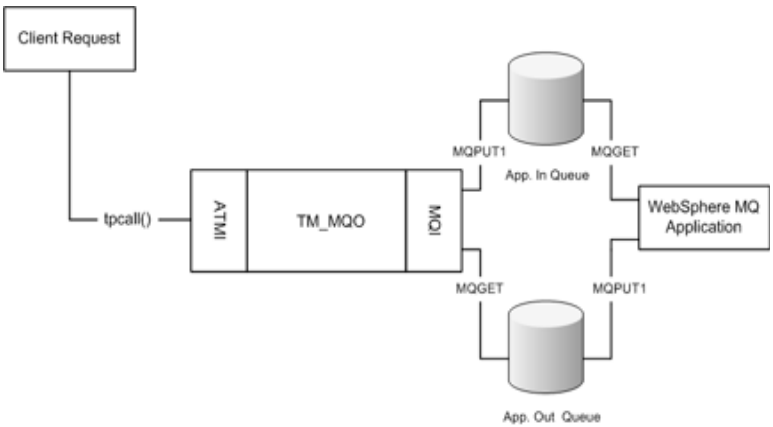
Overview of the BEA MQ Adapter for Tuxedo

The BEA MQ Adapter for Tuxedo 10.0 provides communication between IBM WebSphere MQ applications and BEA TUXEDO applications. The MQ Adapter consists of three TUXEDO servers:

- The TM_MQO server manages Tuxedo to WebSphere MQ requests.
- The TM_MQI server manages WebSphere MQ to Tuxedo requests.
- The TMQUEUE_MQM server handles `tpenqueue()` and `tpdequeue()` requests.

These servers are managed in the TUXEDO environment. The following diagram illustrates the flow of data when a `tpcall ()` is issued from a TUXEDO client to an WebSphere MQ application.

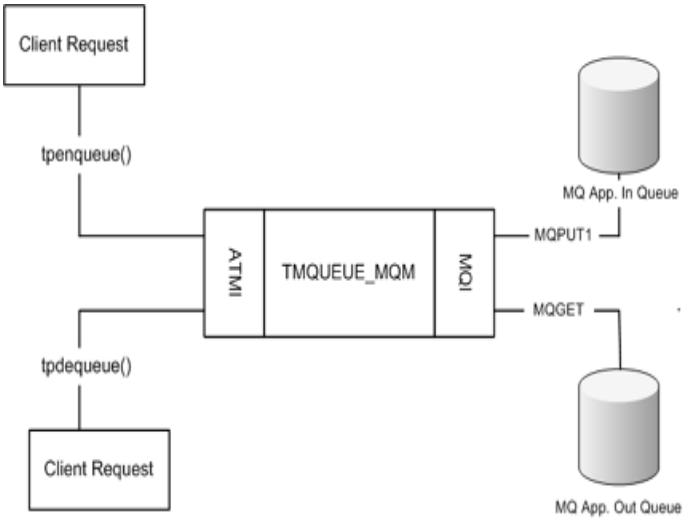
Figure 1-1 Data Flow for tpcall () from TUXEDO to WebSphere MQ



The TUXEDO client calls the service advertised by the Tuxedo to WebSphere MQ server (TM_MQO). The configuration of the service dictates the queue manager name, queue name, and reply queue related to the service. The MQ Adapter then places the request on the WebSphere MQ queue to be processed by the WebSphere MQ application. The MQ Adapter waits for the reply message on the output queue. When the MQ Adapter receives the reply, it returns the response data to the client's outstanding `tpcall()`.

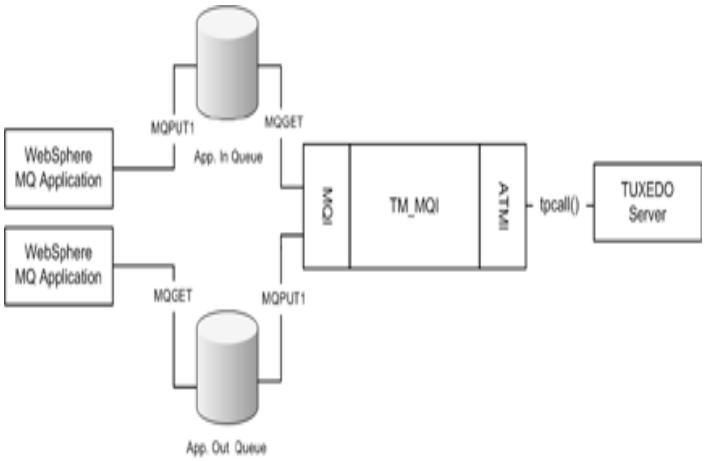
The TMQUEUE _MQM server handles `tpenqueue()` and `tpdequeue()` requests from clients who want to place messages on WebSphere MQ queues. The following diagram shows the enqueuing and dequeuing message flows:

Figure 1-2 Message Flow for `tpenqueue ()` and `tpdequeue ()`



The WebSphere MQ to Tuxedo server (TM_MQI), processes service requests from WebSphere MQ applications to TUXEDO servers. The following diagram illustrates the data flow for inbound requests:

Figure 1-3 Data Flow for Inbound Service Requests



The WebSphere MQ to Tuxedo server (TM_MQI) monitors specified queues for requests. When TM_MQI receives a request, it issues a `tpcall()` request to the designated server. It then places the response data in the reply queue specified in the original request

CHAPTER 2

Building the Tuxedo MQ Adapter

This topic contains the following sections:

Overview

Object files for the Tuxedo MQ Adapter are included as part of the standard Tuxedo distribution. After customizing the `$TUXDIR/udataobj/RM` file with any values specific to the MQ installation being used, the Tuxedo administrator uses the `buildmqadapter(1)` command to build the MQ adapter executables. The Tuxedo administrator also needs to use the `buildtms(1)` command to build the TMS for the WebSphere MQ Resource manager. This chapter describes the process of building the MQ Adapter executables.

Pre-Build Considerations

The MQ Adapter for Tuxedo software is currently available on HP-UX, Solaris, AIX, Linux, and Windows versions of Tuxedo. Ports of Tuxedo to other platforms will include the MQ Adapter if WebSphere MQ is supported by IBM on the target platform.

The Tuxedo administrator should complete the following tasks before building the MQ Adapter TMS and adapter servers:

- Make sure the proper maintenance version of WebSphere MQ is installed on the target platform. For more information about the required maintenance version of WebSphere MQ on each supported platform, see the BEA Tuxedo 10.0 Platform Data Sheets.
- Set up the required WebSphere MQ resource manager.

Setting Up the WebSphere MQ XA Compliant Resource Manager

The Tuxedo installation program attempts to add a sensible default line for WebSphere MQ 6.0 to the `$TUXDIR/udataobj/RM` file when installing Tuxedo. For example, on 32-bit Solaris, the installation program adds the following lines:

```
# WebSphere MQ 6.0
MQSeries_XA_RMI:MQRMIXASwitchDynamic: /opt/mqm/lib/libmqmxa.so
/opt/mqm/lib/libmqm.so
```

(On other hardware platforms, the default library suffixes and locations may be different.)

As the Tuxedo administrator, you should review this information to determine if it is correct for your installation. You may need to modify the RM file information

- If WebSphere MQ is installed in a non-standard location.
- If your application is to be linked with WebSphere MQ client libraries instead of WebSphere MQ server libraries.
- If your application prefers to use the static XA switch MQRMIXASwitch instead of the dynamic XA switch MQRMIXASwitchDynamic.

On Windows NT, there is no standard installation location for WebSphere MQ, so the variable `%MQMDIR%` is used in the `%TUXDIR%\udataobj\RM` file. This variable should be set before building the MQ Adapter servers.

If using a 64-bit version of Tuxedo, be sure to link with libraries in the WebSphere MQ lib64 directory and not in the 32-bit lib directory, and be sure to link with the mqmxa64 library and not the mqmxa library.

Building the Tuxedo MQ Adapter Servers

To build the MQ adapter servers, execute the following commands as the Tuxedo administrator with write permission in the $TUXDIR/bin directory:

```
buildmqadapter -v
```

This command will install the TM_MQI, TM_MQO, and TMQUEUE_MQM servers in the `$TUXDIR/bin` directory. The -v option indicates that buildmqadapter will write the buildserver and compiler command lines used to link the 3 servers as part of its output.

Alternatively the MQ adapter servers can be built one at a time using the following commands:

```
buildTM_MQI -v -o $TUXDIR/bin/TM_MQI
buildTM_MQO -v -o $TUXDIR/bin/TM_MQO
buildTMQUEUE_MQM -v -o $TUXDIR/bin/TMQUEUE_MQM
```

(If building the servers one at a time, it is possible to specify $APPDIR instead of $TUXDIR/bin as the output directory.)

Building the TMS Server for the WebSphere MQ Resource Manager

Run the following command while in the `$TUXDIR/bin` directory to create the resource manager executable:

```
buildtms -o MQXA -r MQSeries_XA_RMI
```

(An alternate file name can be specified instead of MQXA if desired, and the output file can be placed in either $TUXDIR/bin or in $APPDIR.)

Specifying MQ Adapter Server Group in the UBBCONFIG file

Add the following lines to the `*GROUPS` section of the UBBCONFIG file for the group used for the MQ Adapter for Tuxedo servers.

```
MQ_GROUP_NAME TMSNAME=MQXA TMSCOUNT=3
       OPENINFO="MQSeries_XA_RMI:BEA.TEST.MANAGER"
```

In the above example, `MQ_GROUP_NAME` is the name of the group in which the MQ adapter servers are defined. Your application may use whatever name it wants for this group. The number of TMS servers in this group is 3. BEA.TEST.MANAGER is the name of the queue manager being used, and another name should be substituted if your queue manager is named something else.

Post-Installation Procedures

After you complete the installation procedures for the MQ Adapter for Tuxedo, make sure the following has been done before continuing with the MQ Adapter configuration procedures:

- WebSphere MQ programs you intend to use must be built, must be accessible to run, and must have triggers defined, if applicable.
- All WebSphere MQ queues you intend to use must be defined and must be accessible. For more information on these procedures, refer to WebSphere MQ documentation.

CHAPTER 3

Configuring the Tuxedo MQ Adapter

This chapter contains the following topics:

Configuring the Tuxedo Servers

The Tuxedo MQ Adapter consists of three Tuxedo servers: a Tuxedo to WebSphere MQ server (TM_MQO), a WebSphere MQ to Tuxedo server (TM_MQI), and a server for handling `tpenqueue()` and `tpdequeue()` requests (TMQUEUE_MQM). You must identify each of the servers that you intend to use in the TUXEDO `UBBCONFIG` file. In addition, the MQ Adapter requires that certain parameters be set for each server. You define these parameters in a server configuration file. A sample configuration file for each server is shown in this document. You can use these sample configuration files as a base and insert the specific information required for your environment.

The following sections describe how to identify each of the servers in the TUXEDO `UBBCONFIG` file, and how to set up the configuration file required by the MQ Adapter for each of these servers.

Configuring the Tuxedo to WebSphere MQ Server (TM_MQO)

`TM_MQO` routes all requests for WebSphere MQ services from TUXEDO clients. It interacts with WebSphere MQ via the Message Queue Interface (MQI) by enqueuing and dequeuing service

requests and responses. The MQI is a common application programming interface that all WebSphere MQ applications implement.

You define the TM_MQO server in the SERVERS section of the TUXEDO UBBCONFIG file as follows:

Listing 3-1 Syntax for TM_MQO Server Definition in UBBCONFIG

```
*SERVERS
TM_MQO SRVGRP="identifier" SRVID="number"
CLOPT="-- -C configuration_file_name"
```

For `SRVGRP`, `SRVID`, and `CLOPT` parameter syntax and definitions information, see File Formats, Data Descriptions, MIBs, and System Processes Reference.

`CLOPT= "-- -C Tuxedo to WebSphere MQ configfile"` specifies the server's configuration file.

A configuration file provides a list of services and their associated parameters to the server at startup. See Creating the Server Configuration Files for an explanation of the parameters you need to define.

Defining FML32 Fields for the Tuxedo to WebSphere MQ Server

Tuxedo has defined FML32 fields for TM_MQO in the `$TUXDIR/udataobj/Usysfl32` field definition table and in the `$TUXDIR/include/Usysfl32.h` header file. The Tuxedo MQ Adapter uses FML32 functions to manipulate fielded buffers related to errors. For more information about FML32 programming, see the *Programming a BEA Tuxedo ATMI Application Using FML*.

The syntax for the field definition table for TM_MQO is as follows:

Listing 3-2 Syntax for field definition table for TM_MQO

```
# name                  number    type    flags   comments
TPMQ_ADAPTER_ERR        `n'     string      -        -
```

```
TPMQ_ADAPTER_ERR_CODE   `n'     string      -       -
TPMQ_APP_ERR            `n'     string      -       -
```

These FML32 fields are used as follows:

- `TPMQ_ADAPTER_ERR:` stores the details of the errors specific to the Tuxedo MQ Adapter.
- `TPMQ_ADAPTER_ERR_CODE:` stores the category code for the adapter-specific errors.
- `TPMQ_APP_ERR:` stores the details of the errors specific to WebSphere MQ.

These fields are defined in `$TUXDIR/udataobj/Usysfl32` and in `$TUXDIR/include/Usysfl32.h`. (If an interoperating client running Tuxedo 9.1 or earlier wants to use these fields, the client will need to define these fields using the same field numbers as in later versions of Tuxedo.).

The environment variables `FLDTBLDIR32` and `FIELDTBLS32` should be set so that the system will look in `$TUXDIR/udataobj/Usysfl32` for FML32 field definitions.

Configuring the WebSphere MQ to Tuxedo Server (TM_MQI)

TM_MQI forwards messages requested from WebSphere MQ applications to TUXEDO services. The application queues a request to a designated queue that is monitored by TM_MQI. The requested service is specified in the message descriptor. Like TM_MQO, TM_MQI must perform data and semantic transformations on the data stored on a queue before delivering it to a service. It must do the same to replies.

You define the TM_MQI server in the SERVERS section of the TUXEDO UBBCONFIG file as follows:

Listing 3-3 Syntax for TM_MQI Definition in UBBCONFIG

```
*SERVERS
TM_MQI SRVGRP="identifier" SRVID="number" REPLYQ=N
CLOPT="--  -C configuration_file_name"
```

For information about the SRVGRP, SRVID, REPLYQ, and CLOPT parameter syntax and definitions, refer to the BEA TUXEDO Reference Manual.

`CLOPT= "-- -C WebSphere MQ to Tuxedo configfile"` specifies the server's configuration file.

A configuration file provides a list of queues and services and their associated parameters to the server at startup. See Creating the Server Configuration Files for an explanation of the parameters you need to define.

Configuring the TMQUEUE_MQM Server

TMQUEUE_MQM processes the `tpenqueue()` and `tpdequeue()` requests from TUXEDO applications that need to send or retrieve data to or from an WebSphere MQ queue.

You define the TMQUEUE_MQM server in the SERVERS section of the TUXEDO UBBCONFIG file as follows:

Listing 3-4 Syntax for TMQUEUE_MQM Definition in UBBCONFIG

```
*SERVERS

TMQUEUE_MQM SRVGRP="identifier" SRVID="number" REPLYQ=N CLOPT="--  -C
configuration_file_name"
```

For information about the SRVGRP, SRVID, REPLYQ, and CLOPT parameter syntax and definitions, refer to the BEA TUXEDO Reference Manual.

`CLOPT= "-- -C enqueue/dequeue configfile"` specifies the server's configuration file.

A configuration file provides a list of queues and their associated parameters to the server at startup. See Creating the Server Configuration Files for an explanation of the parameters you need to define.

Creating the Server Configuration Files

You must create a configuration file for each of the three MQ Adapter servers. The documentation shows sample files you can use as a base for creating your own configuration files. You can substitute the parameter settings in the sample files with the settings required for your own environment.

Creating the Tuxedo to WebSphere MQ Server Configuration File

The TM_MQO.CFG file controls the operation of the Tuxedo to WebSphere MQ server (TM_MQO). Following are the sections of the TM_MQO configuration file and the parameters you can define for each section. A sample configuration file follows the descriptions.

Note: TM_MQO.CFG is a generic filename. You can name this file anything you choose, but the filename must match the -C configuration_file_name parameter you specify in the TUXEDO UBBCONFIG file. (See Configuring the Tuxedo to WebSphere MQ Server (TM_MQO) for instructions on configuring the TM_MQO server in the UBBCONFIG file.)

The TM_MQO configuration file is divided into the following required sections:

- Queue_Manager

  Defines the queue manager name and logical ID.

  Note: The configuration file can only have one QUEUE_MANAGER section.

- SERVICES

  Defines various parameters for services and messages.

The TM_MQO configuration file has one optional section:

- SERVER

  Defines minimum and maximum settings for services and messages.

  Note: The configuration file can only have one SERVER section.

These sections and the parameters within each section can be in any order in the configuration file, as long as the required sections and parameters are defined.

Defining the TM_MQO QUEUE_MANAGER Section (Required)

The syntax for the QUEUE_MANAGER section of the TM_MQO configuration file is as follows:

Listing 3-5 Syntax for QUEUE_MANAGER section

```
*QUEUE_MANAGER
```

```
NAME=string
LQMID=string
```

Note: The configuration file can only have one QUEUE_MANAGER section.

Required Parameter

The following parameter must be included in the QUEUE_MANAGER section of the TM_MQO configuration file.

`NAME= string`
: Specifies the name of the WebSphere MQ Queue Manager. Can contain up to 48 characters, all uppercase. Refer to WebSphere MQ documentation for the specific format of Queue Manager names.

Optional Parameter

The following parameter is optional:

`LQMID= string`
: Specifies the logical Queue Manager ID. Can contain up to 8 alphanumeric characters, upper and lowercase. If this parameter is specified, it will be included in certain userlog messages.

Defining the TM_MQO SERVICE Section (Required)

The syntax for the SERVICE section of the TM_MQO configuration file is as follows:

Listing 3-6 Syntax for SERVICE section

```
*SERVICE
NAME=string
MQNAME=string
FORMAT=string
TRAN={Y/N}
MAXMSGLEN=integer
REPLYTOQ=string
TIMEOUT=integer
EXPIRE=integer
```

```
PRIORITY=integer
INFIELD=string
OUTFIELD=string
OUTFIELDVNAME=string
```

Required Parameters

The following parameters must be included in the SERVICE section of the TM_MQO configuration file.

`NAME = string`

Specifies the TUXEDO Service Name. The maximum length of this parameter is the same as the maximum length of a Tuxedo service name, which is currently 15 characters of mixed case. Note that the NAME string should be unique among all of the *SERVICE sections in a particular TM_MQO configuration file.

`MQNAME = string`

Specifies the name of the WebSphere MQ Queue. Can contain up to a maximum of 48 characters, all uppercase. Refer to WebSphere MQ documentation for the specific format of Queue names.

`MAXMSGLEN = integer`

Specifies the maximum message length expected for buffers received by this service. This is a required parameter unless DEFMAXMSGLEN is specified in the SERVER section. (See Defining the SERVER Section (Optional) for more information).

`TIMEOUT = integer`

Specifies the amount of time, in seconds, that is allowed for processing of the indicated MQ service. (This is the maximum amount of time that MQGET will wait for a message on the reply queue after the MQ service is invoked with MQPUT1.) The value must be greater than or equal to 0. A value of 0 indicates that the service will not be timed out. This is a required parameter unless DEFTIMEOUT is specified in the SERVER section. (See Defining the SERVER Section (Optional) for more information).

Optional Parameters

The following parameters are optional.

`EXPIRE = integer`

Specifies the amount of time, in tenths of a second, the expiry time for the MQ service request message sent with MQPUT1. The value must be greater than or equal to 0. A value of 0 indicates that the message will not be expired. The default value is 0.

`PRIORITY = integer`

Specifies the WebSphere MQ priority at which requests to this service will be sent. WebSphere MQ has traditionally allowed values of 0 through 9 for priorities. If not specified, the default queue priority represented in MQ by the value MQPRI_PRIORITY_AS_Q_DEF will be used.

`FORMAT = string`

Maps to the WebSphere MQ Message Descriptor Format. Can contain up to 8 characters. Refer to WebSphere MQ documentation for specific format of the Message Descriptor Format field.

`TRAN = {Y|N}`

Specifies whether the Service is transactional. Y indicates transactional; N indicates non-transactional. The default value is N. (A service defined as transactional must be called within a Tuxedo transaction. A service defined as non-transactional must be called outside of a Tuxedo transaction. Messages sent to transactional services are sent with persistence MQPER_PERSISTENT and messages sent to non-transactional services are sent with the default queue persistence MQPER_PERSISTENCE_AS_Q_DEF.)

`REPLYTOQ = string`

Specifies the name of the WebSphere MQ Reply To Queue. Can contain up to 48 characters, all uppercase. Refer to WebSphere MQ documentation for the specific format of Queue names.

`INFIELD = string`

Specifies the FML32 field name of the input data in an FML32 buffer. If you specify this parameter, requests for this service may pass FML32 buffers for input, and the data in INFIELD will be passed as the actual service data. (A request to a service which specifies INFIELD may still pass a non-FML32 data type as input, which will be treated exactly as if INFIELD had not been specified, but a request to a service which specifies INFIELD may not pass NULL data.)

If INFIELD is a FLD_VIEW32 field, TM_MQO will pass only the actual VIEW32 data to MQ and will not pass the FVIEWFLD structure that was returned by FML32. The MQ program must know which view it is expecting to receive. The application should not use fields of type FLD_PTR, FLD_FML32, FLD_MBSTRING, or FLD_FML for the TM_MQO INFIELD or OUTFIELD. These field types will not result in meaningful data being passed to or from the MQ application.

If INFIELD is not specified, then TM_MQO will accept only buffers not of type FML32 as input, and will pass this data directly to WebSphere MQ.

`OUTFIELD = string`

Specifies the FML32 field name of the output data to be used when building an FML32 response buffer. If you specify this parameter, you must also specify the INFIELD parameter. The omission of this parameter causes the response data field name to match the name specified by the INFIELD parameter.

When OUTFIELD is a FLD_VIEW32 field, the data returned by MQ should be VIEW32 data, and not a FVIEWFLD structure. In order to pass VIEW32 data back to a Tuxedo application, TM_MQO must internally fill in a FVIEWFLD structure to add the VIEW32 data to the FML32 reply buffer, and therefore TM_MQO must know the name of the view to be returned. The TM_MQO server will obtain this information in the following manner:

- If the OUTFIELDVNAME parameter is set, the value of this parameter will be used as the view32 name.
- Otherwise, if the INFIELD was a FLD_VIEW32 field, the same viewname used in the INFIELD will be used for the OUTFIELD.
- Otherwise, TM_MQO will log an error message and call `tpreturn()` with TPFAIL.

The application should not use fields of type FLD_PTR, FLD_FML32, FLD_MBSTRING, or FLD_FML for the TM_MQO INFIELD or OUTFIELD. These field types will not result in meaningful data being passed to or from the MQ application.

`OUTFIELDVNAME = view32name`

Specifies the name of the VIEW32 subtype used for output to the FML32 OUTFIELD (or the FML32 INFIELD if OUTFIELD is not configured.) OUTFIELDVNAME is allowed only if OUTFIELD and/or INFIELD are also specified.

Defining the TM_MQO SERVER Section (Optional)

The syntax for the SERVER section of the TM_MQO configuration file is as follows:

Listing 3-7 Syntax for SERVER section

```
*SERVER
DEFTIMEOUT=integer
DEFMAXMSGLEN=integer
MINMSGLEVEL=integer
MAXMSGLEVEL=integer
```

Required Parameters

There are no required parameters in the *SERVER section.

Optional Parameters

`MINMSGLEVEL = integer`
Specifies minimum debug level desired for userlog messages.

`MAXMSGLEVEL = integer`
Specifies maximum debug level desired for userlog messages.

Notes:

- The absolute maximum value for `MAXMSGLEVEL` is 100, however, the highest meaningful value for customer debugging is 30. All the higher values should (only) be used by Oracle Support for deeper analysis or debugging.
- Parameters `MINMSGLEVEL` and `MAXMSGLEVEL` are intended only for application debugging, and setting `MAXMSGLEVEL` to a large value can result in excessive output to the ULOG, therefore they have to be used with caution.

Level range for customers:

- [10]: The log of message flow, mainly `MQPUT` log, or major function entry points only.
- [20]: Detailed level of tracing. The information of MQ, Service, User, CompCode, and Reason are printed.
- [30]: `MQGET` log in `TM_MQI`.

`DEFMAXMSGLEN = integer`
Specifies the default maximum message length expected for buffers received by Tuxedo to WebSphere MQ services. The DEFMAXMSGLEN parameter is overridden by the Service MAXMSGLEN parameter.

`DEFTIMEOUT = integer`
Specifies the default amount of time, in seconds, that is allowed for processing of WebSphere MQ services. The value must be greater than or equal to 0. A value of 0 indicates that the service will not be timed out. The DEFTIMEOUT is overridden by the Service TIMEOUT parameter.

Sample TM_MQ0.CFG

Listing 3-8 Sample Configuration File for TM_MQ0 Server

```
*SERVER
   DEFTIMEOUT=60
   DEFMAXMSGLEN=4096
   MINMSGLEVEL=10
   MAXMSGLEVEL=30

*QUEUE_MANAGER
   LQMID=QM1
   NAME=BEA.TEST.MANAGER

*SERVICE
   NAME=MQTest1
   TIMEOUT=20
   MQNAME=TEST.SAMPLE.ECHO
   REPLYTOQ=TEST.REPLY.QUEUE

*SERVICE
   NAME=MQTest2
   TIMEOUT=15
   REPLYTOQ=TEST.REPLY.QUEUE
   MQNAME=TEST.SAMPLE.TOUPPER

*SERVICE
   NAME=MQTest3
   MQNAME=TEST.NOREPLY.QUEUE
   TRAN=Y
   MAXMSGLEN=1024
```

Creating the WebSphere MQ to Tuxedo Server Configuration File

The TM_MQI.CFG file controls the operation of the WebSphere MQ to Tuxedo server (TM_MQI). Following are the sections of the TM_MQI configuration file and the parameters you can define for each section. A sample configuration file follows the descriptions.

Note: TM_MQI.CFG is a generic filename. You can name this file anything you choose, but the filename must match the -C configuration_file_name parameter you specify in the TUXEDO UBBCONFIG file. (See Configuring the WebSphere MQ to Tuxedo Server (TM_MQI) for instructions on configuring the TM_MQI server in the UBBCONFIG file.)

The TM_MQI configuration file is divided into the following required sections:

- Queue_Manager

  Defines the queue manager name and logical ID.

  Note: The configuration file can only have one QUEUE_MANAGER section.

- QUEUE

  Defines various parameters for the incoming message queue.

- SERVICE

  Defines various parameters for the TUXEDO service.

The TM_MQI configuration file has one optional section:

- SERVER

  Defines the minimum and maximum settings for services and messages.

  Note: The configuration file can only have one SERVER section.

These sections and the parameters within each section can be in any order in the configuration file, as long as the required sections and parameters are defined.

Defining the TM_MQI QUEUE_MANAGER Section (Required)

The syntax for the QUEUE_MANAGER section of the TM_MQI configuration file is as follows:

Listing 3-9 Syntax for QUEUE_MANAGER section

```
*QUEUE_MANAGER
NAME=string
LQMID=string
```

Note: The configuration file can only have one QUEUE_MANAGER section.

Required Parameter

The following parameter must be included in the QUEUE_MANAGER section of the TM_MQO configuration file.

`NAME = string`
: Specifies the name of the WebSphere MQ Queue Manager assigned to the queue to be processed by the server. Can contain up to 48 characters, all uppercase. Refer to WebSphere MQ documentation for the specific format of Queue Manager names.

Optional Parameter

The following parameter is optional in the QUEUE_MANAGER section:

`LQMID = string`
: Specifies the logical Queue Manager ID. Can contain up to 8 alphanumeric characters, upper and lowercase. If this parameter is specified, it will be included in certain userlog messages.

Defining the TM_MQI QUEUE Section (Required)

The syntax for the QUEUE section of the TM_MQI configuration file is as follows:

Listing 3-10 Syntax for QUEUE section

```
*QUEUE
MQNAME=string
MAXMSGLEN=integer
SERVICE=string
```

Required Parameters

The following parameter must be included in the QUEUE section of the TM_MQI configuration file:

`MQNAME = string`

Specifies the name of a WebSphere MQ queue to be processed by the server. Can contain up to 48 characters, all uppercase. Refer to WebSphere MQ documentation for specific format of queue names. Note that the MQNAME string should be unique among all of the *QUEUE sections in a particular TM_MQI configuration file.

Optional Parameters

The following parameters are optional:

`MAXMSGLEN = integer`

Specifies the maximum message length expected for MQ queue buffers received by this server. Any message longer than this length read from the WebSphere MQ queue will be sent to the dead letter queue. If this parameter is not set, the value of the SERVER section DEFMAXMSGLEN parameter will be used. If the SERVER section DEFMAXMSGLEN parameter is also not set, the `MaxMsgLength` value defined for the WebSphere MQ queue will be used.

`SERVICE = string`

Specifies the name of the Tuxedo service that will process requests received on this queue. If not specified, the value of the Format field in the MQMD structure received from WebSphere MQ will be used as the Tuxedo service name. In either case, the service name must be defined in the TM_MQI SERVICE section in order for the Tuxedo service to be invoked.

There are no other parameters in the QUEUE section.

Defining the TM_MQI SERVICE Section (Required)

The syntax for the SERVICE section of the TM_MQI configuration file is as follows:

Listing 3-11 Syntax for SERVICE section

```
*SERVICE

NAME=string

FORMAT=string

TRAN={Y|N}
```

Required Parameters

The following parameters must be included in the SERVICE section of the TM_MQI configuration file.

`NAME = string`
: Specifies the TUXEDO Service Name. The maximum length of this parameter is the same as the maximum length of a Tuxedo service name, which is currently 15 characters of mixed case.

`FORMAT = string`
: Maps the WebSphere MQ Message Descriptor Format field to the service being called. Can contain up to 8 characters. Refer to WebSphere MQ documentation for specific format of the Message Descriptor Format field. Note that the FORMAT string should be unique among all of the *SERVICE sections in a particular TM_MQI configuration file. If the same FORMAT is used for two or more Tuxedo services, it is undefined which service the MQ adapter will map that particular FORMAT to.

Optional Parameter

The following parameter is optional.

`TRAN = {Y|N}`
: Specifies whether the Service is transactional. Y indicates transactional; N indicates non-transactional. Default value is N.
(TM_MQI starts a Tuxedo transaction before reading each message from a WebSphere MQ queue. If TRAN=Y, the call to the Tuxedo service is made within this transaction. If TRAN=N, then the call to the Tuxedo service is made with the TPNOTRAN flag.)

Defining the TM_MQI SERVER Section (Optional)

The syntax for the SERVER section of the TM_MQI configuration file is as follows:

Listing 3-12 Syntax for SERVER section

```
*SERVER
MINMSGLEVEL=integer
MAXMSGLEVEL=integer
DEFMAXMSGLEN=integer
```

```
TPESVCFAILDATA={Y|N}
POLINTERVAL=integer
```

Note: The configuration file can only have one SERVER section.

Required Parameters

There are no required parameters in the SERVER section.

Optional Parameters

`MINMSGLEVEL = integer`
Specifies minimum debug level desired for userlog messages.

`MAXMSGLEVEL = integer`
Specifies maximum debug level desired for userlog messages.

Notes:

- The absolute maximum value for `MAXMSGLEVEL` is 100, however, the highest meaningful value for customer debugging is 30. All the higher values should (only) be used by Oracle Support for deeper analysis or debugging.
- Parameters `MINMSGLEVEL` and `MAXMSGLEVEL` are intended only for application debugging, and setting `MAXMSGLEVEL` to a large value can result in excessive output to the ULOG, therefore they have to be used with caution.

Level range for customers:

- [10]: The log of message flow, mainly `MQPUT` log, or major function entry points only.
- [20]: Detailed level of tracing. The information of MQ, Service, User, CompCode, and Reason are printed.
- [30]: `MQGET` log in `TM_MQI`.

`DEFMAXMSGLEN = integer`
Specifies the default value for the MAXMSGLEN parameter in the *QUEUE section.

`TPESVCFAILDATA = {Y|N}`
Specifies the manner of handling TPESVCFAIL errors from Tuxedo with associated non-zero length data. The default value is N.

If TPESVCFAILDATA=N, then any error occurring on `tpcall()` will cause TM_MQI to send the original data sent to `tpcall` to the MQ dead letter queue, if any.

If TPESVCFAILDATA=Y, then TM_MQI will return data to the caller's reply queue and will not send the original message to the MQ dead letter queue if `tpcall()` fails with TPESVCFAIL and a non-zero length reply. TM_MQI will set Feedback in MQMD to MQFB_APPL_FIRST+TPESVCFAIL in this case. If a Tuxedo service call fails with zero-length data or with any value of `tperrno` other than TPESVCFAIL, the original message will be sent to the Dead Letter Queue.

`POLINTERVAL = integer`

Specifies the polling interval in milliseconds of the MQ Series in queue for `TM_MQI` server.

If not specified, the default value is 500. The recommended range is 50 to 500. Setting this one to lower value (0 to 49) may imply a high server load and to a higher value (501 and up) may lead to performance application decrease as the MQ Series incoming request queue will be check out less often and thus introduce delay to process the MQ Series request.

Sample TM_MQI.CFG

Listing 3-13 shows a TM_MQI Server configuration file example

Listing 3-13 TM_MQI Server Configuration File Example

```
*SERVER
   MINMSGLEVEL=1
   MAXMSGLEVEL=100

*QUEUE_MANAGER
   LQMID=QM1
   NAME=BEA.TEST.MANAGER

*SERVICE
   NAME=SvcToupper
   FORMAT=UPPER

*SERVICE
   NAME=SvcEcho
   FORMAT=ECHO
```

```
   TRAN=Y

*QUEUE
   MQNAME=TEST.SAMPLE.QUEUE1
   MAXMSGLEN=200
*QUEUE
   MQNAME=TEST.SAMPLE.QUEUE2
```

Creating the enqueue/dequeue Server Configuration File

The TMQUEUE_MQM.CFG file controls the operation of the server that handles `tpenqueue` and `tpdequeue` requests (TMQUEUE_MQM). Following are the sections of the TMQUEUE_MQM configuration file and the parameters you can define for each section. A sample configuration file follows the descriptions.

Note: TMQUEUE_MQM.CFG is a generic filename. You can name this file anything you choose, but the filename must match the -C `configuration_file_name` parameter you specify in the TUXEDO UBBCONFIG file. (See Configuring the TMQUEUE_MQM Server for instructions on configuring the TMQUEUE_MQM server in the UBBCONFIG file.)

The TMQUEUE_MQM configuration file is divided into the following required sections:

- Queue_Manager

  Defines the queue manager name and logical ID.

  Note: The configuration file can only have one QUEUE_MANAGER section.

- QUEUE

  Defines various parameters for the message queue.

The TMQUEUE_MQM configuration file has one optional section:

- SERVER

  Defines the minimum and maximum debug level and the default maximum message length.

  Note: The configuration file can only have one SERVER section.

These sections and the parameters within each section can be in any order in the configuration file, as long as the required sections and parameters are defined.

Defining the TMQUEUE_MQM QUEUE_MANAGER Section (Required)

The syntax for the QUEUE_MANAGER section of the TMQUEUE_MQM configuration file is as follows:

Listing 3-14 Syntax for QUEUE_MANAGER section

```
*QUEUE_MANAGER
NAME=string
LQMID=string
```

Note: The configuration file can only have one QUEUE_MANAGER section.

Required Parameter

The following parameter must be included in the QUEUE_MANAGER section of the TMQUEUE_MQM configuration file:

`NAME = string`
: Specifies the name of the WebSphere MQ Queue Manager assigned to the queue to be processed by the server. Can contain up to 48 characters, all uppercase. Refer to WebSphere MQ documentation for the specific format of Queue Manager names.

Optional Parameter

`LQMID = string`
: Specifies the logical Queue Manager ID. Can contain up to 8 alphanumeric characters, upper and lowercase. If this parameter is specified, it will be included in certain userlog messages.

Defining the TMQUEUE_MQM QUEUE Section (Required)

Listing 3-15 shows the syntax for the QUEUE section of the TMQUEUE_MQM configuration file.

Listing 3-15 Syntax for QUEUE section

```
*QUEUE
MQNAME=string
```

```
TUXNAME=string
MAXMSGLEN=integer
```

Required Parameters

The following parameters must be included in the QUEUE section of the TMQUEUE_MQM configuration file.

`MQNAME = string`
: Specifies the name of a WebSphere MQ queue to be processed by the server. Can contain up to 48 characters, all uppercase. Refer to WebSphere MQ documentation for specific format of queue names.

`TUXNAME = string`
: Specifies the name of the queue used for ATMI enqueue/dequeue calls. The maximum length of this parameter is the same as the maximum length of a Tuxedo queue name, which is currently 15 characters of mixed case. Note that the TUXNAME string should be unique among all of the *QUEUE sections in a particular TMQUEUE_MQM configuration file.

`MAXMSGLEN = integer`
: Specifies the maximum message length expected for buffers received from this queue. This parameter overrides the SERVER DEFMAXMSGLEN parameter. If DEFMAXMSGLEN is not specified, then MAXMSGLEN is required.

Defining the TMQUEUE_MQM SERVER Section (Optional)

Listing 3-16 shows the syntax for the SERVER section of the TMQUEUE_MQM configuration file.

Listing 3-16 Syntax for SERVER section

```
*SERVER
MINMSGLEVEL=integer
MAXMSGLEVEL=integer
DEFMAXMSGLEN=integer
```

Note: The configuration file can only have one SERVER section.

Required Parameters

There are no required parameters in the SERVER section.

Optional Parameters

`MINMSGLEVEL = integer`
Specifies minimum debug level desired for userlog messages.

`MAXMSGLEVEL = integer`
Specifies maximum debug level desired for userlog messages.

Notes:

- The absolute maximum value for `MAXMSGLEVEL` is 100, however, the highest meaningful value for customer debugging is 30. All the higher values should (only) be used by Oracle Support for deeper analysis or debugging.
- Parameters `MINMSGLEVEL` and `MAXMSGLEVEL` are intended only for application debugging, and setting `MAXMSGLEVEL` to a large value can result in excessive output to the ULOG, therefore they have to be used with caution.

Level range for customers:

- [10]: The log of message flow, mainly `MQPUT` log, or major function entry points only.
- [20]: Detailed level of tracing. The information of MQ, Service, User, CompCode, and Reason are printed.
- [30]: `MQGET` log in `TM_MQI`.

`DEFMAXMSGLEN = integer`
Specifies the default maximum message length expected for buffers received from WebSphere MQ queues. The DEFMAXMSGLEN parameter is overridden by the QUEUE section MAXMSGLEN parameter.

Sample TMQUEUE_MQM.CFG

Listing 3-17 shows TMQUEUE_MQM Server configuration file example.

Listing 3-17 TMQUEUE_MQM Server Configuration File Example

```
*SERVER
   DEFMAXMSGLEN=4096
```

```
*QUEUE_MANAGER
   LQMID=QM1
   NAME=BEA.TEST.MANAGER

*QUEUE
   TUXNAME=MQTest1
   MQNAME=TEST.NOREPLY.QUEUE

*QUEUE
   TUXNAME=MQEcho
   MQNAME=TEST.SAMPLE.ECHO
   MAXMSGLEN=2048

*QUEUE
   TUXNAME=MQReply
   MQNAME=TEST.REPLY.QUEUE
```

Configuring the WebSphere MQ Queue Manager

You must configure the WebSphere MQ queue manager in order to run the MQ Adapter for Tuxedo. Refer to your WebSphere MQ documentation for specific instructions on configuring queue managers.

CHAPTER 4

Running the Tuxedo MQ Adapter

This chapter contains the following topics:

Booting the Servers

The MQ Adapter servers boot as part of the TUXEDO application using standard TUXEDO utilities, such as `tmboot`. The MQ Adapter reads the server configuration files and attempts to connect to the specified queue manager. Once the MQ Adapter establishes a connection with the queue manager, the Tuxedo to WebSphere MQ server (TM_MQO) advertises the services associated with that queue manager.

Initiating a TUXEDO-to-WebSphere MQ Request

TUXEDO clients can call services advertised by the Tuxedo to MQ server (TM_MQO). A TUXEDO-to-WebSphere MQ request consists of the following actions.

1. The TUXEDO client initiates a request for a service advertised by TM_MQO.
2. The MQ Adapter uses the WebSphere MQ Message Queue Interface (MQI) to forward these requests to the appropriate WebSphere MQ queue.

3. The MQ Adapter retrieves response data (if any) from the designated reply queue and returns this data to the TUXEDO client.

Initiating a WebSphere MQ-to-TUXEDO Request

WebSphere MQ applications can request TUXEDO services via the WebSphere MQ to Tuxedo server TM_MQI. An WebSphere MQ-to-TUXEDO request consists of the following actions.

1. The WebSphere MQ application queues a message requesting the service to a designated queue.
2. The MQ Adapter retrieves the message from the incoming queue.
3. The MQ Adapter forwards the message data to the appropriate service.
4. The MQ Adapter places response data (if any) on the specified reply queue.

Sending and Receiving Queued Messages to and from WebSphere MQ

The TMQUEUE_MQM server processes `tpenqueue` and `tpdequeue` requests from TUXEDO client and server processes.

Processing a tpenqueue Request

The syntax for a `tpenqueue` request is as follows:

```
tpenqueue (qspace, qname, qctl, data, len, flags)
```

A `tpenqueue` request requires all parameters shown previously. Following are brief descriptions of these parameters. For more information, refer to the BEA TUXEDO Programmer's Guide.

`QSPACE`
: The name of the service advertised by TMQUEUE_MQM. This value can be overwritten in the TUXEDO UBBCONFIG file using the `-s` option. For example:

  ```
  -s myname :MQMQUEUE
  ```

  Default is MQMQUEUE.

`QNAME`
: The name of the queue where you want the adapter to place or retrieve messages. Can contain up to 15 characters. Corresponds to a WebSphere MQ queue mapped from the

TUXNAME parameter in the QUEUE section of the TMQUEUE_MQM configuration file.

QCTL

Provides additional information about the message. The supported options you can define for tpenqueue requests are as follows:

TPNOFLAGS
: No options apply to this message

TPQPRIORITY
: Message priority. TUXEDO values for this field can be from 1-100. The MQ values can be 0-9 and are mapped to TUXEDO values as follows: 0=TUXEDO 1-10, 1=TUXEDO 11-20, etc.

TPQCORRID
: Correlation ID. Identifies the response to a request. TUXEDO supports up to 32 bytes. The MQ Adapter supports up to 24 bytes, so TUXEDO values are truncated on tpenqueue requests.

TPQREPLYQ
: Name of the queue where you want to place reply messages. Can contain up to 15 characters.

TPQMSGID
: Return the message ID generated when the message is placed on the queue. TUXEDO supports up to 32 bytes. The MQ Adapter supports up to 24 bytes and pads the ID with nulls.

TPQDELIVERYQOS
: Specifies the delivery quality of service. If a value of TPQQOSPERSISTENT is specified, the MQ persistence is set to MQPER_PERSISTENT. If a value of TPQQOSNONPERSISTENT is specified, the MQ persistence is set to MQPER_NOT_PERSISTENT. If a value of TPQQOSDEFAULTPERSIST is specified or if TPQDELIVERYQOS is not specified, then MQ persistence is set to MQPER_PERSISTENT if the request was made in a transaction and is set to MQPER_PERSISTENCE_AS_Q_DEF otherwise.

TPQEXPTIME_NONE
: Specifies that the queued message should not expire. This corresponds to a value of MQEI_UNLIMITED in the MQMD structure passed to WebSphere MQ.

TPQEXPTIME_ABS
: Specifies an absolute expiration time for the queued message. This value is converted to a format understood by WebSphere MQ and passed to MQ via the Expiry field of the WebSphere MQ MQMD structure.

TPQEXPTIME_REL
: Specifies a relative expiration time for the queued message. This value is converted to a format understood by WebSphere MQ and passed to MQ via the Expiry field of the WebSphere MQ MQMD structure.

Note: The TPQFAILUREQ, TPQBEFOREMSGID, TPQTOP, TPQTIME_REL, TPQTIME_ABS, and TPQREPLYQOS options are not supported for `tpenqueue` requests to TMQUEUE_MQM

DATA
: Data to be placed on the queue

LEN
: Length of the data to be placed on the queue

FLAGS
: Defines the flag settings for the message. The supported flags for `tpenqueue` requests are as follows:

  TPNOTRAN
  : Messages from callers that are in transaction mode are not queued within the same transaction as the caller.

Processing a tpdequeue Request

The syntax for a `tpdequeue` request is as follows:

```
tpdequeue (qspace, qname, qctl, data, len, flags)
```

A `tpdequeue` request requires all parameters shown previously. Following are brief descriptions of these parameters. For more information, refer to the BEA TUXEDO Programmer's Guide.

QSPACE
: The name of the service advertised by TMQUEUE_MQM. This value can be overwritten in the TUXEDO UBBCONFIG file using the - s option. For example:

  ```
  -s myname :MQMQUEUE
  ```

  The default value is MQMQUEUE

QNAME
: The name of the queue where you want the adapter to place or retrieve messages. Can contain up to 15 characters. Corresponds to a WebSphere MQ queue mapped from the TUXNAME parameter in the QUEUE section of the TMQUEUE_MQM configuration file.

QCTL

Provides additional information about the message. The supported options you can define for tpdequeue requests are as follows:

TPNOFLAGS

No options apply to this message

TPQGETBYMSGID

Dequeues the message with the specified message ID, if available. The specified message ID is truncated to 24 bytes and passed in the MsgId field of the MQMD structure when calling the WebSphere MQ MQGET API.

TPQGETBYCORRID

Dequeues the message with the specified correlation ID.

TPQPRIORITY

Message priority. TUXEDO values for this field can be from 1-100. The MQ values can be 0-9 and are mapped to TUXEDO values as follows: 0=TUXEDO 1-10, 1=TUXEDO 11-20, etc.

TPQCORRID

Correlation ID. Identifies the response to a request. TUXEDO supports up to 32 bytes. MQ supports up to 24 bytes, so this ID will always fit in 32 bytes supported by Tuxedo. The remaining 8 bytes are set to null characters.

TPQREPLYQ

Name of the queue where you want to place reply messages. Can contain up to 15 characters.

TPQMSGID

Return the message ID generated when the message is placed on the queue. MQ supports up to 24 bytes, so this ID will always fit in 32 bytes supported by Tuxedo. The remaining 8 bytes are set to null characters.

TPQPEEK

If this flag is set, the specified message is read but is not removed from the queue. This flag implies the TPNOTRAN flag has been set for the tpdequeue() operation. That is, non-destructive dequeuing is non-transactional. Note that it is not possible to read messages enqueued or dequeued within a transaction before the transaction completes.

TPQDELIVERYQOS

If this flag is set, the call to tpdequeue() is successful, and the message was queued with a delivery quality of service, then the flag is stored in ctl->delivery_qos. If the MQGET call indicates a Persistence of

MQPER_NOT_PERSISTENT, ctl->delivery_qos is set to TPQQOSNONPERSISTENT. Otherwise, it is set to TPQQOSPERSISTENT.

Note: The TPQWAIT option is not supported for `tpdequeue` requests to TMQUEUE_MQM

`DATA`

Data to be placed on the queue

`LEN`

Length of the data to be placed on the queue

`FLAGS`

Defines the flag settings for the message. The supported flags for `tpdequeue` requests are as follows:

`TPNOTRAN`

Messages from callers that are in transaction mode are not dequeued within the same transaction as the caller.