WebSphere Application Server for z/OS

Migrating an ND Configuration from V5.1 to V6

A chronicle of an actual migration

This document can be found on the web at:
www.ibm.com/support/techdocs
Search for document number **WP100559** under the category of "White Papers"

Version Date: May 18, 2005
Please see "Document Change History" on page 76 for updates provided in this version of the document.

IBM Washington Systems Center

Don Bagwell
IBM Washington Systems Center
301-240-3016
dbagwell@us.ibm.com

Many thanks go to **John Hutchinson**, **Mike Cox** and **Bob Teichman** of the Washington Systems Center, **Mike Loos** of the WebSphere Technical Sales team, and **Lou Amodeo** from Poughkeepsie development.

Table of Contents

Initial Information and Disclaimers

About this document

This document is *not* intended to provide an exhaustive reference for all things related to the migration of nodes from V5.x to V6.0. The purpose of this document is to familiarize the reader with the process and provide enough insight so the process of migration can be approached confidently.

But there are things we don't go into here, such as how to migrate an IJP configuration to the new V6 architecture. Perhaps at some future date those topics will be included in this document.

The WebSphere Application Server for z/OS Version 6 InfoCenter has a section on migration and should be considered the definitive source for migration information.

How this document is constructed

The method this document employs to do this is a chronicling of the steps taken to migrate an actual configuration at the IBM Washington Systems Center. That configuration was a Network Deployment configuration at the V5.1 W510207 level of maintenance. It consisted of two application server nodes spanning two MVS images in a Sysplex.

This method of illustrating a migration is useful for two reasons:

1. It provides a way for you to see "real" steps performed against a "real" configuration. From that you can map what you read here to your environment and understand things better.
2. It provides a handy framework in which to provide a running commentary of things to be aware of and things to watch out for.

Fundamental assumption underlying this entire document

We assumed the WebSphere Application Server for z/OS Version 6 code was properly installed on the system, including all the steps required to allow this code to operate on the systems.

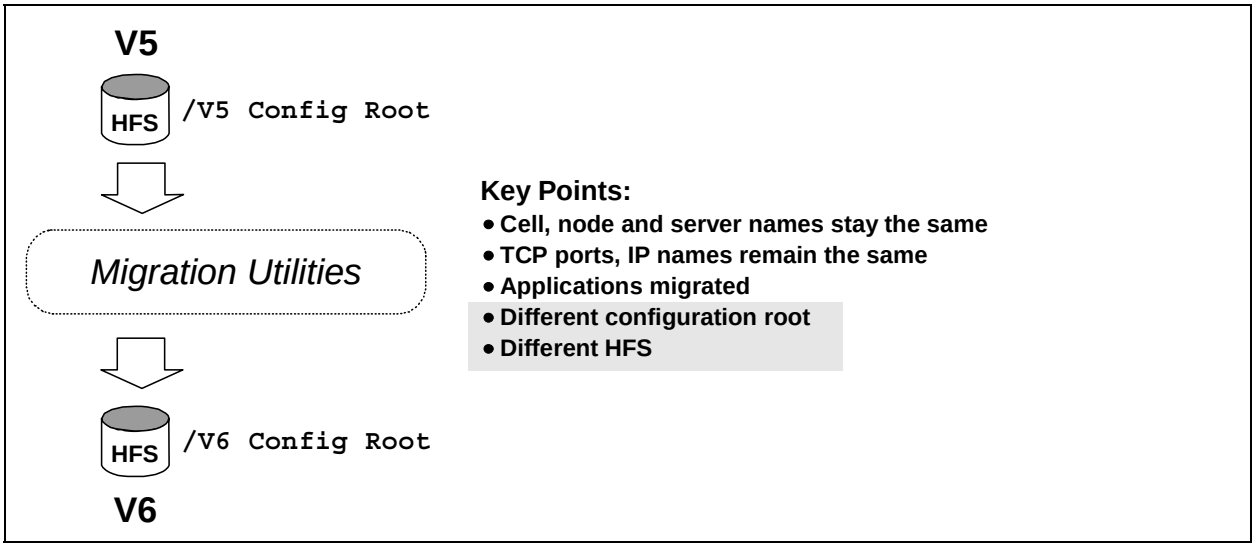
In other words, we assumed the newly migrated servers had what they needed to start, provided we had done the migration properly.

An Overview of the Version 6 Migration Process

The migration process provided to go from V5.x-to-V6 is much improved over the one provided to go from V5.0-to-V5.1. So if you're harboring painful welts from that experience, then cheer up: this will be relatively easy.

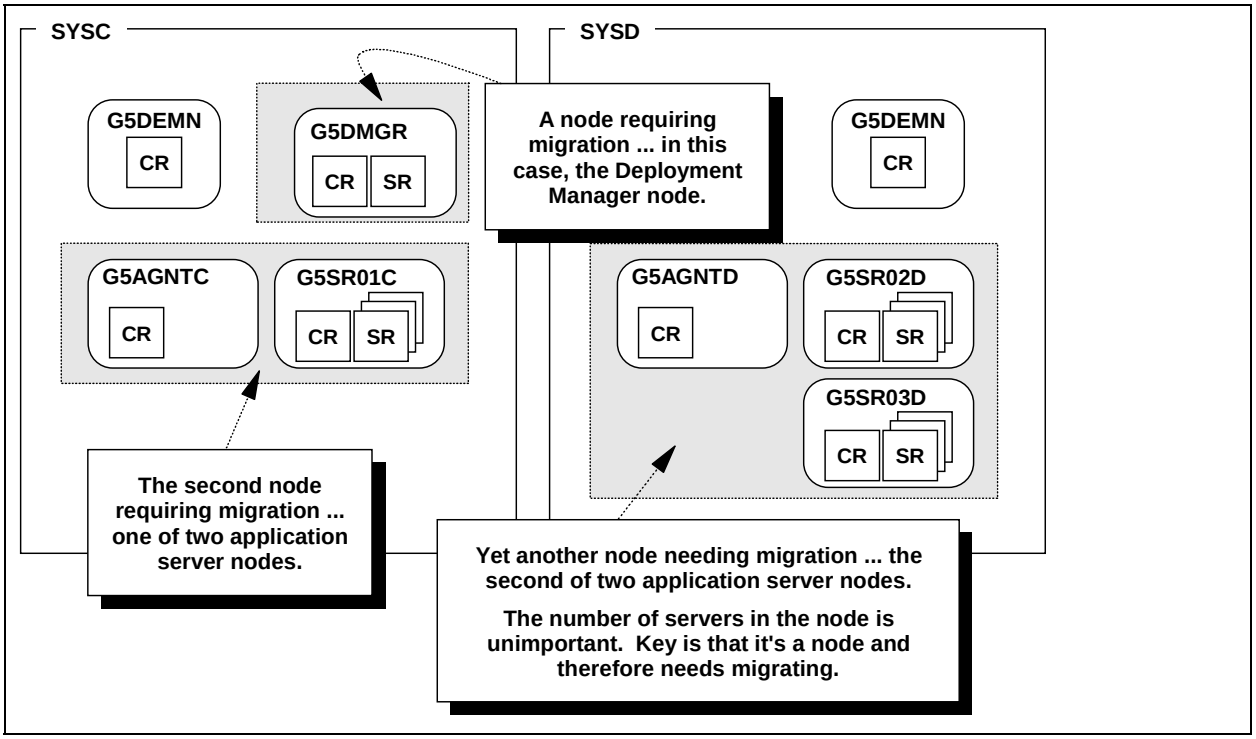
Essential concepts of migration

At the very highest level this is the same as it was for the V5.0-to-V5.1 migration; that is, an existing configuration in an HFS is copied out, transformed, and written into a new HFS:



High level view of what migration entails

Further, the migration process is a node-by-node procedure, just like it was before:



Migration utilities must be run against each node in your configuration, including Deployment Manager

Flow of migration process

Here's a snapshot of what the process is like to migrate a multi-node Network Deployment configuration:

1. *Take an inventory of your existing environment so you have a feel for things like mount points, userids and groups, node directory roots and JCL start procedures*

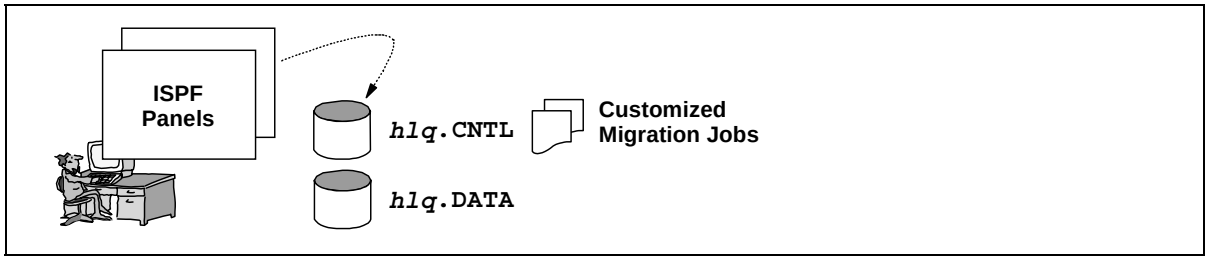
   The information shown under "The "G5CELL" Network Deployment Configuration" starting on page 15 gives you a good idea of what kind of information you'll need.

2. *Back up your "source" configuration HFS*

   The migration process alters that configuration HFS. It's a minor alteration, but one nevertheless. So "just to be sure" you should backup the source configuration HFS using your preferred file backup tool.

   Note: See "Falling back to V5 (or recovering a migrated node)" on page 73 for more on the changes made to the source configuration.

1. *Run through the ISPF customization dialogs that are provided with V6. They will generate customized migration jobs for the node*



ISPF panels capture key information and then generate customized migration jobs

We illustrate that panel-by-panel under each of the node sections in this document.

1. *Migrate the node by running the customized jobs*

   This involves submitted the jobs and checking for RC=0.

2. *Perform post-migration work*

   This involves creating a few additional RACF profiles.

Differences from V5.1 migration process

While many of the concepts are the same, some of the specifics of the V6 migration process are different. Let's cover them here:

One primary migration utility

In the V5.0-to-V5.1 migration process there were five migration utilities: `BBOXMIG1` through `BBOXMIG5`. With the V6 migration process, that's been reduced so that one job is responsible for doing the heavy-lifting of migration.

Note: But that's not to say that *only* one job is run. There are other jobs, but they do more mundane things like create and mount the new HFS, or copy new JCL procedures into PROCLIB. With V6 there's one primary migration job that does what used to take two or three.

ISPF Dialog creates customized migration jobs

The V5.0-to-V5.1 migration process came packaged as a series of JCL jobs in a PDS that you had to hand-modify. With V6 there is an option in the ISPF Customization Dialogs that

will create those migration jobs for you, with all the relevant information updated in all the right spots.

Simpler ISPF panels to define configuration being migrated

The V5.0-to-V5.1 migration process required you to run through the standard customization dialogs to build a "skeleton" configuration. That "skeleton" was used as input to the migration process. It was somewhat confusing because much of the information entered into the "skeleton" configuration was "dummy information" -- things not needed by the migration process but required by the ISPF dialogs.

With V6, the ISPF panels are much simpler, capturing only that information the migration utilities really require. So the process is far less confusing then it was for V5.0-to-V5.1.

Version 6 information asked for in ISPF panels

These utilities will convert a set of V5.x servers into V6.0 servers. The architecture of the V6 server is different from V5, most notably the "High Availability Manager" function. The ISPF panels that are used to generate the migration jobs will ask you for the host value assigned to this new function.

We'll highlight this during the illustration of the panels used to migrate the "G5CELL."

Additional V6 post-migration security work needed

V6 requires a few more security profiles than did V5. That means that after migrating a node, some additional RACF (or other SAF interface security product) work is required. Not much, but a little.

We'll highlight this at the appropriate spots in the illustration of the migration of the "G5CELL" used for this white paper.

New JCL start procedures created during migration job generation

The V6 migration process will ask for names to be used for the JCL start procedures for server controllers of the migrated node. It will then generate new JCL for you and copy that JCL into your PROCLIB. The implication is that new JCL procedures are required for V6, and indeed the new JCL procedures do look quite a bit different. But there's a subtlety here that we will illustrate in this document:

- Yes, new JCL start procedures are needed. At a minimum the `SET ROOT=` value that points to the configuration mount point must be different. If `STEPLIB` statements are used then those also need to be different from V5.
- But the *names* of the JCL start procedures do not need to be different. Keeping the same names as used with V5 provides a key benefit: you won't need to create new `STARTED` profiles. New controller start

What this means is that you'll need to backup your V5 start procedures before running the job that copies in the new JCL.

Again, we'll illustrate this at the appropriate spot in the document.

Mixed-version nodes within a cell on the same MVS image permitted (with limitations)

One of the nice features of V6 is that it permits a Deployment Manager running at the V6 level of code to manage nodes still at the V5 level of code. What's more, a V6 Daemon is capable of supporting -- *on the same MVS image* -- V5.1 servers in a node that's part of that Daemon's cell.

This provides considerable flexibility in the migration process. Back when V5.1 came out, the migration process there required that two or more nodes from a cell on the same MVS

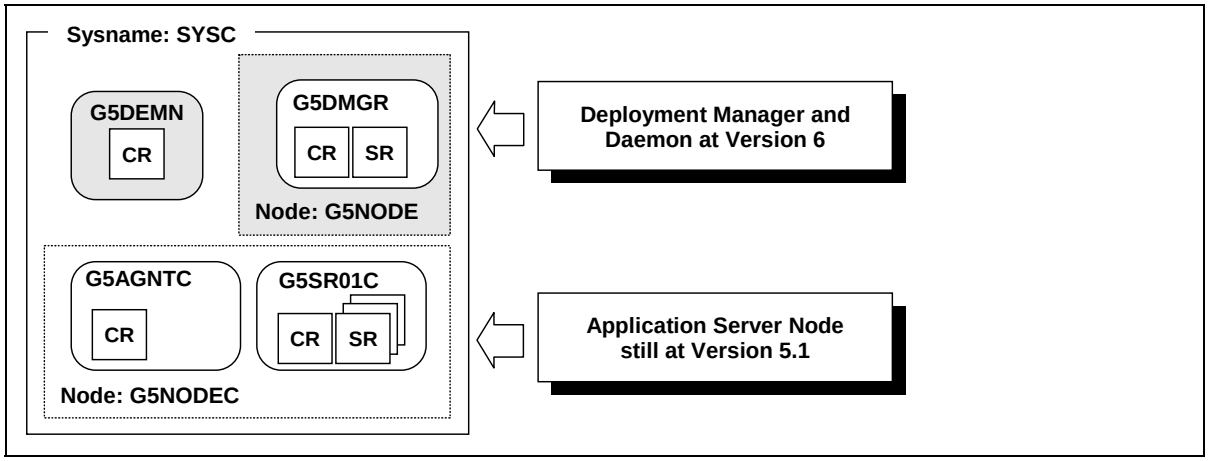
image be migrated "at the same time" (in other words, one immediately after the other). What this meant was that if you had an application server node on the same MVS image as its Deployment Manager, you had to stop both and migrate them both before you could restart either.

The issue was the Daemon server. Two or more nodes from the same cell on the same MVS image share the same Daemon server. If the Daemon server is running at the higher level of code, then the not-yet-migrated node (running at the lower level of code) would have to be compatible with the Daemon. In Version 5.1 this was *not* the case: a Daemon running at V5.1 was not compatible with servers in a node running at V5.0.

Notes: We're being somewhat precise with our language here. Let's lock down some things:

- If the two nodes on the same MVS image were in *different cells*, it would have been okay. That's because they would have been supported by *different Daemon servers*. The issue was when two nodes *from the same cell* on the same MVS image. When that was the scenario, they shared the same Daemon.
- If the two nodes were on different MVS images, that too would have been okay. Again, that would imply different Daemon servers. The white paper WP100441 clearly outlined how a Network Deployment configuration that spanned two or more MVS images could be non-disruptively migrated from V5.0-to-V5.1.

But with Version 6 you had additional flexibility. When migrating from V5.1-to-V6, the Daemon code *is* compatible with both V6 and V5.1, so a configuration like this is possible:



Two nodes from the same cell on the same MVS image at different version levels

But there are limitations:

- The V5 node *must* be at V5.1, and *not* V5.0.x
- If one of the nodes is the Deployment Manager node, it must be at the higher level

Message: If your V5 configuration is still at V5.0.x, then plan on migrating all nodes from a cell on the same MVS image immediately after one another. And do the Deployment Manager first, then the application server node because the DMGR needs to be up and running to migrate an application server node.

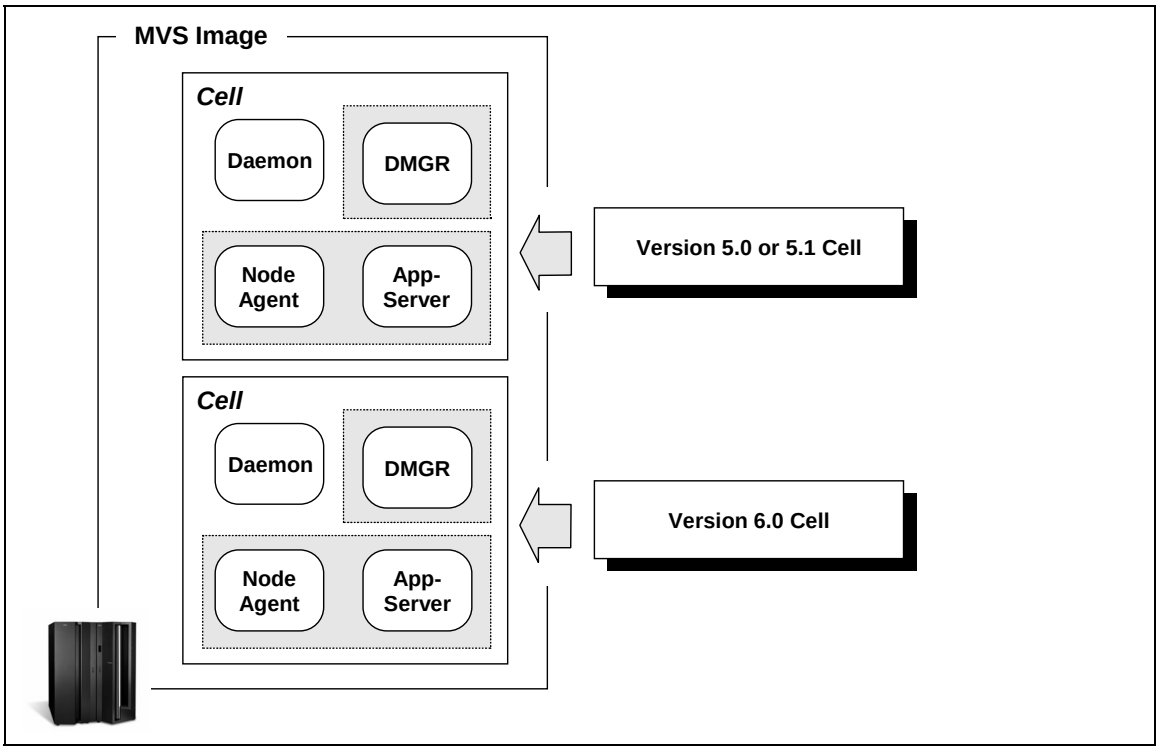
But if your nodes are at V5.1 (and W510207 at a minimum), then you have the flexibility to have a V6 node and V5.1 node from the same cell coexist on the same MVS image.

Other V6 "mixed node" information

We know this topic is likely to generate some interest and discussion. Let's try to get out front of that with a little information here.

Coexistence of separate V5 and V6 cells on same MVS image permitted

There's nothing about V6 that would prevent two different cells -- one at V5 and one at V6 -- from coexisting on the same MVS image (or the same Sysplex for that matter, but on the same MVS image is the more challenging test):



Version 5.x and Version 6.0 cells may coexist on the same MVS image

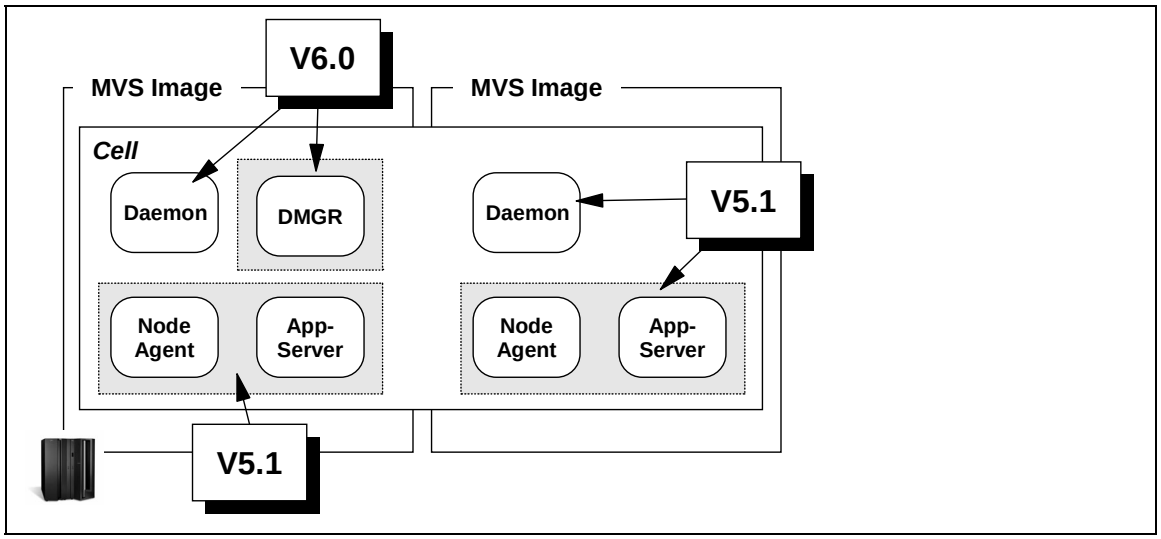
Notes: There are some limitations to this, all fairly common-sense things:

- Both V5 and V6 modules can't be in LPA/LNKLST at the same time
- You can't share JCL start procedures between the V5 cell and the V6 cell when STEPLIB statements are in the JCL

As long as you provide essential separation between the two -- separate mount point, separate HFS, separate JCL start procedures -- the two will happily coexist.

Coexistence of V6 with V5.1 nodes in the same cell or same MVS image permitted

As we stated earlier, a V5.1 node has an increased ability to coexist with V6.0 servers in the same cell or even in the same cell on the same MVS image:



Mixed V5.1/V6.0 cells possible -- same MVS image or other MVS image

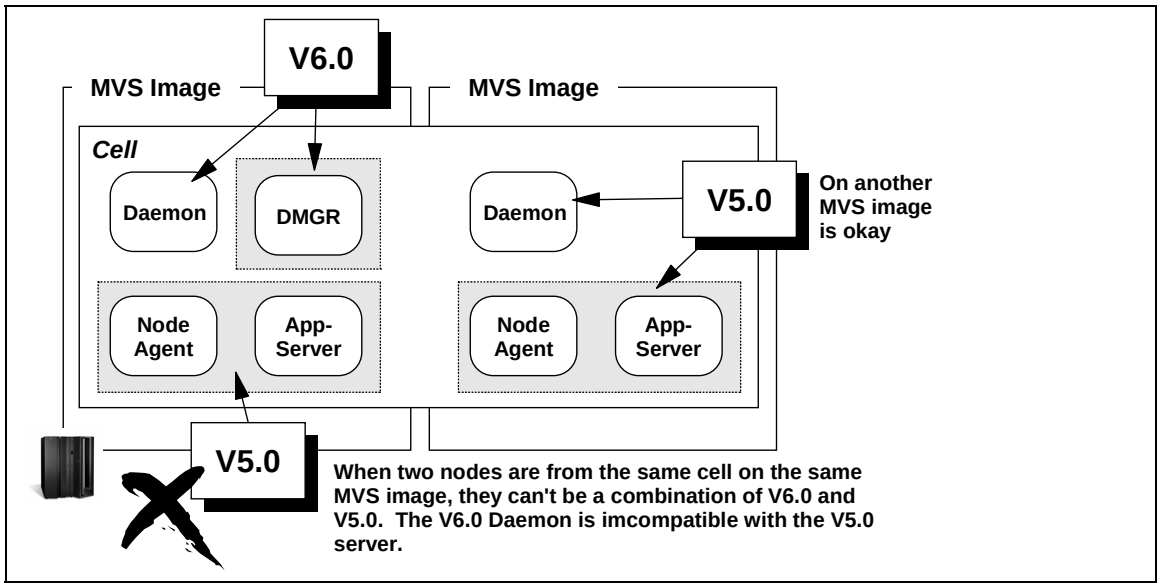
Notes: There are some limitations to this,:

- The Deployment Manager must be at V6.0 to manage V5.1 nodes. The reverse is not permitted -- a V5.1 DMGR can't manage a V6.0 node.
- Both V5 and V6 modules can't be in LPA/LNKLST on the same MVS image at the same time. At a minimum one must be STEPLIBed
- A V6.0 node can't share JCL start procedures with a V5.1 node when STEPLIB statements are used in the JCL.

In addition to this, there are limitations to what kind of management a V6.0 DMGR can perform upon a V5.1 node. See "Some limitations on what you can do with mixed nodes" on page 11 for more.

Coexistence of V6 with V5.0 nodes in the same cell on same MVS not permitted

Here we're a bit more limited. A V6.0 and a V5.0 node from the same cell can't coexist on the same MVS image. The Daemon server -- which must be at the level of code equal to the highest code-level node for that cell on that MVS image -- isn't compatible with a V5.0 node. (A V5.1 node yes, a V5.0 node no.)



Can't have a V5.0 node and a V6.0 node in the same cell on the same MVS image

A cell with mixed V6.0 nodes and V5.0 nodes is permitted, provided they're not mixed on the same MVS image. On another MVS image, where the V5.0 servers can be supported by a V5.0 Daemon, is okay.

Notes: There are some limitations to this,:

- The Deployment Manager must be at V6.0 to manage V5.0 nodes. The reverse is not permitted -- a V5.0 DMGR can't manage a V6.0 node.
- A V6.0 node can't share JCL start procedures with a V5.1 node when STEPLIB statements are used in the JCL.

In addition to this, there are limitations to what kind of management a V6.0 DMGR can perform upon a V5.1 node. See "Some limitations on what you can do with mixed nodes" on page 11 for more.

Some limitations on what you can do with mixed nodes

We've established that mixed node cells are possible. We've established that the Deployment Manager has to be at V6.0 to manage "down-level" nodes. Here a list of things you *can't* do in mixed-node cell environment:

- You can't federate a V5.x node into a cell managed by a V6.0 Deployment Manager. The federation process will detect the mismatch and prevent the federation.
- If you wanted to join a V5 "Base Application Server node" into the V6.0 cell, you would have to first migrate the "BaseApp node" (now called a "Standalone Server" in V6.0 language) to V6.0, and then federate it.
- You can't add servers to a managed down-level node. If you want to add servers to the node you would have to first migrate the node up to V6.0 and then add servers.

There may be more, but you get the point -- free and unfettered management of a down-level node is restricted.

That said, some common things *can* be done:

- You can install applications
- You can change settings like short names and ports
- You can start and stop the server from the Admin Console
- You can start and stop applications from the Admin Console

Questions and Answers

Does it matter if I'm at V5.0 or V5.1?

At a high level, no. The process is essentially the same for both. But when it comes to running nodes at different levels, it does matter. See "Mixed-version nodes within a cell on the same MVS image permitted (with limitations)" on page 7 for more.

Is there a minimum level of maintenance the V5 nodes should be at?

Yes:

Version 5.0.x node W502025

Version 5.1.x node W510207

If your nodes are not at the appropriate level of maintenance, apply the maintenance and make sure at least one server from the node runs applyPTF.sh so the node's configuration is brought up to the minimum maintenance required for migration.

Can a V4 configuration be migrated to V6?

No.

If my cell is at V5.0, should I migrate to V5.1 before migrating to V6?

The advantage to being at V5.1 during migration is an additional degree of flexibility with regard to how the nodes are migrated. It centers around the Daemon servers. A Daemon migrated up to the V6 level of code is capable of managing a V5.1 node from the same cell on the same MVS image. But not a V5.0 node. Therefore, if you have two nodes from a cell on the same MVS image and they're at V5.0, they'll need to be migrated one right after the other (or "at the same time," which means the second one migrated immediately after the first). If the two nodes from a cell on the same MVS image are at V5.1, then the second one can be migrated at your leisure -- the V5.1 node can be started and will happily coexist with a V6.0 Daemon.

That said, migrating from V5.0 to V5.1 is not a trivial undertaking. Migrating from V5.0 to V6 directly is probably the best way to go. But that's something you must decide.

Is there a proper sequence for migrating the nodes of a cell?

Yes. Always migrate the Deployment Manager node first. Version 6 is capable of managing "down" to V5 nodes, but a V5 DMGR can't manage "up."

After the Deployment Manager node is migrated, other nodes may be migrated as you please.

> **Note:** See "Mixed-version nodes within a cell on the same MVS image permitted (with limitations)" on page 7 for a discussion of a key restriction to this.

What about a "Base Application Server" node?

The process is very similar to that of a Network Deployment node. See "Migrated a Base Application Server Node" starting on page 62.

Do the servers in the node have to be down when it's migrated?

Yes. In order to migrate a configuration the servers in the node must be stopped.

Do all the servers in my cell have to be stopped when migrating?

No. In fact when application server nodes are migrated the Deployment Manager must be up and running.

It is quite possible to have some nodes in a cell migrated to V6 while other nodes are still at V5. This provides the ability to provide a "non-disruptive" migration.

> **Note:** See "Mixed-version nodes within a cell on the same MVS image permitted (with limitations)" on page 7 for a discussion of a key restriction to this.

Further, a V6 Deployment Manager is capable of existing with V5.1 nodes for quite some time, so there is no need to feel as if you have to rush a migration to fit within a maintenance window. The V6 DMGR will know that a node it is managing is at the lower level of code, and when configuration changes are made to the node the V6 DMGR will make sure those changes have the V5 format.

Is it possible to migrate a server at a time?

No. The migration process is a node-by-node process.

Is it possible to migrate a configuration when security is enabled?

Yes. The migration ISPF customization panels will ask for the WebSphere Admin ID and password when an application server node is being migrated. This is needed so the migration utility can connect to the running V6 Deployment Manager and synchronize.

Re-use my existing V5 security profiles, or create new?

By "security profiles" we mean things like (in the language of RACF) -- userids and groups, `STARTED` profiles, `SERVER` and `CBIND` profiles, keyrings and certificates.

The answer is to re-use the same profiles. The relationship between the configuration and the underlying security profiles is so tight that trying to map the migrated configuration to a new set of profiles would be extremely challenging. It is far better to use the same profiles.

Perform "post-migration security work" even if global security is not enabled?

Yes. The "post-migration security work" mentioned earlier involves creating a few new profiles needed by a V6 server, regardless of whether global security is enabled or disabled.

Do all servers in a node get migrated when node is migrated?

Yes. The migration utility is capable of determining what servers reside in a node and it will migrate all the servers in the node.

Are the applications in the servers migrated as well?

Yes. The migration utility will make certain the applications installed in the servers are *copied* over.

> **Note:** WebSphere Application Server for z/OS Version 6 has many new application-oriented features, including support for J2EE 1.4. You applications will still run in the new V6 environment, but they will not take advantage of some of these new functions. The migration utilities will not modify the applications to exploit anything new in V6.

What about TCP ports?

The migration will carry over the ports you had assigned to your servers in V5. There is an aspect of this you should know about: a V6 *application server* has six new ports -- above and beyond the V5 number of six -- making the total number of ports per application server now 12. The migration process will carry over the six V5 ports, *and will assign default values for the six new ports*.

You'll probably want to re-map these new ports so they adhere to your port allocation scheme. We cover this issue under "New ports created for V6 application server" on page 47.

May different people perform migration on different nodes at the same time?

It would be best not to attempt this. The `BBOWMG3*` job will try to use the `/tmp/migrate` directory for work space, and it'll attempt to copy the file `bbomigrt2.sh` into the directory. So, depending on whether that file already exists, and what the permissions are on them, a migrate job will either operate or fail. Failure will be indicated by a `RC=256` on the first step ("SETUP") of the `BBOWMG3*` job.

In theory it could work. But it would be safest to assume the migrations be run sequentially and not concurrently.

Need to do 'PRR' processing like we did in V5.0-to-V5.1 migration?

If you have XA connectors installed in your application servers, then the answer is yes. If you do not have XA connectors installed, then the answer is no.

When migrating an application server node, two migration utilities are generated: `BBOWMG1F` and `BBOWMG2F`. They are what perform the PRR ("Peer Resource Recovery") processing for the application servers in a node.

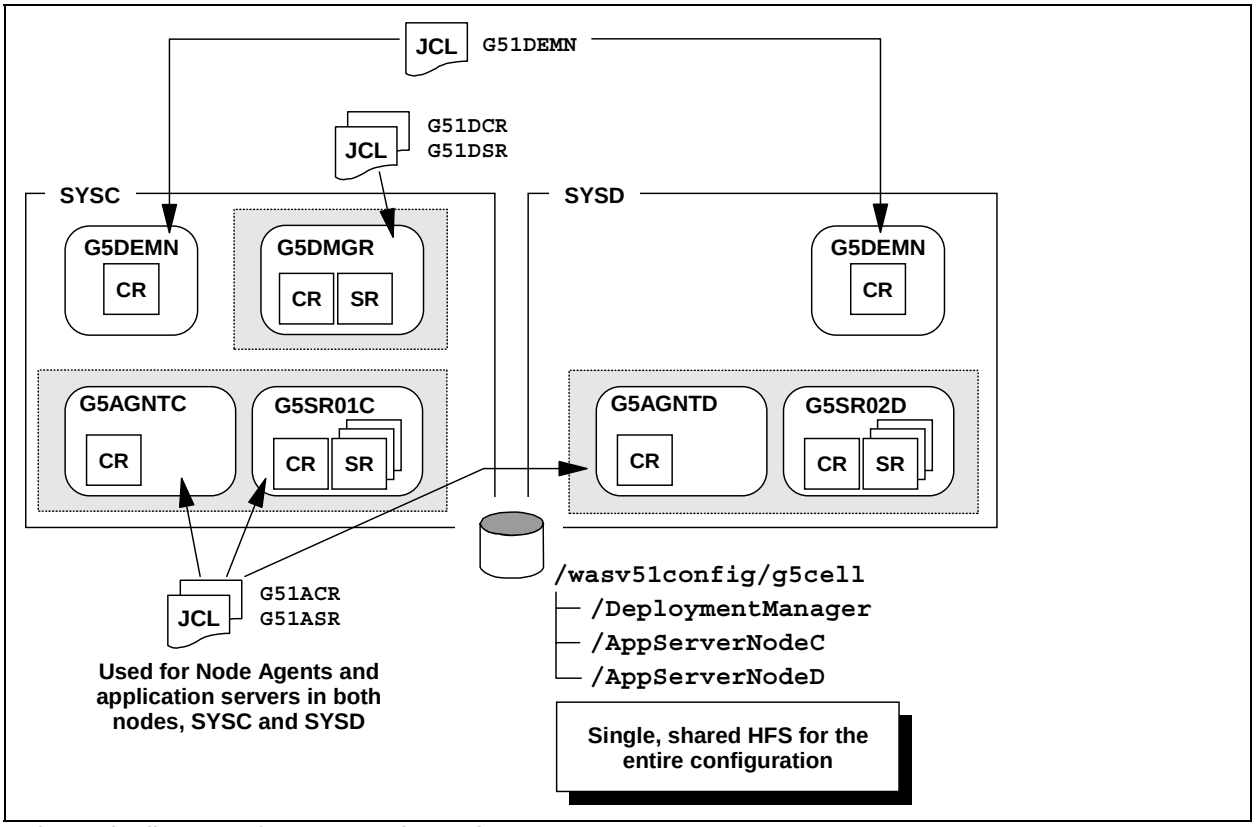
Running those jobs even if you don't have XA connectors installed won't hurt anything. So if you're not sure, then run the jobs.

The "G5CELL" Network Deployment Configuration

This document was written based on the migration of an actual Network Deployment configuration cell at the Washington Systems Center.

Diagram of the cell layout

The cell looked like this:



Schematic diagram of G5CELL migrated to V6

Description of configuration

The G5CELL was a Network Deployment configuration, consisting of a Deployment Manager node on SYSC and two federated application server nodes, one on SYSC and one on SYSD.

The entire configuration was kept in a single, shared HFS.

Common JCL was used. Both Daemon servers used the `G51DEMN` start procedure. The Deployment Manager had its own procs, `G51DCR` and `G51DSR`. All servers in the application server nodes used the same procedures -- `G51ACR` and `G51ASR`. The use of a shared HFS permitted this.

Note: Though, as you'll see, this came back to haunt us. The use of common JCL has the potential to restrict the flexibility of a cell migration. See "The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB" on page 32 for more on this.

Cell information

Cell short name	G5CELL
Version of code	V5.1, maintenance W510207
HFS mount point	/wasv51config/g5cell
HFS data set name	OMVS.WAS51.G5CELL.CONFIG.HFS
Security Enabled	Yes
WebSphere Admin ID	G5ADMIN
WebSphere Admin ID password	G5ADMIN
WebSphere Configuration Group ID	G5CFG
Default RACF keyring name	WASKeyring

Deployment Manager Node

Configuration Root	/wasv51config/g5cell/DeploymentManager
Node short name	G5NODE
DMGR short name	G5DMGR
DMGR JOBNAME	G5DMGR
DMGR controller ID	G5DCRU
DMGR servant ID	G5DSRU
DMGR controller JCL procedure	G51DCR
DMGR servant JCL procedure	G51DSR
DMGR ENV= string	ENV=G5CELL.G5NODE.G5DMGR
Daemon controller ID	G5DEMN
Daemon controller JCL procedure	G5DEMN

Application server node on SYSC

Configuration Root	/wasv51config/g5cell/AppServerNodeC
Node short name	G5NODEC
Node Agent short name	G5AGNTC
Node Agent JOBNAME	G5AGNTC
Application server short name	G5SR01C
Application server JOBNAME	G5SR01C
Common controller ID	G5ACRU
Common servant ID	G5ASRU
Common controller JCL procedure	G51ACR
Common servant JCL procedure	G51ASR
Node Agent ENV= string	ENV=G5CELL.G5NODEC.G5AGNTC
G5SR01C ENV= string	ENV=G5CELL.G5NODEC.G5SR01C
Daemon controller ID	(uses same Daemon as DMGR)

Information continued on next page ...

Application server node on SYSD

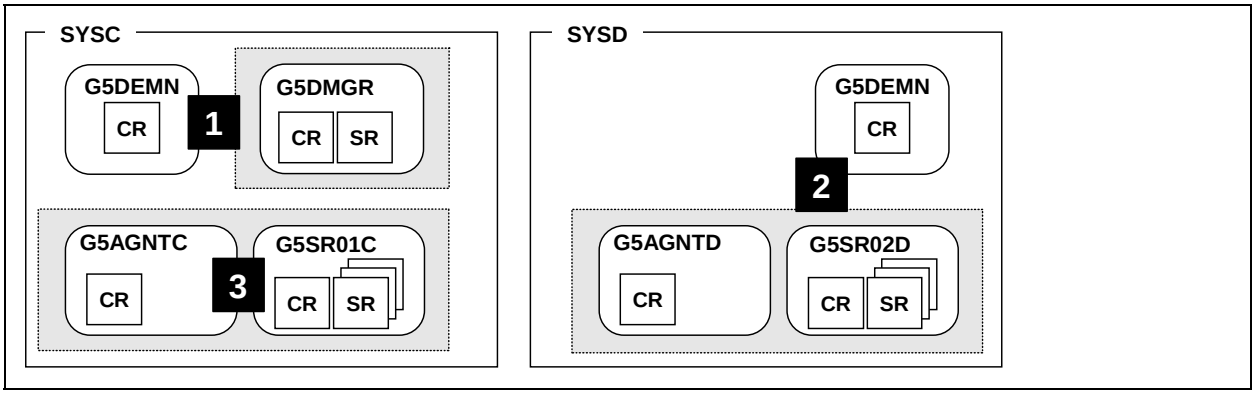
Configuration Root	`/wasv51config/g5cell/AppServerNodeD`
Node short name	`G5NODED`
Node Agent short name	`G5AGNTD`
Node Agent JOBNAME	`G5AGNTD`
Application server short name	`G5SR02D`
Application server JOBNAME	`G5SR02D`
Common controller ID	`G5ACRU` (*same as node on SYSC*)
Common servant ID	`G5ASRU` (*same as node on SYSC*)
Common controller JCL procedure	`G51ACR` (*same as node on SYSC*)
Common servant JCL procedure	`G51ASR` (*same as node on SYSC*)
Node Agent `ENV=` string	`ENV=G5CELL.G5NODED.G5AGNTD`
`G5SR01C ENV=` string	`ENV=G5CELL.G5NODED.G5SR01D`
Daemon controller ID	`G5DEMN`
Daemon controller JCL procedure	`G5DEMN`

Plan for migrated configuration

HFS mount point	`/wasv6config/g5cell`
HFS data set name	`OMVS.WAS6.G5CELL.MIGRATED.HFS`

Planned sequence of node migration

We set out with the plan to migrate the nodes in this order:



Sequence of planned node migration

The rationale for this was as follows:

- The Deployment Manager node had to be migrated first. When a cell has mixed nodes, the Deployment Manager must always be "up level" from the other nodes. The Daemon server on SYSC would be migrated to V6.0 at the same time the Deployment Manager node was.
- SYSD was planned for migration next. Truth was, either SYSC or SYSD could have been migrated next. We chose to migration SYSD next so we could illustrate how a V6.0 node (the Deployment Manager) and a V5.1 node (application server node on SYSC) could coexist in the same cell on the same MVS image.

Migrated the Deployment Manager Node on SYSC

What follows is a step-by-step account of how the Deployment Manager node was migrated.

Preliminary work

- ☐ Took inventory of key information (see "The "G5CELL" Network Deployment Configuration" starting on page 15)
- ☐ Backed up "source" configuration HFS

  In this case we used the following job:

```
//COPYHFS   JOB (ACCTNO,ROOM),'BAGWELL',CLASS=A,REGION=0M
//*
//* DFDSS TO COPY HFS DATASETS
//*
//COPY  EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//HFSOUT   DD UNIT=3390,DISP=OLD,VOL=SER=SMSA09
//SYSIN    DD *
 COPY DATASET(INCLUDE( -
     OMVS.WAS51.G5CELL.CONFIG.HFS -
           )) -
      RENAMEU( -
        OMVS.WAS51.G5CELL.CONFIG.HFS,OMVS.WAS51.G5CELL.MIGCOPY.HFS -
           ) -
      OUTDD(HFSOUT) -
      REPLACEU -
      CATALOG
/*
//
```

Configuration HFS

Backup Copy

Job used to back up G5CELL configuration HFS

- ☐ Backed up JCL start procedures used by Deployment Manager node

  The Deployment Manager node had six JCL members to back up:

```
G51DCR
G51DCRZ
G51DSR
G51DSRZ
G51DEMN
G51DEMNZ
```

  Again, the reason why we backed up the JCL is because we intended to use the *same names* for the newly created V6 JCL start procedures. This reduced the amount of post-migration work required.

- ☐ Insured G5CELL's "Admin ID" could work in the `/tmp` directory

  In our case there was no `/tmp/migrate` directory, so that meant the path was clear for us to invoke our migration job. However, if it exists and is owned by some other ID, you have two options:

  1. Delete the `/tmp/migrate` directory, *or*
  2. Set the permissions of `/tmp/migrate` to `777` and delete the `bbomigrt2.sh` file in that directory

  Note: The bottom line is this: the `BBOWMG3*` job will attempt to copy a the file `bbomigrt2.sh` into the `/tmp/migrate` directory and create another directory there. It needs write access to the `/tmp/migrate` directory and write access to the `bbomigrt2.sh` file.

Invoked ISPF dialogs and customized migration jobs

- Invoked V6's ISPF customization dialogs:

  ```
  ex 'WAS600.WAS.SBBOCLIB(BBOWSTRT)' 'APPL(MIG)'
  ```

  The `'APPL(MIG)'` provided a clean set of ISPF variables for this migration run.

- Selected Option 4 from the primary option menu:

```
1  Configure a security domain.

2  Create stand-alone Application Server nodes.  You must complete
   Option 1 before starting this option.

3  Create Network Deployment cells and nodes.  You must complete
   Option 1 before starting this option.

4  Migrate V5.x Nodes to V6 Nodes.
```

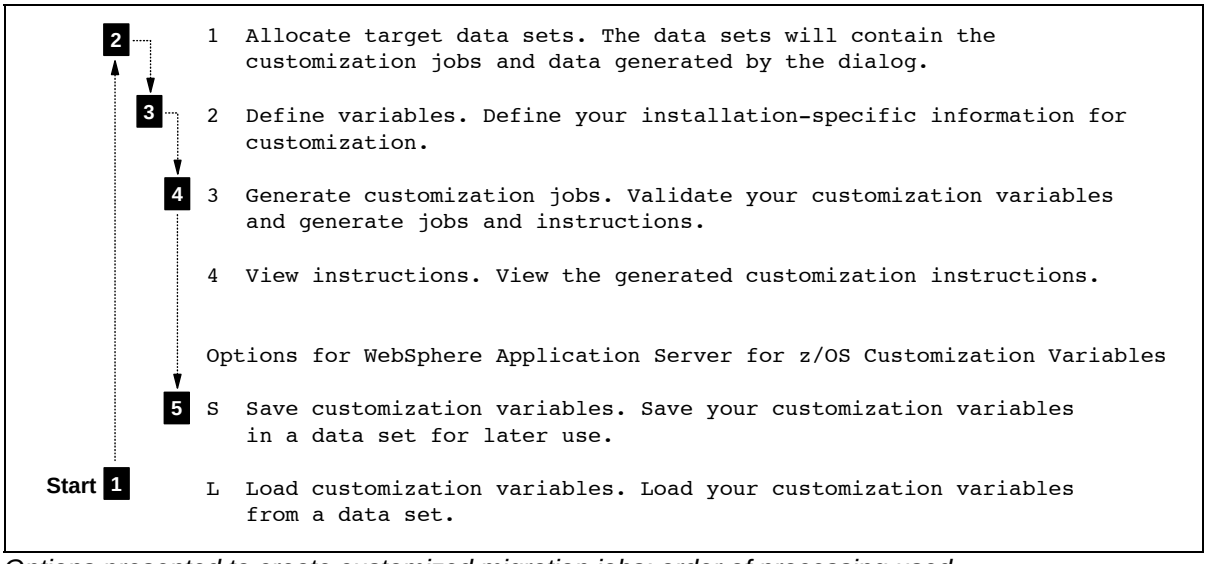
Option 4 -- used to invoke migration customization dialogs

- Selected Option 2, "Migrate a V5.x deployment manager to V6":

```
1  Migrate a V5.x stand-alone application server node to V6.

2  Migrate a V5.x deployment manager to V6.

3  Migrate a V5.x federated node to V6.  You must migrate the cell's
   deployment manager to V6 before migrating any of the cell's
   federated nodes
```

Option 2 -- migrate a Deployment Manager node

- Was presented with the following menu. The processing through the menu took the following path:



Options presented to create customized migration jobs; order of processing used

- Loaded the default customization variables by selecting Option "L":

```
Load Customization Variables

 Specify the name of a data set containing the customization variables,
 then press Enter to continue.

 IBM-supplied defaults are in  'WAS600.WAS.SBBOEXEC(BBOWVARS)'


 Data set name: 'WAS600.WAS.SBBOEXEC(BBOWVARS)'
```

Default customization variables loaded

Note: For the application server nodes `G5NODEC` and `G5NODED`, we'll use a copy of the saved variables from *this* customization run. That'll spare us from entering some of the same information several times. For the first node we loaded the default variables.

- Allocated the target data sets where the customized jobs were stored:

```
Allocate Target Data Sets

Specify a high level qualifier (HLQ) and press Enter to allocate the
data sets to contain the generated jobs and instructions. You can
specify multiple qualifiers (up to 39 characters).

High level qualifier: G5CELL.WP.DMGR                          .CNTL
                                                              .DATA

The dialog will display data set allocation panels. You can make
changes to the default allocations, however you should not change
the DCB characteristics of the data sets.

   .CNTL  - a PDS with fixed block 80-byte records to
            contain customization jobs.

   .DATA  - a PDS with variable length data to contain
            other data produced by the customization dialog.
```

Allocated target data sets

We took the default allocation parameters.

Note: This resulted in the creation of two data sets: `G5CELL.WP.DMGR.CNTL` (a FB 80 PDS where the JCL was stored), and `G5CELL.WP.DMGR.DATA` (a VB 255 PDS where scripts were stored). The "WP" qualifier stood for "White Paper."

- We selected Option 2, "Define Variables."
- We now faced three steps in the customization of the migration jobs:

```
1 - System Locations (directories, HLQs, etc.)
2 - System Environment Customization
3 - Server Customization
```

We worked through them in order: 1, then 2 and finally 3.

☐ We populated the variables for "System Locations 1 of 2":

```
System Locations (1 of 2)

   Specify the following V6 information, then press ENTER to continue.

   For some data sets, specify "Y" if they are in STEPLIB.

 Full Names of Data Sets

   PROCLIB.:  SYS1.PROCLIB

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  WAS600.WAS.SBBOLPA
   SBBOLOAD:  WAS600.WAS.SBBOLOAD
   SBBOLD2.:  WAS600.WAS.SBBOLD2

                                                      Use STEPLIB?
   SCEERUN.:  SYS1.LEMVS.SCEERUN                            Y
   SCEERUN2:  SYS1.LEMVS.SCEERUN2                           Y
   SGSKLOAD:  SYS1.CRYPTO.SGSKLOAD                          Y
              (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
```

Variables supplied for "System Locations 1 of 2"

Note: Use or not use `STEPLIB` according to your local preference.

☐ Next we populated the variables for "System Locations 2 of 2":

```
System Locations (2 of 2)

   Specify the following, then press Enter to continue.

 V6 WebSphere Application Server product directory:
     /u/bagwell/g6inter
```

Variables supplied for "System Locations 2 of 2"

This requires a little explanation. What this panel is asking for is the directory mount point of the V6 SMP/E HFS; in other words, where `WAS600.WAS.SBBOHFS` was mounted. The actual location was `/shared/zWebSphere/V6R0`. We could have entered that value.

However, we entered what you see in the picture above. The value of `/u/bagwell/g6inter` represents an "intermediate symbolic link" we created that pointed to the real V6 product directory.

By pointing to an "intermediate symbolic link" here, the configuration symlinks created during the migration all pointed to this intermediate symbolic link, which then resolved to the *actual* mount point. Why do this? Because it provides the flexibility to isolate nodes for purposes like test, production and maintenance.

Note: See the WP100396 white paper on `www.ibm.com/support/techdocs` for more on the use of "intermediate symbolic links" and their role in providing flexibility.

- ☐ The "System Environment Customization" variables were provided next:

```
System Environment Customization

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /wasv6config/g5cell
   Name...........:  OMVS.WAS6.G5CELL.MIGRATED.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  200
   Secondary allocation in cylinders.:  50
```

"System Environment Customization" variables: the HFS file system information

This information was used to create the BBOWMDMT job, which is designed to create and mount the HFS in which the migrated configuration will reside. The primary and secondary allocation values we used were lower than the default. We have lots of configurations at the WSC and we're trying to save a little space where we can.

You may decide to manually allocate the HFS and mount it yourself. That's okay. Running the BBOWMDMT is not a strict requirement for migrating the node. Having an HFS present is what's important.

Note: It is always necessary to provide the correct "Mount Point" value on this panel, even if you don't plan on running the BBOWMDMT job. This is an important variable used throughout the migration jobs.

- ☐ The "Server Customization 1 of 2" variables were next:

```
Server Customization (1 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.

   V5.x WebSphere Application Server home directory:
      /wasv51config/g5cell
         / DeploymentManager  1

   V6 WebSphere Application Server home directory:
      /wasv6config/g5cell
         / DeploymentManager   2

   Migration Options

     Enable z/OS Migration Tracing:  N
     Enable WASProfile Tracing....:  N
     Enable WASPreUpgrade Tracing.:  N
     Enable WASPostUpgrade Tracing:  N

     Default Backup Directory:  /tmp/migrate/5843/dmgr_backup
     User Specified Backup Directory:
```

"Server Customization" variables: source and target node directories

Notes:

1. This was the mount point and node directory for the V5 Deployment Manager. This value was typed in manually.

2. This was the intended node directory for the migrated V6 Deployment Manager. Notice how the mount point (`/wasv6config/g5cell`) was not open for update. That value was the HFS mount point provided on the previous panel. We chose to maintain the same value -- "DeploymentManager."

☐ The "Server Customization 2 of 2" variables were next:

```
Server Customization (2 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.


   High Availability Manager Host:  9.82.24.71   1
     The High Availability Manager Host MUST resolve to a single
     IP address.  It can not be a multihomed host.

   Daemon Procedure name........:  G51DEMN

   Controller Procedure name....:  G51DCR    2

   Servant Procedure name.......:  G51DSR
```

"Server Customization" variables: High Availability Manager and JCL procedure names

Notes:

1. The "High Availability Manager Host" is a new function of V6 servers, which is why the migration panels are asking for information. The field on the panel will accept either a host name (`www.name.com`) or an IP address (`9.82.24.71`). As you can see on the panel, if you enter a host name, it must resolve to a single IP address.

   For the SYSC system on which this Deployment Manager node resided, *all our hosts were multi-homed*. So we were forced to entered the IP address of SYSC MVS image.

2. New V6 JCL procedures will be created and placed in the *`hlq`*`.CNTL` target data set. The `BBOMDCP` member contains a job that will copy them to your specified PROCLIB (specified on the "System Locations 1 of 2" panel) and rename them to whatever values you supply here.

   We had two choices:

   - Provide new names. That would then require creating new `STARTED` profiles for the controllers.

     Note: Just for the controllers, not the servant procs. It's a simple command. The instruction member `BBOMDINS` has an example of the command required.

   - Keep the same names as used with our V5 configuration. That would eliminate the need for new `STARTED` profiles, but require us to back up the V5 procs since this job would overwrite them.

   We opted for the second choice.

   Note: But this decision came at a cost. See "The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB" on page 32.

- With the variables defined, we worked our way back to the main panel and selected Option 3 to "Generate customization jobs." The panel we received looked like this:

```
Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, ple

 Jobs and data files will get g
   'G5CELL.WP.DMGR.CNTL'
   'G5CELL.WP.DMGR.DATA'
 If you wish to generate custom                    ets, then
 exit from this panel and selec                    s" option.

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the job name for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname JOB (ACCTNO,ROOM),'BAGWELL',CLASS=A,REGION=0M
 //*
 //*
 //*
```

These data set values will be filled in automatically based on what you provided for the "Target Data Sets" earlier. But you should always check this to make sure it is truly where you want the jobs to be placed.

Generating the customization jobs

- When we hit the Enter key the jobs were written out to the target data sets:

```
Processing for data set 'G5CELL.WP.DMGR.CNTL' ...
Member BBOWMG3D successfully created.
Member BBOWMDMT successfully created.
Member BBOMDCR successfully created.
Member BBOMDCRZ successfully created.
Member BBOMDDN successfully created.
Member BBOMDDNZ successfully created.
Member BBOMDSR successfully created.
Member BBOMDSRZ successfully created.
Member BBOMDCP successfully created.
Member BBOMDINS successfully created.

Processing for data set 'G5CELL.WP.DMGR.DATA' ...
Member BBOWBMPT successfully created.
Member BBOWMDRF successfully created.
***
```

Customized jobs written out to the target data sets

☐ Then we selected Option S and saved the customization variables:

```
Save Customization Variables

  Specify the name of a sequential data set to contain the
  customization variables, then press Enter to continue. If the
  data set does not exist, the dialog displays the Allocate New
  Data Set panel, with which you can allocate a data set.


 Data set name: 'G5CELL.WP.DMGR.SAVECFG'
```

Customized jobs written out to the target data sets

We took the default allocation parameters when they were presented to us.

Note: When coming up with names for your data sets, keep a few things in mind:
- Have a separate set of migration data sets for each node being migrated
- The high-level qualifier should indicate in some fashion the node being migrated
- The high-level qualifier for the CNTL, DATA and SAVECFG data sets for a node should be the same

☐ We exited the customization dialogs and did a data set list on our high-level qualifier:

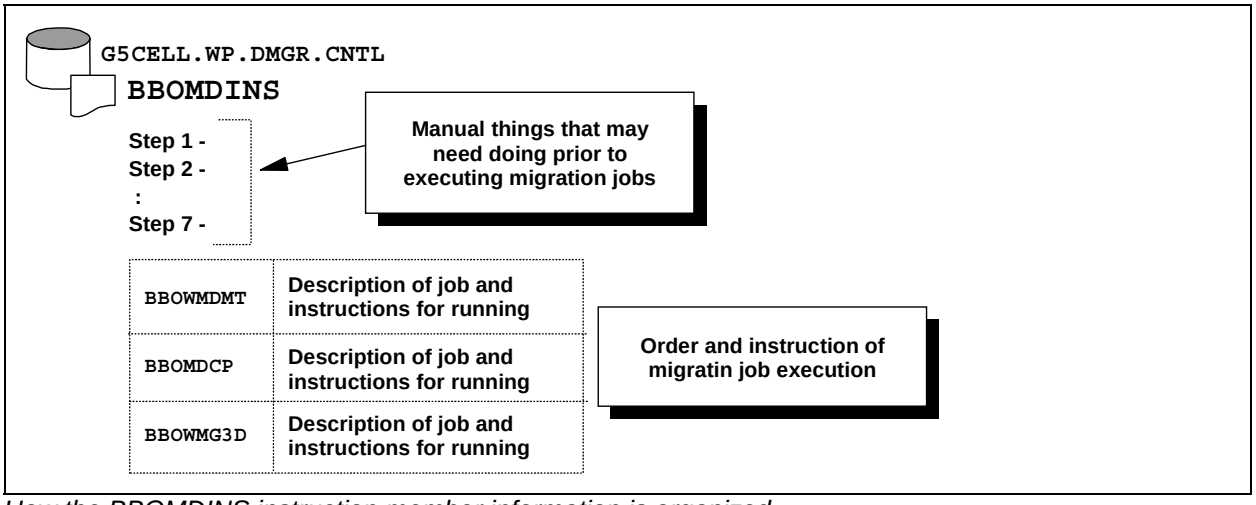
```
DSLIST - Data Sets Matching G5CELL.WP.**
Command ===>

Command - Enter "/" to select action
----------------------------------------
         G5CELL.WP.DMGR.CNTL
         G5CELL.WP.DMGR.DATA
         G5CELL.WP.DMGR.SAVECFG
```

Customized data sets for the migration of the Deployment Manager node

CNTL -- contained the JCL jobs, the new start procedures and an instruction member
DATA -- contained two members with customized scripts
SAVECFG -- contained the variables we supplied during this customization run

Reviewed instruction member BBOMDINS in CNTL data set

The BBOMDINS member of the CNTL data set contains a list of instructions for migrating the node. The instructions are organized in the following manner:



How the BBOMDINS instruction member information is organized

Important: You should review the contents of that member. Do not rely exclusively on this document. In particular, those instructions speak of what needs to be done to your MVS system so WebSphere Application Server for z/OS Version 6 may operate. (Things like APF authorization and HFS auto-mount instructions.)

Important note -- please read

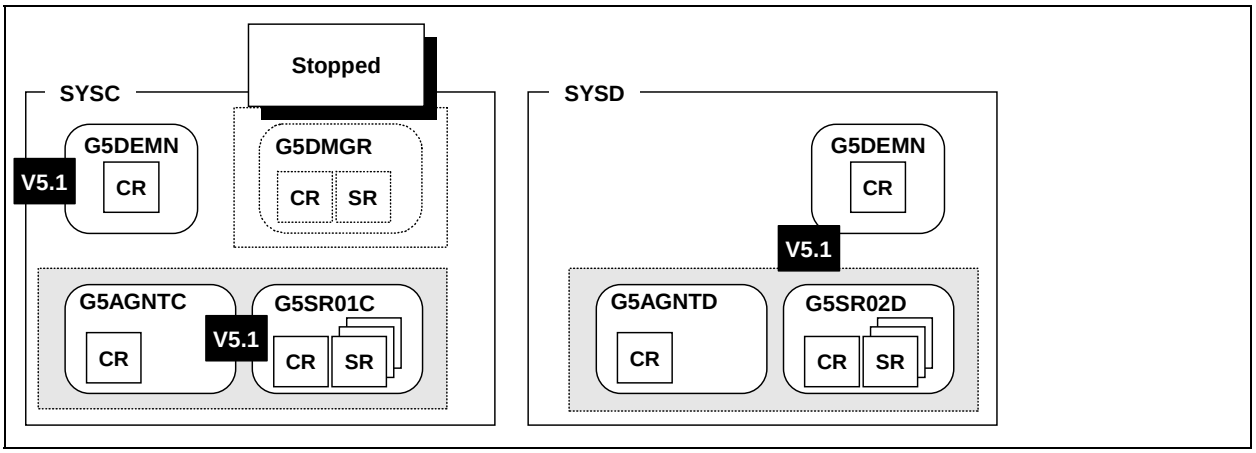
You may run through the ISPF customization dialogs on any MVS system in the Sysplex.

But the migration jobs ***must*** be run on the same MVS image on which the servers of the migrated node will run.

For our Deployment Manager node, that meant the jobs were run on the SYSC system.

Stopped Deployment Manager

The servers in a node being migrated must be stopped while the node is being migrated.



Deployment Manager stopped; Daemon and other servers left started

☐ Issued stop command against Deployment Manager controller:

```
/P G5DMGR
```

Note: This stopped only the Deployment Manager but not its Daemon. Had we stopped the Daemon, the servers in the `G5NODEC` node would have come down as well. To maximize the up-time of those servers, we left the Daemon up for now. Before we re-started the Deployment Manager at V6 we had to stop and restart the Daemon, which resulted in the Node Agent and `G5SR01C` server coming down.

Ran customized jobs

For the migration of a Deployment Manager node, three customization jobs are generated:

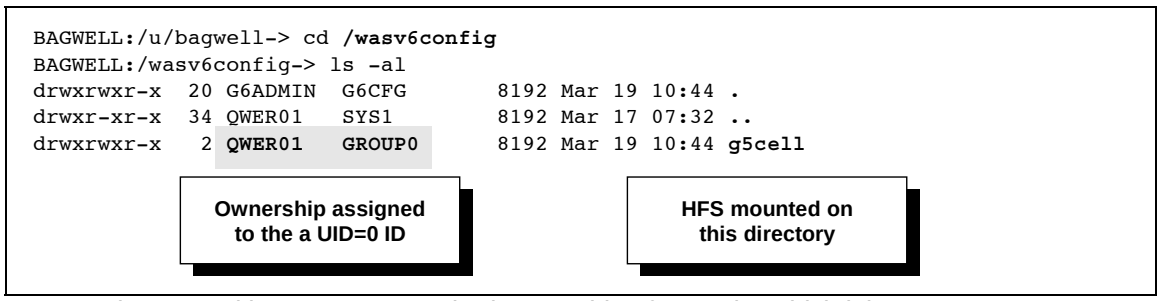
`BBOWMDMT` -- allocates and mounts the new V6 configuration HFS
`BBOMDCP` -- copies new V6 JCL start procedures to the PROCLIB
`BBOWMG3D` -- performs the migration

The BBOWMDMT job

☐ We added a `USER=` and `PASSWORD=` value to the `JOB` card so it would run under a userid we knew to have the authority to mount an HFS file system. The job was submitted and it completed with `RC=0`:

```
$HASP373 BBOWMDMT STARTED - INIT 1    - CLASS A - SYS SYSC
Jobname  Procstep Stepname  CPU Time      EXCPs     RC
BBOWMDMT --None-- ALLOC     00:00:00          0     00
BBOWMDMT --None-- MNTHFS    00:00:00         87     00
```

- We performed a visual inspection of the mount point and found that the ownership to be that of the `UID=0` ID under which we ran the `BBOWMDMT` job:



Mount point created by BBOWMDMT had ownership of ID under which job ran

> **Note:** Here at the Washington Systems Center we have a common directory under which all of us place our separate configuration roots. For V6, that common directory is `/wasv6config`.

The configuration mount point should be owned by the WebSphere Admin ID and WebSphere Configuration Group for the cell configuration mounted there. So this required fixing.

- We switched to superuser and from the `/wasv6config` directory we issued a `chown` command against the `g5cell` mount point directory:

```
chown g5admin:g5cfg ./g5cell
```

An `ls -al` command verified that our mount point now had the proper ownership:

```
BAGWELL:/wasv6config-> ls -al
total 320
  :
drwxrwxr-x   2 G5ADMIN  G5CFG       8192 Mar 19 10:44 g5cell
```

> **Note:** Some people feel it is better to skip `BBOWMDMT` and manually allocate and mount the HFS data set. We'll leave that decision to you. We showed what happens when `BBOWMDMT` is run so you can be aware of this ownership issue. It's easy enough to fix, but you have to know about it.

The BBOMDCP job

The `BBOMDCP` job was configured to copy six members to PROCLIB:

```
//SYSIN     DD *
 C INDD=INPUT,OUTDD=OUTPUT
 S M=((BBOMDDN,G51DEMN,R))
 S M=((BBOMDDNZ,G51DEMNZ,R))
 S M=((BBOMDCR,G51DCR,R))
 S M=((BBOMDCRZ,G51DCRZ,R))
 S M=((BBOMDSR,G51DSR,R))
 S M=((BBOMDSRZ,G51DSRZ,R))
/*
```

Recall that under "Preliminary work" starting on page 18 we backed up those same members from our `SYS1.PROCLIB` data set. We're using the same proc names so we can avoid having to create new `STARTED` profiles for new controller proc names.

> **Note:** You should not try to re-use V5 procs with a V6 configuration. Several things would need to be updated -- the `SET ROOT=` and any `STEPLIBS`. It is better to simply allow the migration utility to create new procs and copy them into PROCLIB.

- ☐ We added a USER= and PASSWORD= value to the JOB card so it would run under a userid we knew to have the authority to copy members into PROCLIB. The job was submitted and it completed with RC=0.
- ☐ A quick visual inspection of one of the copied members verified that the new procs did indeed have the V6 mount point for SET ROOT=

```
BROWSE    SYS1.PROCLIB(G51DCR) - 01.00
 Command ===>
******************************** Top of Data *
//G51DCR  PROC ENV=,Z=G51DCRZ,PARMS=' '
// SET ROOT='/wasv6config/g5cell'
```

The BBOWMG3D job

This job is what performs the actual migration.

- ☐ We insured the Deployment Manager was stopped.

> **Note:** What about the Daemon server that supported the DMGR? We left that running *for now*, but was stopped right after the BBOWMG3D job completed. The servers in the G5NODEC node still relied on this Daemon, so to maximize up-time during the long running of the migration we left it up. *But before the newly migrated Deployment Manager could be re-started, we had to stop and restart the Daemon.*

- ☐ We insured the JOB card had a USER= and PASSWORD= so this job ran under the ID of the "WebSphere Admin ID" of G5ADMIN. This job must run under that ID.
- ☐ We submitted the job. This jobs takes a long time to complete, so be patient.
- ☐ Sixteen minutes later the job had completed with all steps coming back RC=0

```
12.01.00  ---- SATURDAY,  19 MAR 2005 ----
12.01.00   $HASP373 BBOWMG3D STARTED
12.01.01   Jobname  Procstep Stepname  CPU Time    RC
12.01.01   BBOWMG3D --None-- SETUP     00:00:00    00
12.01.02   BBOWMG3D --None-- WRCONFIG  00:00:00    00
12.01.02   BBOWMG3D --None-- WRRESP    00:00:00    00
12.01.22   BBOWMG3D --None-- MKCONFIG  00:00:00    00
12.01.22   BBOWMG3D --None-- VERIFY    00:00:00    00
12.01.25   BBOWMG3D --None-- CRHOME    00:00:00    00
12.10.49   BBOWMG3D --None-- CRPROF    00:00:00    00
12.12.37   BBOWMG3D --None-- PREUPGRD  00:00:00    00
12.16.02   BBOWMG3D --None-- UPGRADE   00:00:00    00
12.17.02   BBOWMG3D --None-- UPPROCS   00:00:00    00
12.17.44   BBOWMG3D --None-- FINISHUP  00:00:00    00
12.17.44   BBOWMG3D --None-- WROUT     00:00:00    00
12.17.44   BBOWMG3D --None-- WRERR     00:00:00    00
12.17.44   $HASP395 BBOWMG3D ENDED
```

Performed post-migration RACF work

The BBOMDINS instruction member speaks of two sets of post-migration RACF work that may be needed:

1. New STARTED profiles

   This is mentioned under "Step 7" of the pre-job instructions. If we had provided *new* JCL start procedure names this would have been necessary. But since we backed up our V5 JCL procs and copied in the V6 procs *with the same names*, no new STARTED profiles were needed.

> **Note:** The BBOMDINS instruction member has an example of the command needed.

1. New keyring for Deployment Manager servant ID.

   One of the instructions found under the `BBOWMG3D` job says the following:

   > *WebSphere for z/OS V6 requires that the Deployment Manager servant has a WASKeyring with the WebSphere CA certificate. If your WebSphere V5 Deployment Manager uses a different userid for the Deployment Manager servant than it does for the controller you must update your RACF configuration.*

   That was the case for the G5CELL -- the Deployment Manager controller ran under a different userid than did the servant: `G5DCRU` vs. `G5DSRU`. Therefore, it was necessary to issue a few RACF commands to provide the `G5DSRU` ID a keyring and connect the WebSphere certificate to it.

☐ Submitted the following RACF command to add a keyring to the `G5DSRU` ID:

```
RACDCERT ADDRING(WASKeyring) ID(G5DSRU)
```

☐ Submitted the following RACF command to connect the Certificate Authority's certificate to the new keyring just created (command entered as one line):

```
RACDCERT ID(G5DSRU) CONNECT(RING(WASKeyring)
                                    LABEL('WebSphereCA') CERTAUTH)
```

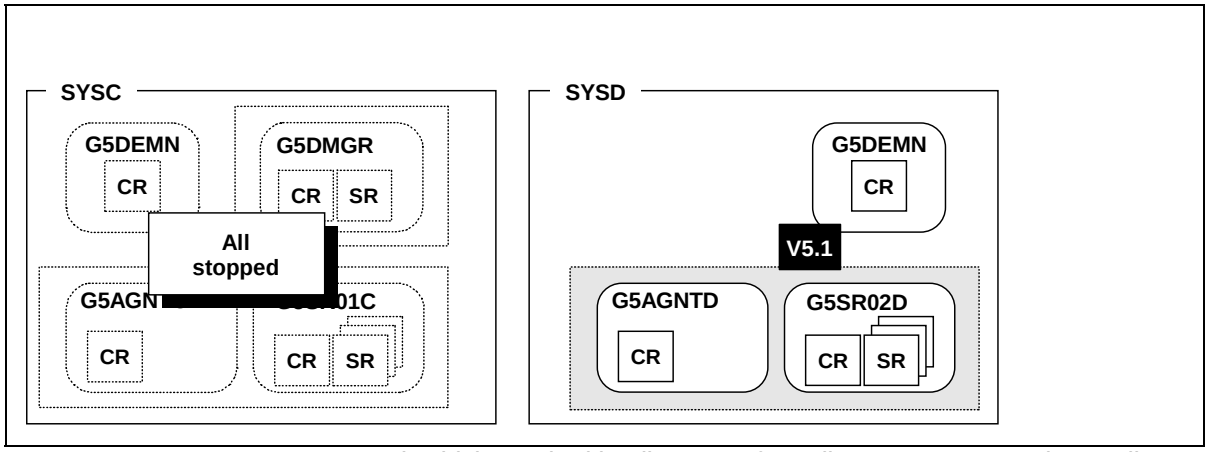
Stopped the Daemon on SYSC

Before we could restart our Deployment Manager at V6 we had to stop the Daemon server. During the migration of the Deployment Manager the Daemon that supported it -- the Daemon on SYSC -- was migrated as well. Recall we had left the Daemon running to maximize the up-time of the application servers on SYSC. Now was the time to stop the Daemon so it and the Deployment Manager could be brought up on V6.

☐ Issued a stop command against the SYSC Daemon:

```
/P G5DEMN
```

This resulted in all the servers on SYSC stopping. The servers on SYSD remained up:



Daemon server on SYSC stopped, which resulted in all servers for cell on SYSC stopped as well

Note: Any servers for *other cells* on SYSC would be unaffected by this stopping of the Daemon server. It's only servers *for that cell* on the same MVS image as the Daemon that are affected when a Daemon is stopped.

Started Deployment Manager

- ☐ Issued the start command for the Deployment Manager:

  ```
  S G51DCR,ENV=G5CELL.G5NODE.G5DMGR,JOBNAME=G5DMGR
  ```

  Note: That is the same start command as was used when the DMGR was at the V5.1 level.

  The Daemon server was then started automatically by the Deployment Manager.

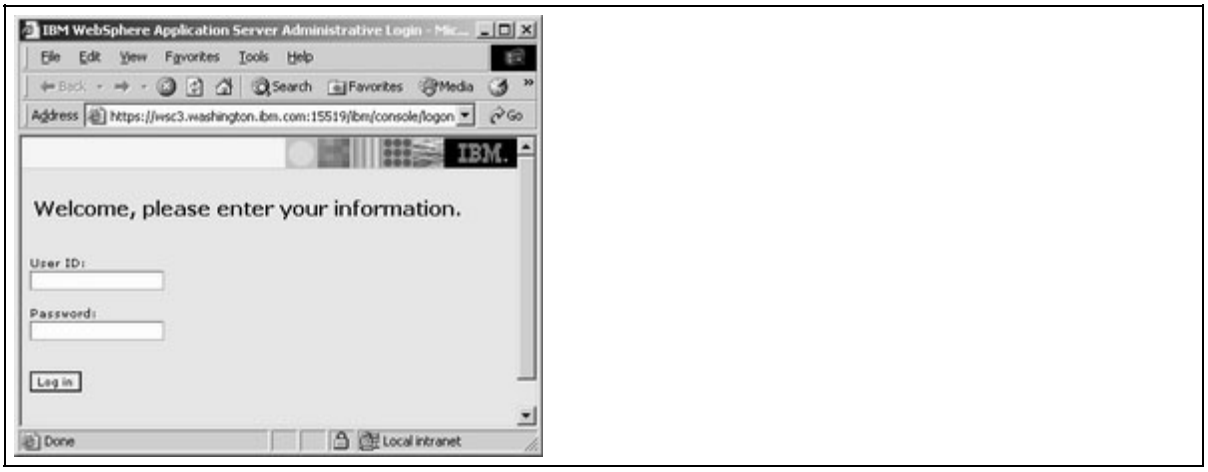
- ☐ We found the following message in the servant region's `SYSPRINT`:

  ```
  WSVR0001I: Server SERVANT PROCESS dmgr open for e-business
  ```

- ☐ We pointed a browser at the HTTP port of the newly-migrated Deployment Manager:

  ```
  http://wsc3.washington.ibm.com:15518/admin
  ```

  and received the following:



Version 6 Admin Console logon panel

Note: We had global security enabled in our V5.1 cell. This logon panel verifies that global security was migrated over to our V6 Deployment Manager as well.

- ☐ We logged on using the G5ADMIN ID and received the main Admin Console panel:

Version 6 Admin Console

Started servers in G5NODEC on SYSC

- ☐ Issued the start command for the Node Agent:

```
S G51ACR,ENV=G5CELL.G5NODEC.G5AGNTC,JOBNAME=G5AGNTC
```

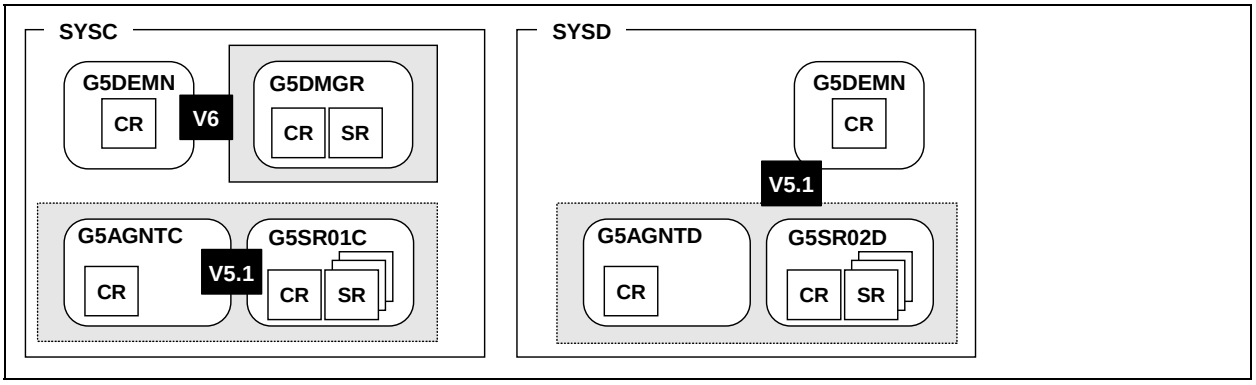
- ☐ Issued the start command for the `G5SR01C`:

```
S G51ACR,ENV=G5CELL.G5NODEC.G5SR01C,JOBNAME=G5SR01C
```

> **Note:** We did this just to verify that a V5.1 node could coexist with a V6 Daemon and Deployment Manager. The server started without difficulty.

Status of the cell at this point in time

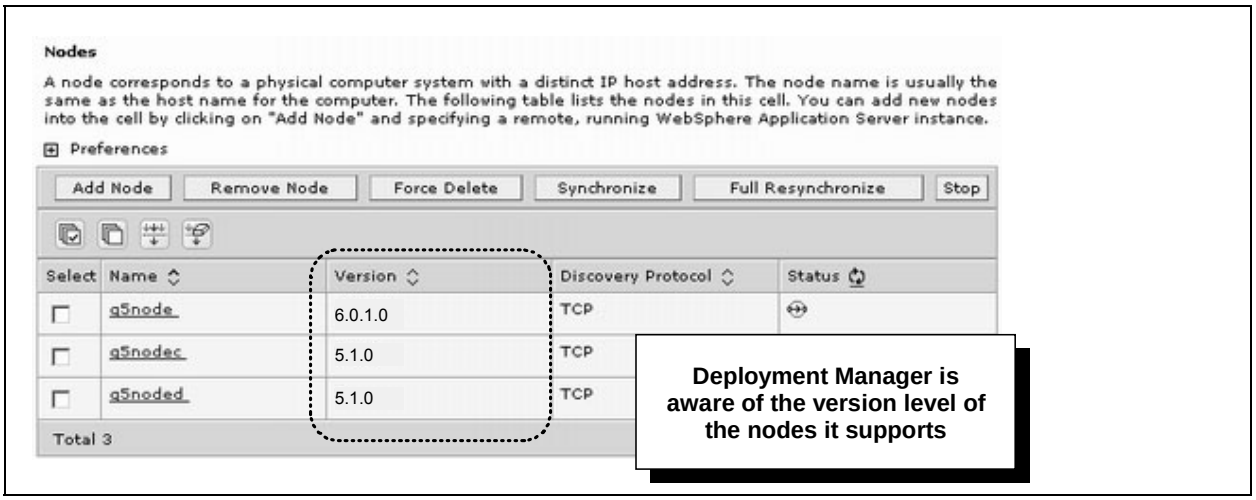
Here's what our cell looked like:



Status of the cell

What the V6 Admin Console showed for the nodes on SYSC and SYSD

We drove into "System Administration" and "Nodes" and saw the following:



Admin Console display for Nodes

Status of the old V5.1 Deployment Manager configuration

The V5 copy of the Deployment Manager is now in a "disabled" state. You would not be able to start it if you tried. You would get a message indicating that the `serverindex.xml` file is not found.

> **Note:** There is a process you may use to "fall back" to V5. It's documented under "Falling back to V5 (or recovering a migrated node)" on page 73.

The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB

We had a potential problem brewing at this point in time, and the problem was the result of three things that when combined formed what might be thought of as the "perfect storm":

1. The cell design of using shared procs between servers, particularly the Daemon servers but to a lesser degree the application servers as well.
2. Our decision to re-use the same JCL proc names for the V6 configuration
3. Our use of `STEPLIB` statements to point to the product libraries

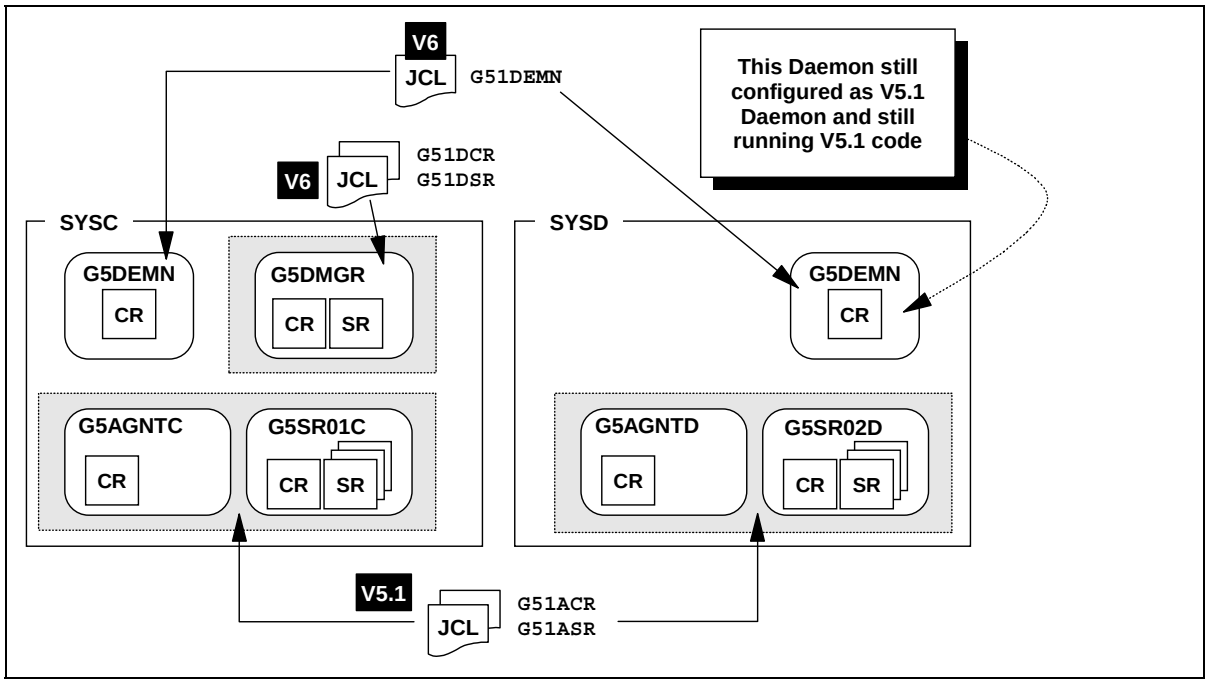
We had just completed migrating the Deployment Manager, and in the process we created new copies of the following JCL start procedures:

- `G51DCR` -- the Deployment Manager's controller start procedure
- `G51ACR` -- the Deployment Manager's servant start procedure
- `G51DEMN` -- the Daemon server's start procedure

The Deployment Manager procs weren't a concern -- those procs were used by the Deployment Manager only, and we had only one Deployment Manager. The problem at this point was the Daemon JCL start procedure. The copy of that in `SYS1.PROCLIB` was now updated to the V6.0 level, with `STEPLIB` pointers to the V6 code libraries.

The Daemon dilemma

Recall our configuration picture:



Common JCL start procedure mapping for this configuration

In the upper-right corner of that picture you see the problem -- our SYSD Daemon was not yet migrated, yet the JCL start procedure that would be used to start that Daemon was already migrated over to a V6 copy of the proc. As long as we didn't restart that Daemon we'd be okay. But if that Daemon was ever stopped and we tried to restart it, it would pick up the V6 copy of the JCL and try to start that Daemon as a V6 Daemon.

Is there a workaround to this? Yes, several:

- If we had the ability to turn back the hands of time we would not have configured the Daemon servers on both MVS images to use the same proc name. Way back when we thought that was clever; we have since come to understand better the pitfalls of doing this. But that was water under the bridge.
- In the customization panels for the Deployment Manager's migration, we chose to use the same JCL start procedure names -- `G51DCR`, `G51DSR` and `G51DEMN`. We could have specified a new JCL procedure name for the Daemon server -- for instance, `G56DEMN`.

  Note: That would have required a new `STARTED` profile -- `G56DEMN.*`

- If we had a procedure library higher in the concatenation than `SYS1.PROCLIB`, we could copy our backup of the V5.1 Daemon JCL proc to that system-specific proclib for SYSD. That way whenever the `G51DEMN` proc was invoked it would pick up that system's copy, which would be the V5.1 copy.
- There would always be the option of replacing the V6 copy of `G51DEMN` in `SYS1.PROCLIB` with our backup V5.1 copy of the proc in the event we ever needed to restart the SYSD Daemon. As long as the SYSC Daemon was up and running, it wouldn't need to access the JCL in proclib.
- Finally, if we had the opportunity to load V6 modules into `LPA/LNKLST` on SYSC and V5 modules into `LPA/LNKLST` on SYSD, we could have used the same Daemon proc (minus `STEPLIB` statements) and gotten around this. For the Daemon this would have been relatively easy, since the Daemon server does *not* have a `setupCmdLine.sh` shell script as part of its configuration (the `setupCmdLine.sh` file for a node may have `STEPLIB` statements in it that have to be taken into account).

  Things get a bit trickier for the application server node JCL procedures because of the `setupCmdLine.sh` `STEPLIB` instances.

  Note: Mapping out how all that's done is a bit beyond this document. For now, just be aware that our use of `STEPLIB`, combined with our use of shared JCL start procedures, combined with our re-use of the same JCL names created this dilemma. But it was *not* a show stopper, as you'll see.

If we had to do PRR processing for servers on SYSD ...

We did *not* exercise PRR (Peer Resource Recovery) during our migration because our servers did not have XA connectors installed. But if they did, it would have required stopping and restarting the servers on SYSD prior to running the migration job.

With regard to this Daemon JCL dilemma we just outlined, it would not have been a problem ... *provided we left the SYSD Daemon up and running*. PRR processing only requires that the application server node servers be stopped and restarted, and restarted at the *pre-migrated V5.1 level*.

As we approached the SYSD node migration, it would have been tempting to drop the Daemon and all the servers as preparation for the migration. After all, the servers have to be down to perform the migration. But had we a need to do the PRR processing (in other words, run the `BBOWMG1F` and `BBOWMG2F` jobs), dropping the Daemon would have been a mistake, given the restriction we faced due to our shared Daemon proc. When the application server was restarted as part of the `BBOWMG1F` processing, it would have tried to restart the Daemon, and it would have picked up the V6 copy of the proc. Things would not have worked, the Daemon would not have started (the Daemon's configuration on SYSD was still at V5.1 and the proc was at V6).

So had we faced the prospect of PRR processing for the G5NODED node, it would have entailed this:

- Stopping Node Agent and application servers in `G5NODED`.
- *Leaving the Daemon up*, running at the V5.1 level.
- Run `BBOWMG1F` against node.
- Start one of the application servers. The server would start and then stop automatically.
- Run `BBOWMG2F` against node.
- Stop the Daemon
- Run `BBOWMG3F` to migrate the whole node, including the Daemon
- Restart the servers, including the Daemon, now at V6

But thankfully we didn't face PRR processing, so none of this was an issue ... for us.

On the horizon: the shared application server proc dilemma

We hadn't yet migrated any application server nodes, so our JCL start procedures for those were still at the V5.1 level (`G5ACR` and `G5ASR`). But the moment one of our two application server nodes gets migrated, the other non-migrated one will face a dilemma. The dilemma again has to do with the triple-combination of shared JCL, re-use of proc names and the use of `STEPLIB` in the JCL.

The dilemma has to do with the fact that WLM is in control of the starting of servant regions. With a common servant region proc -- `G51ASR` -- we ran the risk of WLM trying to start another servant region in a not-yet-migrated node using the `G51ASR` proc that has been migrated to V6. The result would been a server controller running at V5.1 and a servant at V6. Would that have been a problem? Unknown -- we didn't try it. But we suspect it would have been, if not a problem, then at least not a good thing.

What would have been some workarounds to this? Similar to what we saw for the Daemon:

- Go back in time and not use a common proc between nodes on different MVS images.
- Create new procs with new names for the migrated node and pay the cost of creating new `STARTED` profiles for them.
- System-specific proclibs and the careful management of what version of procs reside in which system-specific proclibs.
- Careful management of what's in `LPA/LNKLST` for each system and the removal of `STEPLIB` from the JCL and the `setupCmdLine.sh` file for the node.

Wrap-up: our "gathering storm" may not be your problem

Again, what we just discussed here was the result of three things that came together: common procs between MVS images (*enabled* by our using a shared HFS, but the decision to use the same procs for each node was a separate decision); the decision to use the same JCL proc names for the migrated nodes so we could avoid having to create new `STARTED` procedures; and our use of `STEPLIB` statements in the configuration.

The absence of any one of those could have meant the avoidance of this dilemma.

Still, pointing this out to you was a good way to illustrating something you *might* face.

Migrated the Application Server Node on SYSD

The next step we took was to migrate the application server node on SYSD.

Why SYSD next and not SYSC?

In truth, either could have been migrated next. Because our nodes were at V5.1, we enjoyed the benefit of having a node coexist with a V6 Daemon on the same MVS image. That's why we started the servers in `G5NODEC` earlier -- to prove that very point.

> **Note:** Had our nodes been at the V5.0 level, then `G5NODEC` would have been our next node. At V5.0 those servers could not have been started when the Daemon server that supported it on SYSC was now at V6. So unless we were willing to let those servers stay down, those would have been the next migration.
>
> Happily they were at V5.1, so there was no time pressure.

So to some degree our decision to migrate SYSD next was to illustrate how it was possible to leave an application server node at V5.1 while its Daemon had been migrated to V6.

But in part our decision was due to the dilemma we faced due to our using shared procs and our decision to re-use the same proc names for the migrated nodes. It didn't create a hard requirement to migrate SYSD next, but it was enough of a nagging worry that we went ahead with the SYSD migration next.

For more on this "dilemma," see "The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB" on page 32.

Preliminary work

- ☐ The inventory of the node on SYSD was taken when the Deployment Manager was migrated. It was not necessary to do it again here.
- ☐ The HFS did not need to be backed up since this cell is contained within a single, shared HFS. So when we backed it up during the migration of the Deployment Manager, we backed up the entire cell -- SYSC and SYSD -- in one sweep.
- ☐ Backed up JCL start procedures used by application server node.

  The application servers had four JCL members to back up:

  ```
  G51ACR
  G51ACRZ
  G51ASR
  G51ASRZ
  ```

  Again, the reason why we backed up the JCL is because we intended to use the *same names* for the newly created V6 JCL start procedures. This reduced the amount of post-migration work required.

  > **Note:** What about the Daemon JCL? Recall that we are using a common set of JCL for all Daemon instances. Therefore, we had already backed up `G51DEMN`.

- ☐ Insured G5CELL's "Admin ID" could work in the `/tmp` directory

  This was simply a matter of checking to see that the `/tmp/migrate` directory was still owned by the `G5ADMIN` ID, and the `bbomigrt2.sh` file was also owned by `G5ADMIN`. Provided nobody else ran a migration between our Deployment Manager node and now, things should have been okay.

  In our case, it was okay. Were it not, we simply would have deleted the `/tmp/migrate` directory so this migration would have had a clear path to execute.

Invoked ISPF dialogs and customized migration jobs

- ☐ Invoked V6's ISPF customization dialogs:

  ```
  ex 'WAS600.WAS.SBBOCLIB(BBOWSTRT)' 'APPL(MIG)'
  ```

  The `'APPL(MIG)'` provided a set of ISPF variables for this migration run that were related to the ones used for the Deployment Manager.

- ☐ Selected Option 4 from the primary option menu:

```
 1  Configure a security domain.

 2  Create stand-alone Application Server nodes.  You must complete
    Option 1 before starting this option.

 3  Create Network Deployment cells and nodes.  You must complete
    Option 1 before starting this option.

 4  Migrate V5.x Nodes to V6 Nodes.
```

Option 4 -- used to invoke migration customization dialogs

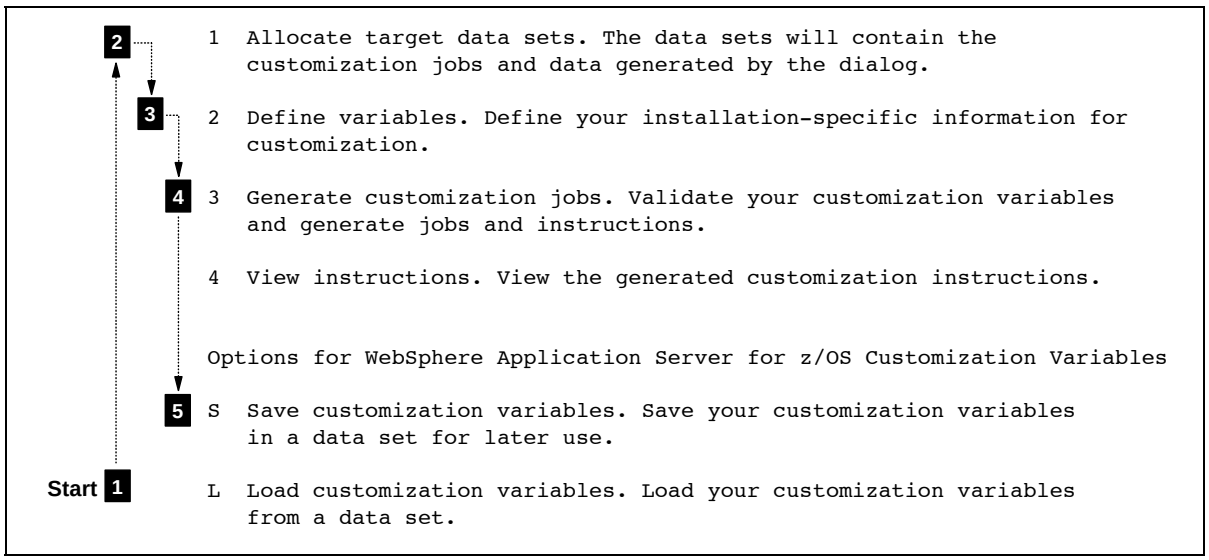
- ☐ Selected Option 3, "Migrate a V5.x federated node to V6":

```
 1  Migrate a V5.x stand-alone application server node to V6.

 2  Migrate a V5.x deployment manager to V6.

 3  Migrate a V5.x federated node to V6.  You must migrate the cell's
    deployment manager to V6 before migrating any of the cell's
    federated nodes
```

Option 3 -- migrate a federated node

Note: Recall the manner in which V5 Network Deployment cells were built -- first a "Base Application Server node" was constructed, then that node was "federated" into the Deployment Manager cell. In a V5 Network Deployment configuration, all application server nodes are "federated nodes."

- ☐ Was presented with the following menu. The processing through the menu took the following path, just as it was with the Deployment Manager:



Options presented to create customized migration jobs; order of processing used

- Loaded the *Deployment Manager's* saved customization variables (the "SAVECFG" member in the `G5CELL.WP.DMGR.*` data set). We used Option "L" to do this:

```
Load Customization Variables

 Specify the name of a data set containing the customization variables,
 then press Enter to continue.

 IBM-supplied defaults are in  'WAS600.WAS.SBBOEXEC(BBOWVARS)'

 Data set name: 'G5CELL.WP.DMGR.SAVECFG'
```

Provided new value here

Deployment Manager's customization variables loaded into SYSD migration

> **Note:** We did this so some of the variables common to all migrations could be brought into this customization run. It wasn't necessary to do this. But it was a handy way to keep from flipping back to see what values were entered before. It required some care to make sure things that required changing were changed. It's easy to get lazy and breeze through panels thinking all the variables are the same.

- Allocated the target data sets where the customized jobs were stored. These were *different* from the ones created for the Deployment Manager:

```
Allocate Target Data Sets

Specify a high level qualifier (HLQ) and press Enter to allocate the
data sets to contain the generated jobs and instructions. You can
specify multiple qualifiers (up to 39 characters).

High level qualifier: G5CELL.WP.NODED                          .CNTL
                                                               .DATA

The dialog will                                    panels. You can make
changes to the d                                   you should not change
the DCB characteristics of the data sets.

   .CNTL  - a PDS with fixed block 80-byte records to
            contain customization jobs.

   .DATA  - a PDS with variable length data to contain
            other data produced by the customization dialog.
```

Provided new value here

Allocated target data sets

We took the default allocation parameters.

> **Note:** This resulted in the creation of two data sets: `G5CELL.WP.NODED.CNTL` (a FB 80 PDS where the JCL was stored), and `G5CELL.WP.NODED.DATA` (a VB 255 PDS where scripts were stored). The "WP" qualifier stood for "White Paper."

- We selected Option 2, "Define Variables."
- We now faced three steps in the customization of the migration jobs:

```
1 - System Locations (directories, HLQs, etc.)
2 - System Environment Customization
3 - Server Customization
```

We worked through them in order: 1, then 2 and finally 3.

- The variables for "System Locations 1 of 2" were those used for the Deployment Manager. They were the same variables and we made no changes to the values on this panel.

```
System Locations (1 of 2)

   Specify the following V6 information, then press ENTER to continue.

   For some data sets, specify "Y" if they are in STEPLIB.

 Full Names of Data Sets

   PROCLIB.:  SYS1.PROCLIB

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  WAS600.WAS.SBBOLPA
   SBBOLOAD:  WAS600.WAS.SBBOLOAD
   SBBOLD2.:  WAS600.WAS.SBBOLD2

                                                      Use STEPLIB?
   SCEERUN.:  SYS1.LEMVS.SCEERUN                            Y
   SCEERUN2:  SYS1.LEMVS.SCEERUN2                           Y
   SGSKLOAD:  SYS1.CRYPTO.SGSKLOAD                          Y
             (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
```

Variables supplied for "System Locations 1 of 2"

Note: Use or not use `STEPLIB` according to your local preference.

- The variables for "System Locations 2 of 2" were also the same as for the Deployment Manager:

```
System Locations (2 of 2)

   Specify the following, then press Enter to continue.

 V6 WebSphere Application Server product directory:
     /u/bagwell/g6inter
```

Variables supplied for "System Locations 2 of 2"

As explained back in the section for the Deployment Manager, what this panel is asking for is the directory mount point of the V6 SMP/E HFS; in other words, where `WAS600.WAS.SBBOHFS` was mounted. We entered a value of `/u/bagwell/g6inter`, which represents an "intermediate symbolic link" we created that pointed to the real V6 product directory.

Note: See the `WP100396` white paper on `www.ibm.com/support/techdocs` for more on the use of "intermediate symbolic links" and their role in providing flexibility.

If we were truly interested in providing maximum flexibility, we would have used a separate "intermediate symbolic" link for each node. Perhaps one that had a system indicator in the directory, such as `/SYSC/wasv5/intermediate`. Had we done that, then the value we provided here for `G5NODED` would have been different than the one we provided for the Deployment Manager.

But as it was, the `G5CELL` was really a test cell -- intended to validate essential concepts -- and not serve as a true model for maximum flexibility and high-availability. So we had one common shared intermediate symbolic link for all nodes in the cell.

- The "System Environment Customization" variables were also the same as provided for the Deployment Manager:

```
System Environment Customization

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /wasv6config/g5cell
   Name...........:  OMVS.WAS6.G5CELL.MIGRATED.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  200
   Secondary allocation in cylinders.:  50
```

"System Environment Customization" variables: the HFS file system information

Here we had to slow down and think for a moment. Our goal was to have all the nodes in one shared HFS, just like we had it for V5.1. That being the case, we then needed only to supply the same value as we did for the Deployment Manager. The `BBOWMMMT` job generated was *not* run -- there was no need to: the HFS was already allocated and mounted.

Note: Even though we didn't plan to run `BBOWMMMT`, it was still necessary to provide the correct "Mount Point" value on this panel. This is an important variable used throughout the migration jobs.

If we had planned to have this node in a *separate* HFS, then a different mount point value and HFS data set name value would have been required. But since we were using a common, shared HFS all we needed to do was maintain the same values we used for the Deployment Manager.

- With the "Server Customization 1 of 2" variables we need to be very careful. This is where things will be very different for each node being migrated. Here's what we entered:

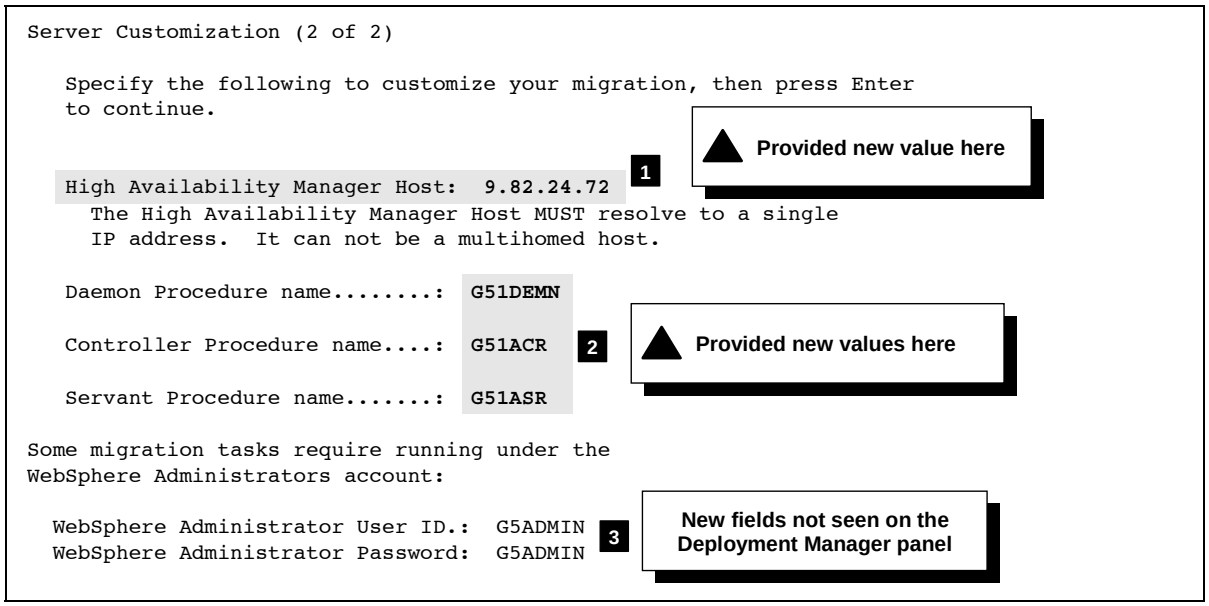
```
Server Customization (1 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.

   V5.x WebSphere Application Server home directory:
      /wasv51config/g5cell
         / AppServerNodeD      1

   V6 WebSphere Application Server home directory:
      /wasv6config/g5cell
         / AppServerNodeD      2

   Migration Options

     Enable z/OS Migration Tracing:  N
     Enable WASProfile Tracing....:  N
     Enable WASPreUpgrade Tracing.:  N
     Enable WASPostUpgrade Tracing:  N

     Default Backup Directory:  /tmp/migrate/5843/dmgr_backup
     User Specified Backup Directory:
```

"Server Customization" variables: source and target node directories

Notes:

1. This was the mount point and node directory for the V5 application server node for SYSD. We overwrote the Deployment Manager's value, which was carried in when we loaded that node's `SAVECFG` file. This value here was typed in manually.
2. This was the intended node directory for the migrated V6 node on SYSD. Notice how the mount point (`/wasv6config/g5cell`) was not open for update. That value was the HFS mount point provided on the previous panel. We chose to maintain the same value for the node -- "AppServerNodeD."

☐ The "Server Customization 2 of 2" variables were next:



"Server Customization" variables: High Availability Manager and JCL procedure names

Notes:

1. The "High Availability Manager Host" is a new function of V6 servers, which is why the migration panels are asking for information. The field on the panel will accept either a host name (`www.name.com`) or an IP address (`9.82.24.72`). As you can see on the panel, if you enter a host name, it must resolve to a single IP address.

   For the SYSD system on which this node resided, *all our hosts were multi-homed*. So we were forced to entered the IP address of SYSD MVS image.

   Note: And it was *absolutely critical* that it be unique within the cell. During one migration run we forgot to change this from SYSC's value of `9.82.24.71` and the server wouldn't start. WebSphere recognized the non-uniqueness of the value and refused to start the server. Note how for SYSD the value was `9.82.24.72`

2. New V6 JCL procedures will be created and placed in the *`hlq`*`.CNTL` target data set. The `BBOMMCP` member contains a job that will copy them to your specified PROCLIB (specified on the "System Locations 1 of 2" panel) and rename them to whatever values you supply here.

   The procs for the application server nodes were different from the Deployment Manager node. `G51DCR` vs. `G51ACR`. So it was necessary to run the `BBOMMCP` job.

   Just as with the Deployment Manager, we chose to use the same names.

> **Note:** Because we used the same procs for the application servers on SYSC and SYSD, this resulted in a slight dilemma when the SYSD servers were migrated. See "On the horizon: the shared application server proc dilemma" on page 34 for more information on this.

But what about the Daemon proc name of `G51DEMN`? Wasn't that the same name we used for the Deployment Manager? Yes. Again, all our Daemon instances were using the same proc name. Specifying the same name here will mean that the V6 proc will be copied into `SYS1.PROCLIB` *again*.

Was this a problem? No. But if you're at all concerned, simply remove the reference to `G51DEMN` and `G51DEMNZ` from the `BBOMMCP` job. That'll prevent the members from being copied a second time.

3. We supplied the "WebSphere Admin ID" and its password here. This is only really necessary if "global security" is enabled, but the panels require the fields to be completed. This information is used during the migration process so the migration process can connect to the Deployment Manager and invoke a node synchronization. If global security is enabled, it needs to be able to authenticate itself properly to the Deployment Manager.

☐ With the variables defined, we worked our way back to the main panel and selected Option 3 to "Generate customization jobs." The panel we received looked like this:

```
Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, ple

 Jobs and data files will get g
   'G5CELL.WP.NODED.CNTL'
   'G5CELL.WP.NODED.DATA'
 If you wish to generate custom                    ets, then
 exit from this panel and selec                    s" option.

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the job name for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname JOB (ACCTNO,ROOM),'BAGWELL',CLASS=A,REGION=0M
 //*
 //*
 //*
```

These data set values will be filled in automatically based on what you provided for the "Target Data Sets" earlier. But you should always check this to make sure it is truly where you want the jobs to be placed.

Generating the customization jobs

- When we hit the Enter key the jobs were written out to the target data sets:

```
Processing for data set 'G5CELL.WP.NODED.CNTL' ...
Member BBOWMG1F successfully created.
Member BBOWMG2F successfully created.
Member BBOWMG3F successfully created.
Member BBOWMMMT successfully created.
Member BBOMMCR successfully created.
Member BBOMMCRZ successfully created.
Member BBOMMDN successfully created.
Member BBOMMDNZ successfully created.
Member BBOMMSR successfully created.
Member BBOMMSRZ successfully created.
Member BBOMMCP successfully created.
Member BBO6CRA successfully created.
Member BBO6CRAZ successfully created.
Member BBOMMINS successfully created.

Processing for data set 'G5CELL.WP.NODED.DATA' ...
Member BBOWBMPT successfully created.
Member BBOWMMRF successfully created.
***
```

Customized jobs written out to the target data sets

- Then we selected Option S and saved the customization variables to a different SAVECFG data set than used by the Deployment Manager:

```
Save Customization Variables

  Specify the name of a sequential data set to contain the
  customization variables, then press Enter to continue. If the
  data set does not exist, the dialog displays the Allocate New
  Data Set panel, with which you can allocate a data set.

 Data set name: 'G5CELL.WP.NODED.SAVECFG'
```

▲ **Provided different value here**

Customized jobs written out to the target data sets

We took the default allocation parameters when they were presented to us.

Note: When coming up with names for your data sets, keep a few things in mind:
- Have a separate set of migration data sets for each node being migrated
- The high-level qualifier should indicate in some fashion the node being migrated
- The high-level qualifier for the CNTL, DATA and SAVECFG data sets for a node should be the same

- We exited the customization dialogs and did a data set list on our high-level qualifier. We saw our new SYSD node data sets and the Deployment Manager node data sets as well:

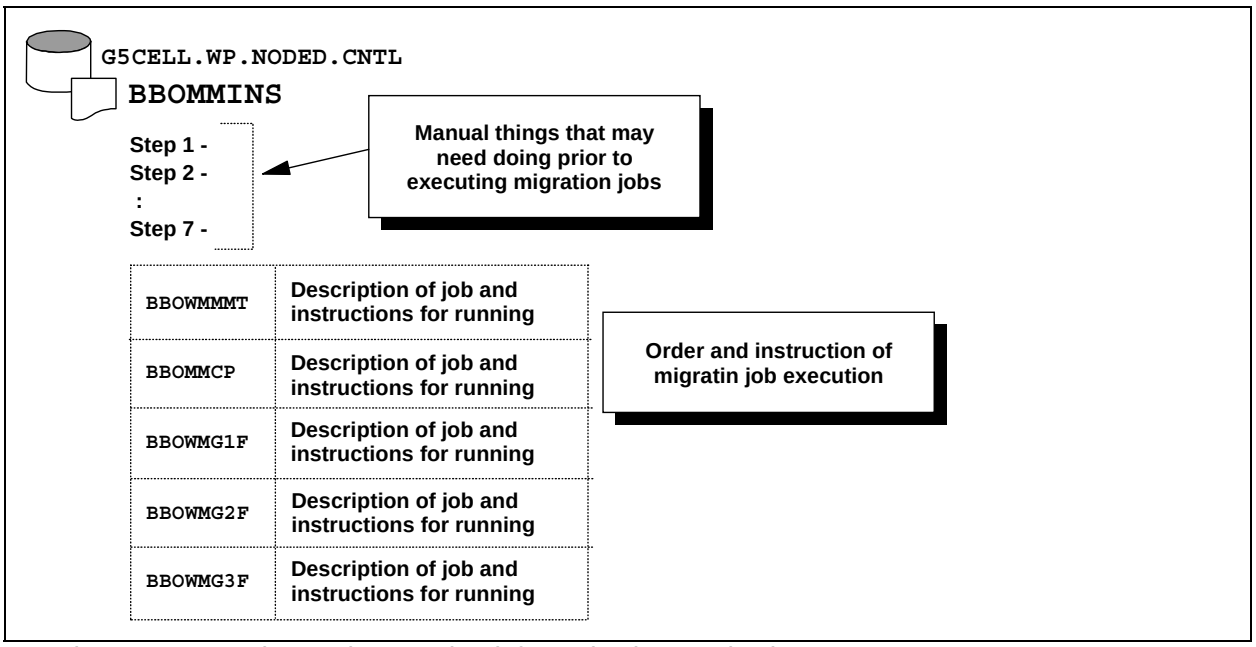
```
DSLIST - Data Sets Matching G5CELL.WP.**
Command ===>

Command - Enter "/" to select action
-----------------------------------------
         G5CELL.WP.DMGR.CNTL
         G5CELL.WP.DMGR.DATA
         G5CELL.WP.DMGR.SAVECFG
         G5CELL.WP.NODED.CNTL
         G5CELL.WP.NODED.DATA
         G5CELL.WP.NODED.SAVECFG
```

Just created during this customization run

Customized data sets for the migration of the SYSD application server node

Reviewed instruction member BBOMMINS in CNTL data set

The `BBOMMINS` member of the CNTL data set contains a list of instructions for migrating the node. The instructions are organized in the following manner:



How the BBOMMINS instruction member information is organized

Important: You should review the contents of that member. Do not rely exclusively on this document. In particular, those instructions speak of what needs to be done to your MVS system so WebSphere Application Server for z/OS Version 6 may operate. (Things like APF authorization and HFS auto-mount instructions.)

Important note -- please read

You may run through the ISPF customization dialogs on any MVS system in the Sysplex.

But the migration jobs ***must*** be run on the same MVS image on which the servers of the migrated node will run.

For our application server node G5NODED, that meant the jobs were run on the SYSD system.

Made sure Deployment Manager was up and running

☐ We checked to make sure the Deployment Manager was up and running, and that it was at the V6 level.

Note: This is a very important prerequisite to migrating a federated node. During migration, the migration utility will establish a TCP connection to the SOAP port of the running Deployment Manager and initiate a node synchronization process. The DMGR has to be there or the migration process fails. This, incidentally, is why the Admin ID and password was asked for in the "federated node" panels -- so if global security is enabled the SOAP connection to the DMGR can be properly authenticated.

Daemon server and all G5NODED servers shut down

☐ On the SYSD system we issued the following command:

```
/P G5DEMN
```

This resulted in the Daemon coming down and all the servers in the G5NODED node coming down as well. The servers in a node must be down to migrate the node.

> **Note:** And we had to be very careful with our command and make sure it was issued against the SYSD system. The JOBNAME value for Daemon servers in a Network Deployment configuration cell will all have the same value -- in this case, G5DEMN. Had we issued that against the SYSC system, we would have brought down our Deployment Manager and the G5NODEC servers as well.

Ran customized jobs

For the migration of a federated node, five customization jobs are generated:

BBOWMMMT -- allocates and mounts the new V6 configuration HFS
BBOMMCP -- copies new V6 JCL start procedures to the PROCLIB
BBOWMG1F -- enables PRR (Peer Resource Recovery) processing
BBOWMG2F -- disables PRR processing
BBOWMG3F -- performs the migration

The BBOWMMMT job

We did *not* run this job for the SYSD application server node migration. The V6 HFS was allocated and mounted when we ran the BBOWMDMT job for the Deployment Manager node.

If we had intended to host this node in a separate HFS, and indicated that on the "System Environment Customization" panel, then we would have run this job.

The BBOMMCP job

The BBOMMCP job was configured to copy six members to PROCLIB:

```
//SYSIN     DD *
 C INDD=INPUT,OUTDD=OUTPUT
 S M=((BBOMMDN,G51DEMN,R))
 S M=((BBOMMDNZ,G51DEMNZ,R))
 S M=((BBOMMCR,G51ACR,R))
 S M=((BBOMMCRZ,G51ACRZ,R))
 S M=((BBOMMSR,G51ASR,R))
 S M=((BBOMMSRZ,G51ASRZ,R))
/*
```

Recall that under "Preliminary work" we backed up those same members from our SYS1.PROCLIB data set. We're using the same proc names so we can avoid having to create new STARTED profiles for new controller proc names.

> **Note:** You should not try to re-use V5 procs with a V6 configuration. Several things would need to be updated -- the SET ROOT= and any STEPLIBS. It is better to simply allow the migration utility to create new procs and copy them into PROCLIB.

What about the G51DEMN and G51DEMNZ members? Weren't they copied into PROCLIB by the BBOMDCP job for the Deployment Manager? Yes, they were. You could copy them again (no harm done) or delete them from this file. We left them.

To run the job, we did the following:

- ☐ We added a USER= and PASSWORD= value to the JOB card so it would run under a userid we knew to have the authority to copy members into PROCLIB. The job was submitted and it completed with RC=0.
- ☐ A quick visual inspection of one of the copied members verified that the new procs did indeed have the V6 mount point for SET ROOT=

```
BROWSE    SYS1.PROCLIB(G51ACR) - 01.00
 Command ===>
****************************** Top of Data *
//G51ACR  PROC ENV=,Z=G51ACRZ,PARMS=' '
// SET ROOT='/wasv6config/g5cell'
```

The BBOWMG1F job

This job goes into the V5 configuration (yes, it does) and updates an XML setting so that when the server is started PRR (Peer Recovery Processing) can take place. Running this job is only required when you have XA connectors installed. For the application server on SYSD we did not, therefore *we did not run this job*.

Note: If you're not sure you can run this job in any case -- it shouldn't hurt anything.

After running this job, one of the application servers in the node needs to be started so PRR can take place. When things work as they're supposed to, the server will come up and then automatically shut down after a minute or so.

For the node on SYSD, we were fortunate we didn't have XA connectors installed. Because of the shared Daemon server procs, things would have been a bit more complicated. It would have all revolved around not shutting down the Daemon server until just the right moment. See "The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB" on page 32 for more on this.

The BBOWMG2F job

This job goes back into the V5 configuration XML and *disables* the change made by BBOWMG1F. Therefore:

1. If BBOWMG1F was *not* run, then you do not need to run BBOWMG2F
2. But if BBOWMG1F *was* run, then running BBOWMG2F is *mandatory*

This job is supposed to be run when no servers in the node are running. And after this job is run, then no server in the node should be started until the node has been migrated.

For us, since we did not run BBOWMG1F, we did not run this job either.

The BBOWMG3F job

This job is what performs the migration.

- ☐ We insured the JOB card had a USER= and PASSWORD= so this job ran under the ID of the "WebSphere Admin ID" of G5ADMIN. This job must run under that ID.
- ☐ We submitted the job. This jobs takes a long time to complete, so be patient.
- ☐ Thirty one minutes later the job had completed with all steps coming back RC=0

```
15.28.35 ---- SATURDAY,  19 MAR 2005 ----
15.28.35  $HASP373 BBOWMG3F STARTED
15.28.38  Jobname  Procstep Stepname  CPU Time   RC
15.28.38  BBOWMG3F --None-- SETUP     00:00:00   00
15.28.38  BBOWMG3F --None-- WRCONFIG  00:00:00   00
15.28.38  BBOWMG3F --None-- WRRESP    00:00:00   00
```

```
15.28.50  BBOWMG3F --None-- MKCONFIG  00:00:00   00
15.28.50  BBOWMG3F --None-- VERIFY    00:00:00   00
15.29.08  BBOWMG3F --None-- CRHOME    00:00:00   00
15.30.54  BBOWMG3F --None-- PREUPGRD  00:00:00   00
15.36.15  BBOWMG3F --None-- CRPROF    00:00:00   00
15.55.15  BBOWMG3F --None-- UPGRADE   00:00:00   00
15.58.39  BBOWMG3F --None-- UPPROCS   00:00:00   00
15.59.47  BBOWMG3F --None-- FINISHUP  00:00:00   00
15.59.47  BBOWMG3F --None-- WROUT     00:00:00   00
15.59.47  BBOWMG3F --None-- WRERR     00:00:00   00
15.59.47  $HASP395 BBOWMG3F ENDED
```

Note: Why did it take so long? Just a theory, but consider this: the shared HFS for this migrated configuration was owned by SYSC when this job was run on SYSD. The overhead of shared HFS might be showing its face in this long execution time.

Performed post-migration RACF work

The BBOMMINS instruction member speaks of two sets of post-migration RACF work that may be needed:

1. New STARTED profiles

   This is mentioned under "Step 7" of the pre-job instructions. If we had provided *new* JCL start procedure names this would have been necessary. But since we backed up our V5 JCL procs and copied in the V6 procs *with the same names*, no new STARTED profiles were needed.

   Note: For the controller JCL only. The BBOMMINS instruction member provides an example.

1. Permitting the servant region ID access to BPX.WLMSERVER.

   One of the instructions found under the BBOWMG3F job says the following:

   > *WebSphere for z/OS V6 requires that the servant's address space is authorized to interface with the z/OS Workload Manager. The following RACF commands must be issued to permit the Application Server's started task userid to have Read access to the WLM Server profile.*

   It was a simple two-command process

- ☐ Submitted the following RACF commands to permit the servant ID (G5ASRU) access to the BPX.WLMSERVER class:

```
PE BPX.WLMSERVER CLASS(FACILITY) ID(G5ASRU) ACCESS(READ)

SETROPTS RACLIST(FACILITY) REFRESH
```

Started servers on SYSD

We chose to start the Node Agent first. Node Agents provide the Deployment Manager a way to "see into" the status of a node. In truth, the sequence of start -- Node Agent or application server -- doesn't matter.

Node Agent

- ☐ Issued the start command for the Node Agent:

```
S G51ACR,ENV=G5CELL.G5NODED.G5AGNTD,JOBNAME=G5AGNTD
```

Note: That is the same start command as was used when the Node Agent was at the V5.1 level of code.

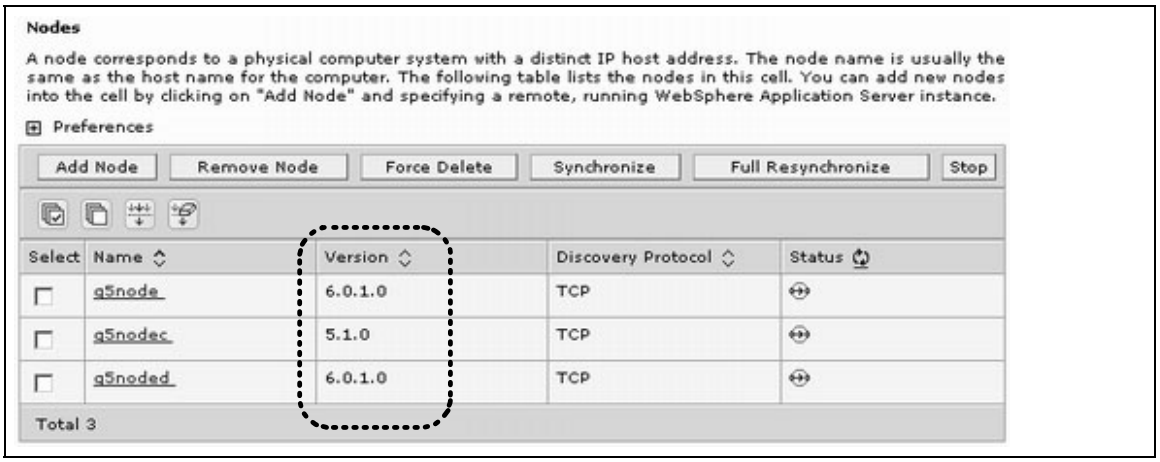
☐ We found the following message in the Node Agent's controller SYSPRINT:

```
WSVR0001I: Server CONTROL PROCESS nodeagent open for e-business
```

Note: There's a reason why we didn't start the server next. Keep reading ...

What the V6 Admin Console showed for the nodes on SYSC and SYSD

☐ We checked the Admin Console to see what it saw regarding its nodes:



Admin Console display for Nodes

The Deployment Manager now saw the `g5noded` node as at V6. The `g5nodec` node was started but not yet migrated to V6. It was still at V5.1

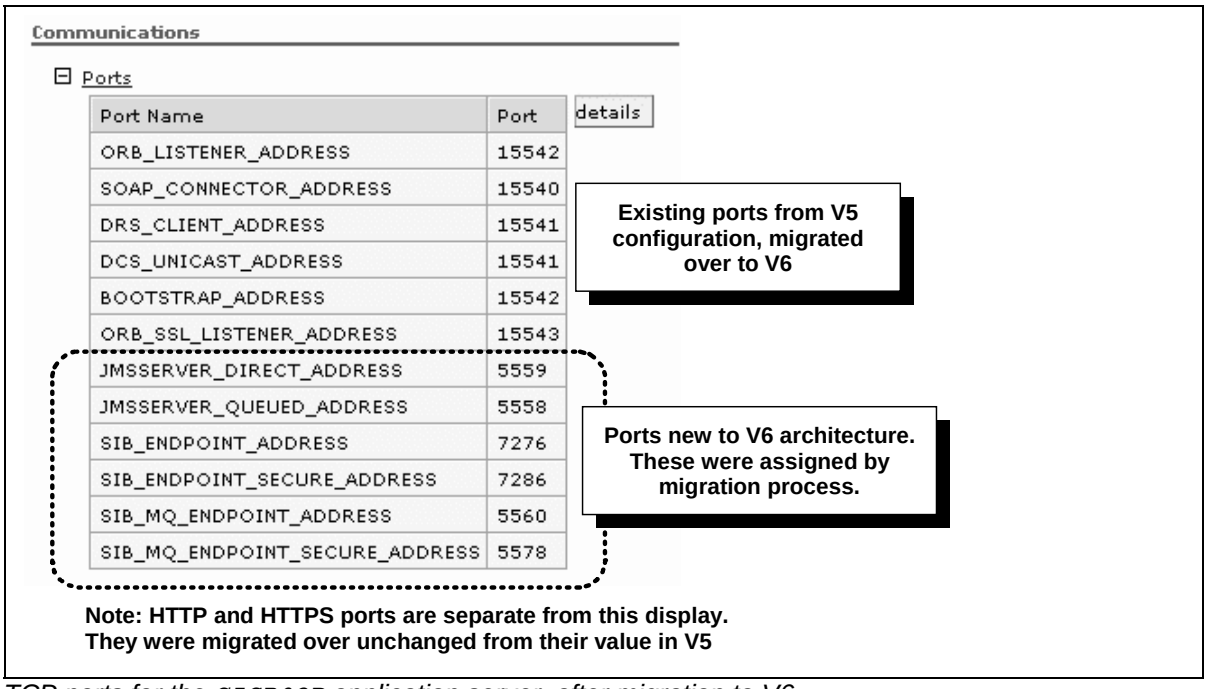
Note: See "Other V6 "mixed node" information" on page 8 for more on what capabilities are present for "mixed node" environments.

New ports created for V6 application server

When an V5 application server is migrated to V6, the migration process will create *additional ports*. These ports are required so the migrated server adheres to the architecture of V6.

Note: This discussion applies only to *application servers*, not the Deployment Manager, Node Agents or Daemon servers. Those servers have the same number of ports, though in V6 they are sometimes called different things.

So when we finished migrating the node on SYSD, we found that the ports for the `G5SR02D` server were as follows:



Port Name	Port
ORB_LISTENER_ADDRESS	15542
SOAP_CONNECTOR_ADDRESS	15540
DRS_CLIENT_ADDRESS	15541
DCS_UNICAST_ADDRESS	15541
BOOTSTRAP_ADDRESS	15542
ORB_SSL_LISTENER_ADDRESS	15543
JMSSERVER_DIRECT_ADDRESS	5559
JMSSERVER_QUEUED_ADDRESS	5558
SIB_ENDPOINT_ADDRESS	7276
SIB_ENDPOINT_SECURE_ADDRESS	7286
SIB_MQ_ENDPOINT_ADDRESS	5560
SIB_MQ_ENDPOINT_SECURE_ADDRESS	5578

TCP ports for the `G5SR02D` application server, after migration to V6

We could have left these as the migration process assigned them. But the migration process had no knowledge of TCP port usage elsewhere on the communication stack, so these numbers may or may not already be in use. As we mentioned earlier, you will probably want to re-map these to numbers you know are good.

The dilemma we faced was this: what numbers should we assign to these new ports?

Note: Changing the ports in the Admin Console is the easy part -- it's a simple point-and-click exercise. The issue here was coming up with a meaningful set of new ports.

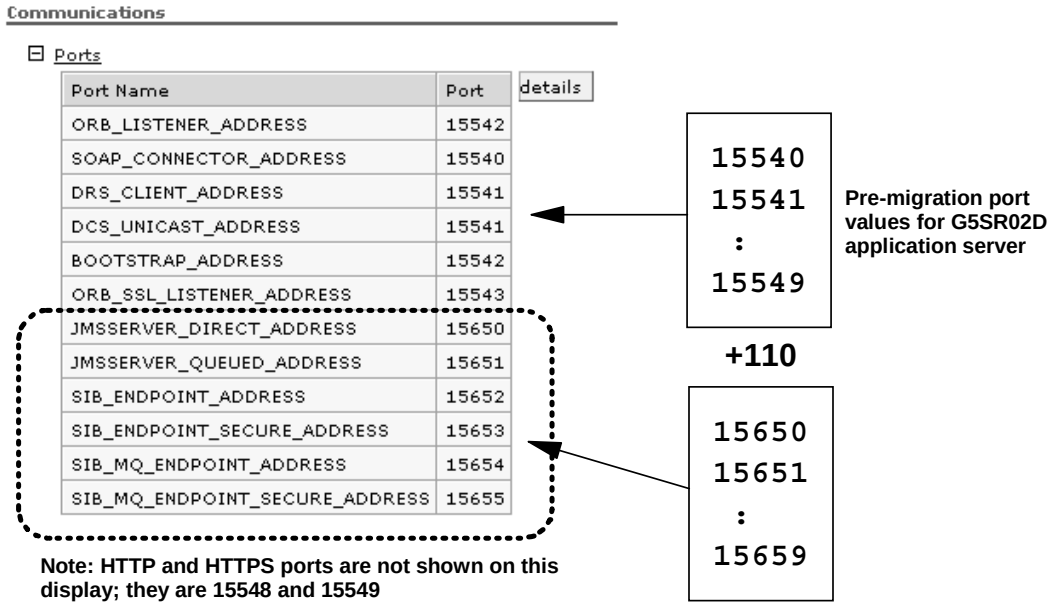
When we built the G5CELL using V5 of the product, we used a port allocation scheme that set aside 10 TCP ports per server. No server used all ten, but it provided a handy way to organize the port assignments into something that made some sense. Here's the assignment matrix we used for the G5CELL:

	Daemons	Dep. Manager	Node Agents	g5sr01c	g5sr02d
JMX Soap		15510	15520	15530	15540
DRS		15511	15521	15531	15541
Bootstrap		15512	15522	15532	15542
ORB IIOP	15500	15512	15522	15532	15542
ORB SSL	15501	15513	15523	15533	15543
Discovery		15514	15524		
Multi-cast			15525		
HTTP		15518		15538	15548
HTTP SSL		15519		15539	15549
	00-09	**10-19**	**20-29**	**30-39**	**40-49**

TCP port assignment matrix used when constructing G5CELL in V5

We now needed six more ports for `G5SR01C` and six more ports for `G5SR02D`. If we simply added six to the top range of `G5SR01C`'s allocation, it would take us into the ports assigned to `G5SR02D`. See the problem?

At the time this document was written, we hadn't worked out an elegant solution for this. So we simply took the server's port range, added 110 to it, then extended the six new ports from there. So for G5SR02D it's new ports became:



Where the six new ports came from

Notes:
- *Please* do not consider this a recommendation or a "best practice." It is simply a way of explaining the issue and offering one solution, though perhaps not the most elegant solution.
- Any time port values are changed for a server, those servers must be restarted if the change was made while they were already up. For us, G5SR02D hadn't yet been started, but the Node Agent had. That's how we could change the server's port values and get the changes synchronized out to the G5NODED node.

☐ Went into the Admin Console and under the g5sr02d server clicked on "Ports."

☐ Changed the port values one by one to the values shown in the previous chart.

☐ Saved the changes to the master configuration and *synchronized out to the nodes*.

Note: Being able to synchronize the changes out to the G5NODED node was critical. That's why the Node Agent had to be up. If we couldn't synchronize, the changes would have remained in the "Master Configuration" and never made it out to the server.

Application server G5SR02D

☐ Issued the start command for the G5SR02D application server on SYSD:

```
S G51ACR,ENV=G5CELL.G5NODED.G5SR02D,JOBNAME=G5SR02D
```

We saw the message:

```
WSVR0001I: Server SERVANT PROCESS g5sr02d open for e-business
```

The SYSC application server JCL procedure dilemma at this point

See "On the horizon: the shared application server proc dilemma" on page 34. For a more comprehensive explanation of the problem we encountered due to our use of shared procs and the re-use of proc names during migration, see "The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB" starting on page 32.

Migrated the Application Server Node on SYSC

The process was very similar to what was done for the application server node on SYSD. Actually, it was somewhat easier as some jobs didn't need to be run because the work needed had already been done earlier (for instance, the `BBOWMMMT` and `BBOMMCP` jobs).

Preliminary work

Again, this fell into three categories, none of which we had to do:

- *Back up HFS* -- didn't need to do this; it was done when the Deployment Manager was migrated. Our single shared HFS meant the HFS needed to be backed up only once.
- *Back up JCL start procedures* -- didn't need to do this either; the application server procs `G51ACR` and `G51ASR` were backed up when the application server node on SYSD was migrated.
- *Insure migration jobs have access to* `/tmp/migrate` -- provided nobody else had changed the ownership on that directory and the `bbomigrt2.sh job`, it should still have been owned by the `G5ADMIN` ID. A quick visual check verified that it was. We were ready to go.

Invoked ISPF dialogs and customized migration jobs

This was also relatively easy. By loading the saved variables from the application server migration on SYSD and changing a few things, we made short work of this effort.

☐ Invoked V6's ISPF customization dialogs:

```
ex 'WAS600.WAS.SBBOCLIB(BBOWSTRT)' 'APPL(MIG)'
```

The `'APPL(MIG)'` provided a set of ISPF variables for this migration run related to the last run.

☐ Selected Option 4 from the primary option menu:

```
1  Configure a security domain.

2  Create stand-alone Application Server nodes.  You must complete
   Option 1 before starting this option.

3  Create Network Deployment cells and nodes.  You must complete
   Option 1 before starting this option.

4  Migrate V5.x Nodes to V6 Nodes.
```

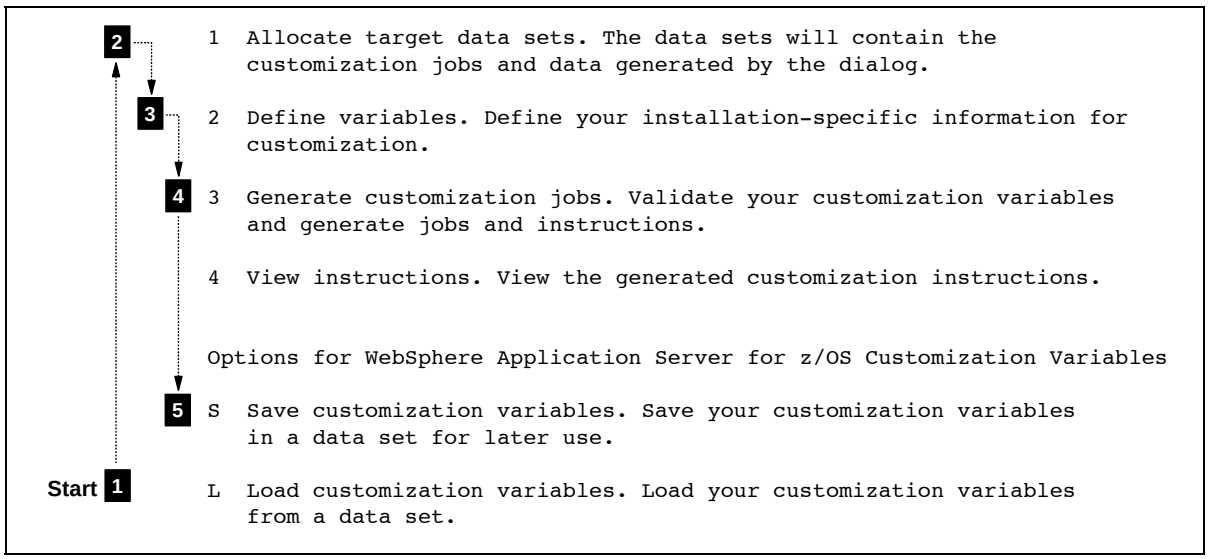
Option 4 -- used to invoke migration customization dialogs

☐ Selected Option 3, "Migrate a V5.x federated node to V6":

```
1  Migrate a V5.x stand-alone application server node to V6.

2  Migrate a V5.x deployment manager to V6.

3  Migrate a V5.x federated node to V6.  You must migrate the cell's
   deployment manager to V6 before migrating any of the cell's
   federated nodes
```

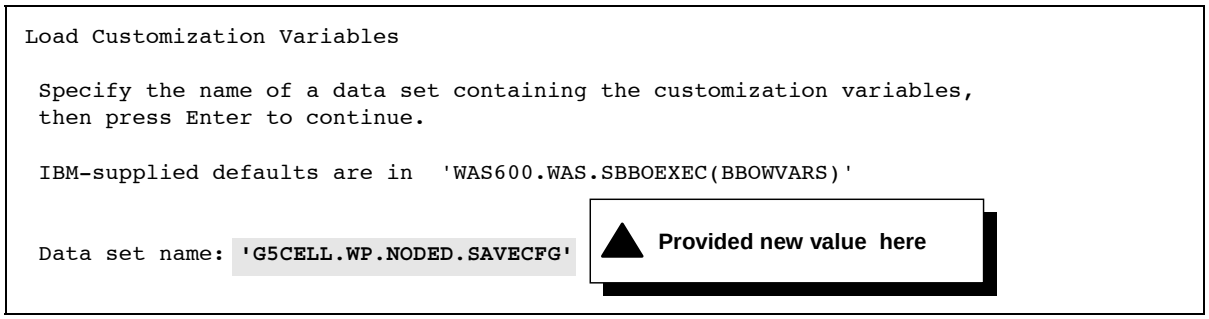
Option 3 -- migrate a federated node

- Was presented with the following menu. The processing through the menu took the following path, just as it was with the node migration for SYSD:



Options presented to create customized migration jobs; order of processing used

- Loaded the saved customization variables from the `G5NODED` migration (the "`SAVECFG`" member in the `G5CELL.WP.NODED.*` data set). We used Option "L" to do this:



G5NODED's customization variables loaded into SYSD migration

Note: We did this so some of the variables common to all migrations could be brought into this customization run. It wasn't necessary to do this. But it was a handy way to keep from flipping back to see what values were entered before.

- Allocated the target data sets where the customized jobs were stored. These were *different* from the ones created for G5NODED:

```
Allocate Target Data Sets

Specify a high level qualifier (HLQ) and press Enter to allocate the
data sets to contain the generated jobs and instructions. You can
specify multiple qualifiers (up to 39 characters).

High level qualifier: G5CELL.WP.NODEC                          .CNTL
                                                               .DATA

The dialog will c                                  els. You can make
changes to the de                                  should not change
the DCB characteristics of the data sets.

   .CNTL  - a PDS with fixed block 80-byte records to
            contain customization jobs.

   .DATA  - a PDS with variable length data to contain
            other data produced by the customization dialog.
```

Provided new value here

Allocated target data sets

We took the default allocation parameters.

Note: This resulted in the creation of two data sets: G5CELL.WP.NODEC.CNTL (a FB 80 PDS where the JCL was stored), and G5CELL.WP.NODEC.DATA (a VB 255 PDS where scripts were stored). The "WP" qualifier stood for "White Paper."

- We selected Option 2, "Define Variables."
- We now faced three steps in the customization of the migration jobs:

```
1 - System Locations (directories, HLQs, etc.)
2 - System Environment Customization
3 - Server Customization
```

We worked through them in order: 1, then 2 and finally 3.

- The variables for "System Locations 1 of 2" were those used for the Deployment Manager and G5NODED. They were the same variables and we made no changes to the values on this panel.

```
System Locations (1 of 2)

   Specify the following V6 information, then press ENTER to continue.

   For some data sets, specify "Y" if they are in STEPLIB.

 Full Names of Data Sets

   PROCLIB.:  SYS1.PROCLIB

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  WAS600.WAS.SBBOLPA
   SBBOLOAD:  WAS600.WAS.SBBOLOAD
   SBBOLD2.:  WAS600.WAS.SBBOLD2

                                                      Use STEPLIB?
   SCEERUN.:  SYS1.LEMVS.SCEERUN                           Y
   SCEERUN2:  SYS1.LEMVS.SCEERUN2                          Y
   SGSKLOAD:  SYS1.CRYPTO.SGSKLOAD                         Y
             (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
```

Variables supplied for "System Locations 1 of 2"

Note: Use or not use STEPLIB according to your local preference.

- ☐ The variables for "System Locations 2 of 2" were also the same as for the Deployment Manager:

```
System Locations (2 of 2)

   Specify the following, then press Enter to continue.

 V6 WebSphere Application Server product directory:
     /u/bagwell/g6inter
```

Variables supplied for "System Locations 2 of 2"

As explained earlier, what this panel is asking for is the directory mount point of the V6 SMP/E HFS; in other words, where WAS600.WAS.SBBOHFS was mounted. We entered a value of /u/bagwell/g6inter, which represents an "intermediate symbolic link" we created that pointed to the real V6 product directory.

Note: See the WP100396 white paper on www.ibm.com/support/techdocs for more on the use of "intermediate symbolic links" and their role in providing flexibility.

If we were truly interested in providing maximum flexibility, we would have used a separate "intermediate symbolic" link for each node. Perhaps one that had a system indicator in the directory, such as /SYSC/wasv5/intermediate. Had we done that, then the value we provided here for G5NODEC would have been different from the value we provided for the node on SYSD.

But as it was, the G5CELL was really a test cell -- intended to validate essential concepts -- and not serve as a true model for maximum flexibility and high-availability. So we had one common shared intermediate symbolic link for all nodes in the cell.

- ☐ The "System Environment Customization" variables were also the same as provided for G5NODED:

```
System Environment Customization

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /wasv6config/g5cell
   Name...........:  OMVS.WAS6.G5CELL.MIGRATED.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  200
   Secondary allocation in cylinders.:  50
```

"System Environment Customization" variables: the HFS file system information

We were building our V6 cell under a common, shared HFS. Therefore, the BBOWMMMT job generated was *not* run -- there was no need to: the HFS was already allocated and mounted.

Note: Even though we didn't plan to run BBOWMMMT, it was *still necessary* to provide the correct "Mount Point" value on this panel. This is an important variable used throughout the migration jobs.

If we had planned to have this node in a *separate* HFS, then a different mount point value and HFS data set name value would have been required. But since we were using a

common, shared HFS all we needed to do was maintain the same values we used for the Deployment Manager.

- With the "Server Customization 1 of 2" variables we need to be very careful. This is where things will be very different for each node being migrated. Here's what we entered:

```
Server Customization (1 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.

   V5.x WebSphere Application Server home directory:
      /wasv51config/g5cell
         / AppServerNodeC        1

   V6 WebSphere Application Server home directory:
      /wasv6config/g5cell
         / AppServerNodeC        2

   Migration Options

     Enable z/OS Migration Tracing:  N
     Enable WASProfile Tracing....:  N
     Enable WASPreUpgrade Tracing.:  N
     Enable WASPostUpgrade Tracing:  N

     Default Backup Directory:  /tmp/migrate/5843/dmgr_backup
     User Specified Backup Directory:
```

▲ **Provided new values here**

"Server Customization" variables: source and target node directories

Notes:

1. This was the mount point and node directory for the V5 application server node for SYSC. We overwrote the `AppServerNodeD` value, which was carried in when we loaded that node's `SAVECFG` file. This value was typed in manually.
2. This was the intended node directory for the migrated V6 node on SYSC. Notice how the mount point (`/wasv6config/g5cell`) was not open for update. That value was the HFS mount point provided on the previous panel. We chose to maintain the same value for the node -- "AppServerNodeC."

☐ The "Server Customization 2 of 2" variables were next:

```
Server Customization (2 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.

   High Availability Manager Host:  9.82.24.71
     The High Availability Manager Host MUST resolve to a single
     IP address.  It can not be a multihomed host.

   Daemon Procedure name........:  G51DEMN

   Controller Procedure name....:  G51ACR

   Servant Procedure name.......:  G51ASR

Some migration tasks require running under the
WebSphere Administrators account:

  WebSphere Administrator User ID.:  G5ADMIN
  WebSphere Administrator Password:  G5ADMIN
```

1 2 3

▲ Provided new value here

"Server Customization" variables: High Availability Manager and JCL procedure names

Notes:

1. The "High Availability Manager Host" is a new function of V6 servers, which is why the migration panels are asking for information. The field on the panel will accept either a host name (`www.name.com`) or an IP address (`9.82.24.71`). As you can see on the panel, if you enter a host name, it must resolve to a single IP address.

   For the SYSC system on which this node resided, *all our hosts were multi-homed*. So we were forced to entered the IP address of SYSC MVS image.

   Note: And it was *absolutely critical* that it be unique within the cell. During one migration run we forgot to change this from SYSD's value of `9.82.24.72` and the server wouldn't start. WebSphere recognized the non-uniqueness of the value and refused to start the server. Note how for SYSC the value was `9.82.24.71`

2. New V6 JCL procedures will be created and placed in the *`hlq`*`.CNTL` target data set. For our cell, the application servers on SYSC used the same procs as the servers on SYSD. Therefore, these procs had already been copied over when the `BBOMMCP` job was run for the migration of `G5NODED`.

   Note: But it was still necessary to insure the correct proc name values were supplied here. The migration utilities used the values supplied here to update the XML for `G5NODEC` so the proc names configured in the XML would be correct.

3. We supplied the "WebSphere Admin ID" and its password here. This is only really necessary if "global security" is enabled, but the panels require the fields to be completed. This information is used during the migration process so the migration process can connect to the Deployment Manager and invoke a node synchronization. If global security is enabled, it needs to be able to authenticate itself properly to the Deployment Manager.

- With the variables defined, we worked our way back to the main panel and selected Option 3 to "Generate customization jobs." The panel we received looked like this:

```
Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, ple

 Jobs and data files will get g
   'G5CELL.WP.NODEC.CNTL'
   'G5CELL.WP.NODEC.DATA'
 If you wish to generate custom                    ets, then
 exit from this panel and selec                    s" option.

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the job name for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname JOB (ACCTNO,ROOM),'BAGWELL',CLASS=A,REGION=0M
 //*
 //*
 //*
```

These data set values will be filled in automatically based on what you provided for the "Target Data Sets" earlier. But you should always check this to make sure it is truly where you want the jobs to be placed.

Generating the customization jobs

- When we hit the Enter key the jobs were written out to the target data sets:

```
Processing for data set 'G5CELL.WP.NODEC.CNTL' ...
Member BBOWMG1F successfully created.
Member BBOWMG2F successfully created.
Member BBOWMG3F successfully created.
Member BBOWMMMT successfully created.
Member BBOMMCR successfully created.
Member BBOMMCRZ successfully created.
Member BBOMMDN successfully created.
Member BBOMMDNZ successfully created.
Member BBOMMSR successfully created.
Member BBOMMSRZ successfully created.
Member BBOMMCP successfully created.
Member BBO6CRA successfully created.
Member BBO6CRAZ successfully created.
Member BBOMMINS successfully created.

Processing for data set 'G5CELL.WP.NODEC.DATA' ...
Member BBOWBMPT successfully created.
Member BBOWMMRF successfully created.
***
```

Customized jobs written out to the target data sets

- Then we selected Option S and saved the customization variables to a *different* SAVECFG data set than used for the G5NODED node:

```
Save Customization Variables

  Specify the name of a sequential data set to contain the
  customization variables, then press Enter to continue. If the
  data set does not exist, the dialog displays the Allocate New
  Data Set panel, with which you can allocate a data set.

 Data set name: 'G5CELL.WP.NODEC.SAVECFG'
```

Provided new value here

Customized jobs written out to the target data sets

We took the default allocation parameters when they were presented to us.

Note: When coming up with names for your data sets, keep a few things in mind:
- Have a separate set of migration data sets for each node being migrated
- The high-level qualifier should indicate in some fashion the node being migrated
- The high-level qualifier for the CNTL, DATA and SAVECFG data sets for a node should be the same

- We exited the customization dialogs and did a data set list on our high-level qualifier. We saw our new SYSC node data sets and the other two nodes as well:

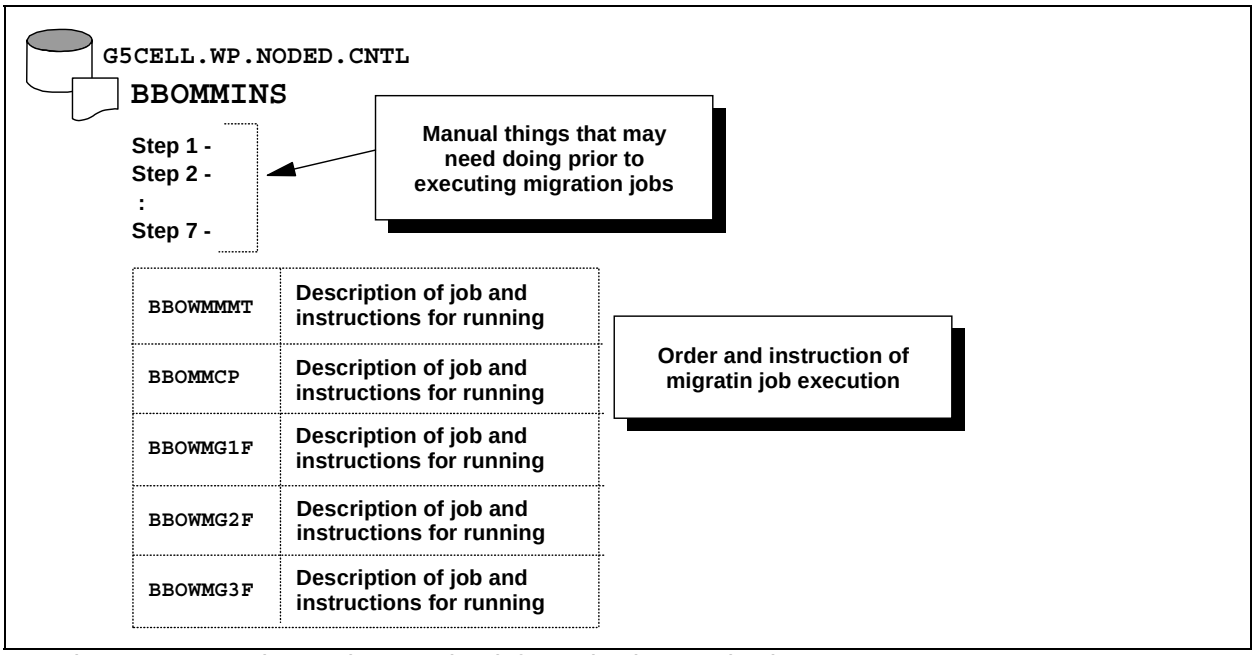
```
DSLIST - Data Sets Matching G5CELL.WP.**
Command ===>

Command - Enter "/" to select action
----------------------------------------
         G5CELL.WP.DMGR.CNTL
         G5CELL.WP.DMGR.DATA
         G5CELL.WP.DMGR.SAVECFG
         G5CELL.WP.NODEC.CNTL
         G5CELL.WP.NODEC.DATA
         G5CELL.WP.NODEC.SAVECFG
         G5CELL.WP.NODED.CNTL
         G5CELL.WP.NODED.DATA
         G5CELL.WP.NODED.SAVECFG
```

Just created during this customization run

Customized data sets for the migration of the SYSC application server node

Reviewed instruction member BBOMMINS in CNTL data set

The BBOMMINS member of the CNTL data set contains a list of instructions for migrating the node, and looked almost identical to the instructions created for the application server node on SYSD. The instructions are organized in the following manner:



How the BBOMMINS instruction member information is organized

Important: You should review the contents of that member. Do not rely exclusively on this document. In particular, those instructions speak of what needs to be done to your MVS system so WebSphere Application Server for z/OS Version 6 may operate. (Things like APF authorization and HFS auto-mount instructions.)

Important note -- please read

You may run through the ISPF customization dialogs on any MVS system in the Sysplex.

But the migration jobs ***must*** be run on the same MVS image on which the servers of the migrated node will run.

For our application server node G5NODEC, that meant the jobs were run on the SYSC system.

Made sure Deployment Manager was up and running

☐ We checked to make sure the Deployment Manager was up and running, and that it was at the V6 level.

Note: This is a very important prerequisite to migrating a federated node. During migration, the migration utility will establish a TCP connection to the SOAP port of the running Deployment Manager and initiate a node synchronization process. The DMGR has to be there or the migration process fails. This, incidentally, is why the Admin ID and password was asked for in the "federated node" panels -- so if global security is enabled the SOAP connection to the DMGR can be properly authenticated.

Stopped all servers in the G5NODEC node

☐ Issued the following stop command on SYSC to stop the Node Agent:

```
/P G5AGNTC
```

☐ Issued the following stop command on SYSC to stop the G5SR01C application server:

```
/P G5SR01C
```

Note: We did *not* stop the Daemon -- that was already migrated to V6.

Ran customized jobs

The BBOWMMMT job

We did *not* run this job. The HFS was already allocated and mounted at the `/wasv6config/g5cell` mount point.

The BBOMMCP job

We did *not* run this job. All the procs that were used by the servers in the `G5NODEC` node (`G51ACR`, `G51ASR` and for the Daemon, `G51DEMN`) had already been copied into PROCLIB by the migration of the other nodes.

The BBOWMG1F job

Our servers did not have XA connectors installed, so running this job was not necessary.

> **Note:** For the migration of the application server node on SYSD we noted how running this utility would have been a bit tricky. See "The gathering storm -- the problem of shared procs, re-using proc names and STEPLIB" on page 32 for more on that topic.
>
> But for SYSC we would *not* have had that problem. Why? Because the servers here used the already-migrated Daemon. Stopping and restarting the servers in `G5NODEC` would not have been a problem.

The BBOWMG2F job

Since we didn't run the `BBOWMG1F` job, running this job was unnecessary.

The BBOWMG3F job

This job is what performs the migration.

- ☐ We insured the `JOB` card had a `USER=` and `PASSWORD=` so this job ran under the ID of the "WebSphere Admin ID" of `G5ADMIN`. This job must run under that ID.
- ☐ We submitted the job. This jobs takes a long time to complete, so be patient.
- ☐ Twenty minutes later the job had completed with all steps coming back `RC=0`

```
14.49.18 --- SUNDAY,    20 MAR 2005 ----
14.49.18 $HASP373 BBOWMG3F STARTED
14.49.19 Jobname  Procstep Stepname  CPU Time    RC
14.49.19 BBOWMG3F --None-- SETUP     00:00:00    00
14.49.19 BBOWMG3F --None-- WRCONFIG  00:00:00    00
14.49.19 BBOWMG3F --None-- WRRESP    00:00:00    00
14.49.32 BBOWMG3F --None-- MKCONFIG  00:00:00    00
14.49.33 BBOWMG3F --None-- VERIFY    00:00:00    00
14.49.35 BBOWMG3F --None-- CRHOME    00:00:00    00
14.50.37 BBOWMG3F --None-- PREUPGRD  00:00:00    00
14.53.05 BBOWMG3F --None-- CRPROF    00:00:00    00
15.06.27 BBOWMG3F --None-- UPGRADE   00:00:00    00
15.08.56 BBOWMG3F --None-- UPPROCS   00:00:00    00
15.09.39 BBOWMG3F --None-- FINISHUP  00:00:00    00
15.09.39 BBOWMG3F --None-- WROUT     00:00:00    00
15.09.40 BBOWMG3F --None-- WRERR     00:00:00    00
15.09.40 $HASP395 BBOWMG3F ENDED
```

Post-migration RACF work

The BBOMMINS instruction member speaks of two sets of post-migration RACF work that may be needed:

1. New STARTED profiles

   This is mentioned under "Step 7" of the pre-job instructions. If we had provided *new* JCL start procedure names this would have been necessary. But since we backed up our V5 JCL procs and copied in the V6 procs *with the same names*, no new STARTED profiles were needed.

   Note: For the controller JCL only. The BBOMMINS instruction member provides an example.

1. Permitting the servant region ID access to BPX.WLMSERVER.

   One of the instructions found under the BBOWMG3F job says the following:

   > *WebSphere for z/OS V6 requires that the servant's address space is authorized to interface with the z/OS Workload Manager. The following RACF commands must be issued to permit the Application Server's started task userid to have Read access to the WLM Server profile.*

   Since our application servers on SYSC and SYSD operated under the same servant ID -- G5ASRU -- there was no need to do this again. We had already done this after we had migrated the G5NODED node on SYSD.

Started servers on SYSC

We chose to start the Node Agent first. Node Agents provide the Deployment Manager a way to "see into" the status of a node. In truth, the sequence of start -- Node Agent or application server -- doesn't matter.

Node Agent

- ☐ Issued the start command for the Node Agent:

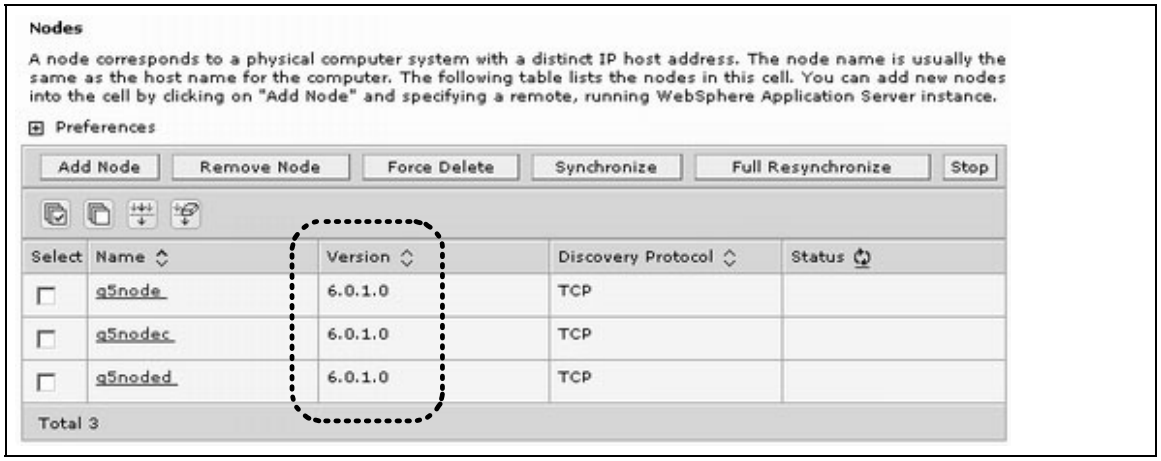
  ```
  S G51ACR,ENV=G5CELL.G5NODEC.G5AGNTC,JOBNAME=G5AGNTC
  ```

  Note: That is the same start command as was used when the Node Agent was at the V5.1 level of code.

- ☐ We found the following message in the Node Agent's controller SYSPRINT:

  ```
  WSVR0001I: Server CONTROL PROCESS nodeagent open for e-business
  ```

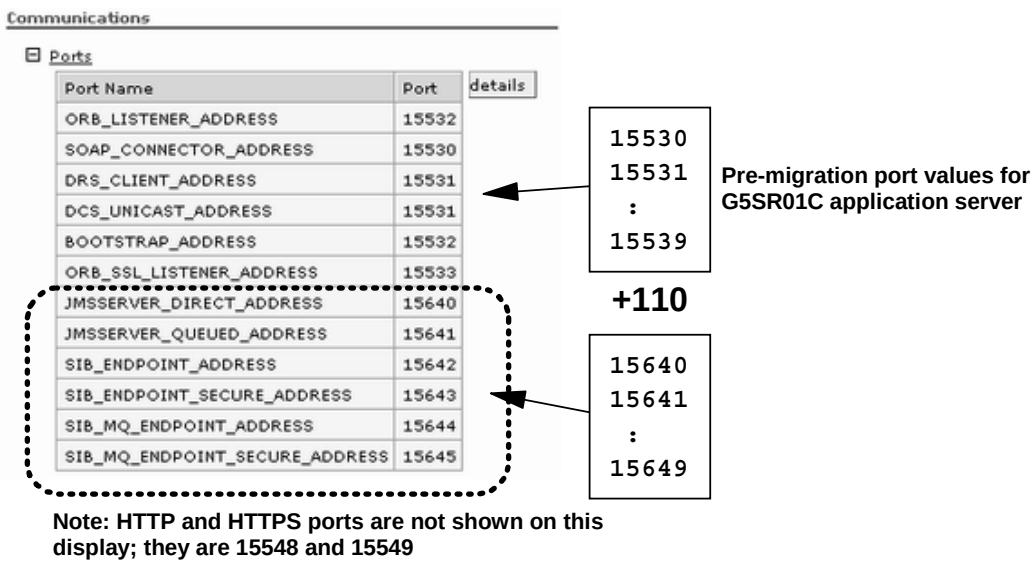
- ☐ We checked the Admin Console to see what it saw regarding its nodes:



Admin Console display for Nodes

As you can see, all nodes were at the V6 level now.

New ports created for V6 application server



Port values assigned to six new ports created for migrated application server

Again, when a server's ports are changed it is necessary to stop and restart the server. Since the application server was not yet started, all we needed to do was start the server.

Note: See "New ports created for V6 application server" on page 47 for more on the new ports created when an application server is migrated.

The change made in the Admin Console made its way out to the node because we had started the Node Agent. That allowed the changes to be synchronized with the node.

Application server G5SR01C

☐ Issued the start command for the `G5SR01C` application server on SYSC:

```
S G51ACR,ENV=G5CELL.G5NODEC.G5SR01C,JOBNAME=G5SR01C
```

We saw the message:

```
WSVR0001I: Server SERVANT PROCESS g5sr01c open for e-business
```

Status at this point

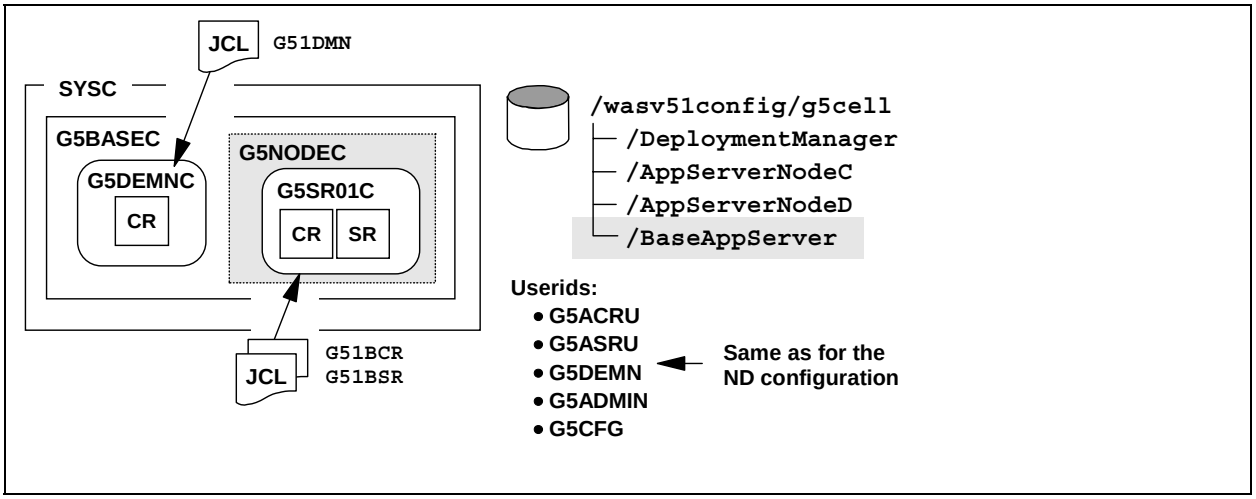
We were finished. Our `G5CELL` was now entirely migrated to V6.

Migrated a Base Application Server Node

A "Base Application Server node" is a standalone application server, with no Node Agent, no Deployment Manager; just a server and a Daemon. In V6, the terminology has been changed so it's now referred to as a "Standalone Server."

The process of migrating a "Standalone Server" is quite similar to that of migrating an application server node. The key difference is there's no Deployment Manager that needs to be up.

Picture of our Base Application Server node



Schematic diagram of G5BASEC migrated to V6

Note: Yes, you see that right -- this Base Application Server node configuration was jammed into the same configuration HFS as the Network Deployment one shown earlier. That's okay -- it illustrates the fact that to the migration utilities it's just a directory full of XML that'll get transformed and migrated.

We'll end up putting the migrated configuration into the `/wasv6config/g5cell` HFS as well. What that means is that we won't have to run the job that allocates and mounts the HFS -- it's already there from earlier.

Preliminary work

- ☐ Determined that we did not need to back up the HFS since this BaseApp server configuration was included in the single shared HFS we backed up earlier.
- ☐ Backed up the three JCL procedures related to this configuration: `G51BCR`, `G51BSR` and `G51DMN`.
- ☐ Performed visual inspection of `/tmp/migrate` directory to make sure migration job had proper write access.

  This configuration was using the same security infrastructure as the Network Deployment configuration, including the admin ID of `G5ADMIN`. So it had proper access.

Invoked ISPF dialogs and customized migration jobs

This was also relatively easy. We loaded the saved variables from the application server migration on SYSC and changed a few of the variables.

- ☐ Invoked V6's ISPF customization dialogs:

  ```
  ex 'WAS600.WAS.SBBOCLIB(BBOWSTRT)' 'APPL(MIG)'
  ```

  The `'APPL(MIG)'` provided a set of ISPF variables for this migration run related to the last run.

- ☐ Selected Option 4 from the primary option menu:

```
1  Configure a security domain.

2  Create stand-alone Application Server nodes.  You must complete
   Option 1 before starting this option.

3  Create Network Deployment cells and nodes.  You must complete
   Option 1 before starting this option.

4  Migrate V5.x Nodes to V6 Nodes.
```

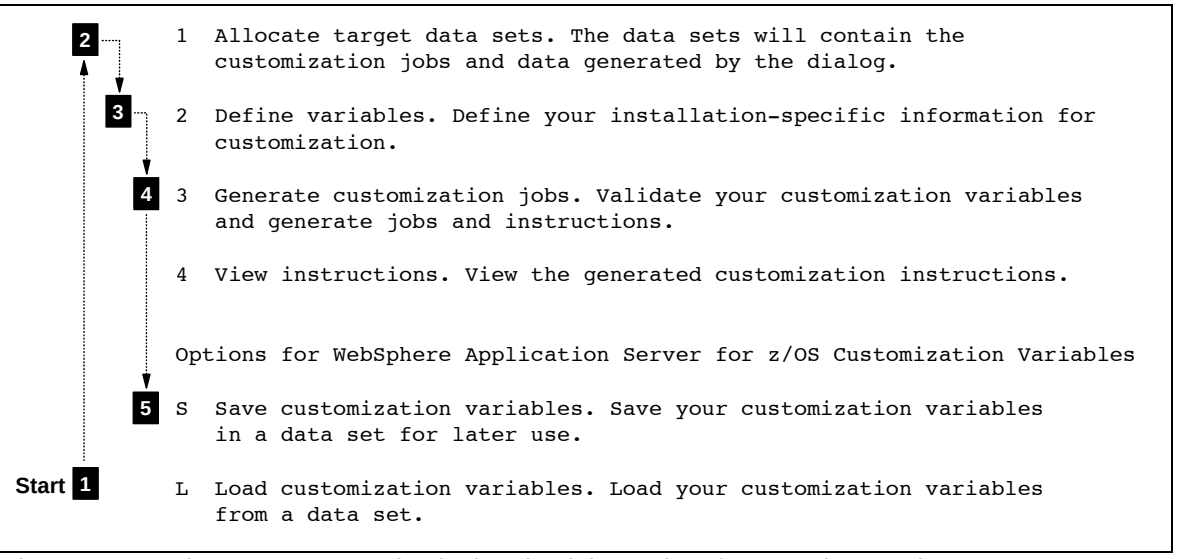
Option 4 -- used to invoke migration customization dialogs

- ☐ Selected Option 1, "Migrate a V5.x stand-alone application server node to V6":

```
1  Migrate a V5.x stand-alone application server node to V6.

2  Migrate a V5.x deployment manager to V6.

3  Migrate a V5.x federated node to V6.  You must migrate the cell's
   deployment manager to V6 before migrating any of the cell's
   federated nodes
```

Option 1 -- migrate a stand-alone application server node

- ☐ Was presented with the following menu. The processing through the menu took the following path, just as it was with the other nodes:



Options presented to create customized migration jobs; order of processing used

- Loaded the saved customization variables from the G5NODEC migration (the "SAVECFG" member in the G5CELL.WP.NODEC.* data set). We used Option "L" to do this:

```
Load Customization Variables

 Specify the name of a data set containing the customization variables,
 then press Enter to continue.

 IBM-supplied defaults are in  'WAS600.WAS.SBBOEXEC(BBOWVARS)'

 Data set name: 'G5CELL.WP.NODEC.SAVECFG'
```

Used the node from SYSC as an input model

G5NODEC's customization variables loaded into BaseApp migration

Note: In truth, any of the other SAVECFG files would have worked equally as well. We chose NODEC simply because the BaseApp server is also on SYSC.

- Allocated the target data sets where the customized jobs were stored. These were *different* from the ones created for G5NODEC:

```
Allocate Target Data Sets

Specify a high level qualifier (HLQ) and press Enter to allocate the
data sets to contain the generated jobs and instructions. You can
specify multiple qualifiers (up to 39 characters).

High level qualifier: G5CELL.WP.STAND                          .CNTL
                                                               .DATA

The dialog will                                  panels. You can make
changes to the d                                 you should not change
the DCB characteristics of the data sets.

   .CNTL  - a PDS with fixed block 80-byte records to
            contain customization jobs.

   .DATA  - a PDS with variable length data to contain
            other data produced by the customization dialog.
```

Provided new value here

Allocated target data sets

We took the default allocation parameters.

Note: This resulted in the creation of two data sets: G5CELL.WP.STAND.CNTL (a FB 80 PDS where the JCL was stored), and G5CELL.WP.STAND.DATA (a VB 255 PDS where scripts were stored). The "WP" qualifier stood for "White Paper."

- We selected Option 2, "Define Variables."
- We now faced three steps in the customization of the migration jobs:

```
1 - System Locations (directories, HLQs, etc.)
2 - System Environment Customization
3 - Server Customization
```

We worked through them in order: 1, then 2 and finally 3.

- The variables for "System Locations 1 of 2" were the same as those used for all the other nodes. We made no changes to the values on this panel.

```
System Locations (1 of 2)

   Specify the following V6 information, then press ENTER to continue.

   For some data sets, specify "Y" if they are in STEPLIB.

 Full Names of Data Sets

   PROCLIB.:  SYS1.PROCLIB

   Run WebSphere Application Server from STEPLIB (Y/N)?  Y
   SBBOLPA.:  WAS600.WAS.SBBOLPA
   SBBOLOAD:  WAS600.WAS.SBBOLOAD
   SBBOLD2.:  WAS600.WAS.SBBOLD2

                                                      Use STEPLIB?
   SCEERUN.:  SYS1.LEMVS.SCEERUN                            Y
   SCEERUN2:  SYS1.LEMVS.SCEERUN2                           Y
   SGSKLOAD:  SYS1.CRYPTO.SGSKLOAD                          Y
             (leave SGSKLOAD blank if all systems are at z/OS 1.6 or above)
```

Variables supplied for "System Locations 1 of 2"

Note: Use or not use `STEPLIB` according to your local preference.

- The variables for "System Locations 2 of 2" were also the same as for the Deployment Manager:

```
System Locations (2 of 2)

   Specify the following, then press Enter to continue.

 V6 WebSphere Application Server product directory:
     /u/bagwell/g6inter
```

Variables supplied for "System Locations 2 of 2"

As explained earlier, what this panel is asking for is the directory mount point of the V6 SMP/E HFS; in other words, where `WAS600.WAS.SBBOHFS` was mounted. We entered a value of `/u/bagwell/g6inter`, which represents an "intermediate symbolic link" we created that pointed to the real V6 product directory.

Note: See the WP100396 white paper on `www.ibm.com/support/techdocs` for more on the use of "intermediate symbolic links" and their role in providing flexibility.

If we were truly interested in providing maximum flexibility, we would have used a separate "intermediate symbolic" link for each node. Had we done that, then the value we provided here for the BaseApp server would have been different than what we provided for `G5NODEC` (our input model).

But as it was, the `G5CELLC` was really a test cell -- intended to validate essential concepts -- and not serve as a true model for maximum flexibility and high-availability. So we had one common shared intermediate symbolic link for all nodes in the cell.

- The "System Environment Customization" variables were also the same as provided for G5NODEC:

```
System Environment Customization

   Specify the following to customize your system environment, then
   press Enter to continue.

 WebSphere Application Server for z/OS Configuration HFS Information

   Mount point....:  /wasv6config/g5cell
   Name...........:  OMVS.WAS6.G5CELL.MIGRATED.HFS
   Volume, or '*' for SMS.:  *
   Primary allocation in cylinders...:  200
   Secondary allocation in cylinders.:  50
```

"System Environment Customization" variables: the HFS file system information

Remember, we were putting *all* our migrated configurations in the same HFS. Therefore, the BBOWMBMT job generated was *not* run -- there was no need to: the HFS was already allocated and mounted.

Note: Even though we didn't plan to run BBOWMBMT, it was *still necessary* to provide the correct "Mount Point" value on this panel. This is an important variable used throughout the migration jobs.

If we had planned to have this node in a *separate* HFS, then a different mount point value and HFS data set name value would have been required. But since we were using a common, shared HFS all we needed to do was maintain the same values we used all along.

- With the "Server Customization 1 of 2" variables we need to be very careful. This is where things will be very different for each node being migrated. Here's what we entered:

```
Server Customization (1 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.

   V5.x WebSphere Application Server home directory:
      /wasv51config/g5cell
         / BaseAppServer      1

   V6 WebSphere Application Server home
      /wasv6config/g5cell
         / StandAppServer     2

   Migration Options

     Enable z/OS Migration Tracing:  N
     Enable WASProfile Tracing....:  N
     Enable WASPreUpgrade Tracing.:  N
     Enable WASPostUpgrade Tracing:  N

     Default Backup Directory:  /tmp/migrate/5843/dmgr_backup
     User Specified Backup Directory:
```

Provided new values here

"Server Customization" variables: source and target node directories

Notes:

1. This was the mount point and node directory for the V5 Base Application Server node. We overwrote the value that was carried in when we loaded the other node's SAVECFG.

2. This was the intended node directory for the migrated V6 stand-alone application server node. Notice how the mount point (`/wasv6config/g5cell`) was not open for update. That value was the HFS mount point provided on the previous panel. We chose to adhere more closely to the V6 terminology - `StandAppServer`.

☐ The "Server Customization 2 of 2" variables were next:

```
Server Customization (2 of 2)

   Specify the following to customize your migration, then press Enter
   to continue.


 High Availability Manager Host:  9.82.24.71   1
     The High Availability Manager Host MUST resolve to a single
     IP address.  It can not be a multihomed host.

   Daemon Procedure name.........:  G51DMN

   Controller Procedure name....:  G51BCR   2

   Servant Procedure name.......:  G51BSR
```

"Server Customization" variables: High Availability Manager and JCL procedure names

Notes:

1. The "High Availability Manager Host" is a new function of V6 servers, which is why the migration panels are asking for information. The field on the panel will accept either a host name (`www.name.com`) or an IP address (`9.82.24.71`). As you can see on the panel, if you enter a host name, it must resolve to a single IP address.

   For the SYSC system on which this node resided, *all our hosts were multi-homed*. So we were forced to entered the IP address of SYSC MVS image. Our input `SAVECFG` file (`G5NODEC`) carried in the correct value, so we didn't need to change this.

   Note: For a stand-alone server with only one server instance, the need for uniqueness of this value within the cell being defined is somewhat moot. But it does have to resolve to the system on which the server will run. For SYSC, that was `9.82.24.71`

2. New V6 JCL procedures will be created and placed in the *`hlq`*`.CNTL` target data set. For our cell, the BaseApp server on SYSC used `G51BCR` and `G51BSR`, and a Daemon proc of `G51DMN`. We opted to use the same names, though the contents of the JCL would be different.

- With the variables defined, we worked our way back to the main panel and selected Option 3 to "Generate customization jobs." The panel we received looked like this:

```
Generate Customization Jobs

 This portion of the Customization Dialog generates the jobs you must
 run after you complete this dialog process. You must complete the
 customization process before you generate the jobs with this step.
 If you have not done this, ple

 Jobs and data files will get g
   'G5CELL.WP.STAND.CNTL'
   'G5CELL.WP.STAND.DATA'
 If you wish to generate custom                    ets, then
 exit from this panel and selec                    s" option.

 All the jobs that will be tailored for you will need a job card.
 Please enter a valid job card for your installation below. The
 file tailoring process will update the job name for you in all the
 generated jobs, so you need not be concerned with that portion of
 the job cards below. If continuations are needed, replace the
 comment cards with continuations.

 Specify the job cards, then press Enter to continue.

 //jobname JOB (ACCTNO,ROOM),'BAGWELL',CLASS=A,REGION=0M
 //*
 //*
 //*
```

These data set values will be filled in automatically based on what you provided for the "Target Data Sets" earlier. But you should always check this to make sure it is truly where you want the jobs to be placed.

Generating the customization jobs

- When we hit the Enter key the jobs were written out to the target data sets:

```
Processing for data set 'G5CELL.WP.STAND.CNTL' ...
Member BBOWMG1B successfully created.
Member BBOWMG2B successfully created.
Member BBOWMG3B successfully created.
Member BBOWMBMT successfully created.
Member BBOMBCR successfully created.
Member BBOMBCRZ successfully created.
Member BBOMBDN successfully created.
Member BBOMBDNZ successfully created.
Member BBOMBSR successfully created.
Member BBOMBSRZ successfully created.
Member BBOMBCP successfully created.
Member BBO6CRA successfully created.
Member BBO6CRAZ successfully created.
Member BBOMBINS successfully created.

Processing for data set 'G5CELL.WP.STAND.DATA' ...
Member BBOWBBPT successfully created.
Member BBOWMBRF successfully created.
***
```

Customized jobs written out to the target data sets

- Then we selected Option S and saved the customization variables to a *different* SAVECFG data set than used for the G5NODEC node, which was our input file:

```
Save Customization Variables

  Specify the name of a sequential data set to contain the
  customization variables, then press Enter to continue. If the
  data set does not exist, the dialog displays the Allocate New
  Data Set panel, with which you can allocate a data set.

 Data set name: 'G5CELL.WP.STAND.SAVECFG'
```

Provided new value here

Customized jobs written out to the target data sets

We took the default allocation parameters when they were presented to us.

Note: When coming up with names for your data sets, keep a few things in mind:
- Have a separate set of migration data sets for each node being migrated
- The high-level qualifier should indicate in some fashion the node being migrated
- The high-level qualifier for the CNTL, DATA and SAVECFG data sets for a node should be the same

- We exited the customization dialogs and did a data set list on our high-level qualifier. We saw our new data sets as well as all the others:

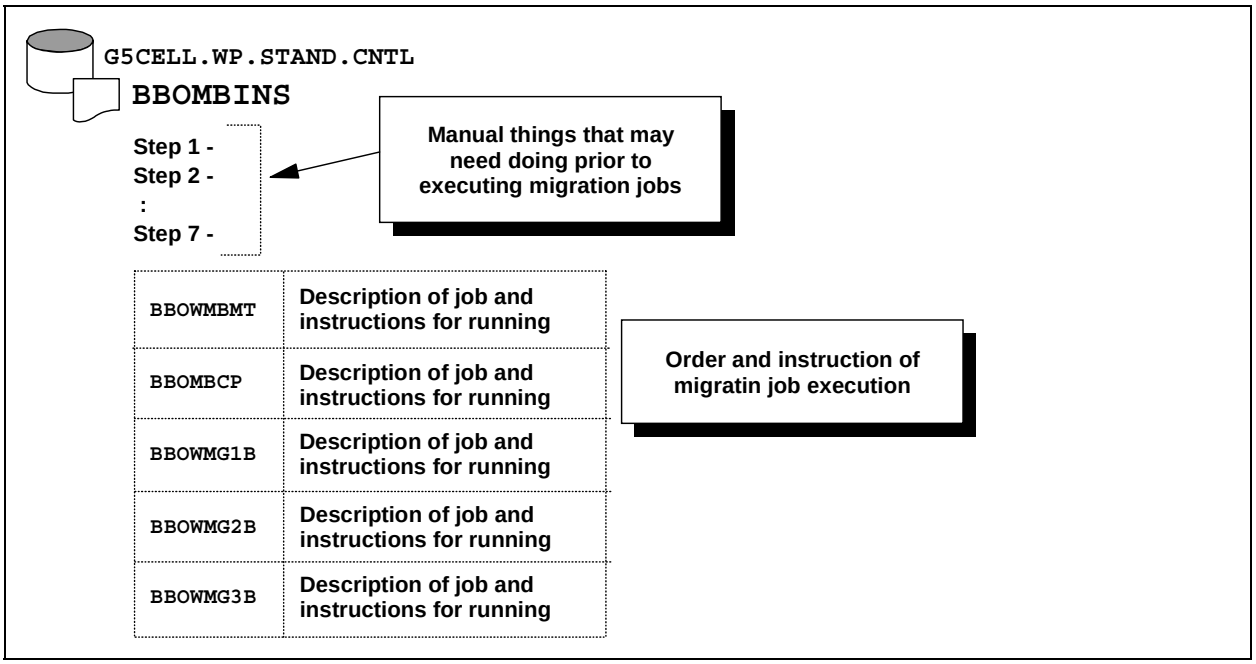
```
DSLIST - Data Sets Matching G5CELL.WP.**
Command ===>

Command - Enter "/" to select action
----------------------------------------
         G5CELL.WP.DMGR.CNTL
         G5CELL.WP.DMGR.DATA
         G5CELL.WP.DMGR.SAVECFG
         G5CELL.WP.NODEC.CNTL
         G5CELL.WP.NODEC.DATA
         G5CELL.WP.NODEC.SAVECFG
         G5CELL.WP.NODED.CNTL
         G5CELL.WP.NODED.DATA
         G5CELL.WP.NODED.SAVECFG
         G5CELL.WP.STAND.CNTL
         G5CELL.WP.STAND.DATA
         G5CELL.WP.STAND.SAVECFG
```

Just created during this customization run

Customized data sets for the migration of the BaseApp server node

Reviewed instruction member BBOMBINS in CNTL data set

The BBOMBINS member of the CNTL data set contains a list of instructions for migrating the BaseApp node, and looked almost identical to the instructions created for the application server node on SYSC or SYSD. The instructions are organized in the following manner:



How the BBOMMINS instruction member information is organized

Important: You should review the contents of that member. Do not rely exclusively on this document. In particular, those instructions speak of what needs to be done to your MVS system so WebSphere Application Server for z/OS Version 6 may operate. (Things like APF authorization and HFS auto-mount instructions.)

Important note -- please read

You may run through the ISPF customization dialogs on any MVS system in the Sysplex.

But the migration jobs ***must*** be run on the same MVS image on which the servers of the migrated node will run.

For our Base Application server node, that meant the jobs were run on the SYSC system.

Stopped server and Daemon

To migrate a Base Application Server node (or now called a "Stand Alone Server") it was necessary to stop all the servers, including the Daemon.

- ☐ Issued the following command to stop the Daemon:

  `/P G5DEMNC`

  The Daemon came down, along with `G5SR01C` server.

  Note: This is a *separate* Daemon from the one used in the Network Deployment configuration earlier. It's defined under a difference cell.

Ran customized jobs

The BBOWMBMT job

We did not run this job since our target HFS was already mounted from before.

The BBOMBCP job

- We added a USER= and PASSWORD= value to the JOB card so it would run under a userid we knew to have the authority to copy members into PROCLIB. The job was submitted and it completed with RC=0.

The BBOWMG1B job

We did not run this job since our Base Application Server did not have any XA connectors.

The BBOWMG2B job

We did not run this job since we did not run BBOWMG1B. Running this 2B is only necessary if you ran 1B.

The BBOWMG3B job

This job is what performs the migration.

- We insured the JOB card had a USER= and PASSWORD= so this job ran under the ID of the "WebSphere Admin ID" of G5ADMIN. This job must run under that ID.
- We submitted the job. This jobs takes a long time to complete, so be patient.
- Twenty three minutes later the job had completed with all steps coming back RC=0

```
15.18.35 ---- TUESDAY,   22 MAR 2005 ----
15.18.35  $HASP373 BBOWMG3B STARTED - INIT 1     SYS SYSC
15.18.35  Jobname  Procstep Stepname  CPU Time     RC
15.18.35  BBOWMG3B --None-- SETUP     00:00:00     00
15.18.35  BBOWMG3B --None-- WRCONFIG  00:00:00     00
15.18.36  BBOWMG3B --None-- WRRESP    00:00:00     00
15.18.44  BBOWMG3B --None-- MKCONFIG  00:00:00     00
15.18.45  BBOWMG3B --None-- VERIFY    00:00:00     00
15.18.46  BBOWMG3B --None-- CRHOME    00:00:00     00
15.35.53  BBOWMG3B --None-- CRPROF    00:00:00     00
15.37.18  BBOWMG3B --None-- PREUPGRD  00:00:00     00
15.39.57  BBOWMG3B --None-- UPGRADE   00:00:00     00
15.40.57  BBOWMG3B --None-- UPPROCS   00:00:00     00
15.41.41  BBOWMG3B --None-- FINISHUP  00:00:00     00
15.41.43  BBOWMG3B --None-- WROUT     00:00:00     00
15.41.43  BBOWMG3B --None-- WRERR     00:00:00     00
15.41.43  $HASP395 BBOWMG3B ENDED
```

Post-migration RACF work

The BBOMDINS instruction member speaks of two sets of post-migration RACF work that may be needed:

1. New STARTED profiles

   This is mentioned under "Step 7" of the pre-job instructions. If we had provided *new* JCL start procedure names this would have been necessary. But since we backed up our V5 JCL procs and copied in the V6 procs *with the same names*, no new STARTED profiles were needed.

   Note: For the controller JCL only. The BBOMBINS instruction member provides an example.

1. Permitting the servant region ID access to BPX.WLMSERVER.

   One of the instructions found under the BBOWMG3F job says the following:

   WebSphere for z/OS V6 requires that the servant's address space is authorized to interface with the z/OS Workload Manager. The following RACF commands must be

issued to permit the Application Server's started task userid to have Read access to the WLM Server profile.

Since this Standalone Server was using the same userids and groups as the Network Deployment Configuration from earlier, we did not need to run this again. But if this Standalone server had a different set of userids, then we would have run these commands:

```
PE BPX.WLMSERVER CLASS(FACILITY) ID(<servant ID>) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Started Standalone server

- ☐ Issued the start command for the `G5SR01C` application server:

```
S G51BCR,ENV=G5CELLC.G5NODEC.G5SR01C,JOBNAME=G5SR01C
```

Mapped new ports created for V6 application server

This process is just as was described under "New ports created for V6 application server" on page 47.

Stopped server and restarted (to use re-mapped ports)

In order to pick up the changes made to the ports, the server had to be stopped and restarted. The Daemon could stay up during this process.

- ☐ Issued the stop command for the `G5SR01C` application server:

```
P G5SR01C
```

- ☐ Issued the start command for the `G5SR01C` application server:

```
S G51BCR,ENV=G5CELLC.G5NODEC.G5SR01C,JOBNAME=G5SR01
```

Other Information

Known issues

- When performing V5.0 to V5.1 Migration, it is recommended you ***not*** have any region constraints in place, such as IEFUSI limits, as these can cause unpredictable JVM errors. The IEFUSI in place can cause a limit on the amount of region space allowed by a system which can limit memory in a fashion of which the user is not aware. This memory limit could stop the JVM from initializing completely during servant startup, which can account for the Shasta error:

  ```
  BBOO0072E Shasta Runtime function loadAndInitVM detected
  ```

Falling back to V5 (or recovering a migrated node)

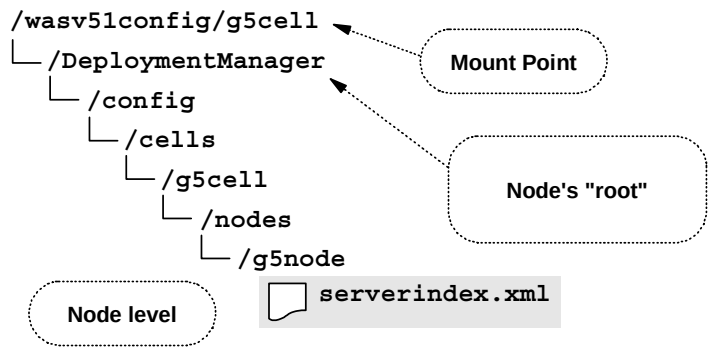
Changes made to the V5 directory

When the `BBOWMG3D/F` job is run, it makes two changes to the source node configuration structure:

> **Note:** This information applies only to Deployment Manager nodes or federated application server nodes. It does not apply to a Standalone, or "BaseApp," server node.

1. It changes `serverindex.xml` file name to `serverindex.xml_disabled`.

   Every node has a `serverindex.xml` file. It's located in the "node level" directory for that node. Here's an example of where the `serverindex.xml` file is for the G5CELL's Deployment Manager node:

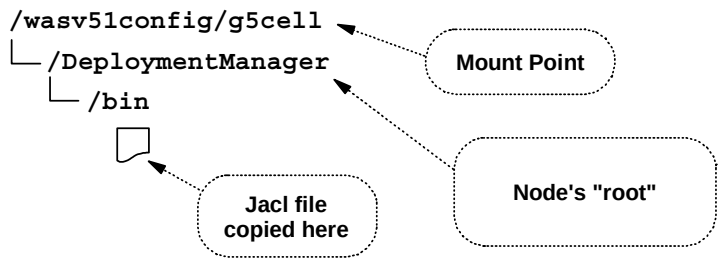


Location of the node's `serverindex.xml` file

2. It copies a Jacl script into that node's `/bin` directory. That Jacl script is a simple input file for WSADMIN script interface processing. The Jacl script is called:

   `migrationDisablementReversal.jacl`

   Every node has a `/bin` directory. It's located right under the node's root directory. Here's an example of where the `/bin` directory is for the G5CELL's Deployment Manager node:



Location of the node's `/bin` directory

The effect of the `serverindex.xml` being renamed is that it renders the V5.x configuration unable to start. The symptom you'll see if you try is a "`serverindex.xml` file not found" error message. So restarting a V5 configuration after it has been migrated involves renaming that `serverindex.xml_disabled` file back to just `serverindex.xml`.

Restoring a migrated configuration -- the manual method

The process is relatively simple:

- ☐ *Stop V6 servers* -- they can't be up at the same time the V5.x copy of the server is up.
- ☐ *Unmount V6 configuration HFS* -- it's best not to have this around to confuse matters. You may wish to delete the HFS file system altogether after unmounting.
- ☐ *Restore procs* -- copy back to PROCLIB the V5.x JCL you backed up earlier.
- ☐ *Change `serverindex.xml` file name* -- go into the "node level" directory of each node that was migrated and rename `serverindex.xml_disabled` to `serverindex.xml`
- ☐ *Start V5.x servers* -- as you normally would

Restoring a migrated configuration -- the "automated" method

Also relatively simple:

- ☐ *Stop V6 servers* -- they can't be up at the same time the V5.x copy of the server is up.
- ☐ *Unmount V6 configuration HFS* -- it's best not to have this around to confuse matters. You may wish to delete the HFS file system altogether after unmounting.
- ☐ *Restore procs* -- copy back to PROCLIB the V5.x JCL you backed up earlier.
- ☐ *Invoke WSADMIN to rename `serverindex.xml`* -- this involves running the WSADMIN scripting interface and pointing the the `migrationDisablementReversal.jacl` file as input:
  - From a Telnet or OMVS session, go to the `/bin` directory of your V5 node
  - Issue the following command:

    ```
    ./wsadmin.sh -f migrationDisablementReversal.jacl -conntype NONE
    ```

    Notes:
    - This is a very simple Jacl script that simply renames the `serverindex.xml` back to its proper name.
    - It's imporant to be in the proper `/bin` directory. That Jacl script will operate on the "node level" copy of `serverindex.xml` in whatever node's `/bin` directory you invoke the WSDAMIN shell script. Pay attention.
    - It's important to have the `-conntype NONE` designation on that
- ☐ *Start V5.x servers* -- as you normally would

Is there a significant different between the two methods

No. One is a manual rename of the `serverindex.xml_disable`, one is "automated."

When the BBOWMG3 job fails*

The `BBOWMG3*` job has 13 steps, each of which much run with `RC=0` for the migration to be successful. What happens if the job runs part-way and fails, leaving an incomplete migration? It's relatively simple. Do the following:

- ☐ Under the V6 configuration HFS mount point, delete the node's "root" (or "home") directory *and all subdirectories*.

  Important: Pay *close attention* to where you are and what you delete when you do this step. Make sure you're the V6 HFS and not the V5, and make sure you delete the node you wish to delete and not some other already-migrated and working node.

  For instance, a Deployment Manager node's home will be `/DeploymentManager`. For the G5CELL the application server node on SYSC was `/AppServerNodeC`.

  Note: The point is this: you don't have to delete the whole HFS ... just the failed node's home.

- ☐ Fix whatever caused `BBOWMG3*` to fail.
- ☐ Resubmit the `BBOWMG3*` job.

Document Change History

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

March 23, 2005	Original document.
March 24, 2005	Clarified the section on restoring a V5 configuration, and added a section on what to do when the `BBOWMG3*` job fails part-way through.
May 18, 2005	Added section titled "Known issues" on page 73 to capture things we come across as more people perform the migration. First item added was a recommendation that no region size constraints (such as IEFUSI exits) be in place when you attempt to run the migration.

End of Document WP100559