

Middleware Resource Center

---

# WebSphere MQ Discipline: Preparing for the Worst

21 September 2004

**A Middleware  
Resource  
Center  
Whitepaper**

Series edited by: Bruce  
Olsen, Executive  
Consultant, Defining  
Technology, Inc.



Defining Technology, Inc.

---

## WebSphere MQ Discipline: Preparing for the Worst

by

Bruce Olsen, Executive Consultant  
Defining Technology, Inc.  
21 September 2004

### Introduction

In the best of all possible worlds, nothing would ever fail, life would go on its merry way, and you'd never have to worry that anything would go wrong with your MQSeries queue managers or the hardware they run on. Unfortunately, we are sometimes played cruel tricks. When that happens, those lucky few of us who have prepared will begin to get the messages through again with the shortest possible outage.

Amongst developers and administrators, those who take a disciplined approach to their tasks experience fewer failures and lose the least time to nuisances like the inability to locate the latest version of a script or backing out changes. This is no less true when managing MQSeries queue managers and their associated objects. In my work, I see far too little exercise of what I like to refer to as the systems management disciplines. I've seen delays in code delivery for want of adequate source code control. I've seen lack of good change and revision management lead to delivery of incomplete binaries. I've seen system administrators frantically looking everywhere for a good copy of a script that got deleted or configuration information under the pressure of rebuilding a server after a failure. Somehow we forget that it takes very little effort to mitigate the effects of the bad things that will inevitably happen.

I'd like to discuss some things you as an MQSeries practitioner can do, even in the absence of some of the commercially available tools, to prepare for the worst.

### Backup

Where WebSphere MQ (since MQSeries Version 5) sets up the basic queue manager objects when the queue manager is created, Version 2 shipped with an MQSeries Command script called `amqscoma.tst` to be used for that purpose. The advantage of a separate script is that it can be customized to local needs. Even though WMQ doesn't include such a script, creating one is easy. IBM has made a SupportPac available that lets you save all objects of a queue manager as well as some of the configuration settings. It queries the attributes of all the objects defined to a queue manager and saves

## WebSphere MQ Discipline: Preparing for the Worst

---

them to a file which you can modify and save for use in subsequently recreating a queue manager. (To get SupportPac MS03, use your browser to access <http://www-306.ibm.com/software/integration/support/supportpacs/>. You will need a C compiler that is supported for your platform to compile the source code to a SAVEQMGR executable. In the latest version, make files are included in the SupportPac for several operating systems.) See <http://www.middleware.org/general/saveqmgr.html> for a sample script based on the output of SAVEQMGR.

Actually, I prefer to have at least two scripts to preserve a queue manager. The first is based on the default configuration created with SAVEQMGR such as the sample script. The second script contains the objects unique to a specific queue manager. These would include application specific objects, general testing objects, channel definitions, transmit queues, queue manager aliases, and the like. Need I mention that these scripts should be carefully backed up?

You will notice at the top of the script are header comments which should be *de rigeur* in any IS shop. Included therein are several variables bracketed by dollar signs (\$). These are keywords for the Revision Control System (RCS) that can be used for source code control. More on that in a moment.

These scripts can provide valuable cleanup when needed and be used to reset all queue manager objects. You'll note that the script contains all attributes of an object and their default values. The reason for that is to enable resetting all values to that default setting:

- If your programmers write programs that set parameters of an object and the programs don't clean up after themselves or if the program abends and doesn't have an opportunity to do so
- If you have a failure that leaves MQ objects in an unclean state.

Obviously, you can also use a set of scripts like this to build queue managers when you upgrade to a newer version of MQSeries.

## Version Control

Scripts created to preserve a queue manager are of diminished value if not kept up to date. Doing so should be part of your normal change procedures.

One valuable tool for managing change is a source code control system. Although there are many good commercially available packages, the lack of one should be no deterrent. RCS is available for free, often already compiled for your specific platform. I've used it on many flavors of UNIX and on Windows NT. RCS, like any source code control tool allows you to:

- Store and retrieve revisions of ASCII text files, including MQSC scripts
- Maintain a history of changes to the file

## WebSphere MQ Discipline: Preparing for the Worst

---

- Allow reverting to an earlier version of a file for back out or other reasons
- Resolve access conflicts when more than one person wants to change a given file

### Getting prepared

The real issue we've been discussing is taking a disciplined approach in configuring and managing your message oriented middleware. Anyone who fails to address issues of systems management discipline is destined for catastrophe. That's not alarmist; it's a statement of fact. In the case of MQSeries, there are many good commercial products that will assist you in those endeavors. Even so, you can manage a relatively small installation with freely available software, including many items from IBM's plethora of SupportPacs.

When should you begin managing the configuration of your MQSeries queue managers and objects? The simple answer is now. The application of any of the systems management disciplines will save a lot of work, frustration, and even embarrassment when disaster strikes. The more rigorously you apply these disciplines across the board, the safer you will be. It is unfortunate that so many IS professionals have not been taught the value of a disciplined approach that some of us with gray hair learned the hard way so many years ago. Back in the days of mainframes we used to talk about RAS or Reliability, Availability and Serviceability. The cycle of short time to market in the software industry today has led to a mentality of fixing problems with the three finger salute. The concept of rebooting a PC at the time a software defect causes the system to hang doesn't translate well to central application and database servers. Likewise, our communications infrastructure should be robust and as bullet proof as we can make it. This includes such facilities as our message oriented middleware products.

These easy steps will save time, not only in your daily MQSeries administrative tasks, but in the event that something bad happens, which unfortunately may be sooner than you think.



*Bruce Olsen is a senior independent consultant specializing in IT Strategy, Architecture and Planning*

The Middleware Resource Center is a service of Defining Technology, Inc.

Copyright © 1999-2004 by the author. All rights reserved.