.NET 환경에서의 MSMQ 테스팅

전재규, 석승학, 유재형

KT 네트워크 기술연구소

{Jkchun, Suksh,styoo}@kt.co.kr

MSMQ TESTING IN THE .NET ENVIRONMENT

Jae-Kyu Chun[O], Seung-Hak Suk, Jae-Hyoung Yoo

KT Network Technology Laboratory

요 약

We need to communicate with other applications using various methods and when communicating, we need to guarantee delivery. In the .NET environment, we generally use and develop MSMQ which creates outgoing queues for messages waiting to be sent and incoming queues for messages waiting to be received. MSMQ is message-oriented middleware whose queues, according to the developer, are very reliable, simple and more efficient than other skills. However, when developing a huge system, we found out that some problems occur when operating various queues(private/public). This paper shows the results of various tests using MSMQ, such as the status of mqsvc, CPU or memory usage, and whether or not the messages are sent and received. Also, we learned how to use MSMQ in order to develop Message Queuing-based applications.

1. Introduction

The Microsoft Message Queuing (MSMQ) technology enables applications running at different times to communicate across heterogeneous networks and systems that may be temporarily offline. Applications send messages to the queues and read messages from the queues. MSMQ provides guaranteed message delivery, efficient routing, security, and priority-based messaging. You don't have to think about the details, and the system architecture takes care of the queuing processes even if the client and server are not running at the same time. It can be used to implement solutions for both asynchronous and synchronous messaging scenarios. MSMQ allows applications to use components that communicate with one another using queued messages[3][4].

Software products that have these features are often referred to in the industry as message-queuing software, store-and-forward software, or message-oriented *middleware*. *Message Queuing is not a database* and does not provide database functionality, but can be used together with a database, for example, for message storage.

System administrators can use Message Queuing to efficiently manage large, complex computers networks and message queues. Through Message Queuing, you can achieve more reliable communication and more efficient use of network resources. Developers can focus on business logic instead of networking issues because Message Queuing effectively provides guaranteed network communication[1][4].

Applications can use MSMQ to send and process messages regardless of whether the receiving application is running or reachable over the network. This might be because of network problems or planned occurrences, such as mobile users who only connect to the network occasionally. Applications can use transactional messaging in MSMQ, and in this delivery mode, MSMQ makes sure that messages are delivered exactly on time, and in the order in which they were sent.

However, we found out that in cases of generating send-receive messages between outside systems, as well as when communicating between components inside systems when the receiving applications are not working or delaying, problems can occur in the whole system due to the vast accumulation of messages in the Queue. Therefore we aim to make sure of MSMQ's stability through thorough MSMQ testing.

This paper is organized as follows; The first section provides a brief overview of MSMQ and the test environment, as well as the partial MSMQ application source. The next section shows the various test results when applied to multiple test cases and also provides the status of the system such as CPU, memory, mqsvc service and send-receive messages. The last section concludes with the overall problems and concerns regarding MSMQ when developing queuing-based applications.

2. Test Environment

We have been developed a huge system using an applications server(AP) and a DataBase(DB) server, Biztalk server, which supports the operation system. In particular, direct oracle access, web service, BTS(BizTalk Server) EAI(Enterprise Application Integration), socket, XML RPC(Remote Procedure Call) and MSMQ are used as the communication methods between the systems. Among these, the MSMQ communication method is the most appropriate to use in developing applications easily.
To create an accurate testing environment, we configured