Customizing z Systems Development and Test Environment for z/OS ADCD December 2016 Edition

Contents

Customizing the December 2016 Edition z/OS 2.2 ADCD for z Systems Development and Test Environment

Learn how to customize the z/OS® 2.2 Application Developers Controlled Distribution for IBM® z Systems™ Development and Test Environment December 2016 edition.

This z/OS 2.2 distribution is a unique distribution that was created specifically for sites that are entitled to IBM z Systems Development and Test Environment, based on the Application Developers Controlled Distribution (ADCD). It is referred to in this IBM Knowledge Center as the z/OS 2.2 ADCD. It contains many of the products and subsystems you expect in a z/OS system. Most of these products and subsystems run without any customization, and some need some customization or initialization to make them work. These customizations are centered on changes that are needed to establish TCP/IP connectivity and update the infrastructure to allow z/OS 2.2 to properly run on IBM z Systems Development and Test Environment.

Restriction: Any z/OS ADCD for z Systems Development and Test Environment, and any related software, are supplied as a convenience. IBM does not support the operating system distributions that are made available with z Systems Development and Test Environment. You can apply fixes to any z/OS distribution through normal SMP/E facilities.

Before customizing this z/OS 2.2 ADCD, you must first install IBM z Systems Development and Test Environment, and download and install the z/OS 2.2 ADCD. For instructions, see Installing December 2016 Edition z/OS 2.2 ADCD for z Systems Development and Test Environment . Then, to customize the z/OS 2.2 ADCD distribution, do the steps that are described in the next three topics. The last two topics contain reference information about the z/OS 2.2 ADCD distribution.

Defining the device map for z/OS 2.2 ADCD

The IBM z Systems Development and Test Environment allows the customization of the z Systems resources available within the virtualized environment. The resources can be defined in a device map or *devmap*.

The sections and syntax of device maps are explained in detail in the "1090 Control Files" section of the zPDT Guide and Reference.

The system volumes that represent the z/OS distribution are defined in an `awsckd` stanza, and are mounted at arbitrarily chosen addresses within the range of valid addresses that are defined by the z/OS distribution's IODF. Historically in a z/OS ADCD, the xxRES1 and xxSYS1 volumes are mounted at the same addresses that are documented in the zPDT Guide and Reference, and address 0AA3 is always reserved for volume xxDBAR.

The following examples, when combined, define a working device map with this z/OS ADCD for IBM z Systems Development and Test Environment. If you are using the coupling facility or Rational® Tokens, see Enabling the coupling facility and Setting up Rational Tokens.

```
[system]
    memory 4096m
    processors 3
    3270port 3270               # port number for non-SNA (coax) 3270

    [manager]
    name aws3274 0001           # define non-SNA (coax) 3270 terminals
    device 0700 3279 3274 mstcon
    device 0701 3279 3274 tso1
```

For the network adapter definitions, the following example was added. One OSA is defined for connectivity to the host Linux system, called a "tunnel" OSA, and one is defined for establishing remote connectivity by using an ethernet adapter on the Linux host computer. Your setup might differ. For more information about configuring OSAs, see "Configuring a working z/OS system" on page 6 and the sections on connectivity in the zPDT Guide and Reference.

```
[manager]          # define network adapter (OSA) for communication with Linux
    name awsosa 0024 --path=A0 --pathtype=OSD --tunnel_intf=y # QDIO mode
    device 400 osa osa
    device 401 osa osa
    device 402 osa osa

    [manager]          # define network adapter (OSA) for communication with network
    name awsosa 22 --path=F0 --pathtype=OSD    # QDIO mode
    device 404 osa osa
    device 405 osa osa
    device 406 osa osa
```

Since the volumes are in the `/home/ibmsys1/z1090/disks/` directory, the DASD definitions might look like the following example:

```
[manager]
    name awsckd 0002
    device 0A80 3390 3390 /home/ibmsys1/z1090/disks/C2RES1
    device 0A81 3390 3390 /home/ibmsys1/z1090/disks/C2CFG1
    device 0A82 3390 3390 /home/ibmsys1/z1090/disks/C2SYS1
    device 0A83 3390 3390 /home/ibmsys1/z1090/disks/C2C511
    device 0A84 3390 3390 /home/ibmsys1/z1090/disks/C2C521
    device 0A85 3390 3390 /home/ibmsys1/z1090/disks/C2C531
    device 0A86 3390 3390 /home/ibmsys1/z1090/disks/C2DBB1
    device 0A87 3390 3390 /home/ibmsys1/z1090/disks/C2DBB2
    device 0A88 3390 3390 /home/ibmsys1/z1090/disks/C2DIS1
    device 0A89 3390 3390 /home/ibmsys1/z1090/disks/C2DIS2
    device 0A8A 3390 3390 /home/ibmsys1/z1090/disks/C2IMD1
    device 0A8B 3390 3390 /home/ibmsys1/z1090/disks/C2IME1
    device 0A8C 3390 3390 /home/ibmsys1/z1090/disks/C2IMU1
    device 0A8D 3390 3390 /home/ibmsys1/z1090/disks/C2KAN1
    device 0A8E 3390 3390 /home/ibmsys1/z1090/disks/C2PAGA
    device 0A8F 3390 3390 /home/ibmsys1/z1090/disks/C2PAGB
    device 0A90 3390 3390 /home/ibmsys1/z1090/disks/C2PAGC
    device 0A91 3390 3390 /home/ibmsys1/z1090/disks/C2PRD1
    device 0A92 3390 3390 /home/ibmsys1/z1090/disks/C2PRD2
    device 0A93 3390 3390 /home/ibmsys1/z1090/disks/C2PRD3
    device 0A94 3390 3390 /home/ibmsys1/z1090/disks/C2BLZ1
    device 0A95 3390 3390 /home/ibmsys1/z1090/disks/C2RES2
    device 0A97 3390 3390 /home/ibmsys1/z1090/disks/C2USS1
    device 0A98 3390 3390 /home/ibmsys1/z1090/disks/C2USS2
    device 0A99 3390 3390 /home/ibmsys1/z1090/disks/C2W801
    device 0A9A 3390 3390 /home/ibmsys1/z1090/disks/C2W802
    device 0A9B 3390 3390 /home/ibmsys1/z1090/disks/C2W803
    device 0A9C 3390 3390 /home/ibmsys1/z1090/disks/C2W804
    device 0A9D 3390 3390 /home/ibmsys1/z1090/disks/C2W805
    device 0A9E 3390 3390 /home/ibmsys1/z1090/disks/C2W851
    device 0A9F 3390 3390 /home/ibmsys1/z1090/disks/C2W852
    device 0AA0 3390 3390 /home/ibmsys1/z1090/disks/C2W853
    device 0AA1 3390 3390 /home/ibmsys1/z1090/disks/C2W854
    device 0AA2 3390 3390 /home/ibmsys1/z1090/disks/C2W855
    device 0AA3 3390 3390 /home/ibmsys1/z1090/disks/C2DBAR
    device 0AA4 3390 3390 /home/ibmsys1/z1090/disks/C2W901
    device 0AA5 3390 3390 /home/ibmsys1/z1090/disks/C2W902
    device 0AA6 3390 3390 /home/ibmsys1/z1090/disks/SARES1
    device 0AA9 3390 3390       # Available for dynamic mounts
    device 0AAA 3390 3390       # Available for dynamic mounts
    device 0AAB 3390 3390       # Available for dynamic mounts
```

Figure 1. Device map example. Device map example

You can verify the device map with the `awsckmap` command. All of the disk volume images that are referenced in the device map must exist before you verify the device map. Assume that the device map file has the following name:

```
/home/ibmsys1/z/myDeviceMap
```

With the name in the preceding example, the device map can be verified with the command:

```
awsckmap /home/ibmsys1/z/myDeviceMap
```

Sample program to create device map

A sample program that is called `create_devmap.pl` is available in the `ConfigGuideSample` directory where you installed z Systems Development and Test Environment. If Perl is installed on your Linux system, you can use the `create_devmap.pl` program to generate a sample device map that is based on your current decompressed 3390 disk images, memory configuration, and available network parameters.

Consider the output of `create_devmap.pl` to be a starting point from which you can create a final device map. The syntax for the `create_devmap.pl` command is shown here:

```
perl <pathtocommand>/create_devmap.pl pathtodisks > generateddevmap
```

In the preceding example, *pathtocommand* is the location of the `create_devmap.pl` file, and *pathtodisks* is the location of your 3390 disk images. *generateddevmap* is the name of the file to contain the new device map.

If you already have a static IP address that is assigned for your virtual z/OS machine and a z/OS host name that can be resolved to that address by Linux, such as through a connected Domain Name Server or static configuration, you can add the `-h <hostname>` parameters after the `pathtodisks` parameter. Adding the parameter causes the script to attempt to generate comments that contain more accurate TCP/IP configuration samples based on your network.

```
perl <pathtocommand>/create_devmap.pl pathtodisks -h
       hostname_of_zos > generateddevmap
```

The `create_devmap.pl` program creates a `memory` line based on existing hardware and configuration of your Linux machine. Verify that the amount of memory that is requested is appropriate for your situation.

The device map that is created by `create_devmap.pl` defines OSA devices based on the first Tun/Tap and Wired CHPIDs found that use the **find_io** command, and a set of sample z/OS TCP/IP definitions that would correspond to the OSA device definitions in the generated device map. These TCP/IP configuration statements can be used as a starting point for your TCP/IP configuration, but probably require changes to match your network. Verify that the device addresses and device names in your final VTAM® definitions, TCP/IP profile, and device map all correspond to the correct network adapter types.

The device map that is created by `create_devmap.pl` also contains 3390 device statements for files in the *pathtodisks* directory that are over 800 MB, have six-character names, and are verified to be disk images by the `alcckd` command.

Starting and stopping the z/OS 2.2 ADCD

Learn how to start and stop the z/OS 2.2 ADCD.

z Systems Development and Test Environment is started and stopped, and the z/OS system is IPLed, by using commands that are entered on a Linux console that is running under the user ID that is configured to run z Systems Development and Test Environment.

Start the z Systems Development and Test Environment

The sample `runzpdt` script available in the `ConfigGuideSample` directory can be used to start z Systems Development and Test Environment with any z/OS distribution.

This script is described in Starting the z Systems Development and Test Environment.

When you run this z/OS 2.2 ADCD, the load parameter that is specified with the -l switch (the -l is a lowercase L) must correspond to a valid load parameter defined in the distribution. For information about the load parameters available with this

distribution, and which major subsystems they start, see "z/OS 2.2 ADCD for z Systems Development and Test Environment" on page 20.

Note: Each loadparm in this z/OS ADCD has an associated parmlib member that is named COMMND*xx*, where *xx* corresponds to a LOAD*xx* loadparm. That parmlib member defines all of the subsystems that are started when that loadparm is IPLed. The COMMND*xx* parmlib member runs an SVTAM*xx* procedure, which uses a VTAM*xx* parmlib member to start another set of subsystems. You can see all of the subsystems that are started for all loadparms by looking at the COMMND*xx* and VTAM*xx* parmlib members in ADCD.*.PARMLIB.

The first time this z/OS distribution is started, start z Systems Development and Test Environment with the CS loadparm. This loadparm initiates a cold start, and does not start other subsystems. After you verify basic z/OS capabilities, you can then customize the system by using the instructions in "Configuring a working z/OS system" on page 6 and create and start different loadparms as needed. As with any z/OS system, warm starts are less disruptive and preserve the JES job spool. Initiate® a warm start when possible.

When you perform an IPL of the supplied z/OS distribution system for the first time, you might see some errors during the IPL process. For example, SMF files might be full. If you are prompted with the IXC420D REPLY I TO INITIALIZE SYSPLEX ADCDPL, . . .message, respond with R 0,I, that is, the letter R, a blank, a zero, a comma, and the letter I.

Stopping the z Systems Development and Test Environment

If possible, always shut down z/OS cleanly. Typically, shutting down cleanly begins by logging off all TSO users, and starting a procedure that shuts down all active subsystems.

This z/OS 2.2 ADCD contains shutdown scripts for most systems available and active in the distribution. They are named SHUT*xx*, where *xx* corresponds to the VTAM*xx* parmlib member that was used to start the subsystems for a particular IPL.

Note: Use SHUT00 when the z/OS 2.2 ADCD was started with loadparm CS or WS, and SHUTALL when the z/OS 2.2 ADCD was started with loadparm AL. When an **S SHUTxx** command is issued from the MVS™ master console, these scripts shut down all systems that were automatically started by the corresponding VTAM*xx* parmlib member. These shutdown scripts are contained in ADCD.*.PROCLIB, where * varies according to the release started.

While the shutdown scripts attempt to stop all active subsystems started by that loadparm, some subsystems might require further input to be stopped. You might have to respond to z/OS console messages, such as when IMS™, TSO, or z/OS UNIX are stopped. You can see what programs are still running by entering the D J,L console command. Ensure VTAM and all subsystems are stopped.

After you run the appropriate SHUT*xx* script and all systems are stopped, stop your JES system. The loadparms in your z/OS 2.2 ADCD distribution always start JES2. A **$pjes2** command that is entered from the MVS console stops JES2.

After JES ends, z Systems operation can be stopped by entering the **QUIESCE** command. After this command completes, stop the zPDT® system by entering this command in the Linux window:

```
$ awsstop
```

This command produces several messages. It might be necessary to press **Enter** to obtain the Linux prompt. Any 3270 windows can be closed.

Configuring a working z/OS system

Learn how to configure a working z/OS 2.2 ADCD system for z Systems Development and Test Environment.

These topics guide you through procedures to help you make customizations to a z/OS 2.2 ADCD. These customizations include establishing TCP/IP connectivity to your z/OS system, increasing your `/tmp` and `/u` directories, and optional tasks such as creating new TSO user IDs.

A brief introduction to z/OS system configuration

In the simplest case, z/OS is configured by changing partitioned data set (PDS) members in the data sets `SYS1.PROCLIB`, `SYS1.PARMLIB`, and a few other important data sets including site-specific partitioned data sets.

Most configuration (parmlib) member names consist of a predefined name with a two character suffix added. A common convention is to refer to the suffix as *xx*, so you often see references to `LOADxx`, `IEASYSxx`, and so on. Configuration files refer to other members by a keyword and suffix number. For example, a member that is called `IEASYSRC` might identify the main z/OS UNIX configuration file with the line `OMVS=RC`, which means that z/OS UNIX finds the member that contains its configuration parameters by starting with a predetermined name, `BPXPRM`, and add the suffix `RC` (resulting in member name of `BPXPRMRC`). The keywords in the configuration files *are not* usually the same as the member name prefixes.

Important: Do not directly update data sets such as `SYS1.PARMLIB` and `SYS1.PROCLIB`. Some installation-specific libraries are searched before the `SYS1` data sets, leaving the `SYS1` libraries with IBM supplied defaults.

This z/OS 2.2 ADCD defines several sets of alternative libraries. The configurations for the distribution itself are stored in a set of libraries that start with the qualifiers ADCD.*. The set of libraries that are used to store z Systems Development and Test Environment configurations start with the qualifiers FEU.*. This z/OS distribution also provides a set of libraries for you to use, which start with the high-level qualifier USER. This z/OS distribution is already set up to read from most of the USER libraries before the FEU.* libraries and the ADCD.* libraries. The USER.* libraries are initially empty, and you can use them for your own changes. They are on a separate volume, C2CFG1. Avoid updating the ADCD.* and SYS1.* libraries whenever possible. If you choose to place data in the USER data sets, or create other user data sets or customizations in xxCFG1, these data sets can be easily migrated to a new system. Since volume names change with each release of a z/OS ADCD for z Systems Development and Test Environment, add the *xx*`CFG1` volume to the devmap for the new system, and then copy any customized files back to the newer xxCFG1 volume.

The customizations described here use `USER.*` libraries, which tend to be commonly available in any z/OS distribution. While you follow these instructions, all changes are made in `USER.*` libraries, except for adding LOADPARM members to `SYS1.IPLPARM`.

Note: Subsystem configuration changes, such as CICS® definitions, also change original libraries such as the CICS CSD.

The IPL process: LOADxx and IEASYSxx members

When z/OS is started, it looks in data set `SYS1.IPLPARM` for a member called `LOADxx`. The `xx` value is specified in the IPL statement of the start script that was created. A `LOADxx` member defines various settings to start the system, such as the parmlib concatenation that indicates which data sets are to be searched for other configuration members. The `LOADxx` member also defines which `IEASYSxx` member is to be used. `IEASYSxx` is considered the starting point for system configuration because it contains pointers to other parmlib members that are used during the IPL process.

Tip: Create a loadparm that starts the system without pointing to any alternative libraries that contain customized parmlibs, proclibs, and so on. By creating a loadparm if you make a mistake in your customizations that causes z/OS to not start, you can try to perform an IPL of the system with the loadparm that does not point to the alternative libraries. This method gives you TSO access, where you can modify the customizations to correct any mistakes.

If you make a mistake that causes z/OS not to start, you can try to perform an IPL of the system with CS or WS as the last two digits of the LOADPARM. These suffixes start the system with a simpler configuration. CS initiates a cold start, which clears the JES spool, and WS initiates a warm start. For example, to use a LOADPARM of CS, change the IPL command in your startup script to **`ipl 0a80 parm 0a82cs`**.

In an emergency, you can perform an IPL of the stand-alone system, which provides a basic system, but enables you to change your configuration data sets, because it does not share anything with the normal z/OS setup. The stand-alone system does not contain TCP/IP or Unix system service support. The system cannot be used to install products or apply maintenance through SMP/E. When you use the stand-alone system, you must specify the volume name of data sets you want to edit, since all volumes other than the residence volume for the stand-alone system are not cataloged. Do not edit the stand-alone system configuration. You can perform an IPL of the stand-alone system by mounting volume SARES1 and replacing the IPL statement with **`ipl 0A98 parm 0A98SA`**. Replace **`0A98`** with the address of volume SARES1. Use **`S SHUTSA`** to shut down the SARES1 system.

PROCLIB: System procedure library

Parmlib members contain only configuration information. The procedures that start the various subsystems and servers are found in a different concatenation called PROCLIB. As an example, this z/OS 2.2 ADCD contains `SYS1.PROCLIB`, `ADCD.*.PROCLIB`, `FEU.*.PROCLIB`, and `USER.PROCLIB` in its PROCLIB concatenation. To include any PROCLIB data set in the PROCLIB concatenation, you must modify the MSTJCLxx member of parmlib and your JES procedure.

For more information, see "Modifying TCP/IP procedures to point to `USER.TCPPARMS`" on page 14 and Enabling use of `USER.PROCLIB` and IPLing.

Configuration overview

Learn how to configure the z/OS system so that you isolate the customized data in your z/OS distribution volumes and establish TCP/IP communications.

The following steps are described:

- Create new z/OS UNIX file systems for /tmp and /u.
- Customize TCP/IP settings to establish network connectivity.

You might also want to make some additional changes that are commonly made.

- Create TSO user IDs.
- Customize ISPF defaults and the ISPF main panel
- Change console defaults
- Streamline startup and shutdown scripts
- Create an NJE connection to existing z/OS systems

Place these customizations on the single volume C2CFG1 volume that already contains your USER.* libraries. Placing these customizations on a single volume helps with future migrations to new versions of z/OS ADCD distributions.

Logging on to TSO

After you IPL, you can use the x3270 emulator on the host Linux to start a TSO session.

Enter this command:

```
x3270 -port 3270 localhost &
```

This example shows an alternative format of the x3270 command, which produces a larger screen size:

```
x3270  -model 4 localhost:3270 &
```

You can also use an emulator that is not native to the host Linux. For more information, see "Logging on to TSO from emulators not native to the host Linux" on page 17.

Log on to TSO with a valid user ID in your z/OS distribution. For this distribution, log on with the IBMUSER account, with initial password IBMUSER. You are prompted to change the password. This account has the administrative RACF® privileges that you need for certain customizations. You might want to make a few minor changes to your ISPF session before you start working. If you are accustomed to using TSO naming conventions in ISPF, then to ensure that you do not write out files with unexpected high-level qualifiers, enter the command

```
TSO PROFILE PREFIX(tso user id)
```

Creating new zFS files for /tmp and /u

Management of z/OS UNIX file systems in z/OS is a complex area. This procedure provides a simplistic design for new file systems. Considerations such as space requirements and alternative mount points might require a more in-depth plan.

This z/OS ADCD provides fairly small file systems for the /tmp and /u directories. Small file systems can cause problems, particularly when you are installing z/OS software by using SMP/E, or when programs create large memory dumps on the z/OS UNIX file system. Allocate 100 primary cylinders and 20 secondary cylinders for these file systems. If the current file systems for /tmp and /u are not large enough, replace them. To create new file systems for /tmp and /u, create two new zFS files and associate these new files with the /tmp and /u directories. Catalog these two new zFS files on the C2CFG1 volume with your other customizations. Since a user catalog named USERCAT.Z22C.USER is on the C2CFG1 volume of the z/OS 2.2 ADCD as shipped, it can be used for cataloging the two new zFS files. By

avoiding use of the master catalog, during migration to a new z/OS ADCD, you then need only to import the user catalog and replicate the alias definitions so that catalog entries are restored. Use this procedure for creating new zFS files for /tmp and /u:

- Create aliases for the high-level qualifiers of the zFS files
- Create the zFS files
- Modify the appropriate BPXPRMxx parmlib member to mount the new zFS files with the /tmp and /u directories
- Copy the existing /tmp and /u directories to the new /tmp and /u directories that are mounted with the new zFS files
- Optionally create a new load parm to contain these changes
- Shut down the system and reIPL

The next topics describe the procedures for each step, including sample JCL where appropriate. In these procedures, the two z/OS UNIX file systems are created with a high-level qualifier of CUST on volume C2CFG1.

Creating ALIASes for the high-level qualifiers of the zFS files

Start by creating an alias for CUST and any other qualifiers that are used to create data sets here. Create the alias and the file systems on a volume with your other customizations so that it is easy to migrate to a new z/OS distribution.

When you create new users, also create an alias for the user ID in the same catalog. This sample JCL shows you how.

```
//Q12ALIAS JOB (ACCT),MSGCLASS=H,NOTIFY=&SYSUID.
//*-------------------------------------------------
//DEFALIAS EXEC PGM=IDCAMS,REGION=0M
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
    DEFINE ALIAS (NAME(CUST) RELATE(USERCAT.Z22C.USER))
/*
```

Creating the zFS files

Use the JCL in this topic to create two new zFS files on volume C2CFG1 named CUST.ZFS.U and CUST.ZFS.TMP.

Adjust the space allocations as needed.

```
//Q13ZFS JOB 'IBMUSER',CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H
//ZFSALLOC EXEC PGM=IDCAMS,REGION=0M
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
    DEFINE CLUSTER( -
             NAME(CUST.ZFS.TMP) -
             VOLUME(C2CFG1) -
             LINEAR -
             CYL(100 20) -
             SHAREOPTIONS(3) -
                  )
    DEFINE CLUSTER( -
             NAME(CUST.ZFS.U) -
             VOLUME(C2CFG1) -
             LINEAR -
             CYL(100 20) -
             SHAREOPTIONS(3) -
                  )
/*
//ZFSFORMT EXEC PGM=IOEAGFMT,REGION=0M,COND=(0,LT,ZFSALLOC),
//           PARM='-aggregate CUST.ZFS.TMP -compat'
```

```
//SYSPRINT DD SYSOUT=*
//ZFSFORMT EXEC PGM=IOEAGFMT,REGION=0M,COND=(0,LT,ZFSALLOC),
//            PARM='-aggregate CUST.ZFS.U -compat'
//SYSPRINT DD SYSOUT=*
```

Modifying BPXPRM00 parmlib member

After you create CUST.ZFS.TMP and CUST.ZFS.U, copy ADCD.Z22C.PARMLIB(BPXPRM00) into USER.PARMLIB.

Comment out the existing mount of the zFS for tmp, and replace it with a mount of CUST.ZFS.TMP. For example,

```
/* MOUNT    FILESYSTEM('ZFS.&ADCDLVL..TMP') */
/*          TYPE(HFS)                        */
/*          MODE(RDWR) NOAUTOMOVE             */
/*          MOUNTPOINT('/&SYSNAME./tmp')     */

MOUNT    FILESYSTEM('CUST.ZFS.TMP')
         TYPE(ZFS)
         MODE(RDWR) NOAUTOMOVE
         MOUNTPOINT('/&SYSNAME./tmp')
```

Make a similar change for the mount of /u:

```
/* MOUNT    FILESYSTEM('ZFS.&ADCDLVL..USERS') */
/*         TYPE(HFS)                 */
/*         MODE(RDWR) NOAUTOMOVE     */
/*          MOUNTPOINT('/u')         */

MOUNT    FILESYSTEM('CUST.ZFS.U')
         TYPE(ZFS)
         MODE(RDWR) NOAUTOMOVE
         MOUNTPOINT('/u')
```

Copying the existing /tmp and /u directories

Finally, you must copy the existing /tmp and /u directories to contain the new /tmp and /u file system, and ensure that everyone has proper access permission to the new file systems.

The z/OS UNIX command line can be accessed with the TSO OMVS command, and then you can enter these commands. Be sure to verify that each command works correctly before you proceed to the next command.

```
cd /
mkdir /tempmnt
/usr/sbin/mount –f cust.zfs.tmp /tempmnt
cp -r /tmp/ /tempmnt
ls /tempmnt
chmod 777 /tempmnt
/usr/sbin/unmount /tempmnt
/usr/sbin/mount -f cust.zfs.u /tempmnt
cp -r  /u/ /tempmnt
ls /tempmnt                 # to verify that the copy worked
chmod 777 /tempmnt
/usr/sbin/unmount /tempmnt
rm -r /tempmnt
```

Creating a loadparm for customizations

You might want to isolate your customizations into a new loadparm. For example, you can create an RC loadparm that uses your new zFS files for /tmp and /u.

- Instead of modifying BPXPRM00, copy it and rename it to BPXPRMRC, and modify BPXPRMRC.

- Copy the IEASYSxx member that currently starts the subsystems you want to start to USER.PARMLIB. Rename it to IEASYSRC to match the two character loadparm of BPXPRMRC. Update the **OMVS=** parameter to replace the existing two character loadparm with RC.
- Make a copy of the existing LOADxx member in SYS1.IPLPARM that currently starts the subsystems you want to start. Rename it to LOADRC, and store it back into SYS1.IPLPARM. Modify the **SYSPARM** value to RC.

Shutting down the system and re-IPLing

Shut down the system and perform an IPL. Use your loadparm that points to the modified BPXPRMxx, or your newly created load parm.

When the system restarts, you can enter the z/OS UNIX command line and enter the **df -k** command to ensure that your file systems are being used and mounted correctly.

After the shutdown and re-IPL is complete, you have larger zFS files for /tmp and /u.

Setting up TCP/IP

z/OS running on z Systems Development and Test Environment can communicate with your network by using TCP/IP. You can use standard 3270 terminal emulators, FTP, Developer for z Systems, and other services to move data to and from your z/OS system.

Since mainframes are confined to data centers, TCP/IP on z/OS does not act as a DHCP client. It does not automatically configure itself to a TCP/IP address supplied by the network. Therefore, it is necessary to configure a few settings to get TCP/IP to communicate with the network. Several methods of configuring TCP/IP are described in zPDT Guide and Reference.

This example uses the method referred to as Scenario 4 in Chapter 7, “LANs” in the zPDT Guide and Reference. This method allows the z/OS system and the Linux system to communicate with your network, each with their own IP address and with the ability to share the ethernet adapter. By each having their own IP address, both systems can simultaneously use functions such as FTP with minimal configuration change. It also eliminates the need to create firewall *holes* for your z/OS ports or provide Network Address Translation functions in your firewall.

The examples show the changes that you must make to the TCP/IP data sets in your z/OS distribution to gain that TCP/IP network connectivity. The examples assume that existing TCP/IP parmlib members and procedures are copied as new members into your USER.* data sets, and modified. By using this method, the parmlib members and procedures of your original z/OS distribution are unmodified. Alternatively, you can change the existing TCP/IP procedures and parmlib members without creating new members.

TCP/IP and LAN configuration is site-dependent. The exact steps that are outlined here might not work at your site because of local network configuration, firewalls, Linux dependencies, or hardware restrictions. You might need the services of a network administrator to configure z/OS within your network.

z/OS 2.2 ADCD network configuration

The supplied z/OS 2.2 ADCD by default activates member PROF1 in ADCD.Z22C.TCPPARMS as the TCPIP.PROFILE. It activates one OSA with device

name ADM1ETP, an IP address of 192.168.0.61, and a default route to 192.168.0.1. You need to change this configuration work in your network by either defining this OSA as a tunnel or defining this OSA to share a Linux ethernet adapter, are explained in the "LANs" chapter of zPDT Guide and Reference. In either case, with only one OSA activated, you have connectivity restrictions in this configuration. If you define an external OSA without a tunnel OSA, you cannot communicate to the Linux host. If you define a tunnel OSA without an external OSA, all traffic must come through the Linux host, and is subject to Linux routing restrictions, firewalls, and considerations when both Linux and z/OS use the same ports.

Setting up Linux routing

Before you configure TCP/IP, obtain a static IP address for z/OS. The z/OS IP address must be within the same subnet as your Linux machine. It does not matter if the Linux machine has a DHCP or static IP address if both the z/OS and Linux address are in the same subnet. This z/OS environment is configured to use both the static IP address you obtained and an address of 10.1.1.2. The address of 10.1.1.2 is used to communicate with the Linux machine and cannot be seen by other machines on your network. The following examples show how to configure z/OS so that the external network connects to the machine by address 9.12.200.20, and Linux connects to z/OS by address 10.1.1.2. z/OS can connect to the Linux machine that is using the address 10.1.1.1. The connection between z/OS and Linux is called a tunnel.

Note: You can use the `--tunnel_ip` parameter in the `awsosa` stanza to define a local address other than 10.1.1.1. In the customizations below, it is assumed the `--tunnel_ip` parameter is not defined and defaults to 10.1.1.1.

If you use the `create_devmap.pl` program that is described in "Sample program to create device map" on page 3, and you have a z/OS host name that can be resolved to your static z/OS IP address by Linux, such as through a connected Domain Name Server or static configuration, you can add the **`–h hostname`** parameter to the invocation of the program. This parameter causes `create_devmap.pl` to generate sample contents of z/OS host configuration files that are tailored to your network. It generates the TCPIP.DATA and TCPIP.PROFILE statements by using the z/OS IP address that corresponds to the entered `hostname`, and dynamically finds the name server addresses based on the Linux ethernet adapter that is found by using `find_io`. It also generates sample VTAMLST members to define two OSAs; a tunnel OSA and an OSA to be used with one of the Linux ethernet adapters. The contents are shown as comments at the end of the generated device map and are examples only. Because of the variability and complexity of network configuration, the samples that are shown in the generated device map might need modification to work in your environment.

This example assumes that you have the following OSA definitions in your devmap to define the tunnel OSA and one ethernet OSA to be used by TCP/IP:

```
[manager]          # define network adapter (OSA) for communication with Linux
name awsosa 0024 --path=A0 --pathtype=OSD --tunnel_intf=y   # QDIO mode
device 400 osa osa
device 401 osa osa
device 402 osa osa
[manager]          # define network adapter (OSA) for communication with network
name awsosa 22 --path=F0 --pathtype=OSD    # QDIO mode
device 404 osa osa
device 405 osa osa
device 406 osa osa
```

Modifying TCPPARMS files

Learn how to modify two TCPPARMS files to establish TCP/IP connectivity by using an IP address that is defined for z/OS: PROFILE.TCPIP and TCPIP.DATA.

PROFILE.TCPIP:

Copy ADCD.Z22C.TCPPARMS(PROF2) to USER.TCPPARMS(PROFILE) and modify the DEVICE, LINK, HOME, and route definitions to use the OSAs defined in your devmap, with the addresses and netmask information that is correct for your network. For example, given a z/OS IP address of 9.12.200.20, and a netmask of 255.255.255.0, a TCP/IP PROFILE member might look like the following example.

Note: Comments and other statements were removed for brevity. Also, the IP address of the Linux system is not needed here. The gateway address usually ends in .1 or .0, but that might be different on your network.

This example also includes the definitions for the tunnel, the 10.1.1.2 address, which is used to communicate with z/OS from the Linux machine.

```
ARPAGE 5
    DATASETPREFIX TCPIP
    AUTOLOG 5
        FTPD JOBNAME FTPD1   ; FTP Server
        PORTMAP              ; Portmap Server
    ENDAUTOLOG
    PORT
               ((( ports removed for brevity )))
    SACONFIG DISABLED
      DEVICE PORTA MPCIPA
      LINK ETH1  IPAQENET PORTA
      HOME 10.1.1.2 ETH1

      DEVICE PORTB MPCIPA
      LINK ETH2  IPAQENET PORTB
      HOME 9.12.200.20 ETH2

     BEGINROUTES
     ROUTE 10.0.0.0       255.0.0.0       =         ETH1   MTU 1492
     ROUTE 9.12.200.0     255.255.255.0   =         ETH2   MTU 1492
     ROUTE DEFAULT        9.12.200.1                ETH2   MTU 1492
     ENDROUTES
    ITRACE OFF
    IPCONFIG NODATAGRAMFWD
    UDPCONFIG RESTRICTLOWPORTS
    TCPCONFIG RESTRICTLOWPORTS
    START PORTA
    START PORTB
```

Copy ADCD.Z112S.VTAMLST(OSATRL2) to USER.VTAMLST(OSATRL2) and remove any comments within it so that it looks like the next example.

The device name in the TCP/IP profile member must match the port names that are specified in USER.VTAMLST(OSATRL2). In this example, these port names are PORTA and PORTB. Also, verify that your devmap correctly defines the device addresses in the READ, WRITE, and DATAPATH statements of USER.VTAMLST(OSATRL2). For more information about the device map, see "Defining the device map for z/OS 2.2 ADCD" on page 1.

```
OSATRL1 VBUILD TYPE=TRL
    OSATRL1E TRLE LNCTL=MPC,READ=(0400),WRITE=(0401),DATAPATH=(0402),      X
                  PORTNAME=PORTA,                                          X
```

```
                MPCLEVEL=QDIO
OSATRL2E TRLE LNCTL=MPC,READ=(0404),WRITE=(0405),DATAPATH=(0406),      X
                PORTNAME=PORTB,                                        X
                MPCLEVEL=QDIO
```

To activate this configuration, copy ADCD.Z112S.VTAMLST(ATCCON00) to USER.VTAMLST(ATCCON00) and change the word OSATRL1 to OSATRL2.

TCPIP.DATA:

Copy `ADCD.Z22C.TCPPARMS(TCPDATA)` to `USER.TCPPARMS` and set the `HOSTNAME`, `DOMAINORIGIN`, and `NSINTERADDR` values.

```
TCPIPJOBNAME TCPIP
    HOSTNAME RDT900
    DOMAINORIGIN  RTP.IBM.COM
    DATASETPREFIX TCPIP
     NSINTERADDR  9.0.0.1
     NSINTERADDR  9.0.0.11
    RESOLVEVIA UDP
    LOOKUP LOCAL DNS
    RESOLVERTIMEOUT 10
    RESOLVERUDPRETRIES 1
    ALWAYSWTO NO
```

The z/OS 2.2 ADCD ships with a HOSTNAME of S0W1 and a DOMAINORIGIN of DAL-EBIS.IHOST.COM. These work with the shipped WebSphere® Application Server configuration. If you chose to define a different HOSTNAME and DOMAINORIGIN, and used the create_devmap.pl program with the -h option and your chosen HOSTNAME to create your devmap, the output devmap will contain the customized TCPIP.DATA parameters for HOSTNAME and DOMAINORIGIN.

If you choose a HOSTNAME or DOMAINORIGIN arbitrarily, be sure that the DOMAINORIGIN is not a real domain name and that the combination of the HOSTNAME and DOMAINORIGIN does not constitute an existing DNS name. Use the Linux `ping` or `nslookup` commands to ensure that your choice of names is not found by your DNS server. Identifying your computer as another computer, or as a member of an existing but incorrect domain, can cause problems that are difficult to diagnose, such as timeouts, pauses, and connection failures in many areas, including 3270 connections and z Systems Development and Test Environment. Some systems, including components of z Systems Development and Test Environment, require that z/OS can locate itself by name.

Ensure that the `NSINTERADDR` parameter correctly identifies a DNS server. Incorrect name server specifications can also cause long delays and errors that are difficult to diagnose. Stand-alone systems should not specify `NSINTERADDR` values.

If you cannot use a Domain Name Server (DNS) to resolve IP addresses of other systems or of the z/OS system, you can create a local hosts file and refer to it with a GLOBALIPNODES statement. The setup of this file is described in detail in *TCP/IP implementation volume 1: Base functions* (SG24-7798).

Modifying TCP/IP procedures to point to `USER.TCPPARMS`

An easy way to use ISPF to find Procedures that reference the TCPPARMS data sets is to display a member list of each proclib data set in your active `MSTJCLxx` parmlib member (typically `MSTJCL00`), and then type these commands:

```
SRCHFOR TCPPARMS
    SORT PROMPT
```

Copy the following members from `ADCD.Z22C.PROCLIB` to `USER.PROCLIB`. Be careful not to replace any members you already changed in `USER.PROCLIB`.

```
FTPD
PORTMAP
TCPIP
RESOLVER
NFSC
NFSS
```

Modify each member to change references from the existing TCPPARMS data set to USER.TCPPARMS for any members that you duplicated in USER.TCPPARMS. Do not change the member names except as noted in the TCP/IP procedure.

For example, locate this line in FTPD:

```
//SYSTCPD DD DISP=SHR,DSN=ADCD..&SYSVER..TCPPARMS(TCPDATA)
```

Change it to

```
//*SYSTCPD DD DISP=SHR,DSN=CUST.TCPPARMS(TCPDATA)
    //SYSTCPD DD DISP=SHR,DSN=USER.TCPPARMS(TCPDATA)
```

Remember a modified TCP/IP profile member name was placed in USER.TCPPARMS, so the PROFILE DD statement is similar to this example:

```
//PROFILE  DD DISP=SHR,DSN=USER.TCPPARMS(PROFILE)
```

If you prefer to set up system-wide use of a common set of TCP/IP settings, set up the RESOLVER procedure to point to your TCP/IP definitions. Create member USER.TCPPARMS(RESOLVER) to contain the following statements.

```
GLOBALTCPIPDATA('USER.TCPPARMS(TCPDATA)')
    GLOBALIPNODES('USER.TCPPARMS(IPNODES)')
    COMMONSEARCH
```

Create member USER.TCPPARMS(IPNODES) to contain the following statements similar to the following but customized with your network names and address. If you used the create_devmap.pl program with the **-h** option to create your devmap, the output devmap contains the customized IPNODES file. The format of each entry in the IPNODES file is IP address followed by one or more host names. The order of the entries is not significant.

```
# This first entry is the ip address of the z/OS host with its associated hostnames
    # In this example, two hostnames are configured; a shortened alias and a fully
          qualified hostname.
    # Only one hostname is required, typically the fully qualified name
    #
    9.12.200.0   RDT900 RDT90.IBM.COM
    #
    # This entry is the ip address of the OSA tunnel to the Linux host.
    # The hostname should be the hostname of the Linux operating system.
    # Multiple hostnames may be configured if needed
    #
    10.1.1.1     LINUX
    #
    # Typically, 127.0.0.1 is configured as hostname LOCALHOST
    #
    127.0.0.1    LOCALHOST
```

Copy ADCD.Z22C.PROCLIB(RESOLVER) to USER.PROCLIB(RESOLVER), and change the SETUP DD statement from:

```
//SETUP DD DISP=SHR,DSN=ADCD.Z22C.TCPPARMS(GBLRESOL),FREE=CLOSE
```

To

```
//SETUP DD DISP=SHR,DSN=USER.TCPPARMS(RESOLVER),FREE=CLOSE
```

Modifying VTAMLST members:

Create a VTAMLST member to define a TRL major node that matches the OSAs you are using. Member OSATRL2 in ADCD.*.VTAMLST matches the sample devmap. If you create a new member to match your OSAs, add that member to USER.VTAMLST. To activate this major node when VTAM starts, add the VTAMLST member name to the VTAM configuration list (member ATCCONxx in VTAMLST) with which you start VTAM.

```
OSATRL1 VBUILD TYPE=TRL
OSATRL1E TRLE LNCTL=MPC,READ=(400),WRITE=(401),DATAPATH=(402),         X
               PORTNAME=PORTA,MPCLEVEL=QDIO
OSATRL2E TRLE LNCTL=MPC,READ=(404),WRITE=(405),DATAPATH=(406),         X
               PORTNAME=PORTB,MPCLEVEL=QDIO
```

Optional tasks

Learn about optional configuration tasks for z Systems Development and Test Environment.

Creating a user volume

Some users might want to create one or more user volumes to isolate configuration data for some of their products. For example, you might want to isolate all of your configuration files for IBM z Systems Development and Test Environment or you might want a new volume for a particular application or a newer version of an existing product. To create a new volume, do these following steps. This example creates a volume that is named USER00, placing it in the same directory as the other z/OS 2.2 ADCD disks:

- Create a volume in emulated 3390 format.

  ```
  alcckd /home/ibmsys1/z1090/disks/USER00 –d3390-9
  ```

  Note: You can use -d3390-3 or other volume types and sizes if wanted. See the alcckd command section of the
- Add the new volume to your device map

  ```
  device 0CCC 3390 3390 /home/ibmsys1/z1090/disks/USER00
  ```
- After you start IBM z Systems Development and Test Environment, initialize the new volume
  - Vary the volume offline

    ```
    V 0CCC,OFFLINE
    ```
  - Create and submit a member of a partitioned data set with the following JCL

    Note: Customize the job card and VTOC parameters as needed. The VTOC parameters for a 3390-3 and a 3390-9 are shown.

    ```
    //J01INIT JOB (ACCT),MSGCLASS=H,NOTIFY=&SYSUID.,REGION=0M
    //* MOD-3: VTOC(0,1,974) INDEX(65,0,50)
    //* MOD-9: VTOC(0,1,2939) INDEX(196,0,150)
    //FORMAT EXEC PGM=ICKDSF
    //SYSPRINT DD SYSOUT=*
    //SYSIN DD *
         INIT UNIT(CCC) NOVALIDATE NVFY VOLID(USER00) PURGE -
         VTOC(0,1,2939) INDEX(196,0,150)
    /*
    ```
  - If prompted on the console, reply U to the console command (R xx,U) to allow MVS to alter the contents of the volume. This job ends with return code 0.
  - Vary the volume online:

    ```
    V 0CCC,ONLINE
    ```

- Create a user catalog for the new volume by creating and submitting a member of a partitioned data set with the following JCL. Modify the job card as needed.

```
//J02CATLG JOB (ACCT),MSGCLASS=H,NOTIFY=&SYSUID.
//DEFCAT EXEC PGM=IDCAMS,REGION=0M
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DEFINE USERCATALOG (NAME(USERCAT.VUSER00) ICFCATALOG -
    CYLINDERS(10 5) VOLUME(USER00) )
/*
```

  Note: If no user catalog is used, all cataloged data sets are cataloged in the master catalog, which is replaced when you upgrade your z/OS system. To avoid placing entries in the master catalog, you can create aliases for high-level qualifiers for data sets that you want to catalog in the USER00 user catalog. During migration, you need only to import the user catalog and replicate the alias definitions so that catalog entries are restored.

Ensure that all new data sets are written to a new volume

You can increase storage for all new permanent data sets by adding one or more newly created volumes to the volume attribute list (VATLST*xx* member of the parmlib). The suffix *xx* of your volume attribute list parmlib member is the value found for the VAL= parameter in the IEASYSxx parmlib member corresponding to your loadparm. Copy the appropriate VATLSTxx member of the parmlib from ADCD.*.PARMLIB to USER.PARMLIB and add the volume to any existing definitions as shown. In this example, the new USER00 volume is added with a permanently resident mount attribute, a use attribute of storage, and mount messages to be entered. The contents of this member must be in specific columns, so be sure to use the copied contents as a model. See the VATLSTxx section of the z/OS 2.2.0 MVS Initialization and Tuning Reference for the format and explanation of all fields and parameters in the volume attribute list.

```
VATDEF IPLUSE(PRIVATE),SYSUSE(PRIVATE)
B2SYS1,0,0,3390      ,Y
USER00,0,0,3390      ,Y
```

Changes to VATLST00 will not be effective until you perform the next IPL.

Logging on to TSO from emulators not native to the host Linux

After TCP/IP connectivity is established, you can use your favorite 3270 emulator software, such as IBM Personal Communications Manager (PCOMM), to connect to the non-SNA (coax) 3270 device emulator that is provided by z Systems Development and Test Environment. When you connect from outside the Linux system that is hosting z Systems Development and Test Environment, use the Linux TCPIP address and 3270 as port. (The actual port number is defined in the device map.)

Creating TSO user IDs

Learn how to create TSO user IDs for running z Systems Development and Test Environment.

A z/OS ADCD for IBM z Systems Development and Test Environment provides a number of predefined user IDs. “z/OS 2.2 ADCD for z Systems Development and Test Environment” on page 20 contains the list of all predefined user IDs available to you.

If more TSO user IDs are required, they can be created through a series of commands. Typically, you want to use user IDs that have OMVS segments and an

associated z/OS UNIX user directory when you are running Developer for z Systems. The example commands that are shown here create a user ID in the RDZUSERS group with an OMVS segment. Replace #userid, #name, and #password with appropriate values, and do not remove the quotation marks in the commands.

From a CLIST, REXX exec, or TSO command line, enter the following commands. The commands create the user ID, provide an OMVS segment, and assign an account number, default logon procedure, and region size. They also protect data sets with a high-level qualifier that belongs to the user from being accessed by other users. Finally, the commands create an alias in the master catalog to indicate that the users data sets are cataloged in the user catalog on volume S1CUST. Be sure that the substitutions in the HOME( ) and PROGRAM() parameters are in lowercase.

```
ADDGROUP RDZUSERS OMVS(AUTOGID)
    ADDUSER #userid DFLTGRP(RDZUSERS) NAME('#name') PASSWORD(#password)
    ALTUSER #userid OMVS(HOME(/u/#userid) PROGRAM(/bin/sh) AUTOUID)
    ALTUSER #userid TSO(ACCTNUM(ACCT#) PROC(TSOLOGON) SIZE(4096) COMMAND('ISPF'))
    ADDSD '#userid.*' UACC(NONE)
    PERMIT ACCT# CLASS(ACCTNUM) ID(#userid) ACCESS(READ)
    DEFINE ALIAS (NAME('#userid') RELATE('USERCAT.Z22C.USER'))
```

The new user's z/OS UNIX directory must be created. From a z/OS UNIX command line, type the following commands. You can access z/OS UNIX by typing TSO OMVS on the command line of any ISPF screen. You exit z/OS UNIX with the exit command. Again, replace #userid with the name of the new user ID in lowercase.

```
mkdir /u/#userid
    chown #userid:RDZUSERS /u/#userid
```

Defining a new logon procedure

A common customization in z/OS systems is to alter the logon procedure that TSO users use. Do not alter the ISPFPROC logon procedure because errors might prevent you from being able to fix problems later on.

Create USER.PROCLIB(TSOLOGON) based on ADCD.Z22C.PROCLIB(ISPFPROC), and make your modifications to TSOLOGON instead of ISPFPROC. Be sure to change the identifier on the EXEC from ISPFPROC to TSOLOGON and to override the default VOLSER used by the ISPFCL CLIST.

```
//TSOLOGON EXEC PGM=IKJEFT01,REGION=0M,DYNAMNBR=175,
    //             PARM='%ISPFCL VOL(C2CFG1)'
```

To allow all users to use the TSOLOGON procedure, enter the following TSO commands.

```
RDEFINE TSOPROC TSOLOGON UACC(READ)
    SETROPTS RACLIST(TSOPROC) REFRESH
```

Altering system startup and shutdown scripts

This z/OS 2.2 ADCD has shutdown scripts that contain a series of commands to shut down all active subsystems.

You might want to make these changes to the scripts.

The shutdown scripts stop OMVS automatically. Explicitly stopping zFS can expedite the shutdown process, but requires a response to a prompt on the operator console. If you want to stop zFS before you shut down OMVS, add this change:

```
F OMVS,STOPPFS=ZFS
F OMVS,SHUTDOWN
```

Some startup and shutdown scripts have PAUSE commands that you might find are too long or too short for your system. Many can be safely changed to pause for a shorter amount of time.

You can change startup scripts the same way. Other changes might include not starting particular subsystems, changing pause times, and so forth.

Defining JES NJE connectivity

Your z/OS distribution is a stand-alone system with no connection to other z/OS systems. However, you might want to connect it to one or more of your z/OS systems to transfer data to customize and use the z Systems Development and Test Environment system.

Beginning with z/OS 1.7, JES supports NJE over TCP/IP, which makes setting up a connection between two systems an easy task. The following operator commands, to be run on the z Systems Development and Test Environment system, name the local system RUT0 and define a connection to M168.

```
$TNODE(Z21S),NAME=RUT0
    $TLINE1,UNIT=TCP
    $SLINE1
    $ADDNETSRV1,SOCKET=LOCAL
    $SNETSERV1
    $TNODE2,NAME=M168
    $ADDSOCKET(REMOTE),NETSRV=1,LINE=1,NODE=2,IPADDR=M168.RTP.IBM.COM
    $SN,SOCKET=REMOTE
```

Give similar operator commands on the M168 system to complete the setup. Since this system is an existing system, the command to define the local node name was skipped. The following commands also assume that M168 does not have spare line or node definitions, so new ones (line 5 and node 20) are created.

```
$ADDLINE5,UNIT=TCP
    $SLINE5
    $ADDNETSRV1,SOCKET=LOCAL
    $SNETSERV1
    $TNJEDEF,NODENUM=20
    $TNODE20,NAME=RUT0
    $ADDSOCKET(REMOTE),NETSRV=1,LINE=5,NODE=20,IPADDR=RDz9100.RTP.IBM.COM
    $SN,SOCKET=REMOTE
```

The z Systems Development and Test Environment system can use the existing NJE definitions on the M168 system to connect to other NJE nodes in your network. Enter the following operator commands on the z Systems Development and Test Environment system to connect to the IPO1 system (node 3) through the previously defined M168 system (node 2).

```
$TNODE3,NAME=IPO1
    $ADDCONNECT,NODEA=2,NODEB=3
```

For more information, see the "Network Job Entry" section of the *z/OS JES2 Initialization and Tuning Guide*.

z/OS 2.2 ADCD for z Systems Development and Test Environment

Learn what's new with this distribution from past z/OS distributions, the products that are included with this distribution, the volumes that can be downloaded and their general contents, the migration guidelines, including the parmlib and proclib structure of the distribution, the loadparms available with this distribution and which subsystems they start, the user IDs and passwords available with the system, and the maintenance service levels of the products.

Customization help and location of product PGMDIRs

Although the products in this ADCD were customized, you might find some products are not fully customized or are customized with options that you need to change.

Included in this release are files in the format of *p*.SVSC.l where *p* is the product high-level qualifier and l is the library name. The purpose of these files is to provide PGMDIRs, readme file, and installation instructions. This ADCD is built from a driver system that uses different volume and library names. Thus, the names that are referenced in the SVSC files are different from the ADCD. You might need to make translations on names to implement in the ADCD. Included on DVD1 on the RES1 volume are data sets with an HLQ of MVS.ZOSRxx that document MVS. The MVS PGMDIR is contained in this grouping.

What's new in this release

Learn which modifications and enhancements are incorporated into this z/OS 2.2 ADCD for z Systems Development and Test Environment December 2016 edition.

- z/OS 2.2 and related products were upgraded to PUT1606/RSU1607.
- The following products were removed:
  - IBM Ported Tools for z/OS V1.3
  - DB2® V10
  - CICS Interdependency Analyzer for z/OS V5.3
  - OMEGAMON® XE for DB2 Performance Expert
  - OMEGAMON XE on z/OS V5.3 (all but OMEGAMON Monitoring for JVM)
  - IBM InfoSphere® Optim™ Data Tools Runtime Client
- The following products are upgraded with the latest available release/mod level:
  - IBM Rational Team Concert™ V6.0.2
  - IBM Developer for z Systems V14.0.0
  - IBM Explorer for z/OS V3.0.1
  - IBM COBOL for z/OS V6.1.0
  - IBM Enterprise PL/I for z/OS V5.1.0
  - IBM IMS Enterprise Suite for z/OS, V3.2.1
  - IBM DB2 Administration tool for z/OS V11.2
  - IBM Debug for z Systems V14.0.0
  - CICS Transaction Gateway for z/OS V9.2
  - IBM DB2 Administration Tool for z/OS V11.2.0
  - IBM DB2 Object Comparison Tool for z/OS, V11.2.0
  - IBM Tivoli® OMEGAMON XE for IMS on z/OS V5.3.0
  - z/OS Log Forwarder (of OMEGAMON XE on z/OS Performance Management Suite for z/OS V5.4)

  - IBM Tivoli Asset Discovery for z/OS (TADz) ID (of OMEGAMON XE on z/OS Performance Management Suite for z/OS V5.4)
- The following products are added to the ADCD z/OS 2.2 December 2016 Edition:
  - WAS V9.0
  - Websphere MQ V9.0
  - IBM Workload Scheduler V9.3
  - OMEGAMON Monitoring for JVM V5.3
- All of z/OS base, z/OS products and middleware volume size was increased to mod-9 (10,017 cylinders) to provide more free space for expansion.

Products contained in this release

Lists the products contained in this release.

z/OS 2.2 December 2016 Edition

Lists the products available in the z/OS 2.2 December 2016 Edition.

Product Number	Description	Fmid	Dependent FMID	HLQ
5650-ZOS	GDDM-PGF	HGD3201	HGD3201	SYS1
5650-ZOS	High Level Assembler Toolkit	HMQ4160	JMQ416A	SYS1
5650-ZOS	Infoprint Server - IP PrintWay™ Base	HMOS705	HMOS705	SYS1
5650-ZOS	BULK DATA TRANSFER SNA NJE	HBD6602	JBD6202	SYS1
5650-ZOS	C/C++ HOST PERFORMANCE ANALYZER	H24P111	H24P111	SYS1
5650-ZOS	BULK DATA TRANSFER BASE	HBD6602	JBD6201	SYS1
5650-ZOS	RMF™ Base	HRM77A0	HRM77A0	SYS1
5650-ZOS	DFSORT Base	HSM1M00	HSM1M00	SYS1
5650-ZOS	SDSF Base	HQX77A0	HQX77A0	SYS1
5650-ZOS	Security Server - RACF Base	HRF77A0	HRF77A0	SYS1
5650-ZOS	XL C/C++ Base	HLB77A0	HLB77A0	SYS1
5650-ZOS	JES3	HJS77A0	HJS77A0	SYS1
5650-ZOS	Hardware Configuration Manager	HCM1F10	HCM1F10	SYS1
5650-ZOS	Infoprint Server - NetSpool Base	HNET7A0	HNET7A0	SYS1
5650-ZOS	Infoprint Server - Print Interface Base	HOPI7A0	HOPI7A0	SYS1

5650-ZOS	BOOKMANAGER READ/MVS - ENU	HBKM300	JBKM310	SYS1
5650-ZOS	ESCON Director	HSWF100	HSWF100	SYS1
5650-ZOS	GDDM NLS ENU	HGD3200	JGD3219	SYS1
5650-ZOS	OSA/SF	H0GI400	H0GI400	SYS1
5650-ZOS	Distributed File Service Base	H0H2410	H0H2410	SYS1
5650-ZOS	IBM Alternate Library for REXX on zSeries	HWJ9143	HWJ9143	SYS1
5650-ZOS	Metal C Runtime Library	HSD7780	HSD7780	SYS1
5650-ZOS	Integrated Security Services - Open Crypto Enhance	HRO7740	HRO7740	SYS1
5650-ZOS	GDDM NLS ENU	HGD3200	HGD3200	SYS1
5650-ZOS	Library Server	HBKQ400	HBKQ400	SYS1
5650-ZOS	Integrated Security Services - Enterprise Identity	HIT7750	HIT7750	SYS1
5650-ZOS	BDT	HBD6602	HBD6602	SYS1
5650-ZOS	SMP/E Base	HMP1J00	HMP1J00	SYS1
5650-ZOS	High Level Assembler Base	HMQ4160	HMQ4160	SYS1
5650-ZOS	z/OS Font Collection - Chinese, Japanese, Korean	HFNT11J	HFNT11J	SYS1
5650-ZOS	IOCP	HIO1104	HIO1104	SYS1
5650-ZOS	z/OS Font Collection	HFNT110	HFNT110	SYS1
5650-ZOS	BOOKMANAGER READ/MVS - ENU	HBKM300	HBKM300	SYS1
5650-ZOS	SMP/E Planning and Migration Assistant Software In	HBCNC00	HBCNC00	SYS1
5650-ZOS	FFST™	HFST101	HFST101	SYS1
5650-ZOS	3270 PC File Transfer Program	HFX1112	HFX1112	SYS1
5650-ZOS	SMP/E Planning and Migration Assistant	HBCND0B	HBCND0B	SYS1

5650-ZOS	ICKDSF - Device Support Facilities, Base	EDU1H01	EDU1H01	SYS1
5650-ZOS	Environmental Record Editing and Printing	EER3500	EER3500	SYS1
5650-ZOS	MICR/OCR	EMI2220	EMI2220	SYS1
5650-ZOS	TIOC	ETI1106	ETI1106	SYS1
5650-ZOS	ICKDSF - Device Support Facilities, ISMF/MODS	EDU1H01	FDU1H07	SYS1
5650-ZOS	ICKDSF - Device Support Facilities, ISMF/ENU	EDU1H01	FDU1H08	SYS1
5650-ZOS	BCP Base	HBB77A0	HBB77A0	SYS1
5650-ZOS	z/OSMF Software Deployment	HSMA224	HSMA224	SYS1
5650-ZOS	Runtime Library Extensions Base	HTV77A0	HTV77A0	SYS1
5650-ZOS	BCP - Support for Unicode Base	HUN77A0	HUN77A0	SYS1
5650-ZOS	z/OSMF Plug-in A	HSMA228	HSMA228	SYS1
5650-ZOS	z/OSMF Plug-in B	HSMA229	HSMA229	SYS1
5650-ZOS	z/OSMF Incident Log	HSMA225	HSMA225	SYS1
5650-ZOS	z/OSMF Configuration Assistant	HSMA22A	HSMA22A	SYS1
5650-ZOS	z/OS File System Base	HZFS420	HZFS420	SYS1
5650-ZOS	z/OSMF Core Functions	HSMA220	HSMA220	SYS1
5650-ZOS	z/OSMF ISPF	HSMA221	HSMA221	SYS1
5650-ZOS	z/OSMF Plug-in C	HSMA22B	HSMA22B	SYS1
5650-ZOS	IBM Tivoli Directory Server for z/OS Base	HRSL420	HRSL420	SYS1
5650-ZOS	Communications Server SNA	HVT6220	HVT6220	SYS1
5650-ZOS	IBM z™/OS Management Facility - Capacity Provisioni	HSMA226	HSMA226	SYS1

5650-ZOS	Integrated Security Services-Network Authenticatio	HSWK420	HSWK420	SYS1
5650-ZOS	TSO/E Base	HTE77A0	HTE77A0	SYS1
5650-ZOS	z/OSMF Workflow	HSMA227	HSMA227	SYS1
5650-ZOS	IBM z/OS Management Facility - RM	HSMA222	HSMA222	SYS1
5650-ZOS	z/OSMF WLM	HSMA223	HSMA223	SYS1
5650-ZOS	z/OS UNIX System Services Application Services Bas	HOT77A0	HOT77A0	SYS1
5650-ZOS	Common Information Model	HPG77A0	HPG77A0	SYS1
5650-ZOS	BCP - Program Management Binder	HPM77A0	HPM77A0	SYS1
5650-ZOS	BCP - Capacity Provisioning	HPV77A0	HPV77A0	SYS1
5650-ZOS	OpenSSH for z/OS	HOS2220	HOS2220	SYS1
5650-ZOS	Communications Server X11R4 XWindows	HIP6220	JIP622X	SYS1
5650-ZOS	TSO/E ENU	HTE77A0	JTE77AE	SYS1
5650-ZOS	Language Environment® Base	HLE77A0	HLE77A0	SYS1
5650-ZOS	Cryptographic Services - PKI Services	HKY77A0	HKY77A0	SYS1
5650-ZOS	IBM Knowledge Center for z/OS	HKCZ100	HKCZ100	SYS1
5650-ZOS	JES2 Base	HJE77A0	HJE77A0	SYS1
5650-ZOS	Communications Server IP	HIP6220	HIP6220	SYS1
5650-ZOS	IBM HTTP Server	HHAP90P	HHAP90P	SYS1
5650-ZOS	ISPF Base	HIF7P02	HIF7P02	SYS1
5650-ZOS	Network File System Server and Client	HDZ222N	HDZ222N	SYS1
5650-ZOS	Data Facility System Managed Storage Base & ENU	HDZ2220	HDZ2220	SYS1

5650-ZOS	Enhanced Cryptographic Support - ICSF	HCR77B0	HCR77B0	SYS1
5650-ZOS	Hardware Configuration Definition Base	HCS77A0	HCS77A0	SYS1
5650-ZOS	Cryptographic Services - System SSL Base	HCPT420	HCPT420	SYS1
5650-ZOS	Cryptographic Services - OCSF Base	HCRY740	HCRY740	SYS1
5650-ZOS	Communications Server Security Level 3	HIP6220	JIP622K	SYS1
5650-ZOS	z/OS Security Level 3 - System SSL Security Level	HCPT420	JCPT421	SYS1
5650-ZOS	z/OS Security Level 3 - IBM TDS for z/OS Security	HRSL420	JRSL421	SYS1
5650-ZOS	z/OS Security Level 3 - Network Authentication Srv	HSWK420	JSWK421	SYS1
5650-ZOS	z/OS Security Level 3 - OCSF Security Level 3	HCRY740	JCRY741	SYS1
5655-P97	Encryption Facility DFSMSdss Encryption	HCF773D	HCF773D	SYS1
5655-P97	Encryption Facility Encrypt Ser	HCF7740	HCF7740	SYS1
5655-W97	IBM WebSphere MQ for z/OS USS	HMS8000	JMS8008	CSQ800
5655-W97	IBM WebSphere MQ for z/OS Base	HMS8000	HMS8000	CSQ800
5655-W97	IBM WebSphere MQ for z/OS US English	HMS8000	JMS8001	CSQ800
5655-W97	IBM WebSphere MQ for z/OS Uppercase English ENP	HMS8000	JMS8004	CSQ800

5655-MQ9	IBM WebSphere MQ for z/OS USS	HMS9000	JMS9008	CSQ900
5655-MQ9	IBM WebSphere MQ for z/OS Base	HMS9000	HMS9000	CSQ900
5655-MQ9	IBM WebSphere MQ for z/OS US English	HMS9000	JMS9001	CSQ900
5655-MQ9	IBM WebSphere MQ for z/OS Uppercase English ENP	HMS9000	JMS9004	CSQ900

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-M32	UPLOAD FOR z/OS	HPRF417	HPRF417	APS450
5655-M32	PSF for z/OS Base	HPRF450	HPRF450	APS450
5655-M32	Download for z/OS	HPRF450	JPRF452	APS450
5655-M32	Code Page	H0HL15C	J0HL144	APS450
5655-M32	Code Page	H0HL15C	H0HL15C	APS450
5655-M32	Code Page	H0HL15D	H0HL15D	APS450
5655-M32	Code Sample	H0HL140	H0HL140	APS450
5655-M32	Code Page	H0HL15D	J0HL14S	APS450
5655-M32	Code Page	H0HL141	J0HL143	APS450
5655-M32	Code Page	H0HL141	H0HL141	APS450
5655-M32	ACIF	HQN4450	HQN4450	APS450
5655-M32	AFP Download Plus	HPRF450	JPRF451	APS450
5698-SA3	IBM AUTOMATION CONTROL FOR Z/OS - TEP	HKAH35T	HKAH35T	AUT350
5698-SA3	SA z/OS I/O Operations	HWRE35D	HWRE35D	AUT350
5698-SA3	SA z/OS Base Automation	HWRE350	HWRE350	AUT350
5698-SA3	SA z/OS CICS AUTO	HWRE350	JWRE35C	AUT350
5698-SA3	SA z/OS UNIQUE	HWRE350	JWRE35F	AUT350
5698-SA3	SA z/OS IMS AUTO	HWRE350	JWRE35I	AUT350
5655-103	DITTO/ESA base	H0GB310	H0GB310	DIT130

5655-R29	IBM Rational COBOL Runtime	HACZ601	HACZ601	ELA601
5655-Q50	IBM Debug Tool Base	HADRE00	HADRE00	EQAE00
5655-Q50	IBM Problem Determination Tools Common Component	HVWR170	HVWR170	EQAE00
5695-013	IBM Compiler for REXX on zSeries	HWK0140	HWK0140	FAN140
5695-014	IBM Library for REXX on zSeries	HWJ9140	HWJ9140	FAN140
5695-014	IBM Alternate Library for REXX on zSeries	HWJ9143	HWJ9143	FAN140
5724-T07	IBM Developer for z Systems	HHOPE00	HHOPE00	FELE00
5724-T07	IBM Developer for z Systems Host Utilities	HAKGE00	HAKGE00	FELE00
5655-EX1	IBM Explorer for z/OS V3.0.1	HALG300	HALG300	FELE00
5655-Q12	PDTCC	HVWR170	HVWR170	FMND10
5655-Q12	FILE MANAGER Z/OS BASE	HADLD10	HADLD10	FMND10
5655-Q12	FILE MANAGER Z/OS CICS	HADLD10	JADLD16	FMND10
5655-Q12	FILE MANAGER Z/OS DB2	HADLD10	JADLD12	FMND10
5655-Q12	FILE MANAGER Z/OS IMS	HADLD10	JADLD14	FMND10
5655-Q11	FAULT ANALYZER/ BASE	HADQD10	HADQD10	IDID10
5655-PL5	IBM Enterprise PL/I for z/OS	H270510	H270510	IEL510
5655-PL5	IBM Enterprise PL/I for z/OS FEATURE	H270510	J270511	IEL510
5655-EC6	IBM COBOL FOR Z/OS BASE	HADB610	HADB610	IGY610
5655-EC6	IBM COBOL FOR Z/OS HFS	HADB610	JADB61H	IGY610
5655-EC6	IBM COBOL FOR Z/OS US ENGLISH	HADB610	JADB611	IGY610

5635-ISP	IBM DATA SET COMMANDER FOR Z/OS	HIQI810	HIQI810	IQI810
5655-J51	XML Toolkit for z/OS	HXML1A0	HXML1A0	IXM110
5655-DGG	IBM 31-Bit SDK for z/OS Java™ Technology Edition	HJVA800	HJVA800	JVA800
5655-W43	IBM 31-Bit SDK for z/OS Java Technology Edition	HJVA710	HJVA710	JVA710
5655-DGH	IBM 64-Bit SDK for z/OS Java Technology Edition	HJVB800	HJVB800	JVB800
5655-W44	IBM 64-Bit SDK for z/OS Java Technology Edition	HJVB710	HJVB710	JVB710
5697-NV6	IBM AUTOMATION CONTROL FOR z/OS	HNVL12B	HNVL12B	NET621
5697-NV6	IBM Tivoli NetView® Full Version	HNV621B	HNV621B	NET621
5697-NV6	IBM AUTOMATION CONTROL FOR z/OS English	HNVL12B	JNVL12E	NET621
5697-NV6	IBM Tivoli NetView English	HNV621B	JNV621E	NET621
5655-Y20	CICS Transaction Gateway for z/OS	HCTG920	HCTG920	CTG920
5655-Y24	CICS VSAM RECOVERY FOR Z/OS V5.1.0	HCCV510	HCCV510	DWW510
5655-Y24	CICS VSAM RECOVERY FOR Z/OS V5.1.0	HCCV510	JCCV51E	DWW510
5655-Y24	CICS VSAM Transparency for z/OS - Base	HCCV520	HCCV520	DWW520
5655-Y24	CICS VSAM Transparency for z/OS - English	HCCV520	JCCV52E	DWW520
5724-V04	RATIONAL BUILD AGENT for z/OS	HRBA602	HRBA602	BLZ602

5724-V04	RATIONAL TEAM CONCERT - BUILD SYSTEM TOOLKIT for z/OS	HRBT602	HRBT602	BLZ602
5724-V04	RATIONAL COMMON COMPONENTS	HRCC602	HRCC602	BLZ602
5724-V04	RATIONAL TEAM CONCERT - RATIONAL DEVELOPER SUBSET	HRDV602	HRDV602	BLZ602
5668-806	VS FORTRAN LIBRARY MVS	HFL2602	HFL2602	AFF260
5668-806	MVS IAD	HFR2602	HFR2602	AFF260
5668-806	COMPILER MVS	HFT2602	HFT2602	AFF260
5668-806	LIBRARY MVS DEP MODULES	HFL2602	JFL2611	AFF260
5668-806	MVS IAD (TSO)	HFR2602	JFR2611	AFF260
5668-806	MVS IAD (ISPF)	HFR2602	JFR2620	AFF260
5668-806	COMPILER MVS DEP MODULES	HFT2602	JFT2611	AFF260
5668-806	PUBS POST SCRIPT	HFT2602	JFT2612	AFF260
5668-806	PUBS BOOK MANAGER	HFT2602	JFT2613	AFF260
5655-Y29	CICS Deployment Assistant	HGEM530	HGEM530	CPH530
5725-M54	UrbanCode Deploy for z/OS 6.2.0	HRUC620	HRUC620	BUZ620
5655-CEE	IBM z/OS Connect Enterprise Edition V2.0	HZCN200	HZCN200	BAQ200
5655-X09	IBM Sterling Connect Direct for z/OS	HDGA520	HDGA520	DGA520
5655-X09	IBM Sterling Connect Direct for z/OS Standard Edit	HDGA520	JDGA520	DGA520

CICS V5.3

Describes the CICS v5.3 features shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-Y04	CICS - C feature	HCI7000	JCI7003	DFH530
5655-Y04	CICS - COBOL feature	HCI7000	JCI7001	DFH530
5655-Y04	CICS - PL/1 feature	HCI7000	JCI7002	DFH530
5655-Y04	CICS REXX Development System	H0B7110	H0B7110	DFH530
5655-Y04	CICS REXX Common for CICS/ESA	H0Z2110	H0Z2110	DFH530
5655-Y04	CICS TS Licence	HCTS530	HCTS530	DFH530
5655-Y04	CICS - JAVA	HCI7000	JCI700D	DFH530
5655-Y04	CICS - WAS Liberty Profile	HCI7000	JCI700L	DFH530
5655-Y04	CICS REXX Runtime Facility	H0B5110	H0B5110	DFH530
5655-Y04	CICS Service Flow Feature for CICS/TS	HCIZ300	HCIZ300	DFH530
5655-Y04	CICS - System Manager	HCI7000	JCI700M	DFH530
5655-Y04	CICS - WS Security	HCI7000	JCI700W	DFH530
5655-Y04	CICS - Base	HCI7000	HCI7000	DFH530
5655-Y50	CICS Transaction Server Feature Pack for Modern Ba	HCIF51B	HCIF51B	DFH530
5655-Y47	CICS Transaction Server Feature Pack for Dynamic S	HCIF51D	HCIF51D	DFH530
5655-Y49	CICS TS Feature Pack for Security Token Extensions	HCIF51A	HCIF51A	DFH530
5655-Y48	CICS Transaction Server Feature Pack for Mobile Ex	HCIF51C	HCIF51C	DFH530

CICS V5.2

Describes the CICS v5.2 features shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ

5655-Y04	CICS - C feature	HCI6900	JCI6903	DFH520
5655-Y04	CICS - COBOL feature	HCI6900	JCI6901	DFH520
5655-Y04	CICS - PL/1 feature	HCI6900	JCI6902	DFH520
5655-Y04	CICS REXX Development System	H0B7110	H0B7110	DFH520
5655-Y04	CICS REXX Common for CICS/ESA	H0Z2110	H0Z2110	DFH520
5655-Y04	CICS TS Licence	HCTS520	HCTS520	DFH520
5655-Y04	CICS - JAVA	HCI6900	JCI690D	DFH520
5655-Y04	CICS - WAS Liberty Profile	HCI6900	JCI690L	DFH520
5655-Y04	CICS REXX Runtime Facility	H0B5110	H0B5110	DFH520
5655-Y04	CICS Service Flow Feature for CICS/TS	HCIZ300	HCIZ300	DFH520
5655-Y04	CICS - System Manager	HCI6900	JCI690M	DFH520
5655-Y04	CICS - WS Security	HCI6900	JCI690W	DFH520
5655-Y04	CICS - Base	HCI6900	HCI6900	DFH520
5655-Y50	CICS Transaction Server Feature Pack for Modern Ba	HCIF51B	HCIF51B	DFH520
5655-Y47	CICS Transaction Server Feature Pack for Dynamic S	HCIF51D	HCIF51D	DFH520
5655-Y49	CICS TS Feature Pack for Security Token Extensions	HCIF51A	HCIF51A	DFH520
5655-Y48	CICS Transaction Server Feature Pack for Mobile Ex	HCIF51C	HCIF51C	DFH520

CICS V5.1

Describes the CICS v5.1 features shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-Y04	CICS - PL/1 feature	HCI6800	JCI6802	DFH510
5655-Y04	CICS - C feature	HCI6800	JCI6803	DFH510

5655-Y04	CICS - WS Security	HCI6800	JCI680W	DFH510
5655-Y04	CICS - COBOL feature	HCI6800	JCI6801	DFH510
5655-Y04	CICS REXX Runtime Facility	H0B5110	H0B5110	DFH510
5655-Y04	CICS REXX Common for CICS/ESA	H0Z2110	H0Z2110	DFH510
5655-Y04	CICS - JAVA	HCI6800	JCI680D	DFH510
5655-Y04	CICS REXX Development System	H0B7110	H0B7110	DFH510
5655-Y04	CICS Service Flow Feature for CICS/TS 3.2	HCIZ300	HCIZ300	DFH510
5655-Y04	CICS - WAS Liberty Profile	HCI6800	JCI680L	DFH510
5655-Y04	CICS - System Manager	HCI6800	JCI680M	DFH510
5655-Y04	CICS - Base	HCI6800	HCI6800	DFH510
5655-Y50	CICS Transaction Server Feature Pack for Modern Ba	HCIF51B	HCIF51B	DFH510
5655-Y47	CICS Transaction Server Feature Pack for Dynamic S	HCIF51D	HCIF51D	DFH510
5655-Y49	CICS TS Feature Pack for Security Token Extensions	HCIF51A	HCIF51A	DFH510
5655-Y48	CICS Transaction Server Feature Pack for Mobile Ex	HCIF51C	HCIF51C	DFH510

DB2 V11

Describes the DB2 v11 features shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-W87	IBM DB2 UTILITY SUITE FOR Z/OS	HDBBB10	JDBBB1K	DSNB10
5615-DB2	DB2 JDBC/SQLJ Z/OS	HDBBB10	JDBBB12	DSNB10
5615-DB2	DB2 ENGLISH Z/OS	HDBBB10	JDBBB14	DSNB10

5615-DB2	DB2 ODBC Z/OS	HDBBB10	JDBBB17	DSNB10
5615-DB2	DB2 BASE Z/OS	HDBBB10	HDBBB10	DSNB10
5697-Q04	DB2 SPATIAL SUPPORT	H2AF310	J2AG310	DSNB10
5697-Q04	JSON capabilities for DB2 for z/OS	H2AF310	J2AS220	DSNB10
5615-DB2	DB2 RACF EXIT Z/OS	HDREB10	HDREB10	DSNB10
5615-DB2	IRLM	HIR2230	HIR2230	DSNB10
5615-DB2	DB2 IMS ATTACH Z/OS	HIYBB10	HIYBB10	DSNB10
5615-DB2	DB2 SUBSYS INIT Z/OS	HIZBB10	HIZBB10	DSNB10
5697-Q04	DB2 INT COMP UNICODE	H2AF310	H2AF310	DSNB10
5697-Q04	DB2 Adapter for z/OS Connect	H2AZ330	H2AZ330	DSNB10
5655-DAT	IBM DB2 Administration tool for z/OS	H0IHB10	H0IHB10	ADBB10
5655-DOC	IBM DB2 Object Comparison Tool for z/OS	H25GB10	H25GB10	ADBB10
5615-DB2	QMF™ CLASSIC EDITION - ENGLISH	HSQBB10	HSQBB10	QMFB10
5615-DB2	Z/OS APPLICATION CONNECTIVITY TO DB2 FOR Z/OS	HDDA211	HDDA211	DDA211

IMS 14.1

Describes the IMS v14.1 features shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5635-A05	IMS	HMK1400	HMK1400	DFSE10
5635-A05	IMS JAVA ON DEMAND FEATURES	HMK1400	JMK1406	DFSE10
5635-A05	IMS DATABASE MANAGER	HMK1400	JMK1401	DFSE10
5635-A05	IMS DLT	HMK1400	JMK1405	DFSE10
5635-A05	IMS EXT. TERMINAL OPT	HMK1400	JMK1403	DFSE10
5635-A05	IMS RLT	HMK1400	JMK1404	DFSE10
5635-A05	IRLM	HIR2230	HIR2230	DFSE10

5635-A05	IMS TRANSACTION MANAGER	HMK1400	JMK1402	DFSE10

IMS 13.1

Describes the IMS v13.1 features shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5635-A04	IMS	HMK1300	HMK1300	DFSD10
5635-A04	IMS JAVA ON DEMAND FEATURES	HMK1300	JMK1306	DFSD10
5635-A04	IMS DATABASE MANAGER	HMK1300	JMK1301	DFSD10
5635-A04	IMS DLT	HMK1300	JMK1305	DFSD10
5635-A04	IMS EXT. TERMINAL OPT	HMK1300	JMK1303	DFSD10
5635-A04	IMS RLT	HMK1300	JMK1304	DFSD10
5635-A04	IRLM	HIR2230	HIR2230	DFSD10
5635-A04	IMS TRANSACTION MANAGER	HMK1300	JMK1302	DFSD10

IMS Utilities

Describes the IMS utilities shipped with the z/OS v2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-TDA	IMS ENTERPRISE SUITE BASE SERVICES	HAHF320	HAHF320	AEW321
5655-TDA	IMS ENTERPRISE SUITE MOBILE GATEWAY COMPONENT	HAHF320	JAHF32A	AEW321
5655-TDA	IMS ENTERPRISE SUITE IMS SOAP GATEWAY	HAHF320	JAHF321	AEW321
5655-TDA	IMS ENTERPRISE SUITE JAVA MESSAGE SERVICES API	HAHF320	JAHF322	AEW321

5655-TDA	IMS ENTERPRISE SUITE CONNECT API FOR C/JAVA	HAHF320	JAHF323	AEW321
5655-TDA	IMS ENTERPRISE SUITE IMS MOBILE FEATURE PACK	HAHF324	HAHF324	AEW321

WebSphere Application Server V9.0

Lists information about the WebSphere Application Server v9 available with this z/OS 2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-WAS	WebSphere Application Server for z/OS	HBBO900	HBBO900	WAS900

WebSphere Application Server V8.5

Lists information about the WebSphere Application Server v8.5 available with this z/OS 2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-W65	WebSphere Application Server for z/OS	HBBO850	HBBO850	WAS850
5655-W65	IBM WebSphere SDK Java Technology Edition	HBJA700	HBJA700	WAS850

WebSphere Application Server V8.0.0

Lists information about the WebSphere Application Server V8.0.0 available with this z/OS 2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5655-W65	WebSphere Application Server for z/OS	HBBO800	HBBO800	WAS800

Tivoli Workload Scheduler 9.3.0

Lists information about the Tivoli Workload Scheduler 9.3.0 available with this z/OS 2.2 ADCD.

Product Number	**Description**	**Fmid**	**Dependent FMID**	**HLQ**
5698-T08	IWS FOR Z/OS AGENT	HWSZ930	HWSZ930	TWS930

Product Number	Description	Fmid	Dependent FMID	HLQ
5698-T08	IWS FOR Z/OS ENGINE ENGLISH	HWSZ930	JWSZ93B	TWS930
5698-T08	IWS FOR Z/OS ENGINE	HWSZ930	JWSZ932	TWS930
5698-T08	IBM Workload Scheduler for z/OS - E2E Enabler	HWSZ930	JWSZ933	TWS930
5698-T08	IBM Workload Scheduler for z/OS - zConnector	HWSZ930	JWSZ934	TWS930

Tivoli OMEGAMON V5.4.0

Lists the Tivoli OMEGAMON v 5.4.0 features available in the z/OS 2.2 ADCD.

Product Number	Description	Fmid	Dependent FMID	HLQ
5698-B48	ITCAM for Application Diagnostics	HAAD710	HAAD710	KAN54A
5698-B48	ITCAM for Application Diagnostics, TEMA	HKYN710	HKYN710	KAN54A
5698-B48	ITCAM for Application Diagnostics Common Services	HAAD71C	HAAD71C	KAN54A
5698-T07	OMEGAMON XE for CICS on z/OS	HKC5530	HKC5530	KAN54A
5698-T07	OMEGAMON XE for CICS TG on z/OS	HKGW530	HKGW530	KAN54A
5698-T07	End-to-End	HKET620	HKET620	KAN54A
5698-T01	OMEGAMON Monitoring for JVM	HKJJ530	HKJJ530	KAN54A
5698-T02	OMEGAMON XE for IMS on z/OS	HKI5530	HKI5530	KAN54A
5698-B23	OMNIMON Base	HKOB730	HKOB730	KAN54A
5698-B23	OMEGAMON XE for WebSphere MQ Configuration	HKMC730	HKMC730	KAN54A

Product Number	Description	Fmid	Dependent FMID	HLQ
5698-B23	OMEGAMON XE for WebSphere MQ Monitoring	HKMQ730	HKMQ730	KAN54A
5698-B23	Generic Configuration	HKCF730	HKCF730	KAN54A
5698-B23	OMEGAMON XE for WebSphere Message Broker Monitorin	HKQI730	HKQI730	KAN54A
5698-A79	Installation/ Configuration Assistance Tool	HKCI310	HKCI310	KAN54A
5698-A79	Tivoli Enterprise Monitoring Server	HKDS630	HKDS630	KAN54A
5698-A79	TMS:Engine	HKLV630	HKLV630	KAN54A
5698-AA8	Log Forwarder	HGLA220	HGLA220	KAN54A
5698-AA8	PERF Mgmt TADz ID	HPMS540	HPMS540	KAN54A

Reported problems, fixes, and maintenance

None.

Build structure

Describes the build structure of the volumes shipped with the z/OS v2.2 ADCD.

C2RES1 and C2RES2

C2RES1 is a 3390-9 volume that contains the base MVS system software target, or run-time libraries, and other system data sets required to IPL the system and use TSO/E and ISPF. This is the MVS IPL volume.

C2RES2 is a 3390-9 volumes that are an extension of the SYSRES volume C2RES1. The volume is also required for IPL.

C2SYS1

This 3390-9 volume contains data sets that you might change, either through normal use of the system or user customization. System control data sets, such as SYS1.IPLPARM,IODF (I/O Definition File), and system's master catalog (CATALOG.Z22C.MASTER) reside on this volume. This volume is required to IPL. C2SYS1 contains the usercatalog USERCAT.Z22C.PRODS containing entries for z/OS related products

C2CFG1

This 3390-9 volume contains data sets for storing user configuration. The 'USER.**' data sets reside on this volume, they are cataloged in the master catalog. The RACF database reside on this volume.

C2USS1 and C2USS2

C2USS1 is a 3390-9 volume that contains the Fonts, Version and zOSMF zFS files for UNIX System Services of z/OS. This volume is required to IPL.

C2USS2 is a 3390-9 volume that contains root zFS and zFS files for UNIX System Services of z/OS. This volume is required to IPL.

C2PRD1, C2PRD2 and C2PRD3

C2PRD1 to C2PRD3 are 3390-9 volumes that contains all z/OS products.

C2DIS1 and C2DIS2

These volumes contain the base MVS system software distribution libraries. They contain the DLIB (Distribution Library) and zone CSI's. These volumes are needed to install service or products on the MVS system using SMP/E. These volumes are all 3390-9.

C2PAGA, C2PAGB and C2PAGC

These volumes contain page data sets. The C2PAGA volume has the PLPA, Common and one Local page data set. Rest of the 2 volumes each have one Local page data set. The loadparms that bring up DB2 or WAS have been defined to use all the above 3 page volumes, all other loadparms use only the first two volumes. If you do not want to use all of these local page data sets, you can modify the PAGE parameter on IEASYSnn member, accordingly.

C2DBB1, C2DBB2 and C2DBAR

C2DBB1 is a 3390-9 volume that contains DB2 V11 related data. C2DBB1 contains the DB2 usercatlog, USERCAT.Z22C.DB2VB. Also C2DBB1 contains all DB2 target libs AND DB2 V11 utility products. This volume is not required to IPL but is required to bring up DB2 V11.

C2DBB2 is a 3390-9 volume that contains DB2 catalog and directories with HLQ DSNCB10, which are SMS managed. The volume is not required for IPL but is needed to bring up DB2.

C2DBAR is a 3390-9 volume defined in a new esoteric called DBARCH. This esoteric has been defined in DB2, for allocating the archive logs into this storage volume. This volume is required, if you want to bring up DB2 V11.

C2DBA1, C2DBA2 and C2DBAR

C2DBA1 is a 3390-9 volume that contains DB2 V10 related data. C2DBA1 contains the DB2 usercatlog, USERCAT.Z22C.DB2VA. Also C2DBA1 contains all DB2 target libs and DB2 V10 utility products. This volume is not required to IPL but is required to bring up DB2 V10.

C2DBA2 is a 3390-9 volume that contains DB2 catalog and directories with HLQ DSNCA10, which are SMS managed. The volume is not required for IPL but is needed to bring up DB2.

C2DBAR is a 3390-9 volume defined in a new esoteric called DBARCH. This esoteric has been defined in DB2, for allocating the archive logs into this storage volume. This volume is required, if you want to bring up DB2 V10.

C2C531

C2C531 is a 3390-9 volume that contains CICS 5.3 Target, DLIB and all related data for CICS 5.3. USERCAT.Z22C.CICS530 resides on this volume. This volume is not required to IPL but is required to bring up CICS 5.3 and apply CICS maintenance.

C2C521

C2C521 is a 3390-9 volume that contains CICS 5.2 Target, DLIB and all related data for CICS 5.2. USERCAT.Z22C.CICS520 resides on this volume. This volume is not required to IPL but is required to bring up CICS 5.2 and apply CICS maintenance.

C2C511

C2C511 is a 3390-9 volume that contains CICS 5.1 Target, DLIB and all related data for CICS 5.1. USERCAT.Z22C.CICS510 resides on this volume. This volume is not required to IPL but is required to bring up CICS 5.1 and apply CICS maintenance.

C2W901 and C2W902

C2W901 and C2W902 are 3390-9 volumes that contains target and distribution libraries for Websphere Application Services.

C2W851, C2W852, C2W853, C2W854 and C2W855

C2W851, C2W852, C2W853, C2W854 and C2W855 are 3390-9 volume that contains target and distribution libraries for Websphere Application Services.

C2W801, C2W802, C2W803, C2W804 and C2W805

C2W801, C2W802 and C2W803 are 3390-9 volumes that contains target and distribution libraries for Websphere Application Services.

C2W804 and C2W805 are 3390-9 volumes that contain pre-configured zFS files required for manually starting WAS.

C2IME1, C2IMD1 and C2IMU1

C2IME1 is a 3390-9 volume that contains IMS Target, DLIB and all related data for IMS 14. USERCAT.Z22C.IMS14 resides on this volume. This volume is not required to IPL but is required to bring up IMS and perform maintenance on IMS.

C2IMD1 is a 3390-9 volume that contains IMS Target, DLIB and all related data for IMS 13. USERCAT.Z22C.IMS13 resides on this volume. This volume is not required to IPL but is required to bring up IMS and perform maintenance on IMS.

C2IMU1 is a 3390-9 that contains IMS Utilities. These volumes are not required to IPL but is required for IMS Utilities and perform maintenance on IMS Enterprise Suite.

B2IMU1 and B2IMU2 are 3390-9 10000 cylinder volumes that contain IMS Utilities. These volumes are not required to IPL but is required for IMS Utilities and perform maintenance on IMS Enterprise Suite.

C2KAN1

C2KAN1 is a 3390-9 volume that contains Target and DLIB libraries for Tivoli OMEGAMON v5.4.0.

C2BLZ1

C2BLZ1 is a 3390-9 volume that contains Target and DLIB libraries for Rational Team Concert 6.0.2. This volume is not required to IPL but is required to bring up RTC.

SARES1

This 3390-9 volume contains a single volume stand alone system. This volume can be used to IPL and logon to a TSO/ISPF session.

The disk volume can assist in building LPAR environments and correct errors that prevent system IPL.

Recommended use of this volume is to install the volume and leave accessible. The volume should not be altered. The volume should stay in a background mode and be available for emergency type of situations. If other z/OS, OS/390®, or LPAR partitions contain errors, the SARES1 volume could be used to solve the problem.

The stand alone system does not contain TCPIP or Unix system service support. The system cannot be used to install products or apply maintenance via SMP/E.

To IPL the standalone res volume:

Load Address: 0Axx

Load Parms: 0AxxSAM

Migration Guidelines

The following guidelines will make it easier for you to replace this level of system software with new levels built the same way.

Only IBM -supplied system software should reside on C2xxxx volumes except for C2SYS1. C2SYS1 contains RACF, IPL, and catalog datasets that are user dependent.

The ADCD does not contain a generalized migration utility or process. Each user has unique requirements. It is recommended that you build a migration plan based on your unique needs. The ADCD does have some assistance. The ADCD uses a system of concatenated libraries (see below) The highest level of concatenation is USER.xxxxxxxx. The ADCD distributes these libraries empty. It is recommended that the user place changes or overrides in these libraries. It is further recommended that the USER.xxxxxxxx libraries be backed up regularly. When a new ADCD release is installed the USER.xxxxxxxx libraries can be copied from backups to the new USER.xxxxxxxx libraries to provide assistance in migration of programs and parameters.

The ADCD is distributed with a standard library concatenation for the following:

- LINKLST
- PROCLIB (including TSO procedures)
- CLIST
- ISPPLIB (TSO panels)
- LPALIB
- PARMLIB
- VTAMLST
- VTAMLIB
- VTAM source

The configured order of concatenation is user, ADCD developers, and z/OS system data sets. For example, the LINKLST concatenation would be as follows:

- USER.LINKLIB
- ADCD.Z22C.LINKLIB
- SYS1.LINKLIB

USER.xxxxx libraries have all been built on C2CFG1 which is the only volume that should contain user data.

USER.xxxxx will not be changed by ADCD or System processes; thus, user updates in USER.xxxxx will be retained between release levels. ADCD.Z22C.xxxxxx libraries are allocated on C2SYS1 and are the libraries that are used by ADCD developers. No RACF rules exist on these libraries; however, changes to these libraries could destroy customization necessary to bring up many products. SYS1.xxxxxxx libraries should NEVER be updated by users or ADCD development. System libraries should ONLY be updated through SMP/E install, maintenance processes, or like processes.

Note: Changes to System libraries or ADCD pre-customization may corrupt your system and prevent or delay IBM support activites.

The correct procedure for a user update to system data is to:

- Copy the system data from SYS1.xxxxx or ADCD.xxxx to USER.xxxxx
- Edit, compile, or run user program against the USER.xxxx item
- Re-IPL, recycle system service, or relogon to TSO

Removing a user update would be the reverse of the above procedure.

LOADPARMS Options

Lists available preconfigured distributed LOADPARMS.

Note: Start JES2 cold the first time you start the system.

Table 1. Listing of available preconfigured distributed LOADPARMS.

LOADPARM	Description
CS	CLPA and Cold start of JES2. Base z/OS system functions, that is, no CICS, DB2, IMS, Websphere Application Server, and so on.
WS	CLPA and Warm start of JES2. Base z/OS system functions, that is, no CICS, DB2, IMS, Websphere Application Server, and so on.
00	CLPA and Warm start of JES2. Base z/OS system functions, that is, no CICS, DB2, IMS, Websphere Application Server, and so on.
CI	CLPA and Warm start of JES2. Loads CICS 5.3 and 5.2 libraries. Starts CICS 5.3 and IBM Developer for z Systems.
DB	CLPA and Warm start of JES2. Loads DB2 V11 libraries. Starts DB2 V11 and IBM Developer for z Systems.
IM	CLPA and Warm start of JES2. Loads IMS 14 and 13 libraries. up Starts IMS 14 and IBM Developer for z Systems.
IZ	CLPA and Warm start of JES2. Starts up z/OSMF and IBM Developer for z Systems.
WA	CLPA and Warm start of JES2. Loads Websphere Application Server 8.5 libraries. Starts IBM Developer for z Systems. Websphere Application Server needs to be manually started.

Table 1. Listing of available preconfigured distributed LOADPARMS. (continued)

LOADPARM	Description
AL	CLPA and Warm start of JES2. Loads all middleware libraries. Starts CICS 5.3, DB2 V11, IMS 14, z/OSMF and IBM Developer for z Systems.
DC	CLPA and Cold start of JES2. Loads CICS, DB2 libraries. Starts CICS 5.3, DB2 V11, UCD 6.2.0, RTC 6.0.2, and IBM Developer for z Systems.
DW	CLPA and Warm start of JES2. Loads CICS, DB2 libraries. Starts CICS 5.3, DB2 V11, UCD 6.2.0, RTC 6.0.2, IBM Developer for z Systems.

Console PF Key Settings

Describes the default PF key assignments.

Action/Command	PFKey
Display Devices	PF1
Display 3270 Devices	PF2
Clear Top of Screen	PF3
Create a 10-line Display Area	PF4
Display Address Space Information	PF5
Display Outstanding Reply Requests and Error Messages	PF6
Display PF Keys	PF7
Scroll Display Area	PF8
Display TSO Users	PF9
Display Active Address Spaces	PF10
Display Active Jobs	PF11
Clear Bottom Screen	PF12

USERIDS

Lists the TSO User IDs and passwords that are set up on your system.

Table 2. Predefined USERIDS.. The following TSO User IDs and passwords are set up on your system:

User ID	Password
ADCDMST (RACF special authority)	SYS1 or ADCDMST
IBMUSER (RACF special authority)	SYS1 or IBMUSER
SYSADM (DB2 and RACF special authority)	SYS1 or SYSADM
SYSOPR (DB2 and RACF special authority)	SYS1 or SYSOPR
ADCDA - ADCDZ	TEST
WEBADM	WEBADM
OPEN1 through OPEN3	SYS1
ZOSMFAD	ZOSMFAD

Maintenance Service Levels

All the products on the ADCD consist of maintenance that is in a closed status. A PTF that is still in open status or has other than a document hold at the time the ADCD was built would not be added to the ADCD. The following PUT levels might be minus PTFs that were open at build time. Also, some functions might be better than the PUT level due to the addition of RSUs and individual PTFs. In general, the following is valid.

- All functions of the base z/OS 2.2 November 2016 are at RSU level 1607.
- All functions of the CICS TS 5.3 are at PUT level 1606.
- All functions of the CICS TS 5.2 are at RSU level 1607.
- All functions of the CICS TS 5.1 are at RSU level 1607.
- The functions of the DB2 11 base are at RSU level 1607.
- The functions of the DB2 10 base are at RSU level 1607.
- All functions of the IMS 14.1.0 are at RSU level 1607.
- All functions of the IMS 13.1.0 are at RSU level 1607.
- All functions of the IMS Utilities are at PUT level 1606.
- Tivoli OMEGAMON v 5.4.0 is at RSU level 1607.

Hints and tips to manage the z/OS 2.2 ADCD

Learn with the examples in this sections about page data sets available, starting IMS and running an IVP transaction, WLM configuration provided, ACS routines provided, recreating the coupling data sets, allocating DB2 archive logs into an esoteric, steps for dumping SMF data into GDG (Generation Data Group) data sets, using the esoteric devices, and how to start IBM Healthchecker for z/OS, IBM z/OS Management Facility (z/OSMF), and WebSphere Application Server.

Page data sets

This ADCD system provides you with three volumes that are dedicated for page data sets, C2PAGA through C2PAGC.

The following list shows the content of the volumes.

- C2PAGA
  - `SYS1.S0W1.PLPA.PAGE`
  - `SYS1.S0W1.COMMON.PAGE`
  - `SYS1.S0W1.LOCALA.PAGE`
- C2PAGB `SYS1.S0W1.LOCALB.PAGE`
- C2PAGC `SYS1.S0W1.LOCALC.PAGE`

The preconfigured loadparms to start DB2 and WebSphere Application Server use all of the above listed page data sets. The remaining loadparms use only the data sets on volumes C2PAGA and C2PAGB. Depending upon your requirement, you can add or remove these page data sets by using the following parameter in the IEASYSxx member after you copy it to `USER.PARMLIB`.

```
PAGE=(SYS1.S0W1.PLPA.PAGE,
      SYS1.S0W1.COMMON.PAGE,
      SYS1.S0W1.LOCALA.PAGE,
      SYS1.S0W1.LOCALB.PAGE,L),
```

Starting IMS and running an IVP transaction

The instructions for starting and stopping IMS can be found at http://dtsc.dfw.ibm.com/MVSDS/'HTTPD2.DFSE10.PUBLIC.SHTML(INDEX)'.

This page has pointers to the levels of IMS V13 and V14. On each level, click **Issues, Hints and Tips** to see the instructions for starting and stopping. The IVP jobs are available on the ADCD system, in the data set `DFSE10.INSTALIB` for IMS V14, and `DFSD10.INSTALIB` for IMS V13. You must run each series in a specific manner to create the particular IVP environment and data to be able to follow each IVP.

WLM configuration provided on ADCD z/OS

The WLM configuration of application environments for DB2 Stored Procedures for DBBG are self-documented in the WLM application environment display.

For DBBG (DB2 V11), the comment describes the usage of each environment.

```
DBBGENV    Default
DBBGENVC   DSNWLM_DSNACICS
DBBGENVD   DSNWLM_DEBUGGER
DBBGENVG   DSNWLM_GENERAL
DBBGENVJ   DSNWLM_JAVA
DBBGENVM   DSNWLM_MQSERIES
DBBGENVO   DSNWLM_JAVA_BIGMEM
DBBGENVP   DSNWLM_PGM_CONTROL
DBBGENVR   DSNWLM_REXX
DBBGENVU   DSNWLM_UTILS
DBBGENVW   DSNWLM_WEBSERVICES
DBBGENVX   DSNWLM_XML
DBBGENV1   DSNWLM_NUMTCB1
DBBGRFSH   Stored Procedure REFRESH
```

ACS routines provided on ADCD

DB2 V11 requires that its catalog and directory are SMS-managed. The ACS routines that are available on ADCD z/OS system were created to handle HLQ DSNCB10, which is the HLQ of the DB2 V11 Catalog and Directory. The logic is to use two qualifiers; the first is DSNCB10, and the second is DSNDBC or DSNDBD. The DSNDBC and DSNDBD indicate the cluster and data portion of the VSAM data set of a DB2 table space. Other data sets for DB2 V11 have a HLQ of DSNB10, and these data sets are not SMS-managed. A DB2 STOGROUP called SYSSMS was created for the Catalog and Directory table spaces with a VCAT of DSNCB10 and VOL of (*). (*) is for SMS-managed STOGROUP. The DB2 STOGROUP, SYSDEFLT uses VCAT of DSNB10 and lists actual volumes, and it is not SMS-managed. The installation is configured to use SYSSMS when it creates the catalog and directory table spaces. If you create and specify a STOGROUP of your own or default to SYSDEFLT STOGROUP, your table spaces fall into non-SMS managed volumes as specified in the STOGROUP.

Recreating the coupling data sets

The following coupling data sets are provided on ADCD.

```
SYS1.ADCDPL.CDS01
 SYS1.ADCDPL.CDS02
SYS1.ADCDPL.CFRM.CDS01
SYS1.ADCDPL.CFRM.CDS02
SYS1.ADCDPL.LOGR.CDS01
```

```
SYS1.ADCDPL.LOGR.CDS02
SYS1.ADCDPL.OMVS.CDS01
SYS1.ADCDPL.OMVS.CDS02
SYS1.ADCDPL.WLM.CDS01
SYS1.ADCDPL.WLM.CDS02
```

In case any of these data sets are corrupted or become unusable, delete them and use the JCL that is available in member `ALOCCOPL` in `'ADCD.LIB.JCL'` data set to allocate them again.

Note: To perform this procedure, you must have thorough knowledge of setting up a SYSPLEX environment. These steps are a bit involved and could cause disruption of normal functioning of your ADCD system.

Allocating DB2 archive logs into an esoteric device

Additional storage volume C2DBAR was defined and is pointed to by a new esoteric DBARCH for DB2 archive logs. You can find instructions for adding more volumes to this esoteric in this topic. Ensure to map this volume C2DBAR to device# 0AA3 in your devmap so that it is in sync with the IODF definition.

In this edition of ADCD z/OS, the loadparms to start DB2 were configured to use the esoteric device DBARCH for holding the DB2 archive logs. This means that the archive logs go only into this volume (or volumes if you add more to the esoteric). You still need to manage the space on the volume, and delete archive logs that are no longer needed. The fourth qualifier in the data set name is a date of the format D*yynnn* where *yy* is the year and *nnn* is the Julian day of the year. When you start DB2, examine the console log to ensure that it does not read any archive logs. If no archive logs are used at the start, you have a clean start of DB2, and then you can delete all the prior archive logs. If you are running DB2 and running out of space in the esoteric DBARCH volumes, check the SDSF console log for DB2 and determine the start date. Then, you can delete archive logs before the start date of the last clean start of DB2. You must monitor the space available on your DBARCH esoteric volumes. To achieve better space management, turn off logging on some of your table spaces that can be recovered by just dropping, re-creating, and reloading. You can also reduce logging by using the LOG NO parm in your load utility job. You must reset the COPY pending flag with the NOCOPYPEND parm in the load utility or the REPAIR utility. And note that these options are used only for table spaces that you can recover on your own without DB2 recovery. A thoughtful management of DB2 archive logs is important in any DB2 environment: test, development, or production.

Also note that even though you delete the DSNx10.DBxG.ARCLOG1.Date.Time.**A0000000#** data sets, you need to keep enough of the DSNx10.DBxG.ARCLOG1.Date.Time.**B000000#** data sets to cover two successful restarts of DB2. These data sets are the backups for your BSDS data sets. If anything happens to your BSDS, you need these backups to restore your BSDS.

The DBARCH esoteric is coded into DB2 ZPARM by the link-edit JCL member DSNTIJUZ in DSNx10.NEW.SDSNSAMP. You can find the following parameters coded under the DSN6ARVP section of DSNTIJUZ. If you want to create and update with a different esoteric other than the one provided, you can find the information here.

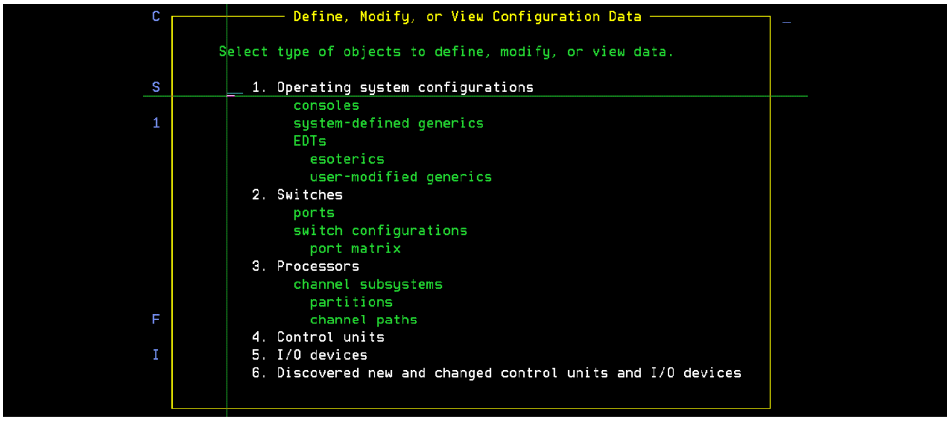
```
TSTAMP=YES,  X
UNIT=DBARCH, X
UNIT2=
```

To change an esoteric, you need to update the IODF file. The default active IODF in the ADCD system is SYS1.IODF99. When you modify it, a temporary work IODF is created, which can be named as your preference, such as SYS1.IODF03.WORK. When the changes are made, you can use this work file to create a production IODF file, such as SYS1.IODF03. After you change the loadparm to use SYS1.IODF03, you can re-LPI the system to check if the changes you made are working properly. You can either continue to use SYS1.IODF03 or copy it to SYS1.IODF99. If you choose to copy it to IODF99, you need to backup SYS1.IODF99, and then delete SYS1.IODF99. After you delete SYS1.IODF99, you can copy SYS1.IODF03 to SYS1.IODF99, and update the loadparm member to use SYS1.IODF99.

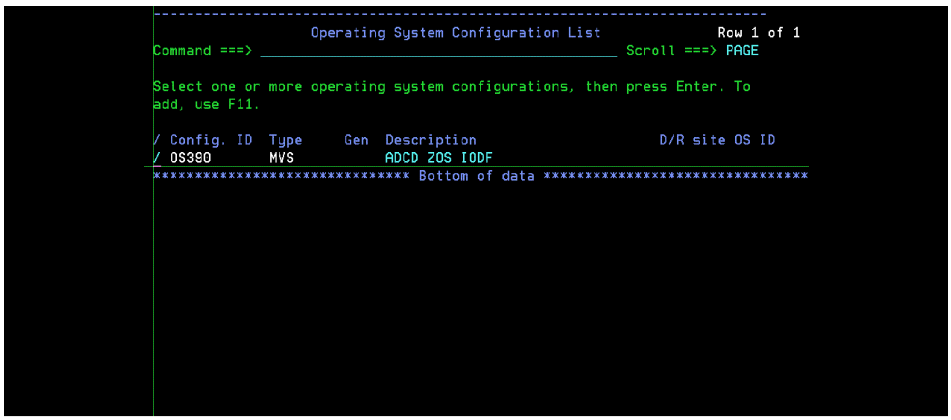
To add more volumes to the esoteric DBARCH, follow these steps:

1. From the ISPF primary option menu, select option **M.4. Hardware Configuration Tools**.

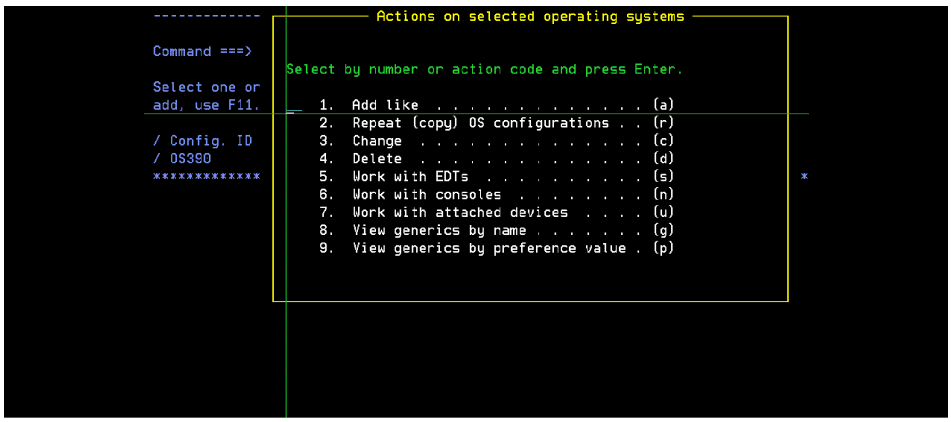
2. In the above **I/O definition file**, type SYS1.IODF99, and then select option **1. Define, modify, or view configuration data**.



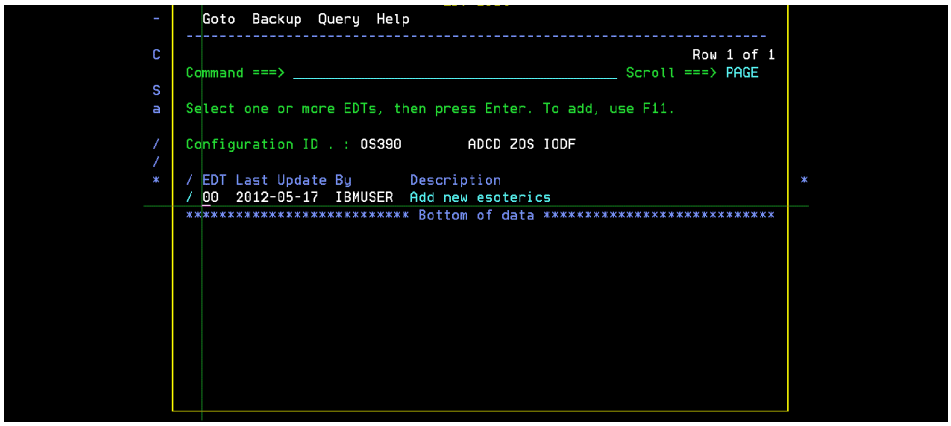
3. In the **Operating System Configurations List** window, type / to select **Config. ID OS390**, and then press Enter.



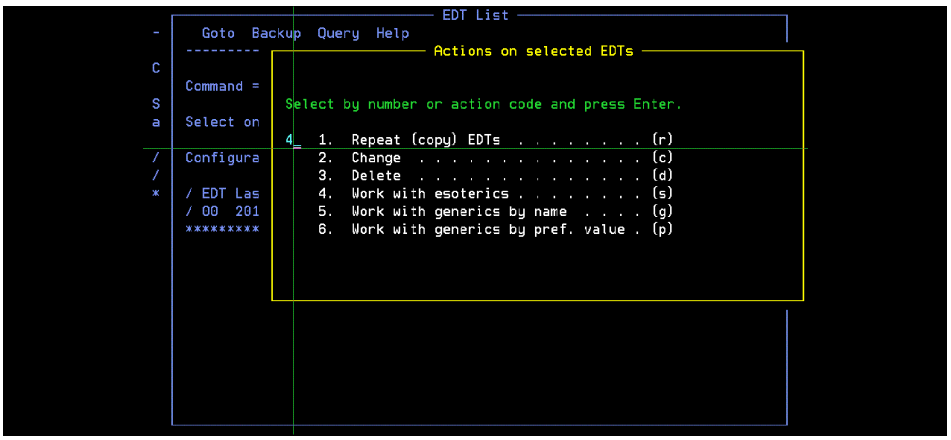
4. When a pop-up window opens as shown below, select option **5. Work with EDTs.**



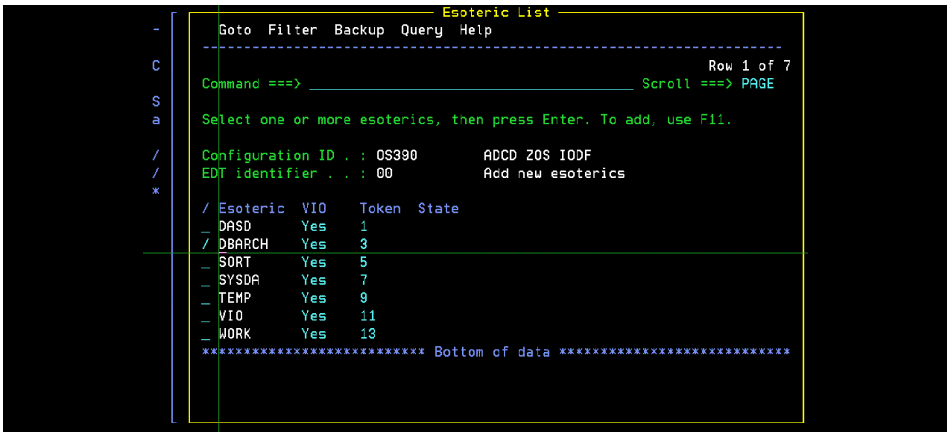
5. When a list of EDTs shows, type / to select **EDT# 00**, and then press Enter.



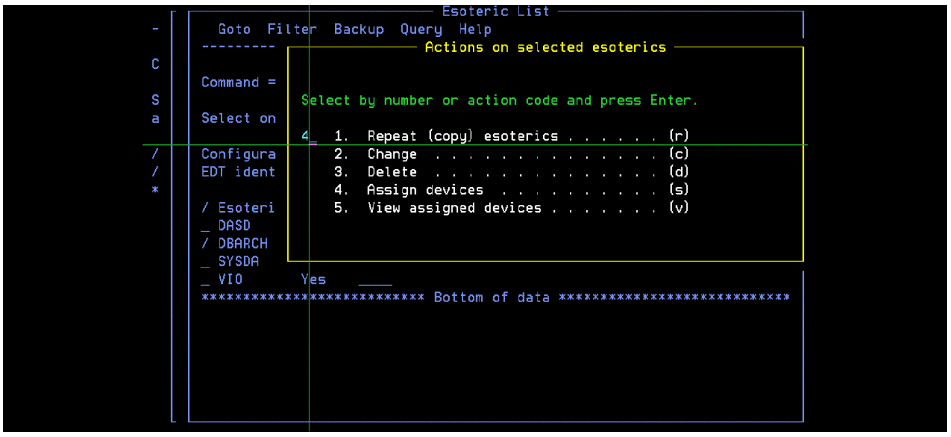
6. When a pop-up window opens as below, select option **4. Work with esoterics**.



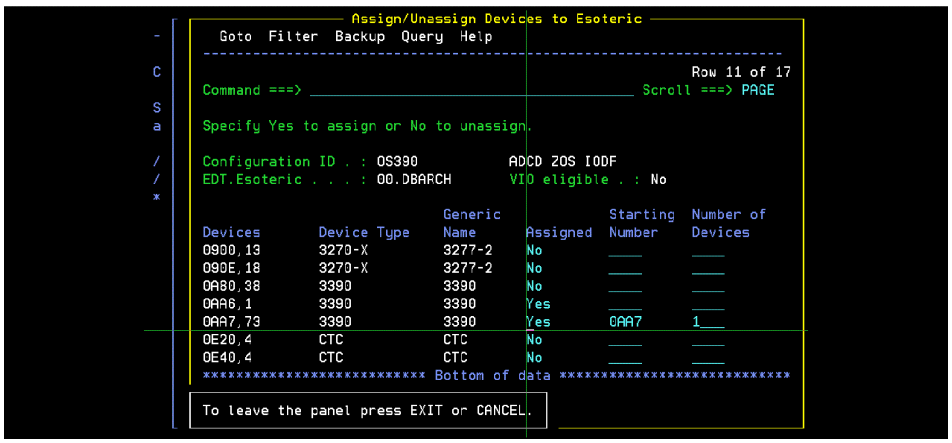
7. When the available esoterics are listed as shown below, type / to select **Esoteric DBARCH**, and then press Enter for further options.



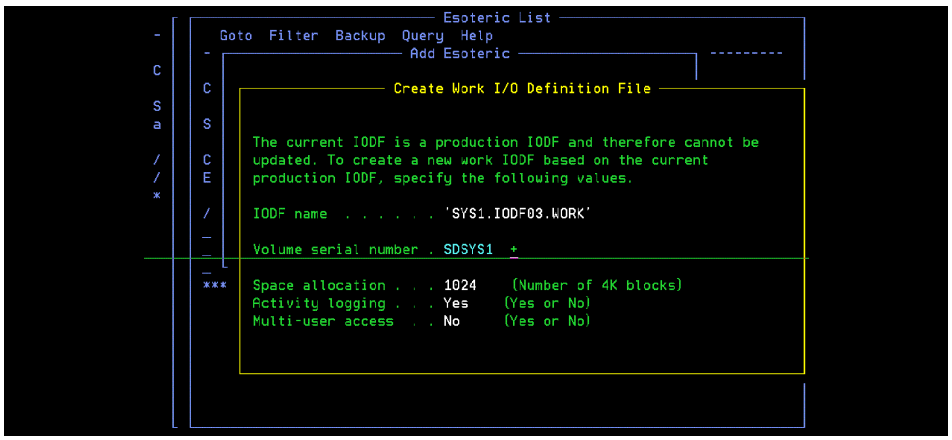
8. In the Actions on selected esoterics window, select option **4. Assign devices**.



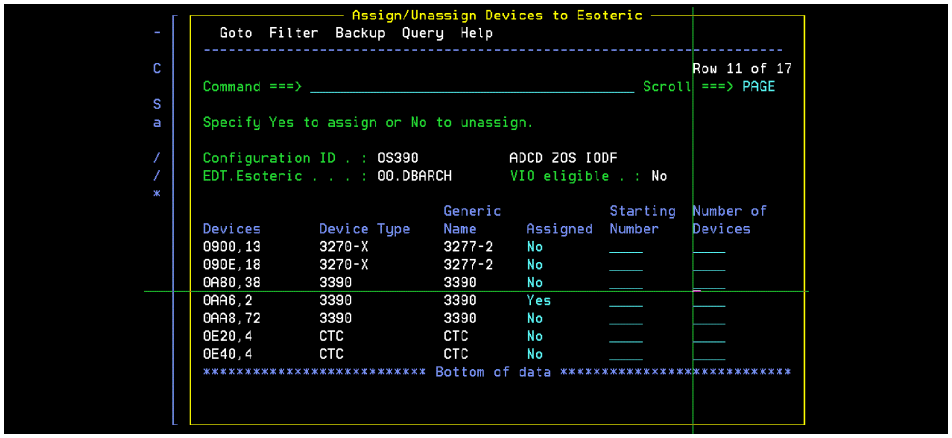
9. When the available devices are listed as shown below, select the appropriate device or device range that you want to add to the DBARCH esoteric. For example, to add one more device at address 0AA7, change the status of 'Assigned' column from 'No' to 'Yes' across the device range that you want to add, and also specify **Starting Numbers** and **Number of Devices** to add a range. Then press enter.



10. When a pop-up window opens as below, type a name for the temporary work IODF, for example, `SYS1.IODF03.WORK`, and then press Enter.

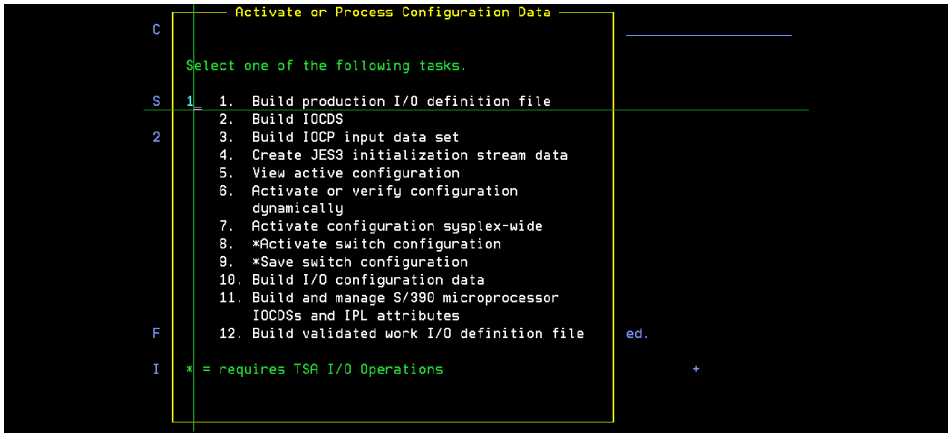


Then The Assign/Unassigned Devices to Esoteric window is updated to show the list of devices assigned. Because device# 0AA6 is already assigned, it is updated to include two devices in that range: 0AA6 and 0AA7.



11. Return to the primary option menu of HCD, type `SYS1.IODF03.WORK` in the **I/O definition file** field, and then select option **2. Activate or process configuration data**.

12. In the Activate or Process Configuration Data window, select option **1. Build production I/O definition file**.

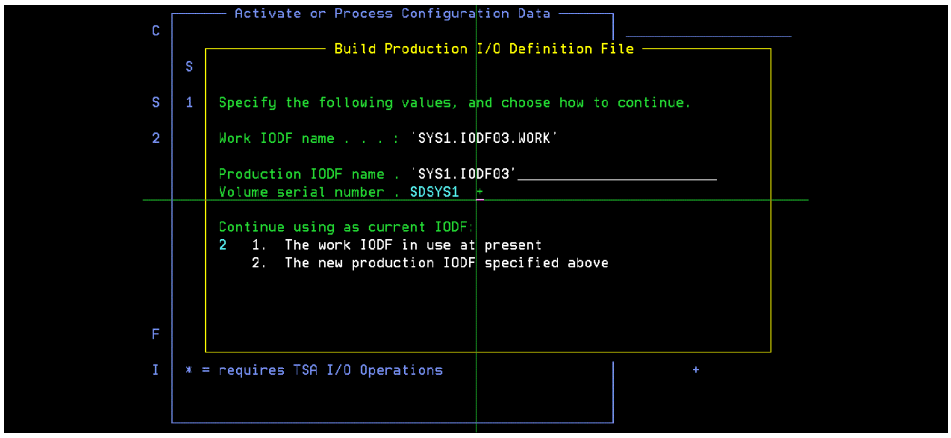


The changes that are made are verified. Any errors or warnings that are displayed at this stage can be ignored.

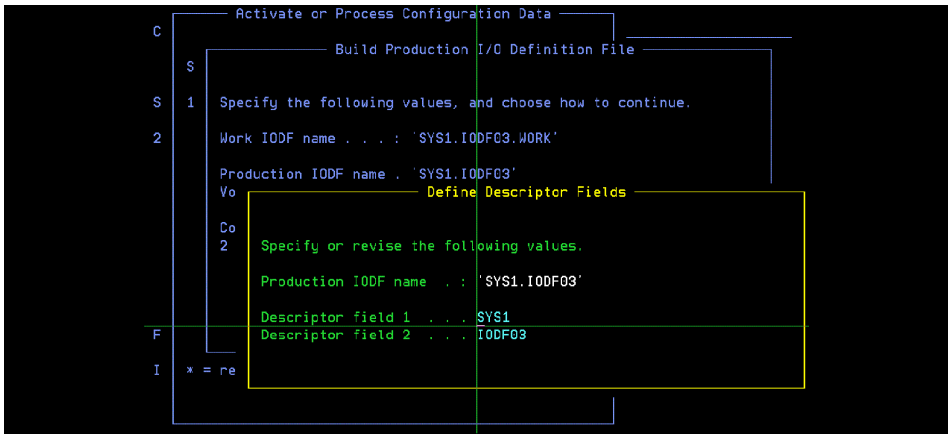
13. Press F3 to return to the Activate or Process Configuration Data window, and select option **1. Build production I/O definition file**.
14. To create the production I/O definition file:
    a. In the **Work IODF name** field, type `SYS1.IODF03.WORK`.
    b. In the **Production IODF name** field, type `SYS1.IODF03`.
    c. In the **Volume serial number** field, specify the same volume where IODF99 is.

d. In the **Continue using as current IODF:** field, specify option **2. The new production IODF specified above**.

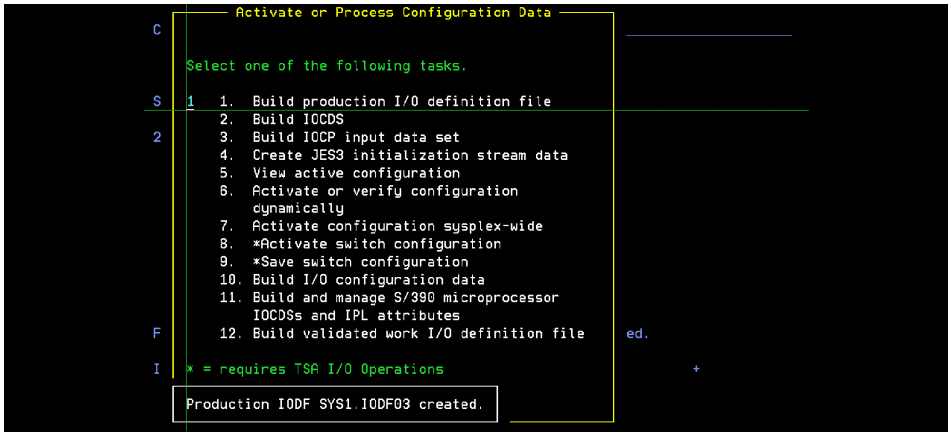
e. Press Enter.



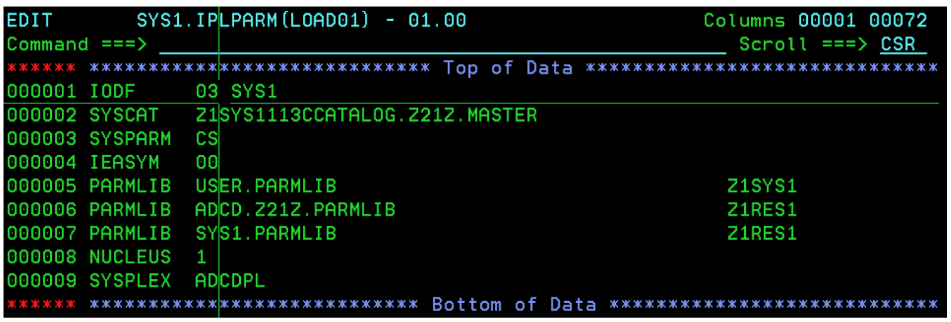
15. When a pop-up window shows as below, specify values for descriptor fields 1 and 2, and then press Enter.



When the creation is successful, the message `Production IODF SYS1.IODF03 created.` is displayed as below.



16. Exit HCD, and then create a new loadparm member by using LOADCS as a template as shown in this example. To use IODF03, change the IODF parm value 99 - 03.



17. IPL the system with loadparm `01` to ensure that IODF03 works without error. If you want to continue using IODF03, modify other loadparms as required. Otherwise, if you want to copy the changes to IODF99, backup IODF99 by using option **6. Maintain I/O definition files**, and then option **2. Copy I/O definition file**. Then, delete file IODF99 and repeat options 6 and 2 to copy IODF03 into IODF99.

Steps for dumping SMF data into GDG data sets

The system is configured to clear SMF data by using the procedure SMFCLEAR. This procedure is triggered by the exit routine IEFU29, and the source code of which is available in `ADCD.LIB.JCL`. If you want to save the SMF data into GDG data sets, modify this exit routine to call procedure SMFDUMPS. Edit the exit routine IEFU29 to change occurrences of string SMFCLEAR to SMFDUMPS, and assemble or link edit the load module by using the JCL member `IEFU29@`, which is also available in `ADCD.LIB.JCL`. To take this change effect, you must re-IPL the system. As a GDG base `SYS1.SMF.DATA` was defined by using the JCL in `ADCD.LIB.JCL(SMFGDG)`, the procedures SMFCLEAR and SMFDUMPS are available in `ADCD.Z22C.PROCLIB`.

Using the esoteric devices

Esoteric devices that are called WORK, TEMP, and SORT are defined and mapped to devices in the range 0600 – 060F. To use these esoterics, create emulated 3390 volumes in your zPDT system, and map them to these device addresses by using the devmap file.

Note that you must update the appropriate `VATLSTxx` member (volume attribute list) with necessary statement, as shown highlighted in the following sample. This example uses volumes WORK01 through WORK12, which are marked as WORK*. This member defines the mount and uses attributes of direct-access volumes.

```
VATDEF IPLUSE(PRIVATE),SYSUSE(PRIVATE)
C2SYS1,0,0,3390 ,Y
WORK* ,0,0,3390 ,Y
```

Starting IBM Health Checker for z/OS

With this edition, IBM Health Checker is preconfigured to start automatically.

To start this utility manually, use the following start command.

```
/START HZSPROC
```

http://www-03.ibm.com/systems/z/os/zos/tools/hchecker/index.html

To stop IBM Health Checker, enter the following command manually, or include it in your shutdown script `SHUTxx` in parmlib.

```
/STOP HZSPROC
```

Related information:

IBM Health Checker for z/OS

Starting IBM z/OS Management Facility (z/OSMF)

A new load parm (IZ) is configured to start the z/OSMF product. These required commands to start z/OSMF are listed below for reference. However, these commands are included in the startup script member VTAMCI/DB/IM/IZ/WA/AL so that z/OSMF starts automatically. The first command starts the CIM (Common Information Model) server, and the second command starts z/OSMF. You must start the CIM server before starting the z/OSMF product. The WebSphere Application Server OEM provides a native application server runtime environment for z/OSMF.

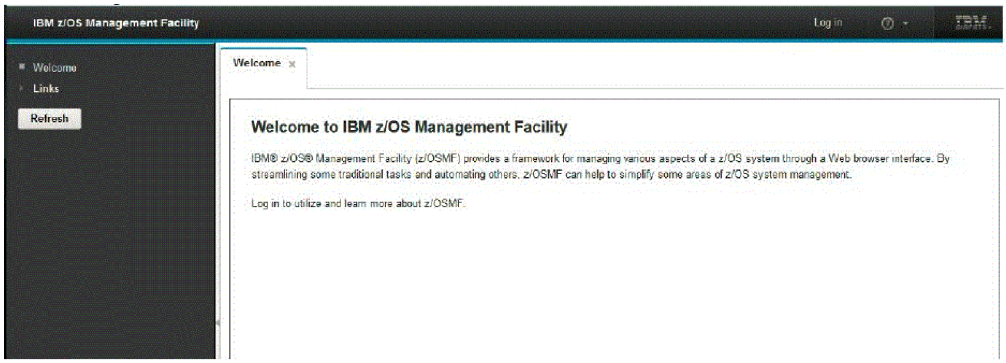
```
S CFZCIM
```

It might take several minutes for these applications to start, which depends on your zPDT system. Refer to the following messages for successful start of the z/OSMF product.

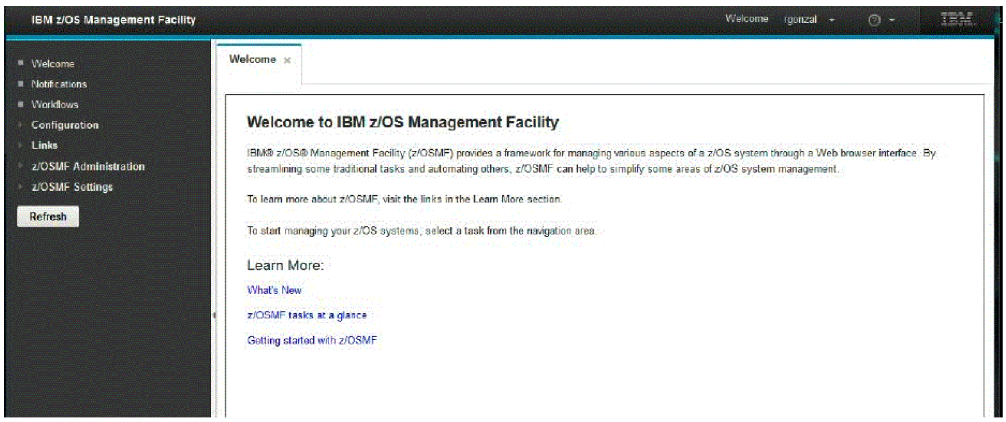
```
S IZUANG1
CWWKB0056I INITIALIZATION COMPLETE FOR ANGEL

S IZUSVR1
IZUG400I: The z/OSMF Web application services are initialized.
+CWWKF0011I: The server zosmfServer is ready to run a smarter planet.
```

To access the z/OSMF console, you can use the URL: https://s0w1.dal-ebis.ihost.com:10443/zosmf

The initial login opens as below.



The welcome screen of z/OSMF console opens as below.



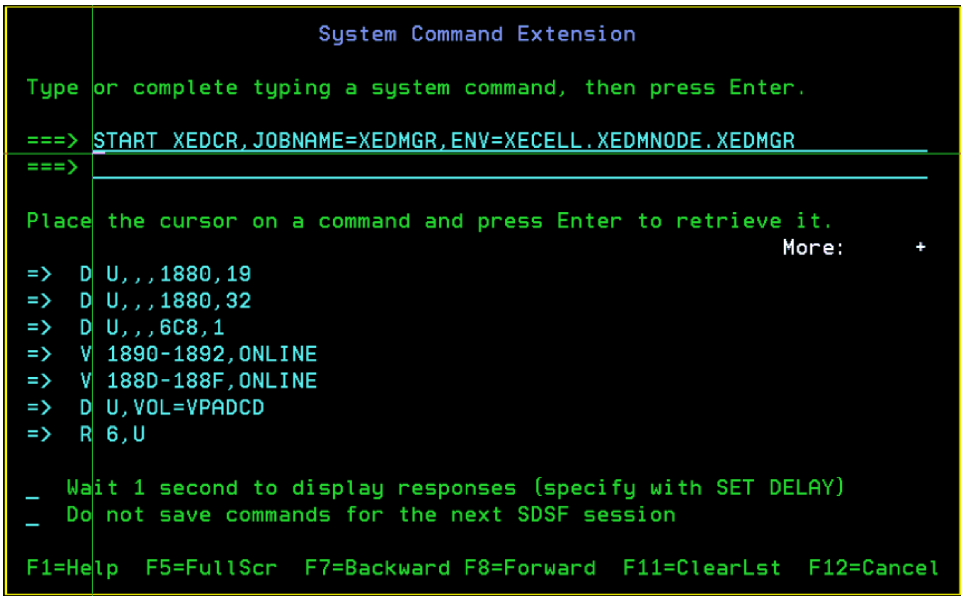
To stop z/OSMF, enter these commands:

```
P IZUSVR1
P CFZCIM
P IZUANG1
```

Starting WebSphere Application Server (WAS V8)

You are recommended to have more than 4 GB real memory on your system and code at least 3000 m (3 GB) in the devmap. IPL the system with loadparm WA and follow these steps to start WebSphere Application Server V8:

1. In SDSF, type / and press Enter to open the System Command Extension window.
2. To start the Deployment Manager, enter this command in the System Command Extension window: `START XEDCR,JOBNAME=XEDMGR,ENV=XECELL.XEDMNODE.XEDMGR`. And then press Enter.



This start command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 478 PROCESS XEDMGR.`

3. To start the Node Agent, enter this command: `START XEACR1,JOBNAME=XEAGNT1,ENV=XECELL.XENODE1.XEAGNT1`.

This command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 346 PROCESS XEAGNT1.`

4. To connect to the Admin console, open a browser and use this URL: https://s0w1.dal-ebis.ihost.com:10443/zosmf..

   A login window with the User ID **xeadmin** opens as below.

5. Click **Log in**.

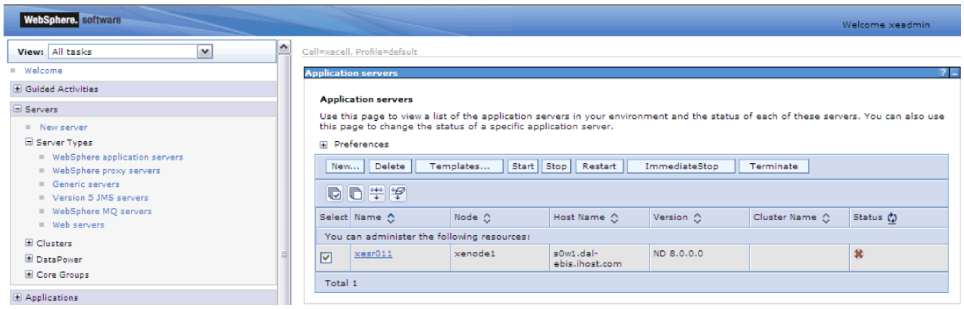
   The Admin console opens.

6. To start the Application Server from the Admin console, follow these steps:

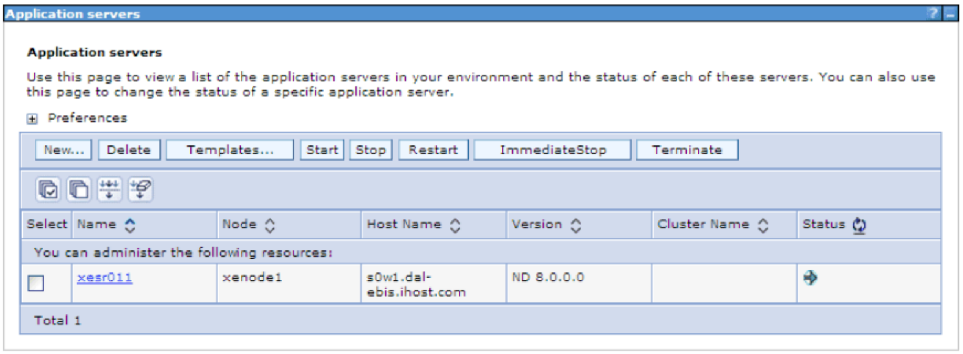
   a. In the navigation list, click **Servers** > **Server Types** > **Websphere application servers**.

   b. In the Application servers view, select the check box beside server **XESR011**.

   c. Click **Start**.



This operation might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 541 PROCESS XESR011.` When the server is started, the **Status** column displays a green arrow:



To stop the application server, do one of these actions:

- From the browser:
  1. Select the server, and then click **Stop**.
  2. On the confirmation window, click **OK**.
- From SDSF, issue this command: `/P XESR011`.

When the application server is shut down, this message is displayed in the system log: `BBOO0002I WEBSPHERE FOR Z/OS CONTROL PROCESS XESR011 ENDED NORMALLY.`

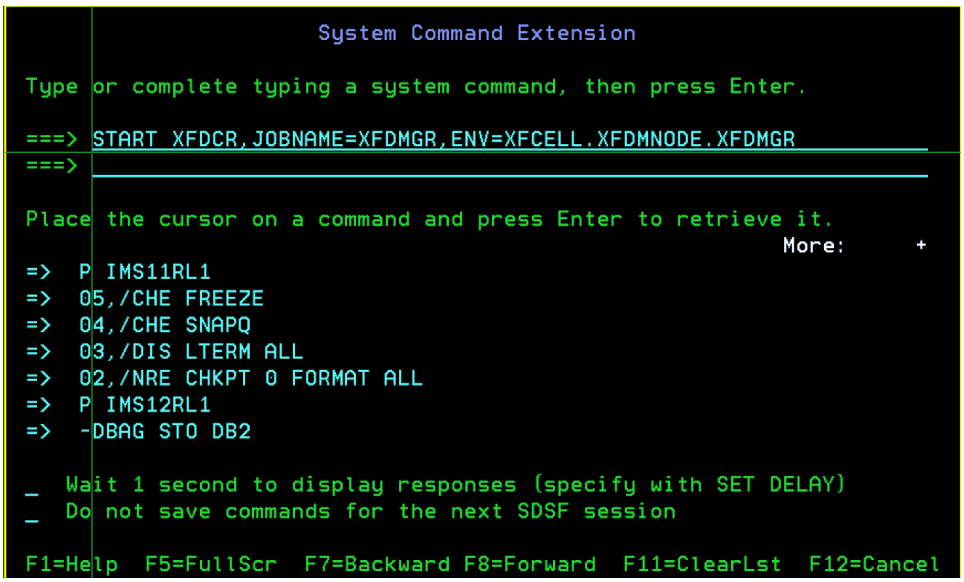
To stop the Deployment Manager and Node Agent, enter this command: `/P XEDEMN`.

The stop command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0008I WEBSPHERE FOR Z/OS DAEMON S0W1 ENDED NORMALLY.`

Starting WebSphere Application Server (WAS V8.5)

You are recommended to have more than 4 GB real memory on your system and code at least 3000 m (3 GB) in the devmap. IPL the system with loadparm WA and follow these steps to start WebSphere Application Server V8.5:

1. In SDSF, type / and press Enter to open the System Command Extension window.
2. To start the Deployment Manager, enter this command in the System Command Extension window: `START XFDCR,JOBNAME=XFDMGR,ENV=XFCELL.XFDMNODE.XFDMGR`. And then press Enter.



The start command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 478 PROCESS XFDMGR.`

3. To start the Node Agent, enter this command: `START XFACR1,JOBNAME=XFAGNT1,ENV=XFCELL.XFNODE1.XFAGNT1.`

   This command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 346 PROCESS XFAGNT1.`

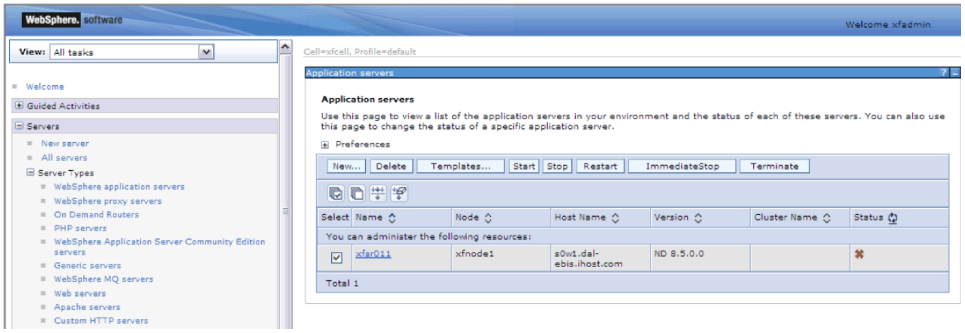
4. To connect to the Admin console, open a browser and use this URL: http://s0w1.dal-ebis.ihost.com:9705/ibm/console.

   A login window with the User ID **xfadmin** opens as below.

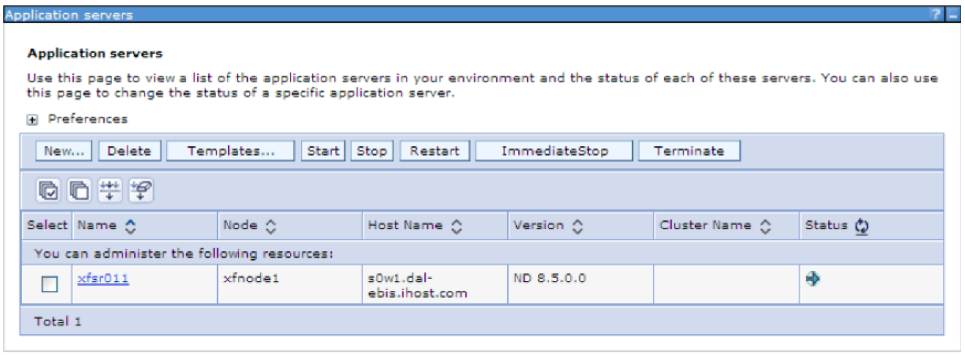
5. Click **Log in**.

   The Admin console opens.

6. To start the Application Server from the Admin console, follow these steps:
   a. In the navigation list, click **Servers** > **Server Types** > **Websphere application servers**.
   b. In the Application servers view, select the check box beside server **XFSR011**.
   c. Click **Start**.



This operation might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 541 PROCESS XFSR011.` When the server is started, the **Status** column displays a green arrow:



To stop the application server, do one of these actions:

- From the browser:
  1. Select the server, and then click **Stop**.
  2. On the confirmation window, click **OK**.
- From SDSF, enter this command: `/P XFSR011`.

When the application server is shut down, this message is displayed in the system log: `BBOO0002I WEBSPHERE FOR Z/OS CONTROL PROCESS XFSR011 ENDED NORMALLY.`

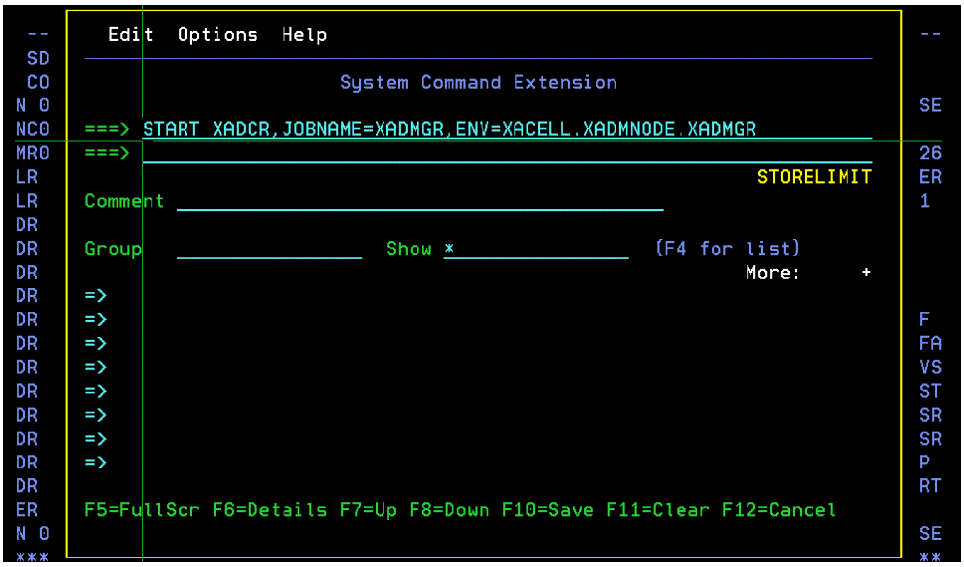
To stop the Deployment Manager and Node Agent, enter this command: `/P XFDEMN`.

The stop command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0008I WEBSPHERE FOR Z/OS DAEMON S0W1 ENDED NORMALLY.`

Starting WebSphere Application Server (WAS V9.0)

You are recommended to have more than 4 GB real memory on your system and code at least 3000 m (3 GB) in the devmap. IPL the system with loadparm WA and follow these steps to start WebSphere Application Server V9.0:

1. In SDSF, type / and press Enter to open the System Command Extension window.
2. To start the Deployment Manager, enter this command in the System Command Extension window: `START XADCR,JOBNAME=XADMGR,ENV=XACELL.XADMNODE.XADMGR`. And then press Enter.



The start command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 478 PROCESS XADMGR.`

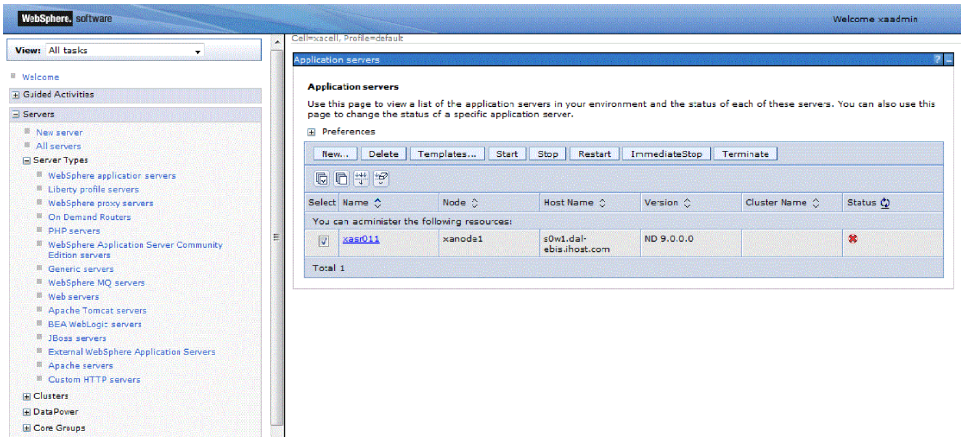
3. To start the Node Agent, enter this command : `START XAACR1,JOBNAME=XAAGNT1,ENV=XACELL.XANODE1.XAAGNT1.`

   This command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 346 PROCESS XAAGNT1.`

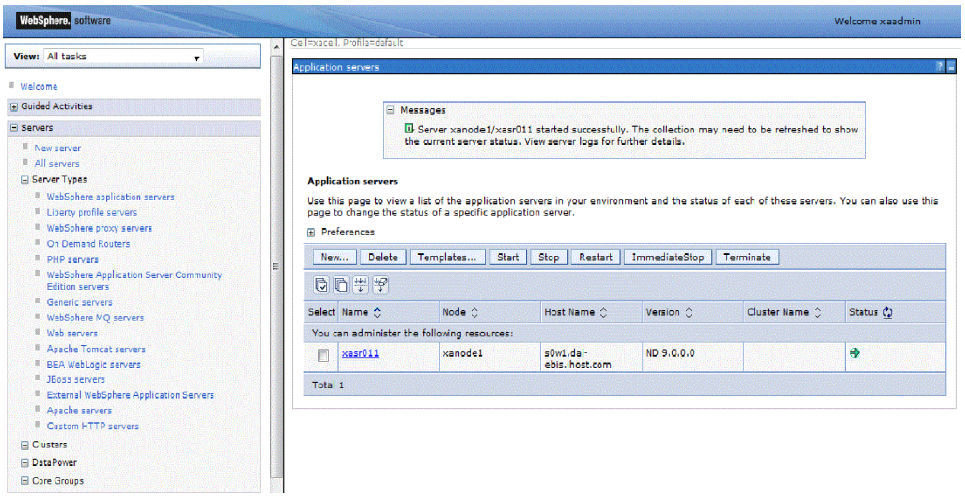
4. To connect to the Admin console, open a browser and use this URL: http://s0w1.dal-ebis.ihost.com:9305/ibm/console.

   A login window with the User ID **xaadmin** opens as below.

5. Click **Log in**.
   The Admin console opens.
6. To start the Application Server from the Admin console, follow these steps:
   a. In the navigation list, click **Servers** > **Server Types** > **Websphere application servers**.
   b. In the Application servers view, select the check box beside server **XFSR011**.
   c. Click **Start**.



This operation might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0019I INITIALIZATION COMPLETE FOR WEBSPHERE FOR Z/OS CONTROL 541 PROCESS XASR011.` When the server is started, the **Status** column displays a green arrow:



To stop the application server, do one of these actions:

- From the browser:
  1. Select the server, and then click **Stop**.
  2. On the confirmation window, click **OK**.
- From SDSF, enter this command: `/P XASR011`.

When the application server is shut down, this message is displayed in the system log: `BBOO0002I WEBSPHERE FOR Z/OS CONTROL PROCESS XASR011 ENDED NORMALLY.`

To stop the Deployment Manager and Node Agent, enter this command: `/P XADEMN`.

The stop command might take several minutes to complete. When it is completed, this message is displayed in the system log: `BBOO0008I WEBSPHERE FOR Z/OS DAEMON S0W1 ENDED NORMALLY.`

Index

Special characters

A

C

E

I

J

L

M

N

O

P

R

S

T

U

W

Z