Transactional MQ

Version	Date	Purpose	Author
1.0	11th Mar 2014	Initial Version	S. Dhilip Kumar

Table of Contents:

Introduction:

This document defines the protocol in which message queue and its clients will communicate with each other. The message queue server will bind to one port and will keep listening. Each client will connect to the server and perform various operation.

Connection Life Cycle:

Current design keeps all the client connections active until the client themselves disconnect.

Protocol Type:

This is a Request - Response in sequence protocol. For this initial prototype we do not implement any sort of re-try mechanism. Client or server sends a message and waits if they

expect a resonpse otherwise they continue performing their jobs.

Operations:

Here is the list of operations that are allowed via this custom protocol.

* Each Operation is represented via a table.
* Each row in the table represents one tcp message either sent from or to server.
* Each row has three column
* First column represents the origin of the message
* Second column represents the message
* Third column represents the size of the message in bytes
* Message to server are Request represented with “-->” symbol
* Message from server are response represented with “<--” symbol

CREATE:

Source	Message	Len
-->	OPERATION TYPE (for CREATE = 0)	1
-->	Queue Name Length (N)	4
-->	Queue Name	N
<--	ACK (0 = Created, 1 = Error)	1

DELETE:

Source	Message	Len
-->	OPERATION TYPE (for DELETE = 1)	1
-->	Queue Name Length (N)	4
-->	Queue Name	N
<--	ACK (0 = Deleted, 1 = Error)	1

OPEN

Source	Message	Len

-->	OPERATION TYPE (for OPEN = 2)	1
-->	Queue Name Length (N)	4
-->	Queue Name	N
<--	MQID (if Invalid = “<NIL>...followed by 59 0’s)	64

The resultant message queue id is nothing but sha256 message digest of the queue name. You should parse the first 5 charecters result to know if its a valid message or not.

CLOSE

Source	Message	Len
-->	OPERATION TYPE (for CLOSE = 3)	1
-->	MQID	64
<--	ACK (0 = Closed, 1 = Failed)	N

En Queue

Source	Message	Len
-->	OPERATION TYPE (for ENQ = 4)	1
-->	MQID	64
-->	MSG Length (N)	4
-->	MSG	N
<--	ACK (1= Sucess, 0 = Failed)	1

De Queue

Source	Message	Len
-->	OPERATION TYPE (for DNQ = 5)	1
-->	MQID	64
<--	MSG Length (N) (if error N = -1)	4
<--	MSG	N

If there are no messages in the queue Message Length is set to -1 which means there is

nothing to send as message, hence client wont have to read the next in TCP recive buffer.

Transaction Start

Source	Message	Len
-->	OPERATION TYPE (for TS = 5)	1
<--	ACK (0 = Sucess, 1= Failed)	1

Transaction End

Source	Message	Len
-->	OPERATION TYPE (for TE = 6)	1
<--	ACK (0 = Sucess, 1= Failed)	1

Transaction Abort

Source	Message	Len
-->	OPERATION TYPE (for TA = 7)	1
<--	ACK (0 = Sucess, 1= Failed)	1

SELECT

Source	Message	Len
-->	OPERATION TYPE (for SEL = 5)	1
-->	MQID	64
<--	Number of messages (-1 if error)	8

If Queue is empty it retuns ‘0’ if Queue is not available or closed then it returns -1.