The WebSphere Application Server for z/OS support team at the Washington Systems Center consists of: John Hutchinson, Mike Kearney, Mike Loos, Louis Wilen, Lee-Win Tai, Steve Matulevich, and Don Bagwell.

Mike Cox, Distinguished Engineer, serves as technical consultant and advisor.

Brian Pierce and Dennis MacDonald provide invaluable system programming support to the team.

Many thanks to Dana Duffield, Mark Luchini, Alex Guo, Harvey McGee and Mickey Scott for their helpful contributions to the content of this document.

For questions or comments regarding this document, e-mail Don Bagwell at dbagwell@us.ibm.com
WP101329 - Migrating to WebSphere Application Server for z/OS V7.0

Table of Contents

Executive Overview ................................................................. 5
The big picture ........................................................................ 5
High-level review of the process .............................................. 5
Frequently Asked Questions ..................................................... 6
  Complicated process? ............................................................ 6
  Supported prior levels? ........................................................... 6
  Mixed levels in a cell okay? ...................................................... 6
  Migrate or build new cell? ....................................................... 6
  Generated jobs? ..................................................................... 7
New configuration file system? New JCL start procedures? .......... 7
How long does a node migration take? ...................................... 8
Recommendations for large, complicated cells? ......................... 8
Stop the servers in the node? ..................................................... 8
What are the common "gotchas" in this process? ......................... 8
What about migrating non-z/OS nodes that are part of a "heterogeneous cell?" ................................................... 9
Any recommended validation prior to attempting migration? .... 9
Any recommended validation after migration? ........................... 9
Minimum system requirements for V7.0? ................................. 10

MMT Introduction and Overview ............................................. 14
No more ISPF panels ............................................................... 14
Relationship of WCT to MMT to PMT ...................................... 14
What about the Application Server Toolkit (AST)? ..................... 14
May I use the WCT to do V6.1 configuration work? .................. 15
The MMT overview ............................................................... 15
  Starting the WCT ................................................................. 15
  Perspectives within the WCT ................................................ 15
  Migration locations and definitions ....................................... 16
  Migration locations ............................................................. 16
  Migration definitions .......................................................... 17
  Migration locations and definitions summary ......................... 17
  Processing a created migration definition ............................. 18

Plan your migration ............................................................... 19
Back up the environment ......................................................... 19
Start WCT and change the MMT perspective ........................... 19
Create migration location ...................................................... 19
Create migration definition .................................................... 21
  Define the type of node being migrated ............................... 21
  Provide a definition name and optional response file .............. 22
  Provide the target data set high level qualifier ...................... 23
  Provide JCL procedure library location and the WebSphere product directory .................................................... 23
  Provide information on the configuration file system to be built ............................................................ 24
Server customization panel, 1 of 2 .......................................... 25
  Deployment Manager and federated node .............................. 25
  Standalone Server node ....................................................... 25
  Numbered notes for the server customization panel 1 of 2 fields ............................................................ 26
Server customization panel, 2 of 2 .......................................... 28
  Deployment Manager and federated node .............................. 28
  Federated node .................................................................. 28
  Standalone node ............................................................... 29
  Numbered notes for server customization panel, 2 of 2 .......... 29

© 2010, IBM Americas Advanced Technical Skills
Washington Systems Center, Gaithersburg, MD

Version Date: Tuesday, March 22, 2011
The Customized Migration Jobs

Overview of the process .................................................. 35
Backup existing configuration .............................................. 35
Manual configuration updates .............................................. 36

BPXFRMmx updates ......................................................... 36
SCERUN, SCERUN2 and SIEALNKE ....................................... 36

STARTED profiles .............................................................. 37
Other SAF profiles ............................................................... 37

"System Update Authority" for WebSphere Admin ID ................. 37

Stop servers in the node? ..................................................... 37
Submit the migration jobs ................................................... 38

BBOM*HFS or BBOM*ZFS ..................................................... 39
BBOM*CP ........................................................................ 39
BBOWMG1* .................................................................... 40
BBOWMG2* .................................................................... 40
BBOWMG3* .................................................................... 41

BBOW*PRO + BBOW*PRE + BBOW*POS rather than BBOWMG3* 41
BBOW*PRO ....................................................................... 41
BBOW*PRE ....................................................................... 42
BBOW*POS ....................................................................... 42

Restart the servers ............................................................... 42

Example of a Multi-LPAR ND Cell Migration

Example cell details ............................................................... 43
Overall cell ....................................................................... 43

DMGR node - Block A in picture ........................................... 43
Federated node on z/OS image A - Block B in picture ............... 44
Second federated node on z/OS image A - Block C in picture .... 44
Federated node on z/OS image B - Block D in picture ............... 44

Migration strategy ............................................................... 44

MMT - locations and definitions .......................................... 45
Pre-plan the values to feed into MMT .................................... 46

Migrating the Deployment Manager node ................................ 46
Creation of jobs using MMT ................................................ 46

Manual configuration work prior to migration ....................... 50

DMGR and Daemon remained up ........................................... 51

Submitted the migration jobs .............................................. 51

BBOMDZFS ..................................................................... 51
Manually created the intermediate symbolic link in the new file system 51
Backed up existing JCL start procedures ................................. 52

BBOMDCP ....................................................................... 52
BBOWMG3D ..................................................................... 52

Stopped the Daemon and restarted the DMGR ....................... 53
Adding STEPLIB to V7 Daemon procedure to V6.1 modules .... 53

Performed basic validation on the new V7 DMGR ................. 53

Restarted the servers in the federated nodes on SYSA .......... 53

Migrating the federated node on SYSA ................................ 54

Creation of jobs using MMT ............................................... 54

Manual configuration work prior to migration ....................... 54

Shut down the servers in the node ...................................... 55
WP101329 - Migrating to WebSphere Application Server for z/OS V7.0

Made sure DMGR was running ........................................................................... 55
Submitted the migration jobs ........................................................................... 55
BBOMMZFS ........................................................................................................ 55
Manually created the intermediate symbolic link in the new file system .......... 56
Backed up existing JCL start procedures ......................................................... 56
BBOWMG1F ........................................................................................................ 56
Started one application server in the node ..................................................... 56
BBOWMG2F ........................................................................................................ 57
BBOMMCP ......................................................................................................... 57
BBOWMG3F ........................................................................................................ 57
Restarted the servers in the federated nodes on SYSA ................................. 58
Migrating the federated node on SYSB ............................................................. 58
Migrating the second federated node on SYSA .............................................. 59

Common Problems and Their Causes ............................................................. 60
MG3* Job run with insufficient authorities ....................................................... 60
Insufficient region size for the MG3* batch job ............................................. 60
Insufficient temporary space for migration utility ......................................... 61
Insufficient space in the target file system .................................................... 61
Default time for job expires for long running migration job ......................... 62
MMT points to intermediate link but link not yet created ......................... 62
Inability to gain access to the running DMGR ............................................. 63

Strategies for Migrating Large Topologies ..................................................... 64
Overview ........................................................................................................... 64
Run the migration as three separate jobs (DMGR, Federated, Standalone) ...... 64
  Why would someone want to do this rather than run the single MG3* job?  65
  Three jobs in flowchart form, with debug and error recovery information  66
Separate migrating applications from the migration of the runtime (DMGR, Standalone) .................................................................................. 67
Creating your V7 DMGR in parallel (DMGR) .................................................. 69
Enabling 64-bit JVMs for the migration process ............................................ 70
  Background -- how the pieces tie together .................................................. 71
  BBOW*PRO, BBOW*PRE and BBOW*POS ... not BBOWMG3* ............... 72
  Step 1 -- ready the environment for shell script edits ............................... 72
  Step 2 -- edit files ......................................................................................... 72
  Step 3 -- run jobs ......................................................................................... 73
Providing 64-bit and large heap for deferred application installs using WSADMIN .................................................................................................. 73
Shared address space for spawned child processes (_BPX_SHAREAS=NO) ........ 74
An illustration of these strategies employed for a hypothetical migration ...... 75
Starting environment ....................................................................................... 75
Deployment Manager migration ...................................................................... 76
  MMT panels ................................................................................................. 76
  Leave entire existing runtime operational .................................................. 77
  Execution of migration jobs ......................................................................... 77
  Running the WSADMIN scripts to deploy the applications ....................... 78
  Starting the new V7 Deployment Manager ................................................ 79
  Disabling the old Deployment Manager .................................................... 80
  Migration of federated nodes ....................................................................... 80

Miscellaneous Information .............................................................................. 81
Acquiring and installing the WCT ................................................................. 81
Updating the level of the WCT ...................................................................... 81
Changes made to the existing node configuration ........................................ 81
  DMGR ......................................................................................................... 81
  Federated Node ......................................................................................... 82
  Standalone ................................................................................................. 82

© 2010, IBM Americas Advanced Technical Skills
Washington Systems Center, Gaithersburg, MD

Version Date: Tuesday, March 22, 2011
Migration Planning Review Outline

- Job to create group ID with sufficient UNIXPRIV authority .......................... 82
- The zmmt.sh shell script ........................................................................... 83
- When the Daemon JCL start procedure is common and shared between images ........ 83
  - What is the problem? ............................................................................ 83
  - What's the solution? ............................................................................ 84
  - Does this even apply to me? ................................................................. 84
  - Single, common Daemon JCL? How is that even possible? .................. 84

- Migration Planning Review Outline ................................................................. 86
- WCT installation and validation .................................................................. 86
- WebSphere z/OS V7 installation and validation ........................................... 86
- Pre-migration SAF work ............................................................................ 86
- MMT customization planning .................................................................... 87
  - Deployment Manager node .................................................................. 87
- Federated application server node (or standalone server) ................................. 88
- Migration execution planning ..................................................................... 89
- Post-migration validation .......................................................................... 90

Document Change History ............................................................................. 91
Executive Overview

WebSphere Application Server for z/OS V7.0 is IBM's latest offering for that product. As in the past, IBM is providing a set of utilities to migrate existing configurations up to the V7.0 level. This document explains how an effective migration is accomplished.

The big picture

Migration involves running a utility job against a configuration. That utility will copy from the existing configuration file system, transform and modify the configuration as required by V7.0, and place the results into a new file system:

Note: Truth is, there's a few utility jobs, but not all are required and one job\(^1\) serves as the main migration utility. The details of this will be explained in this document.

The picture illustrates a key point: migration is a node-by-node process. A cell with, say, a Deployment Manager node and two application server nodes would imply three "migrations" -- one for each node. The DMGR node is always migrated first.

High-level review of the process

Here is an outline of the basic process involved:

- Take some time plan out your migration. We offer some guidance on that under "Migration Planning Review Outline" starting on page 86.
- Use the z/OS Migration Management Tool (MMT) to create the customized JCL batch jobs that perform the migration. We'll go into more detail on what the MMT is under "MMT Introduction and Overview" starting on page 14. We cover the details of creating the jobs under "Panel-by-Panel Review of Job Customization Using MMT" starting on page 19.
- Back up the entire cell's configuration file systems. This is a precaution just in case something does not go properly during the migration.
- Perform any manual updates needed ("Manual configuration updates" on page 36).
- Execute the migration jobs
- Restart the servers in the node at the V7.0 level.

That's the basic outline. This document will supply the details.

\(^1\) Or three smaller jobs that do the same thing as the one larger migration job. See “Run the migration as three separate jobs” on page 64 for more.
Frequently Asked Questions

Complicated process?

"This is a long document. Does that mean the process is complicated?"

Not really ... certainly not once you get used to it. This document is the length it is because it's trying to anticipate and answer questions on specific details. This document also has a lot of "white space" -- wide margins, pictures and diagrams, and spaces between paragraphs.

Once you complete a migration or two you'll come to understand it's really a pretty straightforward process.

Supported prior levels?

"What prior levels of WebSphere z/OS can be migrated up to V7.0?"

The migration utilities for WebSphere Application Server for z/OS V7.0 support the migration of Version 5.1, V6.0 and V6.1 nodes up to V7.0.

Note: V5.0 is not on that list. The V7.0 migration utility does not support the migration of a cell that far back-level. If you still have a V5.0 cell, we would suggest you give serious consideration to simply rebuilding the cell new at the V7.0 level rather than migrating it.

It's not a hard requirement, but the V5.1, V6.0 or V6.1 nodes you are migrating should be a level of maintenance that's relatively current.

Mixed levels in a cell okay?

"May I migrate my DMGR one weekend, then the rest of the cell other weekends?"

Yes, that is possible. What you then have is a "mixed level" cell -- with the Deployment Manager at the V7.0 level and your application server nodes at V7.0 or lower. That works -- a WebSphere Application Server V7.0 Deployment Manager is designed to manage "down" to V6.1, V6.0 and V5.1 nodes.

Note: Two issues need to be brought to your attention:

1. A Daemon server migrated to V7 but still supporting lower level servers on the same LPAR requires access to the lower-version modules. See "Adding STEPLIB to V7 Daemon procedure to V6.1 modules" on page 53.

2. See page 29 for an important note about "script compatibility" when operating in a mixed-level cell.

Migrate or build new cell?

"May I simply build a new cell at V7.0 and avoid migrating my existing cell?"

Yes, you may do that. The idea there is to maintain your existing cell at the existing level, and build a parallel cell at the V7.0 level. This approach has certain benefits, chief among them is the ability to "start fresh" and rebuild the cell from scratch, correcting whatever you wished you had done differently before. The other benefit is this allows your existing cell to remain active while you take whatever time you need to build the new V7.0 cell.

There are several downsides to this approach: all the post-creation customization work you did on your existing cell will need to be done again in the new cell. Do you remember all the configuration changes you made? In a perfect world we'd create WSADMIN scripts for all our post-creation customization work so we could re-customize an environment by rerunning the scripts. But most don't have those scripts, so it requires redoing all the work by hand.

The new cell approach also implies a new cell name since you can't have the same cell short name active in the same Sysplex at the same time. And your applications will need to be
deployed again in the new cell, with your request distribution mechanism modified to start routing work to the new cell.

**Generated jobs?**

"*What jobs are generated by the MMT? What do they do?"*

The MMT will generate a handful of members that get uploaded to the z/OS system and the hlq.CNTL and hlq.DATA data sets you specify. Not all are jobs to be submitted. In fact, depending on the environment, you may have only one real migration job per node.

Here's a brief survey of what to expect:

- One job to allocate and mount a new HFS or ZFS. The MMT will generate one job for each, so the choice of HFS or ZFS is really decided by which job you submit.
- One job to copy the new JCL start procedures into the named proc library.
- Three or four members that hold the new JCL start procedures generated by the MMT. Three members if you're migrating a DMGR node; four if it's a federated node or a standalone server.
- Two jobs related to clearing the XA transaction logs. You'll only see these for federated nodes or standalone servers. The generated instructions indicate they're optional; our belief is it's better to run the job in all cases just to be sure.
- Finally, one main migration job. We often call it the "MG3" job because of the name -- BBOWMG3D for a DMGR node; BBOWMG3F for a federated node; BBOWMG3B for a standalone server.

**Note:** The MMT also provides three smaller jobs that when run in sequence perform the exact same function as BBOWMG3*. These jobs are BBOW*PRO, BBOW*PRE and BBOW*POS.

That's it. The process is relatively simple -- allocate and mount the new configuration file system, copy the new JCL start procedures into the proclib, then migrate the node. We'll supply all the details in this document.

**New configuration file system? New JCL start procedures?**

"*Is V7.0 like past migrations where it was required we allocate a new configuration file system, and required we create and use new JCL start procedures?"*

Yes. The new file configuration file system makes sense -- having the migration utility transform and modify in place the existing configuration file system would be ... well, it would make a lot of people very nervous. It makes far more sense to copy out of the existing and write the new format into a new file system.

V7.0 brings a new JCL start procedure format with it, which implies new JCL start procedures. You simply can't use the prior JCL start procedures with V7.0: you must use the new JCL generated by the MMT. That brings a question to the table -- should you copy the new JCL into your proclib with new names? Or overlay the existing JCL start procedures and reuse the names?

Very quickly, since we discuss this in more detail on page 26, re-using the same names allows you to reuse the same SAF STARTED profiles for the controller regions and may imply no changes to any automation routines. New names may require new STARTED profiles and may well mean changes to automation routines.

Again, see page 26 for more.
How long does a node migration take?

"How much time should I plan on for migrating a node?"

Much depends on the size of the configuration for the node and the resources available to perform the migration. A working estimate is 20 to 30 minutes per node once the main migration job is submitted. Deployment Manager nodes for very large cells may take much longer.

See ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101589 for a very good document that discusses migration performance along with offering many good hints and tips.

**Note:** The task can be broken up into three smaller jobs. See "Run the migration as three separate jobs" starting on page 64.

Recommendations for large, complicated cells?

"Are there any recommendations for how to approach very large and complicated cells? We’ve had problems migrating before and would like to avoid difficulties."

We have a whole section on this topic. See “Strategies for Migrating Large Topologies” starting on page 64.

Stop the servers in the node?

"Is it required that the servers in the node be stopped during the migration?"

No. The migration utility will perform its work while the servers are running. The "old servers" must be stopped before attempting to start the migrated servers.

One exception is an application server node where XA connectors are used. The XA transaction logs need to be flushed before migrating. This involves running the BBOWMG1* and BBOWMG2* jobs, with the servers down after MG2* is run. But the DMGR can be migrated while up, as well as application server nodes where XA connectors are not used.

What are the common "gotchas" in this process?

"It seems there's always something that can go wrong in these things ... what are some of the more common things to be aware of?"

We cover that topic under "Common Problems and Their Causes" starting on page 60.

To answer your question, the most common problems seem to be:

- **Memory issues**
  
  Java out of memory conditions can pop up if for some reason the size of the JVM heap is limited on the system where the migration is run. A recent addition to this document is a description of how to enable 64-bit JVMs if heaps larger than 2GB are needed (see page 70).

- **Space issues**
  
  There are two areas of possible concern -- the *temporary* space the migration utility requires to complete; and the space in the *target file system* for the node being migrated.

  - For the *temporary* space, the MMT is going to default to /tmp/migrate. How much temporary space is needed is a function of the node being migrated. A DMGR node for a cell with several nodes and many very large applications will require a fair amount of temporary space.

    By default the system /tmp probably does *not* have enough space. It may, but it may not.

    You're better off naming a different location and allocating and mounting a file system so the migration utility has lots of room to work. Determine the size of the node configuration file system being migrated and multiply by 1.5.
And make sure the permissions on that location are 777. Not strictly required, depending on the authority of the ID running the job, but for temp space it doesn’t hurt.

- For the target file system, the default MMT values will be 420 primary, 100 secondary. Consider that a minimum. Check the file system allocation size for the node being migrated and use that as a starting point, adjusting upwards for anticipated growth for new applications.

- **User ID authority issues**

  Two areas of concern here as well -- file access authority for the migration utility to read and write its files; and, if security is on, the establishment of the SSL connection back to the running DMGR when a federated node is being migrated.

  Rather than spell out all the details here, we'll simply reference two areas in this document:

  - A summary of the ID requirements for each job on page 31.
  - "The Customized Migration Jobs" starting on page 35, where the ID requirements are discussed in more detail
  - A sample RACF batch job we used to give our WebSphere Admin ID the authority it needed to run the jobs can be found under "Job to create group ID with sufficient UNIXPRIV authority" on page 82.

**What about migrating non-z/OS nodes that are part of a "heterogeneous cell?"**

  You'd use the migration tool that's part included with WebSphere for that platform. There won't be batch JCL to submit, but the other essential things are still applicable:

  - The DMGR must be migrated up to V7.0 first
  - The DMGR must be up and running for a federated node migration to succeed.

**Any recommended validation prior to attempting migration?**

  "What can I do to make sure the migrated servers have the greatest chance of success starting once they're migrated?"

  You may wish to consider configuring a V7.0 cell (standalone or small ND) on the system to validate that all the V7.0 system programmer work has been done and a V7.0 cell is capable of starting.

  What we're looking to avoid is the case where the migrated cell fails to start because of some basic system piece that's missing or incorrect. The natural tendency would be to think the problem is with the migration utility, and that can result in time wasted chasing down problems in the wrong area.

  Validate the basics by creating a validation configuration, then move on to migration.

**Any recommended validation after migration?**

  There are validation things you should plan to do at key steps (we refer to them as “milestones”) during the migration of a Network Deployment cell. For instance, after migrating the DMGR you should of course make sure the new V7.0 DMGR starts, and you can log onto the Admin Console.

  In the planning review section we touch on this under "Migration execution planning" on page 89.

  After the migration is done you will want to do a thorough validation of your new environment. We offer some suggestions under "Post-migration validation" on page 90.
Minimum system requirements for V7.0?

"The previous question has me wondering -- what are the minimum system requirements for running WebSphere Application Server for z/OS Version 7.0?"

The basic requirement is z/OS 1.7 or higher.

Next we will show you the mandatory and conditional requisites.

For the "Target System Requirements" under "Installation requisites," the program directory lists the following mandatory requisites:

**Important!** Please consult the Program Directory that comes with your copy of the product. Do not rely on this information as the final say ... the information in the Program Directory may be updated, and those updates may not be reflected in this document.

Use the program directory that came with the product and consult the PSP bucket for the product. Those will serve as the authoritative guides to installation requirements.

www.ibm.com/support ... search on PSP WASAS700

---

**Figure 8. Mandatory Installation Requisites**

<table>
<thead>
<tr>
<th>Program Number</th>
<th>Product Name and Minimum VRM/Service Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>5694-A01</td>
<td>z/OS with PTFs as indicated</td>
</tr>
<tr>
<td></td>
<td>z/OS V1.7</td>
</tr>
<tr>
<td></td>
<td>UA19007, UA20007, UA20004, UA21373, UA21556,</td>
</tr>
<tr>
<td></td>
<td>UA22890, UA23082, UA23129, UA23197, UA28149,</td>
</tr>
<tr>
<td></td>
<td>UA29651, UA34282, UA39535, UA31615, UA41075,</td>
</tr>
<tr>
<td></td>
<td>UA41251, UA41936</td>
</tr>
<tr>
<td></td>
<td>z/OS V1.8</td>
</tr>
<tr>
<td></td>
<td>UA28150, UA28305, UA29009, UA29652, UA34283,</td>
</tr>
<tr>
<td></td>
<td>UA39406, UA39536, UA39616, UA41076, UA41252,</td>
</tr>
<tr>
<td></td>
<td>UA41937</td>
</tr>
<tr>
<td></td>
<td>z/OS V1.9</td>
</tr>
<tr>
<td></td>
<td>UA39407, UA39537, UA39617, UA41077, UA41253,</td>
</tr>
<tr>
<td></td>
<td>UA41938</td>
</tr>
<tr>
<td></td>
<td>z/OS V1.10</td>
</tr>
<tr>
<td></td>
<td>UA41254</td>
</tr>
<tr>
<td></td>
<td>z/OS Unix System Services and the hierarchical file system (HFS) or zSeries File System (2FS)</td>
</tr>
</tbody>
</table>

| 5655-G52       | z/OS e with PTFs as indicated               |
|                | z/OS e V1.7                                |
|                | UA19007, UA20007, UA20004, UA21373, UA21556,|
|                | UA22890, UA23082, UA23129, UA23197, UA28149,|
|                | UA29651, UA34282, UA39535, UA31615, UA41075,|
|                | UA41251, UA41936                            |
|                | z/OS e V1.8                                |
|                | UA28150, UA28305, UA29009, UA29652, UA34283,|
|                | UA39406, UA39536, UA39616, UA41076, UA41252,|
|                | UA41937                                    |
|                | z/OS e V1.9                                |
|                | UA39407, UA39537, UA39617, UA41077, UA41253,|
|                | UA41938                                    |
|                | z/OS e V1.10                               |
|                | UA41254                                    |
|                | z/OS Unix System Services and the hierarchical file system (HFS) or zSeries File System (2FS) |

It goes on to say "WebSphere Application Server for z/OS V7.0.0 has no conditional installation requisites."
For "Operational requisites," the program directory shows the following mandatory requisites:

<table>
<thead>
<tr>
<th>Program Number</th>
<th>Product Name and Minimum VRM/Service Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Any one of the following:</td>
</tr>
<tr>
<td>5694-A01</td>
<td>Z/OS with PTFs as indicated</td>
</tr>
<tr>
<td></td>
<td>Z/OS V1.7, UA19027, UA20075, UA20094,</td>
</tr>
<tr>
<td></td>
<td>UA21373, UA21556, UA22890, UA23082,</td>
</tr>
<tr>
<td></td>
<td>UA23129, UA23197, UA28149, UA29651,</td>
</tr>
<tr>
<td></td>
<td>UA34262, UA39535, UA39565, UA41075,</td>
</tr>
<tr>
<td></td>
<td>UA41251, UA41936, Z/OS V1.8,</td>
</tr>
<tr>
<td></td>
<td>UA28150, UA28305, UA29009, UA29652,</td>
</tr>
<tr>
<td></td>
<td>UA34263, UA39406, UA39536, UA39916,</td>
</tr>
<tr>
<td></td>
<td>UA41076, UA41252, UA41937, Z/OS V1.9,</td>
</tr>
<tr>
<td></td>
<td>UA39407, UA39537, UA39617, UA41077,</td>
</tr>
<tr>
<td></td>
<td>UA41253, UA41938, Z/OS V1.10,</td>
</tr>
<tr>
<td></td>
<td>UA41254</td>
</tr>
<tr>
<td>5655-G52</td>
<td>Z/OS.e with PTFs as indicated</td>
</tr>
<tr>
<td></td>
<td>Z/OS.e V1.7, UA19027, UA20075, UA20094,</td>
</tr>
<tr>
<td></td>
<td>UA21373, UA21556, UA22890, UA23082,</td>
</tr>
<tr>
<td></td>
<td>UA23129, UA23197, UA28149, UA29651,</td>
</tr>
<tr>
<td></td>
<td>UA34262, UA39535, UA39565, UA41075,</td>
</tr>
<tr>
<td></td>
<td>UA41251, UA41936, Z/OS.e V1.8,</td>
</tr>
<tr>
<td></td>
<td>UA28150, UA28305, UA29009, UA29652,</td>
</tr>
<tr>
<td></td>
<td>UA34263, UA39406, UA39536, UA39916,</td>
</tr>
<tr>
<td></td>
<td>UA41076, UA41252, UA41937, Z/OS.e V1.9,</td>
</tr>
<tr>
<td></td>
<td>UA39407, UA39537, UA39617, UA41077,</td>
</tr>
<tr>
<td></td>
<td>UA41253, UA41938, Z/OS.e V1.10,</td>
</tr>
<tr>
<td></td>
<td>UA41254</td>
</tr>
</tbody>
</table>
And the following *conditional* requisites:

<table>
<thead>
<tr>
<th>Program Number</th>
<th>Product Name and Minimum VRM/Service Level</th>
<th>Function</th>
</tr>
</thead>
</table>
| 5694-A01 or 5655-G52 | PTFs for 64 bit enablement, required by JDK V5  
UG988925  
UG898686  
UG900946  
UG91768  
UG92626  
UA12453  
UA13606  
UG9754  
UG90077  
UG90822 | 64 bit enablement, required by JDK V5 |
| 5694-A01 or 5655-G52 | HTTP Server for z/OS |  |
| 5694-A01 or 5655-G52 | z/OS Version 1.8  
APAR OA22003 (USS)  
APAR OA22064 (MVS)  
or z/OS 1.0 and above | Required for Writable SAF keyring support |
<p>| 5655-147 | CICS TS V2.2 or later | Required if your application environment uses CICS Transaction Server. |
| 5697-E93 | CICS TG V6.0 | Required if your application environment accesses CICS TS through the CICS Transaction Gateway (CICS TG). See <a href="http://www.ibm.com/cics/ctg">www.ibm.com/cics/ctg</a> for more information |
| 5655-C56 or 5655-J38 | IMS V8 or later | Required if your application environment accesses IMS through Java DataBase Connectivity (JDBC) or through IBM Connect/Connector for Java. |</p>
<table>
<thead>
<tr>
<th>Program Number</th>
<th>Product Name and Minimum VRM/Service Level</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>5694-A01 or 5695-G52</td>
<td>PTFs for 64 bit enablement, required by JDK V5 UG88825 UG88986 UG90946 UG91768 UG28266 UA13453 UA13606 UG89754 UG90077 UG90822</td>
<td>64 bit enablement, required by JDK V5</td>
</tr>
<tr>
<td>5694-A01 or 5695-G52</td>
<td>HTTP Server for z/OS</td>
<td></td>
</tr>
<tr>
<td>5694-A01 or 5695-G52</td>
<td>z/OS Version 1.8 APAR OA22093 (USS) APAR OA22094 (MVS) or z/OS 1.9 and above</td>
<td>Required for Writable SAF keyring support</td>
</tr>
<tr>
<td>5655-147</td>
<td>CICS TS V2.2 or later</td>
<td>Required if your application environment uses CICS Transaction Server.</td>
</tr>
<tr>
<td>5655-C56 or 5655-J38</td>
<td>IMS V8 or later</td>
<td>Required if your application environment accesses IMS through Java DataBase Connectivity (JDBC) or through IBM Connect/Connector for Java.</td>
</tr>
<tr>
<td>5675-DB2</td>
<td>DB2 V7.1 or later DB2 V7.1 and APARs PQ64404, PQ86525, PQ89043, PQ68041, UQ88238, PQ88082, PQ87786 UQ88011 PTFs for DB2 V7.1 UK01565, UQ89338, UQ86911, UQ91761, UQ85607, UQ88238, UQ92167 DB2 V8.1 and APARs PK00815, PQ84577, PQ87786, PQ88082, UQ86012, PTF(UK01566) PTFs for DB2 V8.1 UK01566, UQ86912, UQ91762, UQ88239, UQ92168, UQ85537</td>
<td>Required if your application environment uses a DB2 database. Required to use Web Services UDDI and session persistence for Web container or if your application environment accesses DB2 through Java DataBase Connectivity (JDBC) or through IBM Connection/Connector for Java. Use of DB2 requires licensed utilities from IBM or equivalent 5697-E96 DB2 Utilities Suite, or equivalent, is recommended. In addition to DB2.</td>
</tr>
</tbody>
</table>

**Important!** Again ... please consult the Program Directory that comes with your copy of the product. Do **not** rely on this information as the final say.

Use the program directory that came with the product and consult the PSP bucket for the product. Those will serve as the authoritative guides to installation requirements.

www.ibm.com/support ... search on PSP WASAS700
MMT Introduction and Overview

In this section we’ll explore this new things called the z/OS Migration Management Tool, or MMT.

No more ISPF panels

We saw it coming with Version 6.1 ... the eventual elimination of the ISPF configuration panels.

With Version 7.0 the ISPF panels are gone. Now all configuration and migration customization is done with the workstation based configuration tools collectively known as the WebSphere Customization Tools, or WCT.

Relationship of WCT to MMT to PMT

We have three acronyms floating around -- WCT, MMT and PMT. Here’s the relationship between the three:

In summary:

- You install the WCT
- You invoke one of the functions -- PMT or MMT -- depending on what you’re trying to do

As the picture indicates, our focus will be on the MMT function contained within the WCT.

What about the Application Server Toolkit (AST)?

The Application Server Toolkit (AST) is an Eclipse-based workstation tool used to make certain modifications to packaged applications, such as updating application security information, or modifying deployment descriptors.
The V6.1 flavor of the PMT -- the one offered back in the V6.1 days before the WCT was available -- was packaged as part of the AST. That was the packaging and delivery mechanism for the earlier version of the PMT.

Now with V7.0 the AST is no longer used as a way to package and deliver the PMT or MMT. The new WCT is used for that purpose. The AST goes back to doing what it is intended to do -- be an application tool.

This is good news. The AST had a full Eclipse framework as part of its construction, and because of that it was very large. The new WCT, which includes the PMT and MMT, has a smaller Eclipse subset, and the total download size is less than 40MB. It is much smaller to download, installs more quickly and starts more quickly than the AST.

May I use the WCT to do V6.1 configuration work?

Yes. The WCT's PMT function will create customized jobs for V6.1 or V7.0.

The migration capabilities of the MMT are limited to just V7.0.

The focus of this document is on the migration capabilities of the MMT, so we won't go into detail on how to configure V6.1 using the WCT.

The MMT overview

In this section we'll offer an overview of the MMT, with the step-by-step instructions on configuring the migration jobs found under "Panel-by-Panel Review of Job Customization Using MMT" starting on page 19.

Starting the WCT

It's a standard Windows program, so once the product is installed it's started using the normal "Start" button sequence:

Start ⇒ All Programs ⇒ IBM WebSphere ⇒ WebSphere Customization Tools V7.0

Note: See "Acquiring and installing the WCT" on page 81 for specifics.

Perspectives within the WCT

The WebSphere Customization Tool program is based on Eclipse \(^2\). As such, it uses something called "perspective" to contain the various views and panels associated with a particular function. The MMT has its own perspective; the PMT has its own perspective, separate from the MMT.

The way to access the MMT function, once the WCT is started, is to "open the perspective" related to the MMT. There are a few ways that's done.

When you first start your copy of the WCT, you may see something that looks like this:

The block arrows are pointing to tabs that, when clicked, will bring that "perspective" to the front where you can begin your work.

But you may not see those tabs if those perspectives are closed. But don't worry ... you can easily open whatever perspective an Eclipse-based tool provides:

\(^2\) Eclipse is an open standard application framework. You can read more about it at [www.eclipse.org](http://www.eclipse.org). The main point is that the WCT is based on Eclipse, so the terminology and behavior of the WCT will be "Eclipse-like."
If you've ever worked with one of the Rational tools, like Rational Application Developer, you'll know that the number of different perspectives can get very large. Thankfully, the WCT at present has only three -- "Welcome," "PMT" and "MMT."

The WCT is designed to expand and support other functionality. So in the future there may be other perspectives.

Notes:
- If you've ever worked with one of the Rational tools, like Rational Application Developer, you'll know that the number of different perspectives can get very large. Thankfully, the WCT at present has only three -- "Welcome," "PMT" and "MMT."
- The WCT is designed to expand and support other functionality. So in the future there may be other perspectives.

Migration locations and definitions

Once you have the z/OS Migration Management Tool (MMT) perspective opened, you're faced with two more concepts you may not be familiar with:

Notes:
- Rather than limit you to one location where all your definitions had to go, the tool allows you to specify different "locations" so you can organize your various migrations logically.
- For example, imagine you have four cells -- development, test, QA and production. You might choose to define four locations -- one for each cell -- with the migration customizations...
for each cell held separately. That's not a requirement to do it that way, but the flexibility is there.

You add (in other words, “create”) a location using the "Add” button seen to the right of the migration locations list. In the example above the list is empty, which is what you’d see when you first start up your copy of the MMT. You can delete locations with the "Delete” button ... that does not delete the directory; it merely remove the location from the list of locations for the WCT.

In summary, a "migration location” is simply a defined place on a storage device that the MMT will view as a logical collection point for what's explained next -- "definitions."

**Migration definitions**

A migration definition is what we create when we seek to migrate a node. That would include the customized jobs that are uploaded to z/OS and do the migration, as well as the supporting shell scripts and the instruction member. It would also include the file that contains all the variable input you supplied. In short, a migration definition is a collection of all the stuff needed to migrate a node.

We mentioned earlier that migration is a node-by-node process. For example, a cell with a Deployment Manager node and two federated application server nodes would require three definitions -- one for the DMGR, and one for each of the federated nodes.

You could scatter those three definitions across different locations, but that wouldn't make much sense. Better to collect up the definitions that are related and keep them contained in the same location. Use locations to separate logical bundles of definitions.

**Migration locations and definitions summary**

Recapping:

- A location is a defined place on a storage device where migration artifacts (definitions) will be stored.
- A definition is a collection of artifacts used to migrate a node.
- Multiple migration definitions will be needed to migrate a cell if that cell contains multiple z/OS nodes.
- Multiple migration definitions can be kept in the same location. Locations are useful to logically organize definitions are related to one another, for example part of a cell.

Using pictures, the concept might be illustrated like this:
The sequence, when you first start up the WCT will be:

- Open the "MMT" perspective
- "Add" a location -- a simple process that requires a name and a definition of a drive and directory where the location will be created
- Create a migration definition that will be stored in the location. This is done with the "Migrate" button, which we'll explore under "Panel-by-Panel Review of Job Customization Using MMT" start on page 19.

**Processing a created migration definition**

"Process" is a fancy term for "upload to the z/OS system where the customized batch jobs can be submitted to perform the migration."

**Note:** And that captures the key point of all this -- to get the migration utility customized and up on the z/OS so you can submit it to perform the migration. The MMT is simply a tool that runs on the workstation that captures information from you and generates the customized jobs.
Panel-by-Panel Review of Job Customization Using MMT

At this point we'll assume you've installed the WCT and you've verified it starts. Now we can begin the discussion of how to use the MMT to customize the migration utility jobs.

**Plan your migration**

It's useful to map out the migration strategy and capture key information ahead of time.

- Go to "Migration Planning Review Outline" on page 86
- Fill out the information and have it ready for when you use the MMT to configure the migration jobs

**Back up the environment**

- Use your backup tool of choice and backup every node in the cell so that you have a common checkpoint to fall back to if needed. Back up the JCL start procedures as well.

  **Note:** See "Changes made to the existing node configuration" on page 81 for more on what to expect in terms of modifications to the existing configuration when the migration utility is run.

**Start WCT and change the MMT perspective**

This is a simple matter of getting to the proper starting place for the customization work that follows.

- Do the following
  
  **Start ⇒ All Programs ⇒ IBM WebSphere ⇒ WebSphere Customization Tools V7.0**

- Do the following:
  
  - If you see "z/OS Migration Manage Tools" tab, click on it:
  - Otherwise: **Window ⇒ Open Perspective ⇒ z/OS Migration Management Tool**

- Verify what you see looks something like this:

![WebSphere Customization Tools](image)

  You see a section labeled "Migration Locations"

  You see a section labeled "Migration Definitions"

**Create migration location**

A migration location is where the MMT will store the results of the migration job customization work you do. The location can be on a local hard drive or a network drive. See "Migration locations" on page 16 for more of a description.

You may create a new location or tell the tool to use an existing one. For this document we'll illustrate creating a new location.
In the "Migration Locations" section, click the "Add" button:

On the "Add Migration Location" panel provide the key information:

Notes:
1. Two choices -- Add an existing location or create a new. Add existing assumes the directory exists and is already populated with WCT information. Creating a new initialize a new location, either an empty directory you point to or a new directory the WCT will create for you.

2. The name of the location is what will be displayed in the tool. It should be something meaningful to you. For example, MYCELL is the name of the cell being migrated. Inside the MYCELL location will be held the definitions for the nodes in MYCELL being migrated.

3. At the present time the MMT supports only migrations to V7.0.

4. The actual path on the storage device (local drive or network drive) where WCT will create the location. The directory may pre-exist but it must be empty. Or you may name a brand new folder and the WCT will create it for you. The illustration above shows the cell name incorporated as part of the path name, but that's just a suggestion and not a requirement.

Notes:
• The V7 WCT now has a locking mechanism that allows a location to be used by multiple people. The V6.1 zPMT was a bit less sophisticated. Having a location on a network drive with multiple people accessing separate definitions is now okay.

• Locations should be in a place other than the install path for the WCT itself. This is a recommendation to avoid losing definitions accidentally if the WCT is uninstalled and someone wipes out the whole folder structure.
Once the new location is initialized, you should see something like the following:

Notes:
1. The location you just created
2. There should be no definitions in the list because the location is new

Note: If you had a different cell -- let's say it was AZCELL -- you might consider creating a separate location for it. You could keep everything in one big location, but it might get confusing. Different locations allow organizing your definitions into logical groupings.

Create migration definition

Once the location is created, you may now begin the process of creating the migration definition. The definition is what holds all the information about the node being migrated, as well as all the generated components (such as the actual batch jobs) for the migration utilities.

Important: As stated before, migration is a node-by-node process. Each node will require its own migration definition. A cell with three nodes will require three migration definitions. They are created individually. The process we'll show now is for one node; you would repeat the process for each node being migrated, making changes to specific values as needed.

Define the type of node being migrated

- Make sure your new location is highlighted in the "Migration Locations" section
- Click on the "Migrate" button in the "Migration Definitions" section. You should see this:

This shows the three types of nodes that can be migrated:

- **Standalone** -- the node within a cell that consists of a single application server only; there is no Deployment Manager or Node Agent as part of the cell. This is not a Network Deployment cell.
• **Deployment Manager** -- the Deployment Manager node that's part of a Network Deployment cell.

• **Federated Node** -- an application server node in a Network Deployment cell that has been federated into the DMGR cell. This is a node that contains a Node Agent, which is a sure sign the node is a "federated node."

☐ Select the type of node you plan to migrate, then click "Next"

**Note:** The first few panels are common across all three migration types, but then there'll be a some differences. We'll note those when they occur.

**Provide a definition name and optional response file**

☐ Provide a definition name and, optionally, a pointer to a "response file":

![Screenshot of migration management tool](image)

**Notes:**

1. The definition name is what will display in the list of "Migration Definitions" within the selected location. It can be whatever is meaningful to you. Blank spaces are not allowed.

2. A "response file" is a flat file of variable name/value pairs. They are used to populate the input fields for a definition. Response files are optional; if you don't have one it simply means you supply the information manually. The MMT will produce a response file as a means of saving all the input you provide.

**Note:** The PMT -- used to create the runtime -- also takes a response file. The PRS3341 spreadsheet on ibm.com/techdocs is designed to produce the response file for input to the PMT. The spreadsheet does not produce a response file for the MMT. The amount of input needed to produce the migration jobs just isn't that much.
Provide the target data set high level qualifier

□ Provide the high-level qualifier for z/OS data sets that will hold the generated output from the MMT:

For example, a value of **MYCELL.MIG.DMGR** would result in the following data sets being used:

- MYCELL.MIG.DMGR.CNTL
- MYCELL.MIG.DMGR.DATA

You have the choice of having the MMT allocate those data sets at upload, or simply use existing data sets by that name. You'll see that a bit later in this document.

Provide JCL procedure library location and the WebSphere product directory

□ Next comes the specification of the system proc library and the location of the product SMP/E file system:

Notes:
1. Specify whatever proc library you wish the copy job to use when copying in the new JCL.
2. This provides the migration utility with knowledge of where the V7.0 SMP/E product file system is located. Whatever value you supply here gets baked into all the symlinks in the new V7.0 configuration that's created. For that reason you should be a bit careful what value you supply here.

   You *could* supply the actual mount point. That would work. But it would mean the migrated node would be "locked" to the actual mount point location of the product HFS. That would limit your flexibility when rolling maintenance.

   It's better to supply here an intermediate symbolic link value. If you're not familiar with the concept of intermediate symbolic links, see WP100396 on [ibm.com/support/techdocs](http://ibm.com/support/techdocs). In
short, an intermediate symbolic link is a symlink you create that points to the actual V7.0 product HFS location. You then point the MMT at your symlink. That provides a kind of "alias" for the actual location. That alias is what's incorporated into the migrated node's configuration. And with an alias you can then employ the kind of techniques spelled out in the WP100396 white paper.

In the "Example of a Multi-LPAR ND Cell Migration" starting on page 43, we show the creation and use of intermediate symbolic links.

Provide information on the configuration file system to be built

☐ The next panel will be used by the job that allocates and mounts the file system that will hold the node's migrated configuration. Two jobs are generated, one for HFS and one for ZFS, so the choice of which is determined by which job you submit:

<table>
<thead>
<tr>
<th>Mount point:</th>
</tr>
</thead>
<tbody>
<tr>
<td>/wasv7/config</td>
</tr>
</tbody>
</table>

Name:
OMVS.WASV70.CONFIG.HFS

Volume, or * for SMS:
*

Primary allocation in cylinders: 420
Secondary allocation in cylinders: 100

File system type:
- Hierarchical File System (HFS)
- gSeries File System (ZFS)

Notes:
1. This will be the mount point for the new V7.0 configuration file system, not the "from" mount point. That comes later. You should match your naming conventions for WebSphere configuration file system mount points.
2. The configuration file system data set name. It should adhere to your data set naming standards.
3. Whether a specific volume is used or SMS is allowed to decide.
4. The primary and second allocation values seen here are defaults. You should set these values to correspond roughly to your current configuration file system's size and allow for future growth.
5. The MMT will generate a job for both HFS and ZFS. Your choice here will impact the instructions created by the MMT. But WebSphere itself has no concern for the file format of the underlying file system. WebSphere knows about the mount point, and it needs sufficient space. But WebSphere is not aware of HFS or ZFS.
Server customization panel, 1 of 2

Provide the necessary information. The panels for the Deployment Manager, Federated Node and Standalone are slightly different, so study the following pictures and notes.

**Deployment Manager and federated node**

**Deployment Manager**

- **From configuration location:**
  - Mount point: `/WebSphere/VR.MD`
  - Home directory: `DeploymentManager`

- **To configuration location:**
  - Mount point: `/moss/config`
  - Home directory: `DeploymentManager`

- Daemon procedure name: `BB0TDMPV`

- Controller procedure name: `BB0TDGR`

- Serveant procedure name: `BB0TDGR`

- [ ] Replace started procedure command names

- **WebSphere administrator’s user ID:** `XXXXX`

- **WebSphere administrator’s password:** `*******`

- **Confirm password:** `*******`

**Federated Node**

- **From configuration location:**
  - Mount point: `/WebSphere/VR.MD`
  - Home directory: `AppServer`

- **To configuration location:**
  - Mount point: `/moss/config`
  - Home directory: `AppServer`

- Daemon procedure name: `BB0TDMPV`

- Controller procedure name: `BB0TACR`

- Serveant procedure name: `BB0TASR`

- [ ] Replace started procedure command names

- **WebSphere administrator’s user ID:** `XXXXX`

- **WebSphere administrator’s password:** `*******`

- **Confirm password:** `*******`

See numbered notes following screen captures

**Standalone Server node**

**Standalone Server**

- **From configuration location:**
  - Mount point: `/WebSphere/VR.MD`
  - Home directory: `AppServer`

- **To configuration location:**
  - Mount point: `/moss/config`
  - Home directory: `AppServer`

- Daemon procedure name: `BB0TDMPV`

- Controller procedure name: `BB0TACR`

- Serveant procedure name: `BB0TASR`

- [ ] Replace started procedure command names

- **Optional Application Deployment:**
  - [ ] Deploy the default application.
  - [ ] Install the default application that contains the Snooz, Hello, and HitCount servlets.
  - [ ] Deploy the Sample applications.

See numbered notes following screen captures
Numbered notes for the server customization panel 1 of 2 fields

1. This is the node’s “from” mount point and home directory. This is where the migration utility will go to read in the existing configuration that is to be migrated. Provide the proper value for the node being migrated. Case matters, so type carefully.

2. This is the “to” mount point and home directory. This is where the migration utility will write the migrated configuration file system. The mount point field is grayed-out, and is based on the value you supplied on the previous panel. The “home” directory defaults to either DeploymentManager or AppServer, depending on the type of node being migrated. You may override the default for “home” if you’d like, or allow it to default.

3. These fields are asking for the JCL start procedure names to be used by the generated job that copies the new procedures into your procedure library. V7 start procedures have a different format from V5.1 or V6.x, so new procedures will be required. The question is whether you’ll have new names, or you’ll overlay the existing members and keep the same names. There are pros and cons to each approach:

   New Names:
   - [+] New names won’t overlay the existing JCL start procedures in the proclib, so if you need to fall back to the prior version the JCL will still be there.
   - [-] New names may require additional SAF STARTED profiles for the control regions.

   Note: It will depend on how “generic” your existing STARTED profiles are, and how different your new names are from the old. For example, consider the case where very explicit STARTED profiles are in place that require an exact match with the JCL start procedure. New JCL start procedure names will imply new STARTED profiles. But if the STARTED are a bit more generic, it’s possible the new names will match the STARTED profiles and therefore no new STARTED profiles would be needed.
   - [-] Automation routines may need to be updated to reflect the new JCL START procedure name used.

   Reuse existing names:
   - [+] No issue with the STARTED profiles. The ones used before will apply for V7.0 since the JCL start procedure names are exactly the same
   - [+] Automation routines less likely to break, though testing of such things is always recommended when a change like a migration is conducted.
   - [-] Backing up the existing JCL start procedures strongly recommended in case a fallback is required. Failure to back up the existing procedures may mean no easy ability to restart prior-level servers in a fallback ... you would need to recreate the start procedures by running ISPF panels or PMT tool and rerunning *CPY1 job.

4. This check box determines whether or not the JCL start procedure names held in the node configuration XML itself is to be changed.

   Note: It’s possible to start a server through the Administrative Console. To do that, WebSphere itself interacts with z/OS and issues the START command. That requires knowledge of the JCL start procedure to name, and that knowledge is maintained in the XML configuration files. This checkbox determines whether the values held in the XML are updated during the migration, or simply left alone.

There are some somewhat subtle things here we need bring to your attention:
   - If you’re planning to use new JCL start procedure names, then leave this box checked.
   - If you’re planning to reuse the same JCL start procedure names, then uncheck this box.
   - Here’s the subtle one, and it applies to federated nodes -- if your existing configuration uses unique JCL start procedures for each server process type within a node, then uncheck this
box. For example, imagine four servers in a node, and you have a unique JCL start procedure for each server controller and a unique JCL start procedure for each servant. Four servers and each one has its own unique set of JCL start procedures. If this is your scenario, then uncheck this box. That will prevent the migration utility from changing the values in the configuration back to a common set for all servers.

If this is your environment, then it implies some manual steps after the migration to make everything right:

a. Run the job that copies the JCL start procedures into your named proclib. You're going to need V7-format JCL start procedures in any event, so running that job is what will copy the members over to your proclib.

b. Carefully inventory through the migrated V7.0 administrative console the start command string used for each server, making special note of the JCL start procedure name that follows `START` on the command.

c. Makes copies of the V7 JCL procedures and rename them to conform to the names you inventoried through the administrative console.

Following those steps should provide JCL start procedures that align with what the DMGR will name when you start a server through the administrative console.

5. When migrating a Deployment Manager node or a federated node, the migration utility is going to attempt to make contact with the DMGR and it'll need to know a good userid and password that will allow it access. This too requires a few points of explanation:

- If global security is off for the cell, then this value isn't required to do the migration, but the MMT will require the ID field and the password fields to be populated. You may take the defaults just to satisfy the requirement to fill the fields.

- If security is enabled then provide a valid ID and password for accessing the DMGR.

  **Note:** The password value is encrypted in the files that are created, both in the MMT tool and in the uploaded job files.

- The administrative ID and password does not apply when migrating a Standalone server.

6. When migrating a Standalone server you have the opportunity to request that the default application and sample applications be deployed.
Server customization panel, 2 of 2

Provide the necessary information. The panels for the Deployment Manager, Federated Node and Standalone are slightly different, so study the following pictures and notes.

**Deployment Manager and federated node**

<table>
<thead>
<tr>
<th>Specify the following information to customize your migration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔ Migrate to support script compatibility</td>
</tr>
<tr>
<td>✔ Disable previous deployment manager</td>
</tr>
</tbody>
</table>

Application migration preference:
- Migrate applications and use the specified application installation directory
- Migrate applications and use the default application installation directory

Application installation directory: `/wasv/config/DeploymentManager/jprofiles/default/installedApps`

- Migrate and generate administrative scripts to install applications later
- Migrate applications and use the previous application installation directory
- Do not migrate applications

Migration trace options:
- Enable script tracing
- Enable profile creation tracing
- Enable pre-upgrade tracing
- Enable post-upgrade tracing

Migrate administrative console customized "My tasks" settings ("My tasks" is only supported from Version 6.1 forward)
- Migrate the settings for "My tasks" saved in an user defined workspace root location

User defined workspace root location: 5

Temporary directory location: `/tmp/migrate`
Migration definition identifier: `05094424`

**Federated node**

<table>
<thead>
<tr>
<th>Specify the following information to customize your migration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔ Migrate to support script compatibility</td>
</tr>
</tbody>
</table>

Migration trace options:
- Enable script tracing
- Enable profile creation tracing
- Enable pre-upgrade tracing

Migrate administrative console customized "My tasks" settings ("My tasks" is only supported from Version 6.1 forward)
- Migrate the settings for "My tasks" saved in an user defined workspace root location

User defined workspace root location: 5

Temporary directory location: `/tmp/migrate`
Migration definition identifier: `05113011`
Standalone node

Specify the following information to customize your migration.

1. Migrate to support script compatibility

   - Migration migration preference
     - Migrate applications and use the default application installation directory
     - Migrate applications and use the specified application installation directory

     Application installation directory: [default]

     - Migrate and generate administrative scripts to install applications later
     - Migrate applications and use the previous application installation directory
     - Do not migrate applications

2. Migration trace options
   - Enable script tracing
   - Enable profile creation tracing
   - Enable pre-upgrade tracing
   - Enable post-upgrade tracing

3. Migration trace options

4. Migration trace options

5. User defined workspace root location

6. Temporary directory location:

    Migration definition identifier: 05113309

See numbered notes following screen captures.

Numbered notes for server customization panel, 2 of 2

1. This determines whether the migrated configuration will be backwards compatible with the earlier versions of the WSADMIN scripting interface. Saying "Y" to this makes sure any scripts you have will still work unchanged against the V7.0 system. The default is "Y" which means it will be compatible.

   Note: A problem was recently discovered where unchecking this box created a problem in a mixed-level cell where the V7 DMGR was unable to communicate with a downlevel Node Agent. If you intend to run a mixed-level cell, even for a very short while, then leave this box checked so "Y" applies.

   Summary Notes:
   - For mixed-level cells, leave this checked “Y”.
   - If you have WSADMIN scripts that work against HTTP transport definitions, then do the following:
     - Leave the box checked “Y” for this migration
     - Invest time updating your scripts so they are compatible with the V7.0 script standards. See the InfoCenter for more.
     - When you are done updating your scripts, then run the convertScriptCompatibility.sh shell script against each node in your cell. This script removes the backwards compatibility support. See the InfoCenter for more information about this script.

   A reference for what script changes brought about the need for this “compatibility switch” can be found at this website:
   http://www.ibm.com/support/docview.wss?rs=180&uid=swg27008724
   Scroll down and look for the attached PDF with the section titled, "Script Compatibility Document"
2. This checkbox only applies to migrations of the Deployment Manager node. It is intended to make it more difficult to start the earlier DMGR once it has been migrated. Here's what it does:
   - A stop command will be issued against the old Deployment Manager server
   - The current serverindex.xml file will be copied to serverindex.xml_disabled. You can think of that as a backup copy.
   - The ports in the serverindex.xml file will be set to 0, which prevents the DMGR server from starting.

3. The "Application migration preference" section applies to DMGR migrations and Standalone Server migrations. You have five options. Here's some guidance:
   - Default Application Installation Directory -- this will migrate the currently installed applications and place them in the V7 default application location, which is under the /profiles/default/cells/<cell_long>/applications directory.
   - Specified Application Installation Directory -- this will migrate the currently installed applications and place them in the directory you've specified in the input field.
   - Generate Scripts -- this option will not migrate the applications. Instead, it will create WSADMIN scripts that can be used to install the applications at some later time.
   - Use Previous Application Installation Directory -- the applications will simply be migrated and maintained in whatever installation directory they're found in the "from" configuration.
   - Do Not Migrate Applications -- as the name implies, the applications will not be migrated

   **Note:** A Jython script named install_all_apps.jy will be generated in all cases. This script can be used to deploy the applications at a later time if you wish.

4. Use these tracing options at the direction of IBM Support. Otherwise, leave all checkboxes unchecked.

5. The "MyTasks" thing was introduced in V6.1. It provides a way to create a handy list of commonly accessed things in the Administrative Console. When created, it looks something like this:

   ![MyTasks Diagram]

   If none of this looks familiar to you, then take the default setting in the MMT. You very likely do not have any defined "My Tasks" so the default is acceptable.

   But if you have "My Tasks" defined and you wish them to go somewhere other than the default location, then take the second option and specify your location.
6. The temporary directory location is where the migration utility is going to go for space to create the temporary files it needs during the migration run.

**Important!**
- Depending on the size of the configuration, the migration utility may need up to 500 cylinders of temporary space. Allocate and mount a temporary file system of that size and point to that location in this field.

*Lack of space is one of the most common causes of migration job failure!*
- The "Migration definition identifier" is simply a timestamp-derived number that's used to keep one migration separate from others in the specified temporary location.
- If you're performing multiple migrations, or you're rerunning a migration, remember to clean out the temporary directory so you have the maximum space available to you.

**Job statement definition**

- The next panel allows you to create the JOB card that will be used in the generated jobs. You should update this to conform with your normal procedures.

<table>
<thead>
<tr>
<th>//jobname JOB</th>
<th>(ACCTNO,ROOM), 'USERID', CLASS=A, REGION=OM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The ID authority needed is dependent on the job being submitted. The migration instructions created by the MMT will tell you what each job's requirements are, under a heading of "User ID requirement." We show you how to access the instructions under "Migration Instructions" on page 32.

Here's a summary of the ID requirements for each job:

<table>
<thead>
<tr>
<th></th>
<th>DMGR</th>
<th>Federated</th>
<th>Standalone</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BBOM*HFS, or</strong></td>
<td>UID=0 or &quot;System Update Authority&quot;</td>
<td>UID=0 or &quot;System Update Authority&quot;</td>
<td>UID=0 or &quot;System Update Authority&quot;</td>
</tr>
<tr>
<td><strong>BBOM*ZFS</strong></td>
<td>(See note below)</td>
<td>(See note below)</td>
<td>(See note below)</td>
</tr>
<tr>
<td><strong>BBOM*CP</strong></td>
<td>Ability to copy a member into specified proclib</td>
<td>Ability to copy a member into specified proclib</td>
<td>Ability to copy a member into specified proclib</td>
</tr>
<tr>
<td><strong>BBOWMG1</strong>*</td>
<td>n/a</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
</tr>
<tr>
<td><strong>BBOWMG2</strong>*</td>
<td>n/a</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
</tr>
<tr>
<td><strong>BBOWMG3</strong>*</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
</tr>
<tr>
<td><strong>or</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>BBOW*PRO +</strong></td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
</tr>
<tr>
<td><strong>BBOW*PRE +</strong></td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
</tr>
<tr>
<td><strong>BBOW*POS 3</strong></td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
<td>WebSphere Administrator ID</td>
</tr>
</tbody>
</table>

3 See "Run the migration as three separate jobs" on page 64 for more on this option.
Note: The text in the migration instructions for "system update authority" says this:
Whenever "file system update authority" is indicated, the user ID used to run the configuration job must have either uid = 0 or the following UNIXPRIV class profile privileges:

```
CONTROL  access  to  SUPERUSER.FILESYS
UPDATE   access  to  SUPERUSER.FILESYS.MOUNT
READ     access  to  SUPERUSER.FILESYS.CHOWN
READ     access  to  SUPERUSER.FILESYS.CHANGEPERMS
READ     access  to  SUPERUSER.FILESYS.PFSCTL
```

If the UNIXPRIV profile CHOWN.UNRESTRICTED is defined, then the SUPERUSER.FILESYS.CHOWN is not required. For information about the UNIXPRIV class, see the z/OS Unix System Services Planning book.

See "Job to create group ID with sufficient UNIXPRIV authority" on page 82 for an illustration of a job that creates a group with the necessary privileges.

Migration Summary

Nothing to provide here, just:
- Click the "Create" button to create the migration definition
- Click the "Finish" button to complete things and return to the primary panel

Migration Instructions

- For each migration definition you create using the MMT, you should take time to review the "Migration Instructions." The instructions can be viewed through the MMT by clicking on the tab:

```
Make sure your location is selected
```

- Click "Migration Instructions" tab

```
Review/print instructions (Right mouse click and "Print")
```

This document reiterates the instructions found in the MMT tool. But you should be aware of the existence and location of those instructions.

Note: They are also accessible in the uploaded CNTL target data set as member BBOM*INS, where the asterisk is either D for Deployment Manager, M or managed (or federated) node, and B for base (or standalone) application server.
Process (upload) the definition to z/OS

This is what transfers the generated customized jobs from the MMT tool on your workstation up to the z/OS system where they'll be run.

☐ In the MMT do the following:

1. Make sure your location is selected
2. Highlight the definition you wish to upload
3. Click the “Process” button

☐ Select the option to upload to z/OS:

1. Select “Upload to target z/OS system”
2. Click “Next”
Provide the parameters that will allow the upload to z/OS to succeed:

1. Provide the host IP name of the system on which the jobs will be run
2. Provide a user ID and password for the FTP upload
3. Modify the port if you know your system uses something other than 21 for FTP
4. Make a mental note of the data sets that will be created
5. Click "Allocate target z/OS data sets" (leave unchecked if you are pre-allocating manually)
6. Specify volume and unit if you desire, otherwise leave blank
7. Click "Finish"

Watch the progress bar and look for the "Upload Successful" indication:

Click "OK"

You should see the MMT main panel again. Your customized migration jobs are up on the z/OS system ready to be submitted.
The Customized Migration Jobs

At this point you should have a set of data sets on the z/OS system populated with the customized jobs ready for submission. In this section we'll walk you through the process.

**Overview of the process**

This depends on the type of node being migrated, but they all have a similar flow:

<table>
<thead>
<tr>
<th>Deployment Manager Node</th>
<th>Federated Node</th>
<th>Standalone Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Backup the entire configuration</td>
<td>• Backup the entire configuration</td>
<td>• Backup the entire configuration</td>
</tr>
<tr>
<td>• Perform the &quot;manual configuration updates&quot; spelled out in the generated instructions.</td>
<td>• Perform the &quot;manual configuration updates&quot; spelled out in the generated instructions.</td>
<td>• Perform the &quot;manual configuration updates&quot; spelled out in the generated instructions.</td>
</tr>
<tr>
<td>• Run <code>BBOMDHFS</code> (or <code>BBOMD2FS</code>) to allocate and mount the file system that will be the target for the migration.</td>
<td>• Stop the application servers and the node agent in the node being migrated. See note 2.</td>
<td>• Stop the application server and the associated Daemon.</td>
</tr>
<tr>
<td>• Run <code>BBOMDCP</code> to copy the new V7.0 JCL start procedures to the named proclib.</td>
<td>• Run <code>BBOMMHFS</code> (or <code>BBOMMM2FS</code>) to allocate and mount the file system that will be the target for the migration.</td>
<td>• Run <code>BBOMMHFS</code> (or <code>BBOMM2FS</code>) to allocate and mount the file system that will be the target for the migration.</td>
</tr>
<tr>
<td>• Run <code>BBOWMG3D</code> to perform the migration. See note 1.</td>
<td>• Run job <code>BBOWMG1F</code>. This sets an XML flag to indicate the transaction logs are to be cleared.</td>
<td>• Run job <code>BBOWMG1B</code>. This sets an XML flag to indicate the transaction logs are to be cleared.</td>
</tr>
<tr>
<td>• Stop the DMGR's Daemon server. This will stop the DMGR as well as any application servers for the cell on the z/OS image.</td>
<td>• Start one application server in the node. It will start up, clear transaction logs, then stop. See note 3.</td>
<td>• Run job <code>BBOWMG2B</code>. It will set the XML flag back to the normal state.</td>
</tr>
<tr>
<td>• Restart the migrated DMGR. That will restart the migrated Daemon as well.</td>
<td>• Run <code>BBOWMG2F</code>. It will set the XML flag back to the normal state.</td>
<td>• Run <code>BBOWMG3B</code> to perform the migration. See note 1.</td>
</tr>
<tr>
<td>• Restart the application servers on that z/OS image that were stopped when the Daemon was stopped.</td>
<td>• Run <code>BBOMMCP</code> to copy the new V7.0 JCL start procedures to the named proclib.</td>
<td>• Start the Node Agent and the application servers.</td>
</tr>
<tr>
<td>• Make sure the new V7.0 DMGR is up and running.</td>
<td>• Make sure the new V7.0 DMGR is up and running.</td>
<td></td>
</tr>
<tr>
<td>• Run <code>BBOWMG3F</code> to perform the migration. See note 1.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note 1** -- There is a way to perform the migration using three smaller jobs rather than one large one. See "Run the migration as three separate jobs" on page 64.

**Note 2** -- If you do not have XA adapters and you do not plan on running the MG1 and MG2 jobs, then stopping the servers is not required.

**Note 3** -- There is a very specific situation that might cause problems at this point. It has to do with a cells that spans z/OS images where the Daemon JCL start proc name is reused and the physical JCL start procedure member is common and shared between the Daemons on the different z/OS images. It's an unusual thing, but we have seen it. Please see "When the Daemon JCL start procedure is common and shared between images" on page 83 for more on this if you think your configuration matches that brief description.

**Backup existing configuration**

This is "just in case" something doesn't go as planned. There is no "un-migrate" utility. There is only fallback by restore.
For the purposes of backup, consider the whole cell -- every configuration file system that comprises the cell -- as an atomic unit for backup. Snapshot the whole cell environment to backup. This is key because once you've migrated at least one federated node the Deployment Manager has been updated to see that node as at the new level. You can't just fall back that node by restoring that node's file system ... the DMGR would think the node is at V7.0 but the node itself would actually be at the earlier level.

So the rules are:
- If you've migrated only the DMGR, you can fall back by restoring just the DMGR's file system.
- If you've migrated the DMGR and at least one node, you have to fall back the entire cell.

**Manual configuration updates**

The details for these updates are spelled out in the migration instructions generated by the MMT. See "Migration Instructions" on page 32 to understand how to access those instructions.

**Note:** These updates can be done before any migration jobs are submitted. They are all done in anticipation of the migrated servers being started. They are not needed to perform the migration itself, but may be needed when the migrated servers are started.

There are four areas of updates:
1. **Update of BPXPRMxx so V7.0 configuration file system is auto-mounted**
2. **Making key system data sets available via LPA or link list.**
3. **Adding STARTED profiles in support of the V7.0 JCL start procedures copied in**
4. **Adding other SAF profiles that came into being with WebSphere for z/OS V6.1**

**BPXPRMxx updates**

The V7.0 configuration file system will be allocated and mounted by the job generated BBOM*HFS or BBOM*ZFS. To insure they are remounted after any system IPL, updates to the BPXPRMxx member are advised. The generated instructions, either in the MMT or in the BBOM*INS member of hlq.CNTL target data set provides the full details.

This is not strictly required, but it is recommended.

**SCEERUN, SCEERUN2 and SIEALNKE**

The text found in the generated instructions is:

WebSphere Application Server for z/OS customization assumes that the following system data sets are in the system link list or link pack area:

- Language Environment SCEERUN
- SCEERUN2
- System SSL SIEALNKE (z/OS 1.6 and above)

Placing these data sets in the link list or link pack area improves performance and insulates your WebSphere Application Server for z/OS configuration from changes in data set names (for example, when migrating to z/OS 1.6).

If the Language Environment or System SSL load module libraries are not in your system link list or link pack area, you must perform the following steps before starting any WebSphere Application Server for z/OS servers:
- Make sure that the data sets are APF-authorized.
- Complete the optional step below to add the data sets to STEPLIB in the server JCL and setupCmdLine.sh scripts.

If you regenerate server cataloged procedures at any point, make sure that the data sets are added to the new cataloged procedures.

**Note:** Even if they are in the link list concatenation, they may still need to be APF authorized. It depends on the setting for LNKAUTH parameter in IEASYSxx.
STARTED profiles

Whether or not you need to concern yourself with STARTED profile changes depends on two things:

1. Whether you elected to keep the same JCL start procedure names or provide new names during the configuration of the migration jobs
2. How generic your current STARTED profiles are and whether new JCL start procedure names will match and have the proper ID assigned to the started task.

See "Numbered notes for the server customization panel 1 of 2 fields" on page 26 for an explanation of the pros and cons of re-using the same names or coming up with new names.

Ultimately the issue here is fairly straight forward -- the V7.0 started tasks need SAF identities assigned to them. The migration process assumes the same identities as before will be used. So the question is whether the STARTED profiles will still work to do that, or if you need to add additional STARTED profiles to achieve that result.

Other SAF profiles

The key here is what level of WebSphere for z/OS you're coming from when you migrate to V7.0. Because the SAF profiles mentioned in this section were first required for WebSphere z/OS V6.1. So:

- If you're migrating from V6.1 your cell will already have those things, otherwise you'd have had problems long before now
- But if you're migrating from V5.1 you need to take a close look at the instructions, paying attention to the notes on SYNC and TRUSTEDAPPS.

"System Update Authority" for WebSphere Admin ID

The instructions do not explicitly call this out, but we found it helps insure success to make sure the WebSphere Admin ID -- under which we ran the MG3* jobs (or the three smaller jobs, BBOW*PRO, BBOW*PRE and BBOW*POS -- had "system update authority."

Rather than grant that to the ID directly, we accomplished it by creating a Group ID with the proper authorities, then connecting the WebSphere Admin ID to the group. That way we could easily remove the enhanced authority from the Admin ID after migration was done by removing the ID from the group.

See "Job to create group ID with sufficient UNIXPRIV authority" on page 82 for an example of how we accomplished this.

Stop servers in the node?

In general, the servers in a node do not need to be stopped during the migration. It is possible to leave your environment up and running during the execution of the migration jobs.

**Note:** If you're more comfortable stopping the servers in the node during migration, you may certainly do that.

The one exception to this is when XA connectors are involved in a federated node or Standalone and the transaction logs need to be cleared with the MG1* and MG2* jobs. The servers must be stopped during that process.

Bringing up a node migrated to V7 implies stopping the servers for that node still at the earlier level. And that may imply stopping the Daemon server. This takes a bit of explanation because the Daemon server is part of this migration process. We'll break this down by DMGR node and federated node:
In this case the Daemon instance configuration will be from the federated node's file system. After you've migrated the node, stop the Daemon and restart one of the servers in the node. That will start the new V7 Daemon along with the server at V7 as well.

When the servers in a federated node come up on a z/OS image where the DMGR for the cell is already running, then those servers will simply make use of the currently running Daemon. No attempt to start a Daemon will take place.

If, however, a server from that node is started before the DMGR is started, and no Daemon is present, the server will issue a START command for a Daemon. It will get the Daemon instance information from the node's configuration file system. Each federated node has its own /Daemon directory with copies of the Daemon instance information.

Here's the key -- when migrating a DMGR you'll have to stop the Daemon, and that'll stop all servers for that cell on that z/OS image. When restarting things, be sure to restart the DMGR first... that'll insure the migrated V7.0 Daemon is started by the V7.0 DMGR.

In this case the Daemon instance configuration will be from the federated node's file system. After you've migrated the node, stop the Daemon and restart one of the servers in the node. That will start the new V7 Daemon along with the server at V7 as well.

Submit the migration jobs

As stated several places elsewhere, the jobs to be submitted depend on the type of node being migrated. Here's a summary chart we provided earlier. They are listed in the order of submission:

<table>
<thead>
<tr>
<th>Type of Node</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment Manager node</td>
<td>The DMGR's configuration file system has a directory labeled /Daemon which contains the Daemon configuration used when the DMGR is started and it then starts the Daemon. If you perform a migration against the DMGR node, the Daemon configuration in the DMGR's file system is automatically upgraded to V7.0. To start the migrated DMGR at V7.0 requires the supporting Daemon to be started at V7.0 as well. That means you will have to drop the earlier version Daemon. Dropping the Daemon will imply the stopping of the other servers for the cell on that z/OS image. There's no way to avoid an outage for the servers of a cell on that z/OS image when migrating. Only with a cell that spans multiple z/OS images can we avoid outage during migration.</td>
</tr>
<tr>
<td>Federated node on same z/OS image as the DMGR</td>
<td>When the servers in a federated node come up on a z/OS image where the DMGR for the cell is already running, then those servers will simply make use of the currently running Daemon. No attempt to start a Daemon will take place. If, however, a server from that node is started before the DMGR is started, and no Daemon is present, the server will issue a START command for a Daemon. It will get the Daemon instance information from that node's configuration file system. Each federated node has its own /Daemon directory with copies of the Daemon instance information. Here's the key -- when migrating a DMGR you'll have to stop the Daemon, and that'll stop all servers for that cell on that z/OS image. When restarting things, be sure to restart the DMGR first... that'll insure the migrated V7.0 Daemon is started by the V7.0 DMGR.</td>
</tr>
<tr>
<td>Federated node on different z/OS image from the DMGR</td>
<td>In this case the Daemon instance configuration will be from the federated node's file system. After you've migrated the node, stop the Daemon and restart one of the servers in the node. That will start the new V7 Daemon along with the server at V7 as well.</td>
</tr>
</tbody>
</table>
### BBOM*HFS or BBOM*ZFS

| What it does | Allocates and mounts the node configuration file system. |
| Authority required | UID=0 or "File system update." See the generated instructions or page 31 for more on the UNIXPRIV access required for "file system update." |
| Number of steps | BBOM*HFS has two steps; BBOM*ZFS has three. |
| Anticipated time to completion | Less than a minute |
| Other manual activities | It is recommended you check the ownership of the mount point that was created by this job. The mount point must be owned by the WebSphere Administrator ID and the WebSphere Configuration Group. Issue the following command to correct if necessary: `chown <owner>:<group> <mount_point>` Example: `chown MYOWNER:MYCFG /wasv70config/mycl/passport` |
| Required? | No. You may manually allocate and mount the file system if you wish. If you manually create the mount point there are ownership and permission requirements. See the generated instructions for those. The BBOM*HFS or BBOM*ZFS jobs take care of those things automatically. |

### BBOM*CP

| What it does | Copies the generated V7.0 JCL start procedures into the specified procedure library. |
| Authority required | Authority to access and copy members into the specified data set. |
| Number of steps | One |
| Anticipated time to completion | Less than a minute |
| Other manual activities | Two things: 1) backup the existing JCL if opted to use the same names; and 2) make note of any STEPLIBs that may need to be replaced in the new JCL |
| Required? | No. You may manually copy the members from the hlq.CNTL data set to your procedure library if you wish. The names in hlq.CNTL are generic; you must rename as you copy. The BBOM*CP job shows what the "from" and "to" names must be. |

---

4 The MG3* job will set the mount point ownerships if the ID it runs under (the WebSphere Admin ID) has the proper UNIXPRIV authorities. See page 82 for the RACF commands we used to insure this would work.

5 You don't need to worry about STEPLIBing to WebSphere data sets in V7.0. It has no module libraries.
### BBOWMG1*

**Note:** Applies only to federated nodes and standalone servers. Does not apply to DMGR nodes.

<table>
<thead>
<tr>
<th>What it does</th>
<th>BBOWMG1* sets an XML flag so that upon start of one server in the node (starting the server is a separate, manual step) the XA transaction partner logs are cleared.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authority required</td>
<td>WebSphere Administrator ID, or any ID that is connected to the WebSphere Configuration Group ID.</td>
</tr>
<tr>
<td>Number of steps</td>
<td>BBOWMG1* has six steps</td>
</tr>
<tr>
<td>Anticipated time to completion</td>
<td>About a minute</td>
</tr>
<tr>
<td>Other manual activities</td>
<td><strong>Important!</strong> The stop will not be elegant. You may see something like this: BBOO0035W TERMINATING THE CURRENT PROCESS, REASON=C9C218D5</td>
</tr>
<tr>
<td>Required?</td>
<td>Strictly speaking it's only required if you have XA connectors configured. But it will not hurt in any event. We tend to think it's best to run it &quot;just to be sure.&quot; However, it does imply stopping the servers in the node. If you require higher availability and you do not have XA connectors, then skip.</td>
</tr>
</tbody>
</table>

### BBOWMG2*

**Note:** Applies only to federated nodes and standalone servers. Does not apply to DMGR nodes.

<table>
<thead>
<tr>
<th>What it does</th>
<th>BBOWMG2* returns the XML flag to its normal state so that upon server start the XA logs, if XA configured, will be used.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authority required</td>
<td>WebSphere Administrator ID, or any ID that is connected to the WebSphere Configuration Group ID.</td>
</tr>
<tr>
<td>Number of steps</td>
<td>BBOWMG2* has six steps</td>
</tr>
<tr>
<td>Anticipated time to completion</td>
<td>About a minute</td>
</tr>
<tr>
<td>Other manual activities</td>
<td>No. But you should make sure no application servers are re-started between completion of BBOWMG2* and BBOWMG3*, which does the migration. Starting an application server could re-populate the XA transaction logs. You'd have to redo the BBOWMG1*/BBOWMG2* steps again if that happened.</td>
</tr>
<tr>
<td>Required?</td>
<td>If you ran BBOWMG1*, then yes. This becomes required to reset the XML flag back to its normal state.</td>
</tr>
</tbody>
</table>
**BBOWMG3**

<table>
<thead>
<tr>
<th>What it does</th>
<th>Performs the migration of the node.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authority required</td>
<td>WebSphere Administrator ID, or another ID that is connected to the WebSphere Configuration Group ID that also has a keyring with the CA Certificate that signed the DMGR's server certificate attached. It also requires &quot;system update authority.&quot; You can achieve that by connecting the WebSphere Admin ID to a group with the sufficient UNIXPRIV privileges. See &quot;Job to create group ID with sufficient UNIXPRIV authority&quot; on page 82 for an example of a job that does that.</td>
</tr>
<tr>
<td>Number of steps</td>
<td>Twelve steps. See the next section on running three smaller jobs rather than this one big job.</td>
</tr>
<tr>
<td>Anticipated time to completion</td>
<td>One half hour to potentially several hours, depending on the size and complexity of the node being migrated, the number of deployed applications, and the priority the job receives relative to other work on the system.</td>
</tr>
<tr>
<td>Other manual activities</td>
<td>No</td>
</tr>
<tr>
<td>Required?</td>
<td>Yes. No other method of migration is possible.</td>
</tr>
</tbody>
</table>

**BBOW*PRO + BBOW*PRE + BBOW*POS rather than BBOWMG3**

The MMT generates three jobs that, when run in sequence, do the same work as the MG3 job. The difference is the three jobs do it in a more granular fashion.

We have a section dedicated to these three jobs under "Run the migration as three separate jobs" starting on page 64.

We'll offer the same summary overview here as did for the other jobs.

**BBOW*PRO**

<table>
<thead>
<tr>
<th>What it does</th>
<th>Creates the product directory and basic profile in the V7 target file system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authority required</td>
<td>It requires &quot;system update authority.&quot; You can achieve that by connecting the WebSphere Admin ID to a group with the sufficient UNIXPRIV privileges. See &quot;Job to create group ID with sufficient UNIXPRIV authority&quot; on page 82 for an example of a job that does that.</td>
</tr>
<tr>
<td>Number of steps</td>
<td>Seven steps</td>
</tr>
<tr>
<td>Anticipated time to completion</td>
<td>5 to 10 minutes</td>
</tr>
<tr>
<td>Other manual activities</td>
<td>No</td>
</tr>
<tr>
<td>Required?</td>
<td>Yes.</td>
</tr>
</tbody>
</table>
Yes. This requires that BBOW*PRE has been run.

<table>
<thead>
<tr>
<th>What it does</th>
<th>Creates a backup in the temporary location of the node that is being migrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authority required</td>
<td>It requires &quot;system update authority.&quot; You can achieve that by connecting the WebSphere Admin ID to a group with the sufficient UNIXPRIV privileges. See &quot;Job to create group ID with sufficient UNIXPRIV authority&quot; on page 82 for an example of a job that does that.</td>
</tr>
<tr>
<td>Number of steps</td>
<td>Six steps</td>
</tr>
<tr>
<td>Anticipated time to completion</td>
<td>Depends on the size of the existing node being migrated. Figure 10 to 20 minutes</td>
</tr>
<tr>
<td>Other manual activities</td>
<td>No</td>
</tr>
<tr>
<td>Required?</td>
<td>Yes. This requires that BBOW*PRO has been run.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>What it does</th>
<th>Perform the migration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authority required</td>
<td>WebSphere Administrator ID, or another ID that is connected to the WebSphere Configuration Group ID that also has a keyring with the CA Certificate that signed the DMGR's server certificate attached. It requires &quot;system update authority.&quot; You can achieve that by connecting the WebSphere Admin ID to a group with the sufficient UNIXPRIV privileges. See &quot;Job to create group ID with sufficient UNIXPRIV authority&quot; on page 82 for an example of a job that does that.</td>
</tr>
<tr>
<td>Number of steps</td>
<td>Nine steps</td>
</tr>
<tr>
<td>Anticipated time to completion</td>
<td>Very dependent on the size and complexity of the node being migrated.</td>
</tr>
<tr>
<td>Other manual activities</td>
<td>No</td>
</tr>
<tr>
<td>Required?</td>
<td>Yes. This requires that BBOW<em>PRO and BBOW</em>PRE have been run.</td>
</tr>
</tbody>
</table>

Restart the servers

Once a migration for a node is completed, you may start the servers at the new level. The servers at the prior level must be down when you start your V7.0 servers. You’ll see this in the example migration we offer next.
Example of a Multi-LPAR ND Cell Migration

Here's the cell topology we'll be using as the backdrop for the illustration.

**Example cell details**

**Note:** The definitions here are based on what you'd see using the planning spreadsheet found at [ibm.com/support/techdocs](http://ibm.com/support/techdocs) under the number PRS1331.

**Overall cell**

<table>
<thead>
<tr>
<th>Cell Long Name</th>
<th>mycell</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Proclib</td>
<td>SYS1.PROCLIB</td>
</tr>
<tr>
<td>V7.0 SMP/E product location</td>
<td>/usr/lpp/zWebSphere/V7R0</td>
</tr>
<tr>
<td>Intermediate symlinks used?</td>
<td>Yes⁶</td>
</tr>
</tbody>
</table>

**DMGR node - Block A in picture**

<table>
<thead>
<tr>
<th>Node Long Name</th>
<th>mydmnode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version 6.1</td>
</tr>
<tr>
<td>Configuration File System Mount Point</td>
<td>/wasv61config/mycell/mydmnode</td>
</tr>
<tr>
<td>JCL start procedures</td>
<td>MYMGCRR, MYMGSR</td>
</tr>
<tr>
<td>Daemon JCL start procedures</td>
<td>MYDEMN</td>
</tr>
</tbody>
</table>

⁶ The concept of intermediate symbolic links is discussed in WP100396 on ibm.com/support/techdocs. The MMT tool has no capability to generate intermediate symlinks directly. We'll illustrate the manual process in this document.
Federated node on z/OS image A - Block B in picture

<table>
<thead>
<tr>
<th>Node Long Name</th>
<th>mynodea</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version 6.1</td>
</tr>
<tr>
<td>Configuration File System Mount Point</td>
<td>/wasv61config/mycell/mynodea</td>
</tr>
<tr>
<td>JCL start procedures</td>
<td>MYACRA, MYASRA, MYCRAA</td>
</tr>
<tr>
<td>Daemon JCL start procedures</td>
<td>MYDEMNA</td>
</tr>
</tbody>
</table>

Note: We mentioned earlier that each node maintains information about a Daemon instance it will start if it finds no Daemon present when a server is started.

The DMGR's Daemon is likely to be the one most often used because most people start the DMGR first just as a matter of practice.

The node's Daemon instance is essentially the same as the one maintained in the DMGR configuration. But it uses a different JCL start procedure, which as you can see has a slightly different name, with the LPAR identifier on the end.

Second federated node on z/OS image A - Block C in picture

We generally discourage the construction of a second federated node on the same z/OS image, but we recognize there may be a need for it. We'll illustrate that here to show the migration of a V5.1 node.

Note: It should be noted that the Washington Systems Center naming convention tends to break down when a second node for a cell is built on the same z/OS image. Take these names as just that: somewhat awkward names. If you configure a second node on a system, please come up with a better naming convention for it than we did. 😊

<table>
<thead>
<tr>
<th>Node Long Name</th>
<th>mynodea2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version 5.1</td>
</tr>
<tr>
<td>Configuration File System Mount Point</td>
<td>/wasv51config/mycell/mynodea2</td>
</tr>
<tr>
<td>JCL start procedures</td>
<td>MYACRA2, MYASRA2, MYCRAA2</td>
</tr>
<tr>
<td>Daemon JCL start procedures</td>
<td>MYDMNA2</td>
</tr>
</tbody>
</table>

Federated node on z/OS image B - Block D in picture

<table>
<thead>
<tr>
<th>Node Long Name</th>
<th>mynodeb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version 6.1</td>
</tr>
<tr>
<td>Configuration File System Mount Point</td>
<td>/wasv61config/mycell/mynodeb</td>
</tr>
<tr>
<td>JCL start procedures</td>
<td>MYACRB, MYASRB, MYCRAB</td>
</tr>
<tr>
<td>Daemon JCL start procedures</td>
<td>MYDEMNB</td>
</tr>
</tbody>
</table>

Migration strategy

The plan of attack was:

1. Back up the configuration file systems for every node in the cell
2. Migrate the DMGR node

The DMGR is always the first thing migrated in a cell. A DMGR migrated to V7.0 is capable of managing "down" to a V6.1, V6.0.2 or V5.1 node. But the reverse is not true: a V6.1 DMGR is not capable of managing a V7.0 node.
The migration of the DMGR node will bring the Daemon server instance on SYSA to V7.0. That implies a stop and restart of the Daemon server. That will imply a period of outage for all the servers for MYCELL on SYSA. But the servers on SYSB can stay up during that.

That illustrates a key point -- maintaining server availability during migration is accomplished only when the cell spans more than one system image.

3. **Migrate the federated node on SYSA**

   We flipped a coin and decided to go with the node on SYSA first. Truth is, we could have gone with either.

4. **Migrate the federated node on SYSB**

   After the node on SYSA was migrated and back active, we moved over to SYSB.

   The Daemon instance on SYSB was migrated up to V7.0 when the node was migrated.

5. **Migrate the second Federated node on SYSA**

   The second node on SYSA -- again, something we discourage but we see more and more lately -- was migrated last. Imagine we had to wait for an ISV application to be made available that could run on WebSphere z/OS V7.0 ... that would be an example of why someone might do it that way.

   Please note that the servers in this node did take an outage while the DMGR node was migrated and the Daemon instance restarted. But we were able to maintain a node at an earlier level (one supported by our hypothetical ISV application) until we could migrate it up to V7.0.

   **Note:** WebSphere Application Server V5.1 drops from support at the end of September 2008. We are not advocating delaying updating from V5.1 and exposing yourself to loss of support. This is a purely hypothetical illustration. Assume we finished our migration to V7.0 before V5.1 dropped from support.

---

**MMT - locations and definitions**

We used a single location with four definitions:

<table>
<thead>
<tr>
<th>Location</th>
<th>C:\WCT\locations\mycell</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definition for DMGR node</td>
<td>MYDMNODE</td>
</tr>
<tr>
<td>Definition for AZNODEA on SYSA</td>
<td>MYNODEA</td>
</tr>
<tr>
<td>Definition for AZNODEB on SYSB</td>
<td>MYNODEB</td>
</tr>
<tr>
<td>Definition for second node on SYSA</td>
<td>MYNODEA2</td>
</tr>
</tbody>
</table>

**Note:** The MMT will take longer values for definition names. We chose to use the node short name, though that’s not a strict requirement, just an suggestion. It maps the definition name to the node being migrated.
**Pre-plan the values to feed into MMT**

Here are the planning choices we made:

<table>
<thead>
<tr>
<th>Planned Target HLQ</th>
<th>MYCELL.MIG.MYDMNODE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MYCELL.MIG.MYNODEA</td>
</tr>
<tr>
<td></td>
<td>MYCELL.MIG.MYNODEB</td>
</tr>
<tr>
<td></td>
<td>MYCELL.MIG.MYNODEA2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>V7.0 Configuration Mount Points</th>
<th>/wasv70config/mycell/mydmnode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>/wasv70config/mycell/mynodea</td>
</tr>
<tr>
<td></td>
<td>/wasv70config/mycell/mynodeb</td>
</tr>
<tr>
<td></td>
<td>/wasv70config/mycell/mynodea2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>V7.0 Config File System Data Sets</th>
<th>OMVS.WAS700.MYCELL.MYDMNODE.CONFIG.HFS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OMVS.WAS700.MYCELL.MYNODEA.CONFIG.HFS</td>
</tr>
<tr>
<td></td>
<td>OMVS.WAS700.MYCELL.MYNODEB.CONFIG.HFS</td>
</tr>
<tr>
<td></td>
<td>OMVS.WAS700.MYCELL.MYNODEA2.CONFIG.HFS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use Intermediate Symbolic links?</th>
<th>Yes. As mentioned, the MMT does not create them automatically, but the process to do it manually is easy enough.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>They could exist anywhere in the file system, but the logical place to put them is inside the configuration file system for the node. That means we'd first have to allocate/mount the V7.0 node configuration file system (by running the BBOM<em>HFS or BBOM</em>ZFS jobs) and then manually creating the intermediate symbolic links before moving on to the other migration jobs.</td>
</tr>
<tr>
<td></td>
<td>The intermediate links we created were:</td>
</tr>
<tr>
<td></td>
<td>/wasv70config/mycell/mydmnode/mydmnode_wassmpe</td>
</tr>
<tr>
<td></td>
<td>/wasv70config/mycell/mynodea/mynodea_wassmpe</td>
</tr>
<tr>
<td></td>
<td>/wasv70config/mycell/mynodeb/mynodeb_wassmpe</td>
</tr>
<tr>
<td></td>
<td>/wasv70config/mycell/mynodea2/mynodea2_wassmpe</td>
</tr>
<tr>
<td></td>
<td>That naming convention maps to the way the PRS1331 planning spreadsheet and the PMT do intermediate symbolic links.</td>
</tr>
</tbody>
</table>

| JCL Start Procedure Names                | We used the same names. That meant the members already in SYS1.PROCLIB were overwritten. We backed them up before we ran the job that copies the JCL into proclib. |

**Migrating the Deployment Manager node**

The DMGR node is always first in a cell migration.

**Creation of jobs using MMT**

The steps were:

- Launched WCT
- Created the location of C:\WCT\locations\mycell
- Clicked the "Migrate" button and started a definition for a Deployment Manager
- We gave it the definition name of MYDMNODE
- We gave it the high-level qualifier we planned: MYCELL.MIG.MYDMNODE
We specified the proclib and the intermediate symbolic link we planned for the Deployment Manager node.

![Image of the intermediate symbolic link](image)

**Note:** Again, the intermediate symbolic link was not yet created. Here we're simply naming it in the tool. The plan was to run the job to allocate/mount the file system, then manually create the link named on this panel. The migration jobs will "think" the product directory is the link we specify ... the link resolves to the actual named when we create the link.

Intermediate symbolic links provide a way to create an "alias point" between the configuration and the actual SMP/E product file system. The concept of intermediate symbolic links is spelled out in WP100396 on [ibm.com/support/techdocs](http://ibm.com/support/techdocs).

- We provided the information used to allocate and mount the new configuration file system:

![Image of the configuration file system](image)

**Note:** The primary allocation of 420 cylinders should be considered a minimum. Check the actual size of your current node file system and adjust this value accordingly, taking into account any anticipated growth in applications in the future.
The next panel was the first of two panels where the bulk of the migration information was provided:

"From" mount point and home directory

"To" mount point and home directory. Mount point based on previous panel; "Home" could be changed but we did not.

JCL start procedures -- we re-used the same names

See note below

Note: Because we were re-using the same JCL start procedure names, we UNchecked the box. This is per the recommendation in the context help seen when floating your cursor over that box. Unchecking the box means the migration utility will not replace the START strings held in the configuration XML.

If we had elected to use different (new) JCL start procedure names, then it would have been important to leave this box checked so the new JCL names would be placed into the START strings held in the configuration XML.
For the next panel we did the following:

Notes:
1. We did not have scripts we believed would be exposed to script compatibility issues. See page 29 for a note on this and a pointer to a web site with documentation on the compatibility issues the developers had in mind when they put this checkbox on the panel.
2. We believe it’s a good idea to disable the old DMGR once migration is successfully completed. But for this migration we chose to not have the migration utility do it. How the old DMGR is disabled is described on page 30.
3. We took the default here. If you believe your applications are installed somewhere other than default, then you should review the options and make the selection best suited for your environment. See the notes on this back on page 30.
4. We left all boxes unchecked. Check these boxes only when directed to do so by IBM support. The trace options produce significant amounts of output.
5. We took the default for this. We had no “My Tasks” from before.
6. Here we created a separate directory away from the system /tmp and we allocated and mounted a file system at that location that was 1.5 times the current size of the node’s configuration file system. We made sure the permissions on this directory were 777.
For the JOB card we took the default generated by the tool. Our plan was to update the JOB card at the time of execution based on the job authority requirements. As a reminder, those requirements for the DMGR jobs are:

<table>
<thead>
<tr>
<th>Job</th>
<th>Requirement</th>
</tr>
</thead>
</table>
| BBOMDHFS, or BBOMDZFS | UID=0 or "System Update Authority"  
(See notes on page 31) |
| BBOMDCP   | Ability to copy a member into specified proclib                                |
| BBOWMG3D7 | WebSphere Administrator ID                                                   |

We finished up the creation of the jobs in the MMT

We reviewed the migration instructions (see "Migration Instructions" on page 32)

We used the "Process" button to upload the jobs and the supporting members to the z/OS system (see "Process (upload) the definition to z/OS" on page 33)

We validated the data sets had been created and they contained the expected member. Here's what the CNTL data set looked like:

```
BROWSE MYCELL.MIG.MYDMNODE.CNTL
Command ===>
Name
_________________________
BBOMDCP 2
BBOMDCR
BBOMDDN
BBOMDHFS
BBOMDINS
BBOMDSR
BBOMDZFS 1
BBOWDPOS
BBOWDPRE
BBOWDPRO
BBOWMG3D 3
```

Notes:
1. The job that allocates and mounts the ZFS
2. The job that copies the JCL start procedures into the proclib. The JCL start procedures it will copy are found in this CNTL data set as well -- BBOMDCR, BBOMDSR and BBODDN. Those members were renamed during the copy to the names we specified in the MMT.
3. The main migration job.
4. The three smaller migration jobs that when run in sequence -- BBOW*PRO first, then BBOW*PRE, then BBOW*POS -- does the same thing as BBOWMG3*. See "Run the migration as three separate jobs" on page 64 for more on this.

**Manual configuration work prior to migration**

**Note:** Some of these are spelled out in the migration instructions generated by the MMT tool. See "Manual configuration updates" on page 36 for more.

Here’s a summary of the common manual updates needed, and whether they were done for this DMGR or not. If not, why:

---

7 If you choose to use the three smaller jobs rather than the single MG3* job, each of the smaller jobs has the same ID requirement as the MG3* job. See "Run the migration as three separate jobs" on page 64 for more on this topic.
This was a V6.1 node coming up to V7.0. That meant those profiles were already in place. No New security profiles introduced in V6.1 We chose to reuse the same JCL started procedure names, so new STARTED profiles were not needed. No STARTED profiles already in place on our system. But it’s good to check, just to be sure. No SCEERUN, SCEERUN2 and SIEALNKE We desired auto-mount. Yes BPXPRMxx We planned to run the jobs under the WebSphere Admin ID, and to insure success we knew it needed UNIXPRIV. See “Job to create group ID with sufficient UNIXPRIV authority” on page 82 for the RACF commands we ran to accomplish this. Yes Allocated separate temporary space In the “Server customization (Part 2)” panel we pointed to /u/myhome/migtemp as the temporary location for the migration jobs to do their work. The default /tmp may have worked, but we did not want to chance it. We calculated the size for the temp space as: current node configuration file system size times 1.5. Yes ID with UNIXPRIV authority We desired auto-mount. Yes We planned to run the jobs under the WebSphere Admin ID, and to insure success we knew it needed UNIXPRIV. See “Job to create group ID with sufficient UNIXPRIV authority” on page 82 for the RACF commands we ran to accomplish this.

### DMGR and Daemon remained up

To illustrate that the migration process can take place when the DMGR was up, we let it continue running. We had unchecked “Disable previous Deployment Manager” in the MMT so we knew it would not try to stop the running DMGR.

### Submitted the migration jobs

With the jobs up on z/OS ready to run, we proceeded as follows:

**BBOMDZFS**

This job allocated the ZFS that would contain the migrated DMGR configuration. It has an ID requirement of “file system update authority.” See the generated instructions or page 31 of this document for a review of what that implies.

**Note:** Had we specified "HFS" in the MMT, we would have run BBOMDHFS.

The result was:

```plaintext
OMVS.WAS700.MYCELL.MYDMNODE.CONFIG.HFS
OMVS.WAS700.MYCELL.MYDMNODE.CONFIG.HFS.DATA
```

mounted at:

```
/wasv70config/mycell/mydmnode/
```

### Manually created the intermediate symbolic link in the new file system

We mentioned before that we intended to use intermediate symbolic links between our migrated V7.0 configuration and the actual SMP/E product file system. We mentioned that it required a manual step between the allocation of the file system (BBOMDZFS, which we just ran) and the migration job (BBOWMG3D, which had not yet been run).
Here’s a picture of what we did:

![Diagram showing ZFS file system structure]

And here’s how we did it:

- Opened a Telnet session (or OMVS)
- Logged on with an ID that had authority to create symbolic links
- Issued the following command, all as one command (split here because of the length):
  ```
  ln -s /usr/lpp/zWebSphere/V7R0 /wasv70config/mycell/mydmnode/mydmnode_wassmpe
  ```
- Backed up existing JCL start procedures

**Backed up existing JCL start procedures**

In anticipation of running the **BBOMDCP** job, which we knew would copy in JCL start procedures with the same names as our existing DMGR JCL start procedures, we backed up our existing procs. We simply copied them to another data set.

**BBOMDCP**

This job simply copied into **SYS1.PROCLIB** three procs: **MYMGCR**, **MYMGSR**, **MYDEMN**. It overwrote the V6.1 procs by the same names in the proclib. That’s why we backed up those JCL procedures before we started the migration for the DMGR.

**Note:** At this point our **PROCLIB** had the V7 JCL start procedures even though our V6.1 DMGR was still running. That was okay, but we had to keep in mind that we could not stop and restart our V6.1 DMGR at this point. We chose to use the same JCL start procedure names, which meant a restart of our V6.1 server would have picked up V7.0 procs. A better strategy would have been to use new JCL names. We show that in the example under "Strategies for Migrating Large Topologies" starting on page 64.

**BBOWMG3D**

This was the main migration job. This took about a half hour to run, though on your system it may take more or less depending on the size of your DMGR’s configuration and the system resources you have available.

We did a quick checklist before submitting this job:

- Does the JOB card have the WebSphere Admin ID specified (or will the job run under the WebSphere Admin ID through other means)?
- Has the **BBOMDZFS** (or **BBOMDHFS**) job been run so the target configuration file system is allocated and mounted?
- Is the intermediate symbolic link created?

With those in place we submitted the job and waited for it to complete.

---

8 See “Run the migration as three separate jobs” on page 64 if you wish to break this up into smaller increments. For larger cells with many applications, it may be the better way to go.
Stopped the Daemon and restarted the DMGR

The BBOWMG3D job ran to completion, with RC=0 seen for all steps. We had our V7 DMGR and its associated Daemon server.

Note: See "Common Problems and Their Causes" on page 60 for help if your BBOWMG3D did not complete properly.

But before we could start the newly migrated DMGR we had to stop the existing DMGR and the Daemon server. We knew stopping the Daemon server would stop all the servers for that cell on SYSA. There was no way to avoid that outage.

The Daemon was stopped, bringing down all the other servers for the cell on SYSA.

The Deployment Manager was restarted. It restarted its Daemon server instance. When both were initialized both were at V7.0.

Adding STEPLIB to V7 Daemon procedure to V6.1 modules

If you intended to run V6.1 (or earlier) servers in the cell on the same LPAR as a V7 Daemon, you will need to add STEPLIB statements to the V7 Daemon’s JCL start procedure.

A V6.1 (or earlier) server started on the same LPAR as a V7 Daemon will make use of the V7 Daemon. But the servers will still require V6.1 (or earlier) modules accessible through the Daemon.

The solution is relatively simple: until such time you have the nodes migrated over to V7, supply the V7 Daemon JCL start procedure with the earlier version of WAS’s modules:

```
//STEPLIB DD DISP=SHR,DSN=xxxxxxxx.SBBOLD2
//      DD DISP=SHR,DSN=xxxxxxxx.SBBOLOAD
//      DD DISP=SHR,DSN=xxxxxxxx.SBBOLPA
```

Search on the string tmig_migrate_cells_zmmt_step12 in the InfoCenter for more on this.

Performed basic validation on the new V7 DMGR

Simple things at this point:
- Validated we could log onto the Administrative Console
- Made sure the Administrative Console saw the Node Agents as active ("green arrow")
- Made sure the Administrative Console saw the servers as active
- Checked to make sure the applications were seen as active
- Checked the synchronization status for the nodes

Those things are quick tests to see if all the underlying WebSphere communication activities are taking place within the cell.

Restarted the servers in the federated nodes on SYSA

With the Daemon back up, the servers that we had to stop when the Daemon was recycled could now be brought back up. Those servers are still at their original version levels, with the DMGR and its Daemon at V7.0:
Migrating the federated node on SYSA

The next node was the federated node on SYSA -- the V6.1 node, not the second node at V5.1. We did that last.

Creation of jobs using MMT

The panels in the MMT for a federated node very similar to those we saw for the DMGR. We were able to navigate through the MMT in a manner very much like what we did for the DMGR. The values we put into the MMT was different, of course. We used the values we pre-planned for the federated node on SYSA (see "Pre-plan the values to feed into MMT" on page 46). All the other things -- the checkboxes and such -- were selected in the same manner.

And of course the initial migration option we chose for this migration customization was "z/OS migrate a federated node," not the option for a DMGR.

Results:

<table>
<thead>
<tr>
<th>MMT definition:</th>
<th>MYNODEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target data sets on z/OS:</td>
<td>MYCELL.MIG.MYNODEA.CNTL</td>
</tr>
<tr>
<td></td>
<td>MYCELL.MIG.MYNODEA.DATA</td>
</tr>
</tbody>
</table>

Manual configuration work prior to migration

Again a summary of the manual configuration work:
This was a V6.1 node coming up to V7.0.

That meant those profiles were already in place.

No New security profiles introduced in V6.1

We chose to reuse the same JCL started procedure names, so new \textsc{started} profiles were not needed.

No \textsc{started} profiles Already in place on our system. We checked for the DMGR, so no need to check again since this node is on the same system.

No \textsc{sceerun}, \textsc{sceerun2} and \textsc{siealnke}

We desired auto-mount. Yes

BPXPRMxx

Yes We desired auto-mount.

SCEERUN, SCEERUN2 and SIEALNKE

No Already in place on our system. We checked for the DMGR, so no need to check again since this node is on the same system.

\textsc{started} profiles

No We chose to reuse the same JCL started procedure names, so new \textsc{started} profiles were not needed.

New security profiles introduced in V6.1

No This was a V6.1 node coming up to V7.0. That meant those profiles were already in place.

\begin{table}[h]
\centering
\begin{tabular}{|l|l|l|}
\hline
\textbf{Activity} & \textbf{Performed?} & \textbf{Explanation} \\
\hline
ID with UNIXPRIV authority & No & We did this for the DMGR migration. We used the same Admin ID so there was no need to do this again. \\
\hline
Allocated 500 cylinder temporary space & No & We had the temporary space file system allocated from the DMGR migration. \\
\hline
Cleaned temporary space & Yes & The DMGR migration consumed a portion of the temporary space. In order to give this node's migration maximum space to work, we cleared the \texttt{bbomigrt2.sh} file and the directory with the numeric name and all the sub-directories under that. \\
\hline
BPXPRMxx & Yes & We desired auto-mount. \\
\hline
SCEERUN, SCEERUN2 and SIEALNKE & No & Already in place on our system. We checked for the DMGR, so no need to check again since this node is on the same system. \\
\hline
\textsc{started} profiles & No & We chose to reuse the same JCL started procedure names, so new \textsc{started} profiles were not needed. \\
\hline
New security profiles introduced in V6.1 & No & This was a V6.1 node coming up to V7.0. That meant those profiles were already in place. \\
\hline
\end{tabular}
\end{table}

\textbf{Shut down the servers in the node}

The servers in the node being migrated needed to be down because we planned to run the \texttt{MG1*} and \texttt{MG2*} jobs. But the servers in the second node on SYSA stayed up, as did the servers in the node on SYSB.

\textbf{Made sure DMGR was running}

A federated node being migrated requires that the V7 DMGR for the cell be up and running at the time of the migration. The migration utility needs to make contact with the running DMGR and download some files, as well as let the DMGR know that the node is migrated successfully.

\textbf{Submitted the migration jobs}

The process for a federated node is similar to a DMGR, with a small twist.

\textit{BBOMMZFS}

This job allocated the ZFS that would contain the migrated node configuration. It has an ID requirement of "file system update authority." See the generated instructions or page 31 of this document for a review of what that implies.

\textbf{Note:} Had we specified "HFS" in the MMT, we would have run \texttt{BBOMMHFS}.

The result were:

\texttt{OMVS.WAS700.MYCELL.MYNODEA.CONFIG.HFS} \\
\texttt{OMVS.WAS700.MYCELL.MYNODEA.CONFIG.HFS.DATA}

mounted at:

\texttt{/wsv70config/mycell/mynodea/}

\footnotesize{\textsuperscript{9} If no XA connectors, then \texttt{MG1*} and \texttt{MG2*} would not be needed and thus the servers could have remained active during the migration. If you're not sure if \texttt{MG1*} and \texttt{MG2*} are needed, run them just to be safe.}
Manually created the intermediate symbolic link in the new file system

The process here is very similar to what we did for the DMGR, except it’s a separate symbolic link ... this one resides in the file system for MYNODEA. The following picture illustrates this:

```
ln -s /usr/lpp/zWebSphere/V7R0
```

Backed up existing JCL start procedures

In anticipation of running the BBOMMCP job, which we knew would copy in JCL start procedures with the same names as our existing server JCL start procedures, we backed up our existing procs. We simply copied them to another data set.

**BBOWMG1F**

This job (and the next one, BBOWMG2F) are used to clear the XA transaction partner logs. Strictly speaking, this job (and the next one) are optional; that is, if you don't have XA connectors configured, then these jobs are not required.

Here’s the rule -- if you’re not sure, run them. They do no harm and they take only a few extra minutes. They do, however, require a stopping of the servers in the node.

This job is very simple -- it updates an XML file and sets a value from 0 to 1. The presence of that 1 is what tells WebSphere to flush the XA partner logs for the node when a server is started.

This job is run with under the ID authority of the WebSphere Admin ID.

It was a very short-running job.

**Note:** If you run BBOWMG1F, you must then do the next two steps.

Started one application server in the node

We then started one of the application servers in the node. That server will see the "1" we set with the BBOWMG1F job and flush the XA partner logs for the node. When done, the server will automatically stop.

**Important!** The server stop will not be elegant. You may see something like this:

```text
BBOO0035W TERMINATING THE CURRENT PROCESS, REASON=C9C218D5.
```
That's normal. Don't be alarmed.

When the server stopped itself, we proceeded to the **BBOWMG2F** job.

**BBOWMG2F**

Another very simple job, this sets that XML "1" flag back to 0.

This job is run with under the ID authority of the WebSphere Admin ID.

It was a very short-running job.

Here's the key -- with the XA partner logs flushed and the flag set back to 0, we had to make sure we didn't start any servers in the node. If we did, then the XA logs might have been populated with more data.

So we left all the servers down and proceeded on to the migration of the node.

**BBOMMCP**

This job simply copied into SYSLIB four procs: MYACRA, MYASRA, MYCRAA and MYDEMN. It overwrote the V6.1 procs by the same names in the proclib. That's why we backed up those JCL procedures before we started the migration for the node.

**BBOWMG3F**

This was the main migration job. This took about a half hour to run, though on your system it may take more or less depending on the size of your DMGR's configuration and the system resources you have available. We did a quick checklist before submitting this job:

- Is the DMGR **up and running**?
- Does the JOB card have the WebSphere Admin ID specified (or will the job run under the WebSphere Admin ID through other means)?
- Has the **BBOMMZF5S** (or **BBOMMHFS**) job been run so the target configuration file system is allocated and mounted?
- Is the intermediate symbolic link created?

With those in place we submitted the job and waited for it to complete.

---

10 See "Run the migration as three separate jobs" on page 64 for an alternative approach.
Restarted the servers in the federated nodes on SYSA

With the node migrated to V7.0 we could restart the servers. Our cell now looked like this:

---

**Migrating the federated node on SYSB**

This process is nearly identical to the process we just showed for the migration of the node on SYSA.

Three things of note:

1. The Daemon instance on SYSB had its configuration maintained in configuration file system of the node over there on SYSB. Migrating the node to V7.0 migrated the Daemon instance up to V7.0 as well.

   Since we knew the servers in the node had to come down to run the `MG1*` and `MG2*` jobs, it made sense to go ahead and shut down the Daemon as well.

2. Earlier we mentioned a potential "gotcha" involving the case where the Daemon JCL proc names were re-used, and the actual Daemon JCL procedure for a multi-system cell like this was shared between the z/OS images. The write-up for that rare case is under "When the Daemon JCL start procedure is common and shared between images" on page 83.

   We did not have to worry about that. We did reuse the JCL procedure name, but we were not employing a shared JCL proc. Our Daemon procs were LPAR-specific. So we avoided the gotcha.

3. The temporary space (mounted at `/u/myhome/migtemp` had remnants from the federated node migration on SYSA. We cleaned that out.

   We also made sure the temporary file system was owned by SYSB. The migration utility, which we ran on SYSB, does considerable write operations into the temporary space. To insure the best time to completion, we changed the file system ownership to SYSB.
Migrating the second federated node on SYSA

This process was, again, very similar to the other nodes. The generation of the jobs using the MMT and the execution of the jobs on z/OS would be the same.

**Note:** We again cleared the temporary space, and we moved the file system ownership back to SYSA.

The key difference is that when bringing a V5.1 node up to V7.0 the security profiles mentioned in the migration instructions may come into play. Specifically, three areas of focus:

1. BBO.TRUSTEDAPPS
2. BBO.SYNC
3. BPX.WLMSERVER

Whether you need to take any action depends on what’s already defined. Some of that might be in place due to an existing V6.1 component of the cell. For example, the BPX.WLMSERVER requirement was already met for our V5.1 migration because the profile generated during the creation of the V6.1 nodes gave the configuration group **READ** to the profile. Our V5.1 node IDs were connected to that group. Thus the IDs for the migrated node, now at V7.0, enjoyed access to BPX.WLMSERVER.

**BBO.TRUSTEDAPPS** may or may not be covered, depending on how generic the profile is. The default format for that profile is. And, depending on whether "Enable Trusted Applications" is set for the servers. The migration instructions spell out the requirements for this profile very clearly.

The **BBO.SYNC** issue is essentially the same as **BBO.TRUSTEDAPPS**.

Check with your security administrator and have them read the migration instructions. They will be able to assist in determining whether the profiles are needed or not.
Common Problems and Their Causes

In this section we will provide information on some of the more common problems that can be seen when running the migration jobs. We will add to this section and republish the document as more people perform the migration and discover other problems and their causes.

**MG3* Job run with insufficient authorities**

**Symptom:**

The CRHOME step of the MG3* job fails with RC=2048.

**Reason:**

There are two reasons this may occur:

- *Fails fairly quickly* -- mount point ownership does not permit write by WebSphere Admin ID.
- *Fails after running a while, and fails with many "EDC5139I Operation not permitted" messages* -- the WebSphere Admin ID does not have sufficient authority to perform the tasks needed

**Solution:**

The easiest thing to do is to insure the WebSphere Admin ID -- under which the MG3* job should be run -- has all the authority it requires. This would be "system update authority" as spelled out in the migration instructions.

See "Job to create group ID with sufficient UNIXPRIV authority" on page 82 for an example of a job we used to accomplish this. That job creates a new Group ID, grants the authority to the Group, then connects the WebSphere Admin ID to the group.

We took that path because it allowed us to quickly and easily remove all the UNIXPRIV authorities from the WebSphere Admin ID after the migration was done by removing the Admin ID from the Group.

**Insufficient region size for the MG3* batch job**

**Symptom:**

This could manifest itself in several different ways, but the more common is:

- The CRPROF step of the MG3* job fails with RC=2048
- The message "Could not create the Java virtual machine" somewhere in the job output, or
- The string "java.lang.OutOfMemoryError" somewhere in the job output.

**Reason:**

The MMT will generate the MG3* job with REGION=0M. Provided that remains coded like that on the JOB card (in other words, you didn't change that), then something is limiting the amount of storage available. The CRPROF (short for "create profile") step is the first step in the job that fires up a JVM; the earlier steps are mostly setup file work.

There are two basic steps to this:

1. Initialize the JVM, then
2. Expand the JVM heap as the CRPROF work requires additional JVM memory

Failure to achieve both steps will result in a RC=2048 on CRPROF. Both are related to the inability of the job to get the storage it required.

**Solution:**

Correct the storage limitation. In particular, make sure nothing is overriding the REGION=0M on the JOB card, and make sure no IEFUSI exit is limiting requested storage.

Clean up remnants of the failed migration -- clear the temporary location and delete the contents of the target node file system.
**Insufficient temporary space for migration utility**

**Symptom:**

This will probably manifest itself as a RC=2048 failure in step PREUPGRD, because that's where the existing profile is backed up to temporary space. But it may show up in subsequent steps if the space exhaustion occurs there.

In our testing we saw:

MIGR0215W: The migration function cannot copy the file and open the destination file /u/myhome/migtmp/12104351/dmgr_backup/websphere_backup/config/templates/chains/sipcontainer-chains.xml.

But the error message you see may be slightly different.

**Reason:**

On the surface, very simple -- the amount of temporary space was entirely used up.

But the underlying cause might take two forms:

1. The file system -- your private mounted temporary space, or the system /tmp if you're using that -- is at 100%, or
2. The volume on which the temporary space is allocated is full and no further extents are possible.

The second one is less obvious. When debugging space issues, check both.

**Solution:**

Correct the space limitation.

With regard to cleaning up remnants of the failed migration you have to be a little careful. Much depends on when the failure occurred.

We supply a flowchart with recovery and cleanup information under "Run the migration as three separate jobs" starting on page 64. That's for the three jobs PRO, PRE and POS. But remember, those three combined simply do the same thing as MG3. So you can use that information to understand what the recovery/cleanup steps are, depending on where the job failed.

**Insufficient space in the target file system**

**Symptom:**

Depends on when the "out of space" condition occurs. The failure will likely be in CRHOME, CRPROF or UPGRADE.

The key will be looking for a message to indicate which file system the error occurred -- the temporary file system or the V7 target configuration file system. If the messages seem to indicate an inability to create a link, or a directory, or copy a file and the path indicates the V7 target file system, then it means the problem lies there.

**Reason:**

The act of migration involves populating the target file system with a migrated copy of the original configuration. The MMT default allocation settings are for 420 cylinders, which may not be enough depending on the size of the configuration being migrated.

**Solution:**

Take a look at the size of the current configuration file system. Specify the allocation values in the MMT to be at least the same size, and bigger if you anticipate the need for growth in the future.
Default time for job expires for long running migration job

**Symptom:**
S522 abend.

**Reason:**
The migration takes time and it may exceed the allowable time. If it does, the system will abend the job with S522.

**Solution:**
Code `TIME=1440` on the JOB card of the MG3* job to insure there is no time limit applied.

**MMT points to intermediate link but link not yet created**

**Note:** Intermediate symbolic links serve as a kind of "alias" between the configuration and the actual mount point of the product file system. We encourage the use of intermediate symbolic links because they provide enhanced flexibility for things like maintenance and migration. You can read more on intermediate symbolic links in WP100396 on [ibm.com/support/techdocs](http://ibm.com/support/techdocs).

The MMT does not create intermediate symlinks like the PMT does. It's easy enough to work around this -- you create the intermediate symlinks manually. We showed an example of doing that under "Manually created the intermediate symbolic link in the new file system" on page 51.

The problem discussed here is when you name an intermediate symlink in the MMT but for one of the following reasons it won't resolve to the SMP/E product file system:

- Forgot to create the intermediate symbolic link
- Created it but not at the location named in the MMT
- Created it at the location named in the MMT, but the contents of the symbolic link do not properly point to the actual location of the SMP/E product file system

**Symptom:**
The MG3* job fails very quickly and you see `RC=256` in step SETUP.

**Reason:**
The migration utility relies on access to the SMP/E product file system to complete its work. See the gray note box above for reasons why access to the product file system may not be there.

**Solution:**
Correct the intermediate symbolic link so it exists where the MMT expects it to be, and it resolves properly to the actual location of the SMP/E production file system.
Inability to gain access to the running DMGR

Note: This applies only to migrating federated application server nodes, not the DMGR or Standalone Servers. The migration process requires that the DMGR (which should already be migrated to V7) be up and running. Failure to establish a connection to the running DMGR will result in the federated node migration to fail.

There are several reasons why the connection may fail:

- The migrated DMGR for the cell is not running
- The BBOWMG3F job\(^1\) is run under an ID that does not have access to the keyring that holds the CA certificate used to sign the DMGR's server certificate. The WebSphere Admin ID should have that by default, which is why we strongly recommend you run the job under that ID.
- A network problem between the MG3F job and the running DMGR, which should manifest itself in a way similar to when the DMGR is simply not running.
- The password for the administrator ID supplied to the MMT is incorrect. The MMT requires you to enter it twice, so this would not be a typo but rather misunderstanding what the current password actually is.
- The ID supplied to the MMT as the administrator does not actually have administrator privileges

Symptom:

The basic failure indicator is the same in all cases -- RC=2048 in step UPGRADE. The specific messages are similar.

Reason:

For a federated node, the migration utility relies on the DMGR up and running and the ability to connect to it and copy files down from the DMGR. Failure to accomplish the connection will result in RC=2048 in UPGRADE and the message seen above.

Solution:

Ultimately you must get the migration utility to connect to the DMGR. Here's a suggested sequence of debugging activities:

- First, rule out the simple and obvious -- check to insure the DMGR for the cell is indeed up and running. Go one step further -- make sure it's in a fully-initialized state by logging on with the Administrative Console.
- Next, verify that the BBOWMG3F job (or BBOW*POS job) runs under the identity of the WebSphere Admin ID. That is most easily accomplished by coding USER= and PASSWORD= on the JOB card. We recommend the WebSphere Admin ID because by default it will have a keyring and the original CA Cert used during the creation of the cell.
- Finally, go back into the MMT and check to see that the WebSphere Administrator ID supplied is correct. The password field will show as a sequence of dots, so it's hard to visually verify the password. If you're in doubt, carefully retype everything and upload the jobs again and rerun.

\(^1\) Or the BBOW*POS job if you're taking the path of running the three individual jobs in sequence.
Strategies for Migrating Large Topologies

The bigger the configuration -- servers, applications, etc. -- the more effort, resources and time involved with migrating it. There are strategies, learned from experience, to help with approaching these very large environments. We'll discuss these strategies in this section.

Overview

We'll cover the following strategies:

- **Breaking up the main migration job into smaller chunks**
  The \texttt{MG3*} job does a lot of things and can take a very long time. The MMT creates three jobs that do the same thing as \texttt{MG3*}, but do it in smaller increments.  
  \textit{This is applicable to all three node scenarios -- DMGR, federated and Standalone Server.}

- **Separating the application migration from the runtime migration**
  The migration process has a mechanism that creates application EAR files and WSADMIN scripts to install those applications at some later time. This has the benefit of saving time during the running of the migration jobs as well as affording you an good opportunity to do some housekeeping -- after the migration you run only those WSADMIN scripts for applications you really want in your V7 environment.  
  \textit{This is applicable to a DMGR and Standalone Server, but not federated node migrations.}

- **Keeping your existing configuration in place and running while you migrate and create your V7 DMGR configuration**
  It's possible to migrate and construct your V7 DMGR while your existing DMGR and all the application servers keep running. This minimizes the disruption to your current operations and allows you to do the migrations in a more "offline" way.  
  \textit{This is applicable to DMGR migration only.}

- **Enabling 64-bit JVMs if larger heaps are required due to the complexity of the topology**
  By default the JVMs launched by the process will be 31-bit. 31-bit addressing implies a maximum heap of 2GB defined. We have seen cases where the topology was large enough to require a larger heap. That meant modifying a few things to tell the environment to launch a 64-bit JVM.

**Run the migration as three separate jobs (DMGR, Federated, Standalone)**

The MMT will generate three jobs that, when run in the following sequence, results in the exact same thing as the running of the \texttt{MG3*} job:

1. \texttt{BBOW*PRO} -- creates the V7 product directory and creates the V7 default profile  
2. \texttt{BBOW*PRE} -- creates the migration backup directory based on the node being migrated  
3. \texttt{BBOW*POS} -- migrates data from the migration backup directory into the V7 target  

\textbf{Note:} The jobs to create the target HFS (or ZFS) and copy the JCL into the proclib, as well as optionally the \texttt{MG1*} and \texttt{MG2*} jobs to clear the XA logs. Here we're talking about the main migration effort ... either one job (\texttt{MG3*}) or three jobs (\texttt{PRO, PRE and POS}).

In fact, if we look at the steps for the \texttt{MG3*} job and map these three smaller jobs to them we see something like this:
### Why would someone want to do this rather than run the single MG3* job?

The single MG3* job works very well. We see two basic reasons why you might want to consider running the migration as three separate jobs:

1. The *duration* of each job is shorter
2. The opportunity for *more granular error debugging and recovery* is present

The flowchart that comes next helps make these benefits more understandable.

---

**Note:** The separation of steps is not quite as clean as that ... some of the steps in the picture above are really common housekeeping chores and are seen in all three jobs. Specifically, the **SETUP**, **WRCONFIG**, **WRRESP**, **MKCONFIG** and **VERIFY** steps are common to all three.
The following flowchart illustrates the execution of the three smaller jobs, with the error debugging, cleanup and job re-submit needed at each step of the way:

Notes:
1. The **BBOW** job will do work in the defined temporary location (see page 31 for where that temporary location is defined) as well as in the V7 target location. That means all the basic stuff needs to be there -- space in the defined temporary location, the V7 target file system allocated and mounted, job ID with sufficient authorities, etc.

If this fails, you should check the **BBOW**.out and **BBOW**.err that will be found in the temporary file location. Perform debug based on the information in those files. See "Common Problems and Their Causes" on page 60.

Once the problem has been determined and corrected, do the following:
- Cleanup the V7 target directory (leave HFS/ZFS mounted, just clear contents)
- Rerun the **BBOW** job
2. The **BBOW**\^**PRE** job will do work in the defined temporary location only.

If this fails, you should check the **BBOW**\^**PRE.out** and **BBOW**\^**PRE.err** that will be found in the temporary file location. Perform debug based on the information in those files. See "Common Problems and Their Causes" on page 60.

Once the problem has been determined and corrected, do the following:

- Delete the backup directory in the temporary location. You find this with a name such as `dmg_backup` or `base_backup`.
- Do nothing to the V7 target file system. **BBOW**\^**PRE** doesn't touch it.
- Rerun the **BBOW**\^**PRE** job (you do not need to rerun the **BBOW**\^**PRO** job)

3. The **BBOW**\^**POS** job will do work in the defined temporary location as well as in the V7 target location.

If this fails, you should check the **BBOW**\^**POS.out** and **BBOW**\^**POS.err** that will be found in the temporary file location. Perform debug based on the information in those files. See "Common Problems and Their Causes" on page 60.

Once the problem has been determined and corrected, do the following:

- For DMGR or Standalone: simply re-run **BBOW**\^**POS**
- For a federated node:
  - Delete the backup directory in the temporary location
  - Re-run both the **BBOW**\^**PRE** and **BBOW**\^**POS** jobs

4. If you've reached this point all three jobs have completed with RC=0 and the migration for the node has been completed. But you may not be happy with some of the settings you provided in the MMT, such as JCL start procedure names, or the choice to migrate applications, or perhaps one of your post-migration validation tests did not indicate "success" in your eyes. If that's the case, then you may wish to fall back and re-do the migration.

```
Important!  You can delete and re-do a DMGR or Standalone migration easily. But a federated node migration, if successful, has communicated with the DMGR and that DMGR now thinks the node is migrated to V7. You can't simply delete files and do it again because the DMGR will think the node is being migrated again and will reject it.

That means that a federated node that is successfully migrated implies a fallback of the entire cell (DMGR and federated nodes) if you want to re-do one of the federated nodes. This is why we strongly recommend a snapshot backup of the entire cell.
```

5. The cleanup and recovery here is the following:

- Delete the whole temporary file location directory
- Delete the entire V7 target file system contents (leave the HFS/ZFS mounted, just clear the contents)
- Go back to the MMT and make the corrections you desire.
- Upload the jobs to the z/OS system
- Rerun the jobs in order -- **PRO**, **PRE** and **POS**

### Separate migrating applications from the migration of the runtime (DMGR, Standalone)

Migrating a configuration has two basic elements to it -- migrating the runtime configuration, and migrating the deployed applications. For smaller topologies doing both at the same time has proven to work well. But for larger topologies where many applications are deployed, the MMT provides a facility to separate out the applications and deploy them later. It does this by exporting the applications as EAR files and producing a WSADMIN script to do the deploy later.

This provides two benefits:

1. The migration jobs will require fewer resources and take less time if the applications are not part of the migration
2. It gives you an opportunity to do some housekeeping -- applications no longer deemed necessary do not need to be deployed into the new V7 runtime.

Back on page 30 when we saw where the MMT provides the "Application migration preferences":

![Application migration preference](image)

- **Note:** This only applies to DMGR migrations and Standalone Server migrations. Managing applications for federated nodes is done through the DMGR.

Selecting that option results in three things:

1. The applications are not migrated to the V7.0 cell. The *runtime* is, but the applications are not.
2. The applications are written out in EAR file format to the following directory:

<table>
<thead>
<tr>
<th>DMGR</th>
<th>(&lt;V7_mount&gt;/DeploymentManager/profiles/default/installableApps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standalone</td>
<td>(&lt;V7_mount&gt;/AppServer/profiles/default/installableApps)</td>
</tr>
</tbody>
</table>

3. A series of WSADMIN scripts are written out to the backup directory. The backup directory will be under the temporary location you specified in the MMT. For example:

/\(tmp/migrate/09094743/base_backup/\)

**Note:** This is an example of a Standalone Server backup directory, with "09094743" being the date and timestamp generated by the MMT so the directory under /\(tmp\) would be unique.

For example, we had a Standalone with the following applications installed:

- DefaultApplication
- My IVT Application
- QuoteStreamer
- SuperSnoop
- ivtApp
- query

And this yielded the following scripts in the /\(tmp\) backup directory:

- install_all_apps.jy
- install_My_IVT_Application.ear.jy
- install_QuoteStreamer.ear.jy
- install_SuperSnoop.ear.jy

**Notes:**

- The script install_all_apps.jy includes Jython to install all the applications seen at the time of migration.
- The other scripts install just the application specified in the name of the script file.
- It’s interesting to note that DefaultApplication, ivtApp and query were not included in that. Those are WebSphere applications, not customer applications.
The Jython script itself is fairly straight-forward:

- A brief set of comments at the top with instructions on invoking WSADMIN and providing a pointer to the script file and a properties file generated by the migration utility. For example, the instructions in the generated shell script for a Standalone migrated suggested an invocation of the this:

```
/wasv7config/blcell/blnoded/AppServer/profiles/default/bin/wsadmin.sh
-conntype none -f /tmp/migrate/09094743/base_backup/install_all_apps.jy
-p /tmp/migrate/09094743/base_backup/migration.props -lang jython
```

Notes:
1. The location of the migrated V7 standalone's `wsadmin.sh` file
2. A connection type of "none," which means WSADMIN will deploy the application without making any network connection to any running process. That means this can be run while the V7 configuration is down.
3. The `-f` flag points to the location of the Jython script. Note the location within the `/tmp/migrate`, which was the temporary location specified for this test run.
4. The `-p` flag points to the location of a properties file generated by the migration utility and used by the WSADMIN script to do the application deployment.
5. The specification of jython as the scripting language

Inside the script there's an `AdminApp.install()` invocation with a pointer to the `/installableApps` directory where the EAR file was placed.

There's also an `AdminConfig.save()` invocation to save the installation.

More `AdminApp.install()` and `AdminConfig.save()` pairs if the script was the `install_all_apps.jy` script and more than one application was found.

Creating your V7 DMGR in parallel (DMGR)

The DMGR migration is somewhat unique for several reasons:
- It is typically the node type that takes the longest to migrate
- Once migrated, a V7 DMGR can happily manage lower-level federated nodes

So the point of this strategy is to get the V7 DMGR created with as little disruption to your existing environment as possible. Then, when the V7 DMGR is ready to go, you swap the old DMGR out and swap the new one in.

This isn't that complicated a strategy. But there are a few things that can make it go more smoothly:

- **Don't disable the previous DMGR**

  There's a checkbox in the MMT for a DMGR migration that defaults to "Diable the previous Deployment Manager." See page 30 for a discussion of that option and a picture of the MMT panel. The main point is this -- to preserve your ability to easily start the current DMGR if you had to, unchecking this box is advised.

  **Note:** But the responsibility is then on you to make sure that older version of the DMGR is "retired" when you're comfortable with the migrated V7 DMGR's ability to be the operational DMGR.

- **Use new JCL start procedure names for the migrated DMGR**

  The new V7 JCL start procedures won't work to start the older-level DMGR. If you used the same names, the copy job would overlay your existing DMGR procs. Better to maintain maximum flexibility to start either DMGR as needed. This is accomplished by specifying new JCL start procedure names in the MMT and making sure the "Replace started procedure command names" checkbox is selected.

You'll may need a new SAF `STARTED` profile to support your new DMGR JCL start procedure.
• *Leave the existing DMGR and all the other existing runtime up and running*

• *Submit the migration jobs for the DMGR and look for RC=0 signalling success*

At this point you need to pause and take stock of things. You have a V7 DMGR that's ready to start but it can't be started at the same time as your existing DMGR or you'll experience port conflicts.

In addition, the new V7 DMGR can *not* operate with the older Daemon. So the existing Daemon needs to come down, which of course means all the servers for that cell on that z/OS image will also come down.

So give some thought to when you want to do this.

When you're ready, then ...

• *Drop the old Daemon that supports the existing DMGR.*

• *Leave any servers for nodes on other z/OS images running*

• *Start your new DMGR. It will automatically start the migrated V7 Daemon.*

• *Perform preliminary validation testing to make sure the DMGR is talking to pre-migration Node Agents that were still running*

  Look in the V7 Administration Console and make sure the Node Agent has a status of "green arrow" and the status of the node is synchronized, and the servers themselves show a status of green arrow, and any applications show a status of green or partial green.

• *Restart the not-yet-migrated servers on the same z/OS image and check the Administrative console to make sure everything you’d expect to be green is green.*

Your DMGR is now at V7.0 and the rest of your environment is still at the prior level. The only disruption you took was the stop/restart caused by the need to bring the V7.0 Daemon up. No disruption occurred during the running of the migration itself.

You could at this time go back to the prior DMGR ... even leaving the V7 Daemon up if you wished.

---

**Be Careful!** At this point you have two potentially operational DMGRs. Be very aware of which one is up when you make configuration changes, because that's where the changes will be made.

The worst scenario is making a bunch of changes in the old DMGR and a bunch in the new DMGR. There's no facility to match/merge the changes between two different DMGRs. It would be up to you to figure it out and make the changes again in the DMGR you wished to make permanent.

The main point is this -- once you've validated that the migrated V7 DMGR is good and operational, **retire the old DMGR.** See “Changes made to the existing node configuration” on page 81 for an illustration of how the old DMGR would have been disabled had you allowed it with that “Disable previous Deployment Manager” checkbox. Do something similar and you've effectively disabled the old DMGR.

**Enabling 64-bit JVMs for the migration process**

If you find a need to use JVM heaps larger than the 31-bit JVM will allow, then you will need to make a few modifications to get the 64-bit JVM launched.

**Note:** Generally speaking the potential need for a 64-bit JVM applies to the migration of DMGR nodes and Standalone servers, and even then only if the configuration is too large to contain within the default 31-bit JVM. Federated appserver nodes should not require it. You can migrate a federated node with a 64-bit JVM ... it won’t hurt anything ... it just likely isn’t necessary.

Before doing this, make sure the out-of-memory condition you’re experiencing is indeed due to the 31-bit heap limitations and not due to **other** restrictions on the system. For example, a common restriction we see are IEFUSI exits that limit memory allocation.
Once you’ve determined that a 64-bit JVM is needed, then understand this involves two things:

1. Updating the migration jobs so a larger heap size is requested
2. Updating some shell scripts to get the /java64 libraries used rather than the /java libraries

Neither is particularly difficult. It’s best explained through illustration.

**Background -- how the pieces tie together**

There are many connections between the batch JCL and the place where the JVM is actually invoked. We’ll use the PREUPGRD step of the BBOWDPRE job as an illustration. (The UPGRADE step in job BBOWDPOS is the other that may require larger heaps. The file relationships are the same, but with a few different names.)

We will refer back to this diagram several more times.

---

**Figure -- connections from migration JCL to shell scripts**

**Notes:**

1. The JCL step involves the invocation of BPXBATCH and a pointer to the shell script to be run.
2. The bbomigrt2.sh shell script is invoked and the parameter WASPreUpgrade is passed in. That parameter names the function within bbomigrt2.sh to invoke.
3. The bbomigrt2.sh does a little housekeeping then turns and invokes the WASPreUpgrade.sh shell script under the node’s /profiles/default/bin directory. The copy of the shell script here is very small … it simply calls other scripts.
4. One of those scripts is setUpCmdLine.sh, which establishes some key environment settings for the invocation of the subsequent shell scripts.
5. The copy of WASPreUpgrade.sh under the node’s /bin directory is then called. But that is really just a symbolic link.
6. The real copy of WASPreUpgrade.sh is located in the “product file system” … sometimes called the “SMP/E file system.”
BBOW*PRO, BBOW*PRE and BBOW*POS ... not BBOWMG3*

The updates we are about to suggest assume you are running the three separate jobs and not the one big job. See "Run the migration as three separate jobs" on page 64 for more on this topic.

Further, it assumes you run the BBOW*PRO job (which is what creates the skeleton V7 configuration tree), make the 64-bit JVM update changes in the V7 configuration tree, then run BBOW*PRE and BBOW*POS.

You’d have to be a pretty fast typist to make those changes in the middle of BBOWMG3* running those three steps as one continuous job.

Step 1 -- ready the environment for shell script edits

Do the following:

- Rename the WASPreUpgrade.sh and WASPostUpgrade.sh symbolic links under the /bin directory (block 3 in the figure above). This serves as a backup of the symlinks.
- Copy the real WASPreUpgrade.sh and WASPostUpgrade.sh shell scripts from the product file system (block 2 in the figure above) into the /bin directory.
- Insure the just-copied files carry the same owning ID and group as other symlinks in the directory, and the permissions are at least 755.

Step 2 -- edit files

- Do the following to the migration JCL PREUPGRD and UPGRADE steps. The syntax and placement is identical for both; here we’re showing just the PREUPGRD.

```
//PREUPGRD   EXEC PGM=IKJEFT01,REGION=0M,COND=(4,LE)
//SYSTSPRT DD SYSOUT=*  
//STDENV DD * // _CEE_RUNOPTS=TRAP(ON,NOSPIE) //*
//SYSTSIN DD * 
BPXBATCH SH +
  export IBM_JAVA_OPTIONS="-Xms1024M -Xmx4096M"; +
  /tmp/migrate/B2DMGR_1/bbomigrt2.sh WASPreUpgrade +
  /tmp/migrate/B2DMGR_1/17143345/__ +
  1>> /tmp/migrate/B2DMGR_1/17143345/BBOWMG3D.out +
  2>> /tmp/migrate/B2DMGR_1/17143345/BBOWMG3D.err;
/*
```

Notes:
- We're illustrating a JVM heap of minimum 1024MB and maximum of 4096MB. The values you choose may be different
- Syntax matters ... make sure you have the double quotes, dashes, semi-colon and the following + sign to indicate continuation.
- The maximum heap of 4096MB implies 64-bit ... we've not yet shown you how to invoke that. The point is you must invoke a 64-bit JVM if you use 64-bit addressing.
- Make sure you have sufficient real storage available to handle the larger heap.
- See WP101589 on ibm.com/support/techdocs for other JVM options to possibly improve migration performance.
For the WASPreUpgrade.sh file you copied into the node's /bin directory:

```
EDIT WASPreUpgrade.sh
Command ===>
000022 # Set java options for performance
000023 PLATFORM="/bin/uname"
000024 case $PLATFORM in
000025   AIX)
000026       PERF_JVM_OPTIONS="-Xms256m -Xmx512m -Xquickstart";;
000027   Linux)
000028       PERF_JVM_OPTIONS="-Xms256m -Xmx512m -Xj9 -Xquickstart";;
000029   SunOS)
000030       PERF_JVM_OPTIONS="-Xms256m -Xmx512m -XX:PermSize=40m -XX:+UnlockDi
000031   HP-UX)
000032       PERF_JVM_OPTIONS="-Xms256m -Xmx512m -XX:PermSize=40m -XX:+UnlockDi
000033   OS/390)
000034       PERF_JVM_OPTIONS="-Xms1024m -Xmx4096m";;
000035 esac
000036
000037 "$WAS_HOME/java64/bin/java" "$OSGI_INSTALL" "$OSGI_CFG" \\
(above line, but not relevant to this discussion)

Notes:
1. Set the minimum and maximum heaps to the same values you set in the JCL earlier.
2. Originally this was "$JAVA_HOME/bin/java". That variable JAVA_HOME gets set in the setupCmdLine.sh shell script as $WAS_HOME/java. If we'd simply added "64" to the end of what was there initially it would have resolved to $WAS_HOME/java/bin/java64 ... which would have resulted in a failure.
   One option would be to change setupCmdLine.sh and modify the JAVA_HOME setting there.
   The simpler thing is to do what's illustrated above ... take out $JAVA_HOME and substitute in $WAS_HOME/java64.
```

For the WASPostUpgrade.sh file you copied into the node's /bin directory, simply do the exact same as you did for the WASPreUpgrade.sh. The files are nearly identical near the top where these settings are made.

**Step 3 -- run jobs**

With those changes in place, the job will now invoke BPXBATCH and specify the larger heaps. The shell scripts will invoke the 64-bit JVM code. The larger heap settings will be utilized.

**Providing 64-bit and large heap for deferred application installs using WSADMIN**

The migration utility for V7 provides an option to defer installation of applications until a later point. In essence what it does is store the applications to the file system and generate WSADMIN scripts to deploy them later.

If you chose that option you may find you need larger heaps for the process of installing the applications.

The process is similar to what you did for the Pre- and PostUpgrade.sh scripts.

Do the following:

- Make a backup copy of setupCmdLine.sh in the node's /profiles/default/bin directory.
Edit the `setupCmdLine.sh` file and add “64” to the `JAVA_HOME` specification:

```bash
EDIT setupCmdLine.sh
Command ==> 000045 WAS_HOME=/wasv7config/mycell/mynode/...
            000046 JAVA_HOME="${WAS_HOME}/java64"
            000047 WAS_CELL=cel1
```

(Read below this line, but not relevant to this discussion)

- Change to the node’s `/bin` directory (not `/profiles/default/bin`) and rename the symbolic link with name `wsadmin.sh`. This is, in effect, making a backup copy.
- Copy the `real` `wsadmin.sh` from the product file system (block 3 in the figure above) into the `/bin` directory.
- Edit the just-copied `wsadmin.sh` file and set the JVM minimum and maximum heap values to your desired values:

```bash
EDIT wsadmin.sh
Command ==> 000074 # Set java options for performance
            000075 PLATFORM=/bin/uname
            000076 case $PLATFORM in
            000077   AIX)
            000078       PERF_JVM_OPTIONS="-Xms256m -Xmx256m -Xquickstart" ;;
            000079   Linux)
            000080       PERF_JVM_OPTIONS="-Xms256m -Xmx256m -Xj9 -Xquickstart" ;;
            000081   SunOS)
            000082       PERF_JVM_OPTIONS="-Xms256m -Xmx256m -XX:MaxPermSize=128m" ;;
            000083   HP-UX)
            000084       PERF_JVM_OPTIONS="-Xms256m -Xmx256m -XX:MaxPermSize=128m" ;;
            000085   OS/390)
            000086       PERF_JVM_OPTIONS="-Xms256m -Xmx256m -XX:MaxPermSize=128m" ;;
            000087 esac
```

(Read below this line, but not relevant to this discussion)

**Shared address space for spawned child processes (_BPX_SHAREAS=NO)**

The setting for `_BPX_SHAREAS` determines whether spawned child processes run in the *same* address space or a *separate* address space.

- `_BPX_SHAREAS = YES` means spawned processes run in the *same* address space.
- `_BPX_SHAREAS = NO` means spawned processes run in a *separate* address space.

We bring this up for this reason: if your setting turns out to be `YES` and the user region is too small to contain more than a single JVM heap, the migration job will fail.

If you’re certain that `NO` is your system’s setting, you don’t need to worry about this.

But if your system’s setting is `YES`, or you’re not sure, you can force `NO` to be used for the `CRPROF` step with a simple statement. (And yes, you may use this in combination with the statement to increase the heap size.)

Here’s an example of `_BPX_SHAREAS=NO` and the heap size statement used together:

```bash
Note: We show the two used together simply as an illustration. It is **not** a requirement to increase the Java heap when `_BPX_SHAREAS=NO` is used. You **may**, but you are not required to.
```
Notes:
1. The setting of a higher Java heap as an illustration of multiple statements. Note the semi-colon and plus sign at the end of that.
2. The `_BPX_SHAREAS=NO` statement. Note the semi-colon and plus sign at the end of that.

An illustration of these strategies employed for a hypothetical migration

We've introduced three strategies -- breaking the main migration into three smaller steps; separating the application migrations from the runtime migration; and maximizing server uptime during migration migration. Let's paint a hypothetical and see all three in action.

Starting environment

Imagine an environment something like this:

**Note:** No magic to the “V6.0” -- just an arbitrary selection of a supported “from” version.
Deployment Manager migration

**MMT panels**

Key MMT panel settings:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daemon procedure name</td>
<td>MY7DEM</td>
</tr>
<tr>
<td>Controller procedure name</td>
<td>MY7DCR</td>
</tr>
<tr>
<td>Servant procedure name</td>
<td>DY7DSR</td>
</tr>
</tbody>
</table>

This shows the specification of different JCL start procedure names for the DMGR servers and the Daemon.

**Note:** Not strictly required. This is an insurance policy to give maximum flexibility in starting the old DMGR "just in case."

Specify the following information to customize your migration.

1. **Migrate to support script compatibility**
2. **Disable previous deployment manager**

Application migration preference:

- Migrate applications and use the default application installation directory
- Migrate applications and use the specified application installation directory
  
  Application installation directory: `/wasp7/config/DeploymentManager/profiles/default/InstalledApps`

3. **Migrate and generate administrative scripts to install applications later**
   - Migrate applications and use the previous application installation directory
   - Do not migrate applications

Notes:

1. The "Migrate to support script compatibility" is off in this picture though it's not really part of this "large topology" discussion. Let's assume we did not have any scripts that required any concern about compatibility. Hence this is unchecked.

2. This is the main point -- unchecking the "Disable previous deployment manager" box means the migration utility will *not* do the things discussed under "Changes made to the existing node configuration" on page 81.

3. Selecting this radio button means the migration utility will not spend time and resources deploying the applications into the new V7 DMGR. It'll instead put the applications out as EAR files and generate WSADMIN scripts to install them later.

Temporary directory location: `/u/myuser/mig_temp`

Migration definition identifier: 10116369

This shows the temporary location specified. Assume we mounted a 500-cylinder ZFS at that location to insure plenty of room for the utility to do its work.
**Leave entire existing runtime operational**

Nothing is shut down at this point. The entire runtime continues to process work.

**Execution of migration jobs**

**BBOMDZFS and BBOMDCP**

These create the target V7 file system and copy in the new JCL start procedures.

Because new names were used, **BBOMDCP** does *not* overwrite the existing JCL start procedures.

**BBOWDPRO**

This job creates the V7 product directory and creates the V7 default profile, both in the target ZFS allocated and mounted with the **BBOMDZFS** job. It also creates some files down in the `/u/myuser/mig_temp` temporary directory. But *not*, at this point, the "backup" directory we spoke of before.

The **BBOWDPRE** job creates the profile backup directory as we'll see.

**Error and cleanup of BBOWDPRO with re-submit**

Let's assume something went wrong and the **BBOWDPRO** job returned a non-zero return code in one of its steps. Without going into a detailed debugging process here, imagine:

- Looking in the `/u/myuser/mig_temp/BBOWDPRO.err` and `BBOWDPRO.out` files for an indication of the problem.
- Seeing if "Common Problems and Their Causes" on page 60 offers any insights into the problem.
- Correcting the problem.
- Looking at "Three jobs in flowchart form, with debug and error recovery information" starting on page 66 and seeing what the cleanup process is for a failed **PRO** job.
- Performing the cleanup, which involves cleaning up the target V7 directories. The ZFS can stay mounted. It's what's in the ZFS that's cleaned up.
- Resubmitting the job
- Watching it complete successfully this time.

Further validation of this can be found by going to the target V7 ZFS:

`/<mount>/DeploymentManager/profiles/default/logs/AboutThisProfile.txt`

**BBOWDPRE**

This job creates a backup of the existing profile in preparation for the **BBOWDPRES** job performing the migration into the target ZFS.

The backup directory in this scenario:

`/u/myuser/mig_temp/10110309/dmgr_backup`

**Note:** The "10110309" portion of that is the timestamp-based number generated by the MMT. It was shown on the MMT panels under the field where the temporary directory was specified.

The **BBOWDPRE** job makes no changes to the target V7 ZFS.

Had this failed, the cleanup would have been to clear the backup directory in the temporary file system.
This job is what performs the migration of the DMGR. It takes as input the files held in the backup directory, transforms them as part of the migration and copies them into the target V7 ZFS.

Had this failed, cleanup is generally not necessary since this is a DMGR. Simply re-run the BBOWDPOS job after the error has been corrected.

**Note:** If this had been a federated node, then the cleanup plans call for the backup directory to be cleared prior to the resubmission of BBOWDPOS.

**Status at this point**

Assuming the BBOWDPOS job ran successfully, your DMGR is now created at the V7 level.

However, there are no applications deployed into master configuration of the V7 DMGR.

**Note:** Starting this new V7 DMGR at this point would be potentially disruptive if synchronization were to take place. The DMGR sees no applications deployed, which means upon synchronization the applications would be removed from the federated nodes.

The previous version of the DMGR remained active during all of this, as did all of the application servers. No disruption to your existing environment took place.

The picture looks like this:

---

**Running the WSADMIN scripts to deploy the applications**

As explained earlier, when the option to "install applications later" is checked, two things happen:
1. The applications are placed in a directory in EAR file format. The directory:
   
   /<mount>/DeploymentManager/profiles/default/installableApps

2. WSADMIN scripts are created to perform the installation:
   
   /u/myuser/mig_temp/10110309/dmgr_backup

   There is a file called install_all_apps.jy that will install every application in one large job. Individual WSADMIN scripts are also present to install the applications individually.
   
   Running the individual WSADMIN scripts is how you'd perform housekeeping and get rid of applications no longer needed.

   **Key Point:** This process is done entirely "offline" -- the original DMGR is still up and operational, as are all the application servers and applications. If you wanted to take a week or more to perform the application installations that would not affect your existing runtime, which continues to be operationally functional.

### Starting the new V7 Deployment Manager

Once the applications are installed into the DMGR, you may attempt starting the V7 DMGR. Some points:

- The old DMGR and the new can not be started concurrently.
- The new V7 DMGR can not function under the old Daemon server. The old Daemon must be shut down and the new Daemon started by the DMGR when it starts.
- Stopping the old Daemon will mean all the servers for that cell on that z/OS image will stop.
- If some of the applications were *not* deployed using WSADMIN, then if synchronization occurs those applications will be removed from the actual servers.

   **Note:** If you're not certain what the effect of synchronization may be, you *may* wish to consider disabling synchronization for the Node Agents before stopping the existing DMGR and attempting to start the new. Just a thought.

With those thoughts in mind, the process would be:

- Stop the Daemon. This will bring down the old DMGR and all other servers for that cell on this z/OS image.
- Start the new DMGR using the new JCL start procedure. It will issue the necessary START command to start the new V7 Daemon.
- Once the DMGR is up, a quick validation in the Administrative Console would be advised:
  - Check the status of the servers -- they should show green arrow
  - Check the status of the applications -- they too should show green arrow
  - Check the status of the Node Agents -- they should show green arrow

   **Note:** If synchronization is disabled, the status for the nodes may show “not synched.”

- When you're satisfied things look operational, you may wish to enable synchronization and insure the DMGR is capable of synchronizing to the nodes.
- Perform whatever other validation testing you believe is needed.

The picture at this point would be:
**Disabling the old Deployment Manager**

At this point two potentially operational DMGRs are present. It would be possible to stop the V7 DMGR and start the V6 DMGR, even without stopping the V7 Daemon.

Having two potentially operational DMGR's around is somewhat risky. As stated before, the risk is losing track of which is started and making critical configuration changes in one when it was assumed the other was active. The V7 Administrative Console looks enough different from a V5.1 or V6.0 console to catch the eye, but the differences between V6.1 and V7.0 are subtle and would likely be overlooked.

Better, once the V7 DMGR is validated, to retire the older DMGR. An easy, recoverable way to do this is to do manually what the migration utility would do. That is illustrated under "Changes made to the existing node configuration" on page 81.

**Migration of federated nodes**

This may now proceed as illustrated elsewhere in this document. The option of running the three smaller jobs (PRO, PRE and then POS) is available for this migration as well.

Federated nodes may be migrated while the servers are up, unless XA connectors are in use and the MG1* and MG2* jobs are required to flush the transaction logs. If that's the case, then the servers must be stopped as part of that process.

If migrating a federated node on another z/OS image from the DMGR, then the Daemon instance will be migrated and the old Daemon will have to be stopped at some point so the new V7 Daemon can be brought up. Doing that will drop all the servers for the cell on that z/OS image.
Miscellaneous Information

**Acquiring and installing the WCT**

There are two places to acquire the WCT:

1. On the "WCT CD" that shipped with the WebSphere Application Server for z/OS V7.0 product. The program directory for WebSphere Application Server for z/OS V7.0 provides information on the supplied DVD/CD kits. The arrow indicates the DVD/CD on which the WCT can be found:

   - DVD/CD Kit Numbers
     - LK4T-1408-00 - IBM Support Assistant
     - LK4T-1558-00 - WebSphere Customization Tools
     - LK4T-1546-00 - Supplemental Material for AIX
     - LK4T-1547-00 - Supplemental Material for HPUX
     - LK4T-1548-00 - Supplemental Material for Linux
     - LK4T-1549-00 - Supplemental Material for Solaris
     - LK4T-1550-00 - Supplemental Material for Windows

2. At the following IBM web page where the tool can be downloaded:
   
   http://www.ibm.com/support/docview.wss?rs=180&uid=swg24020368

The installation is fairly typical and easy. On Windows the install.exe file launches the installation program, and from there there is but a few steps to install it, much like any other Windows program.

**Updating the level of the WCT**

At the time this document was written the method for updating the level of the WCT is to uninstall the WCT and reinstall it at the newer level. Your locations and definitions do not get uninstalled -- they are considered separate from the product installation.

**Changes made to the existing node configuration**

**DMGR**

Changes to the original DMGR configuration only occur if you tell the migration utility to do it:

Two changes take place:

1. The serverindex.xml file for the DMGR is copied to serverindex.xml_disabled
2. The original serverindex.xml file for the DMGR is updated so all ports have a value of 0

The second thing is what disables the old DMGR from starting. The first is what makes it easy to recover if you have to -- simply delete the serverindex.xml file and rename serverindex.xml_disabled to serverindex.xml.
Federated Node

The MMT has no option to not disable the node being federated. It will by default. The disabling that takes place is very similar to the DMGR:

1. The serverindex.xml file for the node is copied to serverindex.xml_disabled
2. The original serverindex.xml file for the node is updated so the Node Agent ports are all 0

The key difference from the DMGR is in item #2 -- a federated node may have many servers in it, whereas a DMGR node by definition has just the DMGR server itself.

Rather than change every port for every server to 0, the migration utility simply changes the Node Agent ports to 0. The disabled Node Agent won't start, insuring that no updates to that node will take place if you accidentally start the old DMGR.

Standalone

See the description for a "Federated Node" ... it's the same for a Standalone.

Job to create group ID with sufficient UNIXPRIV authority

Earlier we listed the UNIXPRIV profiles that an ID needed to have "system update authority." You could PERMIT the WebSphere Admin ID access to each profile.

Perhaps a better way is to create a Group ID with the access rights, and connect the WebSphere Admin ID to the Group. Then when the migration work is done, you can REMOVE the Admin ID from the group. Having too many IDs with enhanced authorities may not be what you want.

Here's a job we built to create a MYMIG Group ID that possessed the needed authorities. The job also connected the MYADMIN ID to the new group.

```bash
//MIGRATE   JOB (ACCTNO,ROOM),CLASS=A,REGION=0M,
//   USER=******,PASSWORD=******
//STEP1 EXEC PGM=IKJEFT01,DYNAMNBR=20,REGION=2M
//SYSTSPRT  DD SYSOUT=*
//********************************************************************
//SYSTSIN   DD *
ADDGROUP MYMIG OMVS(GID(58105))  1
CONNECT MYADMIN GROUP(MYMIG)  2
PERMIT SUPERUSER.FILESYS ACC(CONTROL) ID(MYMIG) CL(UNIXPRIV)  3
PERMIT SUPERUSER.FILESYS.MOUNT ACC(UPDATE) ID(MYMIG) CL(UNIXPRIV)  4
PERMIT SUPERUSER.FILESYS.CHOWN ACC(READ) ID(MYMIG) CL(UNIXPRIV)  5
PERMIT SUPERUSER.FILESYS.CHANGEPERMS ACC(READ) ID(MYMIG) CL(UNIXPRIV)  6
PERMIT SUPERUSER.FILESYS.PFSCTL ACC(READ) ID(MYMIG) CL(UNIXPRIV)  7
SETROPTS RACLIST(UNIXPRIV) REFRESH  8
/*
Notes:
1. The creation of the new group ID that would carry the UNIXPRIV and file system mount authorities.
   Note: If you already have a group ID with these authorities you don't need to create another. Simply connect the WebSphere Admin ID to that group.
2. The WebSphere Admin ID is connected to this new group.
3. Group is granted CONTROL to FILESYS
4. Group is granted UPDATE to FILESYS.MOUNT
5. Group is granted READ to FILESYS.CHOWN
6. Group is granted READ to FILESYS.CHANGEPERMS
7. Group is granted READ to FILESYS.PFSCTL
8. The class UNIXPRIV is refreshed
```
The zmmt.sh shell script

This utility is found in the /bin directory of a V7.0 configuration profile. It is a non-GUI tool used to generate the migration jobs. It does not require the WCT to be installed on any workstation. The result is a set of jobs just like the WCT produces, so the planning for and execution of the jobs is the same, WCT or zmmt.sh script.

The key to understanding the zmmt.sh script is to think about how it receives the variable input used to generate the jobs. The MMT in the WCT does this using GUI panels and your input. But the zmmt.sh shell script does it by requiring a flat response file with the variables and their values as input. So the question is, how is that response file created?

There’s a very good write-up on the zmmt.sh shell script in the V7 InfoCenter. The URLs for specific topics in the InfoCenter tend to be very long. Perhaps a better way to reference the article is this:

- Go to the InfoCenter:
  http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp
- Search on this string:
  tmig_zmmt_cmd_using.html

That is the main article related to the use of the zmmt.sh utility. Down at the bottom of that page you’ll see a heading for "Prepare the type of response file that you need." Click on the link related to the type of node being migrated, then use the reference information there to manually create the response file you desire.

Or you can use the WCT.

When the Daemon JCL start procedure is common and shared between images

In the V6.0 migration document we labeled this "the perfect storm" -- a problem that crops up only when a number of things come together to create the circumstances that cause the problem.

Here are the three things that have to be present for this "gotcha" to occur:

1. A multi-system ND cell where multiple Daemon instances for the cell exist
2. The migration customization specified the re-use of the JCL start procedure names
3. A single, common Daemon JCL start procedure is used by all the Daemons in the cell

The first two items shown are fairly common, and your migration may well have those things present. The third item is far less common.

What is the problem?

The problem pops up when the Daemon on LPAR A (for example) is migrated to V7.0 and the new JCL start procedure, using the same name as before, is copied into the proclib. The old Daemon JCL start procedure is overwritten.

As long as the Daemon on LPAR B remains running prior to the node on LPAR B being migrated to V7, there's no problem.

But if for whatever reason that Daemon on LPAR B is brought down before the node is migrated, then any attempt to restart the Daemon will fail. Because both Daemons are using the same proc out of a shared proclib, then the new V7 JCL will be picked up and used. But the Daemon instance on LPAR B is still at a pre-V7 level.

V7.0 JCL starting a pre-V7 node will not work ... the Daemon will fail to start.

This is all because of a common, shared JCL proc between the Daemons.
What's the solution?

Very simple -- once you migrate the first node in the cell, be sure to leave the Daemons for the other systems up and running. *Do not do anything that would initiate a restart of those other Daemons!* Migrate the nodes on the other systems and then stop and restart those Daemons once they've been migrated up to V7.

**Note:** The most likely place this will occur is between the BBOWMG1F job and the BBOWMG2F job. The instructions call for the servers to be stopped when BBOWMG1F is run. But the Daemon server does not need to be stopped ... it may remain running. Stop and restart the Daemon only after the BBOWMG3F job runs successfully and the Daemon instance has been migrated up to V7 to match the JCL start proc that would be used.

Does this even apply to me?

<table>
<thead>
<tr>
<th>Is the cell you are migrating a Network Deployment cell that spans at least two z/OS system images?</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Read on</td>
<td>You don't need to worry about this particular issue</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Are you re-using the JCL start procedure names?</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Read on</td>
<td>Using new start procedure names avoids this particular rare gotcha</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Does the Daemon on one LPAR employ the same JCL start procedure as the Daemon(s) on the other LPAR(s)</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Read on</td>
<td>LPAR-specific Daemon proc names is good ... it avoids this issue</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Is the Daemon JCL proc used physically the same proclib member in the same proclib?</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uh oh ... you have the problem</td>
<td>It's possible to have system specific proclibs where the same proc name can be used, but they're really physically separate members in different proclibs. If that's your situation, then you don't have this particular problem.</td>
<td>And it's likely related to using a common, shared configuration file system for the cell</td>
</tr>
</tbody>
</table>

Single, common Daemon JCL? How is that even possible?

There's one way that's likely to happen -- when the configuration file structure used by the nodes in the cell are all held within a single, common, shared file system (HFS or zFS).

Way back in the early V5.0 days we used to recommend that. We've long since stopped recommending a single configuration file system. Now we *strongly* encourage that each node have its own separate configuration file system.

**Note:** Separate configuration files systems for each node improves backup and restore granularity and it promotes greater isolation between nodes. An example of lack of isolation is this problem we're discussing right here.

The reason why a single, common and shared configuration file system is key to allowing this to work is found at the top of the Daemon JCL procs:

```
//B1DEMND PROC ENV=,Z=B1DEMNDZ
// SET ROOT='/wasv6config/b1cell'
//BBODAEMN EXEC PGM=BBODAEMN,REGION=0M
```

That "SET ROOT=" variable is what points to where the configuration file system is mounted. To use the same physical JCL proc for multiple Daemon instances would suggest one SET ROOT= value (mount point), which implies a single, shared file system at that mount point.
If you had separate configuration file systems that would suggest separate mount points. Separate mount points would imply different `SET ROOT=` values, which would imply separate JCL start procedures.

Do you see why having a common, shared configuration file system is really the enabler of this?

**Note:** There may be fancy, tricky ways to accomplish a common JCL even though the nodes employ separate file systems. JCL with system symbolics and mount points with the system name as part of its root is one that comes to mind. But those would be unique, manually created things ... not JCL generated by WebSphere itself.
Migration Planning Review Outline

A planning review is really just a structured walk-through of key issues so any issues or concerns are raised in discussion rather than at the time of execution. Take some time and go through this list with those involved with the migration. Gauge your readiness to perform the migration based on the confidence displayed in discussion that takes place.

WCT installation and validation

- The WCT tool has been acquired and is installed?
  - Yes  No
- The WCT tool has been started at least once and its basic ability to initialize validated?
  - Yes  No
- The WCT has been explored and familiarization with the basic elements -- locations and definitions -- has been accomplished?
  - Yes  No
- A dummy definition has been created and uploaded to z/OS to validate the upload capability (this is to validate access through firewalls to the z/OS system).
  - Yes  No

WebSphere z/OS V7 installation and validation

- WebSphere Application Server for z/OS V7.0 has been installed and all the system programming activities listed in the Program Directory accomplished?
  - Yes  No
- The key prerequisites are in place (z/OS PTFs and other required maintenance)?
  - Yes  No
- A V7.0 configuration has been created and at least one server started to validate the essential elements of the system?
  - Yes  No

Pre-migration SAF work

- Has the WebSphere Admin ID been connected to a Group with the appropriate UNIXPRIV privileges? (See page 82 for an example of RACF to accomplish that.)
  - Yes  No
- If you are planning to use new JCL start procedure names, will the existing STARTED profiles work the new names? If not, have new STARTED profiles been created?
  - Yes  No
- Have you read the MMT migration instruction information regarding TRUSTEDAPPS and SYNC and are those profiles in place to support your migrated V7.0 cell?
  - Yes  No
### MMT customization planning

**Deployment Manager node**

**Existing configuration inventory:**

<table>
<thead>
<tr>
<th>Object</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Name:</td>
<td></td>
</tr>
<tr>
<td>Version prior to migration:</td>
<td></td>
</tr>
<tr>
<td>Configuration mount point:</td>
<td></td>
</tr>
<tr>
<td>Configuration file system data set:</td>
<td></td>
</tr>
<tr>
<td>File system size and utilization:</td>
<td></td>
</tr>
<tr>
<td>JCL Start Procedures</td>
<td></td>
</tr>
<tr>
<td>Control Region:</td>
<td></td>
</tr>
<tr>
<td>Servant Region:</td>
<td></td>
</tr>
<tr>
<td>Daemon:</td>
<td></td>
</tr>
</tbody>
</table>

**V7.0 Migration Customization:**

<table>
<thead>
<tr>
<th>Object</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMT location name:</td>
<td></td>
</tr>
<tr>
<td>MMT definition name:</td>
<td></td>
</tr>
<tr>
<td>Target data set HLQ:</td>
<td></td>
</tr>
<tr>
<td>Location of V7.0 SMP/E mount:</td>
<td></td>
</tr>
<tr>
<td>Use Intermediate Symbolic Link?</td>
<td>Yes</td>
</tr>
<tr>
<td>If yes, symlink location and name:</td>
<td></td>
</tr>
<tr>
<td>Configuration file system mount:</td>
<td></td>
</tr>
<tr>
<td>Configuration file system data set:</td>
<td></td>
</tr>
<tr>
<td>HFS or ZFS</td>
<td>HFS</td>
</tr>
<tr>
<td>Primary/secondary allocation:</td>
<td></td>
</tr>
<tr>
<td>JCL Start Procedures strategy:</td>
<td>Reuse same names</td>
</tr>
<tr>
<td>If new JCL names:</td>
<td></td>
</tr>
<tr>
<td>Control Region:</td>
<td></td>
</tr>
<tr>
<td>Servant Region:</td>
<td></td>
</tr>
<tr>
<td>Daemon:</td>
<td></td>
</tr>
<tr>
<td>&quot;Script Compatibility&quot; an issue?</td>
<td>Yes</td>
</tr>
<tr>
<td>Application migration preferences:</td>
<td>Default</td>
</tr>
<tr>
<td>Preserve any &quot;My tasks?&quot;</td>
<td>Yes</td>
</tr>
<tr>
<td>Temporary space location:</td>
<td></td>
</tr>
<tr>
<td>Temporary space file system size:</td>
<td></td>
</tr>
</tbody>
</table>
Federated application server node (or standalone server)

**Note:** Complete for each federated application server node in the cell.

### Existing configuration inventory:

<table>
<thead>
<tr>
<th>Object</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Name:</td>
<td></td>
</tr>
<tr>
<td>Version prior to migration:</td>
<td></td>
</tr>
<tr>
<td>Configuration mount point:</td>
<td></td>
</tr>
<tr>
<td>Configuration file system data set:</td>
<td></td>
</tr>
<tr>
<td>File system size and utilization:</td>
<td></td>
</tr>
<tr>
<td>JCL Start Procedures</td>
<td></td>
</tr>
<tr>
<td>Control Region:</td>
<td></td>
</tr>
<tr>
<td>Servant Region:</td>
<td></td>
</tr>
<tr>
<td>Adjunct Region:</td>
<td></td>
</tr>
<tr>
<td>Daemon:</td>
<td></td>
</tr>
</tbody>
</table>

### V7.0 Migration Customization:

<table>
<thead>
<tr>
<th>Object</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMT location name:</td>
<td></td>
</tr>
<tr>
<td>MMT definition name:</td>
<td></td>
</tr>
<tr>
<td>Target data set HLQ:</td>
<td></td>
</tr>
<tr>
<td>Location of V7.0 SMP/E mount:</td>
<td></td>
</tr>
<tr>
<td>Use Intermediate Symbolic Link?</td>
<td></td>
</tr>
<tr>
<td>If yes, symlink location and name:</td>
<td></td>
</tr>
<tr>
<td>Configuration file system mount:</td>
<td></td>
</tr>
<tr>
<td>Configuration file system data set:</td>
<td></td>
</tr>
<tr>
<td>HFS or ZFS</td>
<td></td>
</tr>
<tr>
<td>Primary/secondary allocation:</td>
<td></td>
</tr>
<tr>
<td>JCL Start Procedures strategy:</td>
<td></td>
</tr>
<tr>
<td>If new JCL names:</td>
<td></td>
</tr>
<tr>
<td>Control Region:</td>
<td></td>
</tr>
<tr>
<td>Servant Region:</td>
<td></td>
</tr>
<tr>
<td>Adjunct Region:</td>
<td></td>
</tr>
<tr>
<td>Daemon:</td>
<td></td>
</tr>
<tr>
<td>“Script Compatibility” an issue?</td>
<td></td>
</tr>
<tr>
<td>Application installation location:</td>
<td></td>
</tr>
<tr>
<td>Preserve any “My tasks?”</td>
<td></td>
</tr>
<tr>
<td>Temporary space location:</td>
<td></td>
</tr>
<tr>
<td>Temporary space file system size:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Migration execution planning

- Do you have in place a plan to backup the entire WebSphere cell environment?
  □ Yes □ No

- Will the migration be performed during a window of time when all work is suspended to the cell, or will the migration be performed concurrent with real-time work activity in the cell?
  □ Work Suspended □ Concurrent with real-time work

- If concurrent with real-time work, do you have in place a plan to quiesce requests to the node being migrated? Has that methodology been validated?
  □ Yes □ No

- What is the intended node-by-node sequence of execution?
  □ First node migrated should always be the Deployment Manager.
  □ Second node migrated: _____________________________
  □ Third node migrated: ______________________________
  □ Fourth node migrated: ______________________________
  □ Other node migrated: ______________________________

- What are the key checkpoint milestones of your migration plan? What is the key success criteria for each checkpoint milestone?

  Note: A "checkpoint milestone" is a point in time where some key success criteria has been met. It also constitutes a point in time where the fallback strategy may need to be invoked, depending on the success or failure of the migration. Example: "DMGR node migrated and successfully restarted with connectivity to existing pre-migrated nodes verified."

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Success Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Do you have in place a fallback strategy that takes into account the two key "states" of the configuration:
  □ DMGR only node migrated to V7.0
  □ DMGR migrated to V7.0 and one or more federated nodes successfully migrated to V7.0?

  Note: You can't fallback a federated node to a previous level if the DMGR thinks the node is at V7.0. It implies a fallback of the entire cell, DMGR plus all federated nodes.
**Post-migration validation**

Record here the post-migration tests you intend to perform to validate the successful migration of the cell:

<table>
<thead>
<tr>
<th>Validation Test</th>
<th>Succeed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>All the migrated V7.0 servers start successfully?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>V7.0 Administrative Console shows &quot;green arrow&quot; for status of Node Agents?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>V7.0 Administrative Console shows &quot;synchronized&quot; status for each node?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>V7.0 Administrative Console shows V7.0 as the version/maintenance for each node?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>Node Agent log shows successful synchronization?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>V7.0 Administrative Console shows &quot;green arrow&quot; for status of started applications?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>Key user applications start and work as designed?</td>
<td>□ Yes □ No</td>
</tr>
</tbody>
</table>

**User defined tests:**

- □ Yes □ No
- □ Yes □ No
- □ Yes □ No
- □ Yes □ No
- □ Yes □ No
## Document Change History

Check the date in the footer of the document for the version of the document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 17, 2008</td>
<td>Original document.</td>
</tr>
<tr>
<td>September 18, 2008</td>
<td>Updated with Techdoc number and republished.</td>
</tr>
<tr>
<td>September 26, 2008</td>
<td>Provided URL for public download site for the WCT. See page 81.</td>
</tr>
<tr>
<td>December 28, 2009</td>
<td>Several important changes based on experiences with customers migrating very large topologies:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The recommendation that the migration “temporary” space be 500 cylinders was changed to a recommendation it be 1.5 times the size of the node configuration file system being migrated.</td>
</tr>
<tr>
<td></td>
<td>- Added a reference to the WP101589 Techdoc, which provides information on a set of migration performance studies that were conducted. That document also has some excellent advice on avoiding migration problems and speeding up the process. <a href="ibm.com/support/techdocs/atmsastra.nsf/WebIndex/WP101589">ibm.com/support/techdocs/atmsastra.nsf/WebIndex/WP101589</a></td>
</tr>
<tr>
<td></td>
<td>- Created a section on running the migration steps using a 64-bit JVM. By default the JVMs launched for migration will be 31-bit. Those may well work for you. But for larger topologies where larger heaps are necessary, going to a 64-bit JVM may be required. See “Enabling 64-bit JVMs for the migration process” on page 70. Many thanks to Uli Veser of IBM Germany for his guidance on this process.</td>
</tr>
<tr>
<td>January 5, 2010</td>
<td>Added a section on the use of <code>_BPX_SHAREAS=NO</code> to overcome (or avoid) the potential problem of job failure due to multiple child processes being spawned in the same address space.</td>
</tr>
<tr>
<td>April 13, 2010</td>
<td>Added some clarification to the section on using 64-bit JVMs. DMGR nodes and Standalone server nodes are the most likely type to require 64-bit JVMs, and then only if the configuration being migrated is large enough to require it. Federated nodes generally do not require 64-bit JVMs. Also, more clearly spelled out that the manual updates to run with 64-bit JVMs imply running the <code>BBOW*PRO, BBOW*PRE</code> and <code>BBOW*POS</code> jobs in sequence rather than running the <code>BBOWMG3*</code> job.</td>
</tr>
<tr>
<td>February 10, 2011</td>
<td>Added a note about updating the V7 Daemon proc STEPLIB statements pointing to the V6.1 (or earlier) modules. This may be needed in a “mixed version” cell environment. This can be found on page 53.</td>
</tr>
<tr>
<td>March 3, 2011</td>
<td>Based on a problem report from a customer it was discovered that the “script compatibility” switch <strong>must</strong> be set to Y to allow a mixed-level cell to operate. If unchecked the DMGR and downlevel Node Agents are unable to communicate due to an underlying security mechanism that is different between V7 and lower levels. Having the “script compatibility” box checked “Y” will allow mixed-level support. See page 29 for the note on this.</td>
</tr>
<tr>
<td>March 22, 2011</td>
<td>Prior editions of document had the <code>_BPX_SHAREAS</code> variable listed without the leading underscore. The leading underscore is critically important as without it the intended effect is not realized. My sincere apologies to all affected by this oversight.</td>
</tr>
</tbody>
</table>

End of WP101329