JD Edwards EnterpriseOne Tools

Server and Workstation Administration Guide

Release 8.98 Update 4

E14718-03

May 2012

ORACLE

JD Edwards EnterpriseOne Tools Server and Workstation Administration Guide, Release 8.98 Update 4

E14718-03

Contents

3 Administering the UNIX and Linux Servers

4 Administering the Windows Server

13 Troubleshooting the Workstation

14 Troubleshooting the Enterprise Server

A Using the Microsoft Visual C++ 2005 or Higher Level Compiler

Preface

Welcome to the JD Edwards EnterpriseOne Tools Server and Workstation Administration Guide.

Audience

This guide is intended for system administrators who are responsible for administering servers and workstations.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/us/corporate/accessibility/index.html.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

Related Documents

You can access related documents from the JD Edwards EnterpriseOne Release Documentation Overview pages on My Oracle Support. Access the main documentation overview page by searching for the document ID, which is 876932.1, or by using this link:

https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=876932.1

To navigate to this page from the My Oracle Support home page, click the Knowledge tab, and then click the Tools and Training menu, JD Edwards EnterpriseOne, Welcome Center, Release Information Overview.

This guide contains references to server configuration settings that JD Edwards EnterpriseOne stores in configuration files (such as jde.ini, jas.ini, jdbj.ini, jdelog.properties, and so on). Beginning with the JD Edwards EnterpriseOne Tools Release 8.97, it is highly recommended that you only access and manage these settings for the supported server types using the Server Manager program. See the Server Manager Guide on My Oracle Support.

Conventions

The following text conventions are used in this document:

Convention	Meaning
Bold	Indicates field values.
Italics	Indicates emphasis and JD Edwards EnterpriseOne or other book-length publication titles.
`Monospace`	Indicates a JD Edwards EnterpriseOne program, other code example, or URL.

1

Introduction to JD Edwards EnterpriseOne Tools Server and Workstation Administration

This chapter contains the following topics:

1.1 Server and Workstation Administration Overview

Server and Workstation Administration is used to extend an initial installation prototype environment to meet practical requirements and recognizes, addresses, and solves daily issues that arise in a dynamic enterprise. Server and Workstation Administration uses the flexibility of Oracle's JD Edwards Configurable Network Computing architecture to optimize Oracle's JD Edwards EnterpriseOne installation for the enterprise.

1.2 Server and Workstation Administration Implementation

In the planning phase of your implementation, take advantage of all JD Edwards sources of information, including the installation guides and troubleshooting information.

2

Administering the IBM i Server

This chapter contains the following topics:

2.1 Understanding Server Administration for IBM i

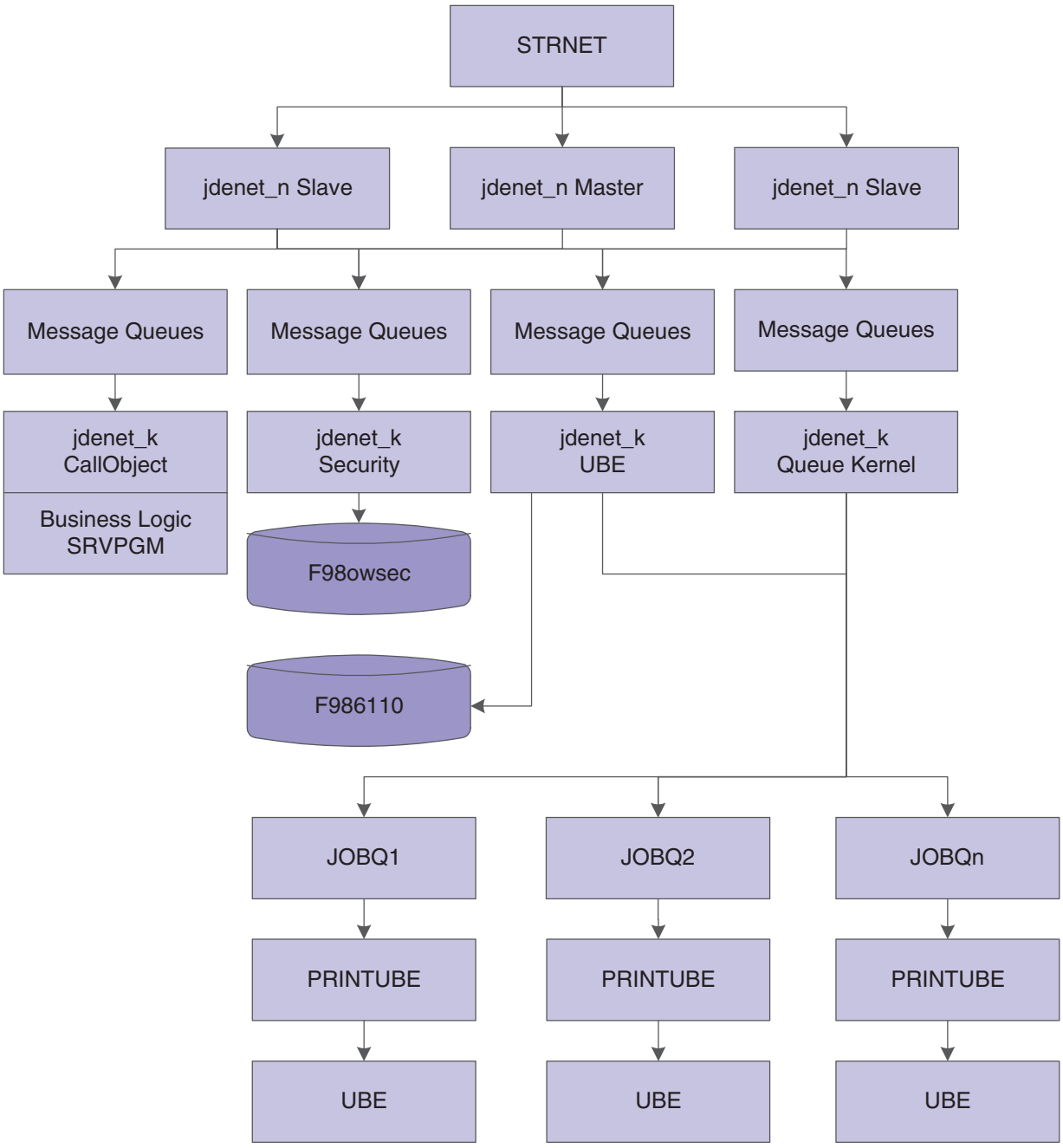
Oracle's JD Edwards EnterpriseOne enterprise servers are supported on the IBM i platform. The IBM i enterprise server can operate in a logic server or database server environment. You need to perform certain administration procedures on the enterprise server to ensure that JD Edwards EnterpriseOne runs properly. This section discusses:

- JD Edwards EnterpriseOne IBM i Architecture and Process Flow for IBM i.
- JD Edwards EnterpriseOnIBM initialization for IBM i.

2.1.1 JD Edwards EnterpriseOne IBM i Architecture and Process Flow for IBM i

This flowchart illustrates the actions that the host server processes perform:

Figure 2–1 IBM i Process Flow



All communications between the client and the host server occur using sockets. The communications between JDENET_N (network processes) and JDENET_K (kernel processes) occur with shared memory.

The process flow is:

1. The STRNET command runs the master NETWORK (JDENET_N) job in a newly started subsystem. The jdenet_n Master process spawns jdenet_n slave and jdenet_k processes (also called kernels) at startup or as they are needed. JD Edwards EnterpriseOne uses a number of different types of kernels to handle different types of processing, even though all of these have the same process name

in the operating system (jdenet_k). The definitions for the number of processes to start and what types to start are stored in the jde.ini file.

2. The JDENET_N process listens to the socket (port) as specified in the jde.ini file by the keywords ServiceNameListen and ServiceNameConnect. These two keywords should be set to the same number, and this number must be the same for every client who wants to connect to the JD Edwards EnterpriseOne server. The definitions for the particular jdenet_k processes to start are also given in the jde.ini file. They are listed in the sections headed by [JDENET_KERNEL_DEFx]. Each of these entries lists the type of jdenet_k processes to start and the maximum number of JDENET_K processes of this type to start.

   The number of JDENET_N slave processes to start is listed in the jde.ini file under the keyword maxNetProcesses. The purpose of these slave processes is to provide parallel processing for the job of listening to the socket and to put the associated messages on the message queues for the JDENET_K processes to finish.

3. JDENET_K processes (kernel processes) do the actual work on the enterprise server. When a JDENET_K process starts, it can be any type of kernel process. The JDENET_N process assigns each kernel process to a certain type.

4. The JDENET_K process that becomes a CallObject kernel has the job of calling business function logic on the server. Business function logic is written in C code and compiled into Service Program (SRVPGM). SRVPGM is loaded onto the JDENET_K processes and then called directly through a C function call.

5. The JDENET_K process that becomes a batch process kernel waits for requests to run batch processes from the client. When a request to run a batch process is submitted, these events occur:

   – JDENET_K (UBE kernel) adds a record to the F986110 database table with a status of W for waiting.

   – JDENET_K (UBE kernel) submits a job to the queue

     If you are using native IBM i job queues, JDENET_K submits a job to the IBM i queue. This job calls the JD Edwards EnterpriseOne program PRINTUBE on the IBM i enterprise server.

     If you are using the JD Edwards EnterpriseOne queue kernel, JDENET_K sends a message to the queue kernel, alerting it that a new job request was submitted. When the job is ready, the queue kernel executes the PRINTUBE program.

6. The PRINTUBE process runs the batch application, and changes the status of the record in the F986110 table to P for processing.

7. If the batch application runs successfully, the software changes the status of the record in the F986110 table to D for done.

   If the batch application fails, JD Edwards EnterpriseOne changes the status of the record in the F986110 table to E for error.

2.1.2 JD Edwards EnterpriseOne Initialization for IBM i

This initialization occurs when you start JD EdwardsEnterpriseOne programs such as PRINTUBE:

- The JD Edwards EnterpriseOne environment name is passed as an argument to the program.

- This environment might be translated to a different environment, based on the settings in the [SERVER ENVIRONMENT MAP] section of the .INI file.
- The software verifies that the environment is a valid entry in the Library ListMaster File table (F0094) and that it has a valid corresponding path code in the Environment Detail - OneWorld table (F00941).
- The Library .INI file setting in the [DB SYSTEM SETTINGS] section indicates where the JD Edwards EnterpriseOne server startup tables, such as Data Source Master (F98611), Object Configuration Master (F986101), and so on, are located.
- Using this information, the software opens the F986101 (OCM) table in the specified database on the server.
- If an override for a given table, BSFN, and so on, or the current user exists, that data source (the OMDATP field in the F986101 table) is used for the given object or user and environment. Otherwise, the data source in which OMOBNM=DEFAULT for the given environment is used. Ignore any inactive records (that is, OMSTSO=NA).

  Note: We highly recommend that you do not have any default (OMOBNM=DEFAULT) records for reports (OMFUNO=UBE) or for BSFNs that are mapped to the server. These records might prevent report interconnections (one report calling another report) from starting correctly.

  Each unique data source in the F986101 table should correspond to one entry in the F98611 table. The corresponding information in the F98611 table must be correct. In particular, the OMDLLNAME field must display the correct SRVPGM (.DLL) for the database to which the data source points:

- DBDR for files located on the IBM i enterprise server.
- JDBNET for files not located on the IBM i enterprise server.

2.2 Starting the Enterprise Server for IBM i

This section provides overviews of the JD Edwards EnterpriseOne library structure and startup options for IBM i, lists prerequisites, and discusses how to:

- Start the enterprise server for IBM i manually.
- Start the enterprise server for IBM i automatically.

2.2.1 Understanding the IBM i Library Structure for JD Edwards EnterpriseOne

You can set up an initial program to create the library list. Also, you should add this library to the top of the library list before you start JD Edwards EnterpriseOne on the enterprise server: releaseSYS (or the system library name). The variable release is the JD Edwards EnterpriseOne release level, such as E900SYS.

The releaseSYS library contains these objects:

Object	Description
INI	The jde.ini file used to initialize JD Edwards EnterpriseOne on the IBM i enterprise server.

Object	Description
*PGM and *SRVPGM	The various programs and service programs required to run the JD Edwards EnterpriseOne IBM i enterprise server.
CHGLIBOWN (*CMD)	A JD Edwards EnterpriseOne utility command used to change ownership of all objects contained in a library.
SHOW (*CMD)	A JD Edwards EnterpriseOne utility command used to display runtime output.
UPDLF (*CMD)	A JD Edwards EnterpriseOne utility command used to modify the maintenance attribute of logical files.
DPSPSTMF (*CMD)	The display stream file, which displays IBM i Integrated File System (IFS) text stream files. The JD Edwards EnterpriseOne log files, jde.log and jdedebug.log, typically reside in a directory called PSFTrelease, where release represents the JD Edwards EnterpriseOne release, such as /PSFT900.
LINKBSFN (*CMD)	The command used to relink business functions to their respective service programs (*SRVPGM). Typically, the system uses this command during an upgrade of the JD Edwards EnterpriseOne system library.
PID2JOB (*CMD)	The Convert Process ID to Job command, which returns the job information when the system passes a process ID to the command. The system writes the process ID in the JDE.LOG files. This command returns job information only while the job is still active.
PORTTEST (*CMD)	The command that runs the JD Edwards EnterpriseOne test program PORTTEST.
RUNUBE (*CMD)	The command that interactively runs a batch program. If you need to run a batch program, use the SBMJOB command to submit the RUNUBE command to batch.
PRINTQUEUE (*FILE)	The file that contains the output from a batch program. This output is stored as ASCII PDF members.
*PGM and *SRVPGM	The programs and server programs required to run the JD Edwards EnterpriseOne network.
JDENET (*JOBQ)	The job queue used by the JD Edwards EnterpriseOne IBM i network jobs.
NETJOBD (*JOBD)	The job description used by JD Edwards EnterpriseOne IBM i network jobs.
JDENET (*CLS)	The class used to create the routing entry for the JDENET subsystem.
ENDNET (*CMD)	The command that ends the JD Edwards EnterpriseOne IBM i network jobs and cleans up the network runtime structures.
IPCS (*CMD)	The utility command that indicates the status of objects used by the JD Edwards EnterpriseOne IBM i network jobs and as a backup method for cleaning up the IPCS objects.
STRNET (*CMD)	The command that starts the JD Edwards EnterpriseOne IBM i network jobs.
CLRIPC (*CMD)	The command used to clear IPC structures.
DSPIPC (*CMD)	The command used to display IPC structures.
PSFTrelease (*SBSD)	The subsystem description under which the JD Edwards EnterpriseOne network jobs run. The variable release is the JD Edwards EnterpriseOne release level, such as PSFT900.

2.2.2 Understanding Startup Options for the Enterprise Server for IBM i

You can start the JD Edwards EnterpriseOne enterprise server for the IBM i either manually or automatically.

You manually start the enterprise server for IBM i by starting JDENet from the command line, and then starting the PORTTEST program, which verifies that the enterprise server software was installed correctly. If it was, PORTTEST initializes an environment and user.

If you start the server automatically, it is recommended that you separate the JD Edwards EnterpriseOne add library list entry (ADDLIBLE) and startup (STRNET) commands from the IBM i startup program. You should create a separate JD Edwards EnterpriseOne startup program and call that program from the IBM i startup program. This action ensures that commands subsequent to the JD Edwards EnterpriseOne add library list entry and startup are not associated with the modified library list. This recommendation also ensures that the JD Edwards EnterpriseOne library list is set correctly before issuing the STRNET command. In addition, the separately-called program provides you with a single location in which to locate and maintain JD Edwards EnterpriseOne startup commands on the IBM i.

2.2.3 Prerequisites

Before you complete the tasks in this section:

- Install JD Edwards EnterpriseOne as described in the *JD Edwards EnterpriseOne Installation Guide.* In that guide, you should have performed all steps up to the Installation Workbench.
- Run the clear CLRIPC command before you start the server to ensure that the server is clear. If you do not run this command prior to starting a server, the startup process will fail.

2.2.4 Starting the Enterprise Server for IBM i Manually

To start the enterprise server for IBM i manually:

1. Sign on to the IBM i as **ONEWORLD.**
2. Start JDENet using this command:

   ```
   STRNET
   ```

3. Start the PORTTEST program using this command to verify that the basic enterprise server software was correctly installed:

   ```
   PORTTEST userID password environment
   ```

   Where userID represents the JD Edwards EnterpriseOne IBM i user ID, password represents the password, and environment represents the environment that you want to test.

   The PORTTEST program initializes an environment and user if JD Edwards EnterpriseOne was correctly installed and configured. The program opens a table and displays up to 99 rows of data. You should see results similar to those in this example:

   ```
   Running porttest for JDESVR on M9ASD2 with password JDESVR
   Initializing Environment M9ASD2,...
   Environment M9ASD2 was initialized successfully.
   Initializing JDESVR/JDESVR (User/Password),...
   ```

```
JDESVR/JDESVR (User/Password) Initialized successfully.
Opening table F986110,...
Opened table F986110 successfully.
Closing table F986110,...
Closed table F986110 successfully.
Opening table F0902,...
Opened table F0902 successfully.
Performing select all on table F0902,...
Select all on F0902 succeeded.
Printing up to 99 records in the table F0902,...
f0902.gbaid f0902.gbawtd
------------ ------------
[98] 00009697 24060973
[97] 00009806 13540877
[96] 00010102 3140380...
[1] 00068798 10000
[0] 00058798 250000
Total number of rows printed = 99
Calling DataDictionary Validation function,...
Data Dictionary Validation Succeed for CO 00001.
Closing table F0902,...
Closed table F0902.
Freeing user JDESVR,...
Freed user JDESVR successfully.
Cleaning up environment M9ASD2,...
Cleaned up environment M9ASD2 successfully.
Congratulations! Porttest completed successfully.
All Done!
BYE!
```

If the table in the environment that you specified is empty, the total number of records that the program prints will equal zero.

4. Enter this command:

```
WRKACTJOB SBS(PSFTrelease)
```

The variable release is the JD Edwards EnterpriseOne release level that the site is using, such as PSFT900.

5. Verify that the entry NETWORK with function PGM-JDENET_N and status of SELW is running (until a net request is performed, the CPU will be 0).

2.2.5 Starting the Enterprise Server for IBM i Automatically

To start the enterprise server for IBM i automatically:

1. Create a CL program.

   You will use this program to establish the appropriate JD Edwards EnterpriseOne library list and execute the command to start the IBM i server job (JDENet).

   The CL program should be similar to:

```
PGM
CHGLIBL LIBL(E900SYS QTEMP QGPL)
STRNET
ENDPGM
```

2. Identify and modify the program called during the IBM i IPL to submit a job to call the previous program.

The program name and location are set in the IBM i system value, QSTRUPPGM.

3. Determine the QSTRUPPGM value by entering this command:

   ```
   DSPSYSVAL SYSVAL(QSTRUPPGM)
   ```

4. Determine where the source of the program is located by executing this command against the library and program (as set in the system value):

   ```
   DSPPGM LIBRARY/PROGRAM NAME
   ```

5. Modify the source of the startup library and program by inserting a SBMJOB command that calls the program created in Step 1.
6. Verify that the startup program is created correctly by recreating it and ensuring that it is created in the library specified by the system value.

   Use CRTCLPGRM and prompt (using F4) for the appropriate parameters.

2.3 Shutting Down the Enterprise Server for IBM i

You can manually shut down the enterprise server for the IBM i with the command, ENDNET. This command is in the system library. For example, ENDNET causes JD Edwards EnterpriseOne to end the JDENet jobs and clean up all JDENet runtime structures.

2.3.1 Prerequisite

Ensure that the library is set correctly before performing this command.

See Also:

- Understanding the IBM i Library Structure for JD Edwards EnterpriseOne.

2.4 Using IBM i Integrated File System Logging Support

To achieve better performance and to allow easier access to log files from the workstation, JD Edwards EnterpriseOne generates log files for the IBM i in the Integrated File System (IFS) rather than the traditional file system on the IBM i.

With IFS, JD Edwards EnterpriseOne generates log files as stream files (STMF) in an IFS directory, based on the IBM i jde.ini file settings.

2.4.1 Example: Easy Access to Log Files

These examples illustrate how to modify the jde.ini file to enable easier access to log files from the workstation:

```
[DEBUG]
DebugFile=jdedebug
JobFile=jde.log
```

JD Edwards EnterpriseOne generates log files in the IFS root directory.

```
[DEBUG]
DebugFile=/psft900_a/jdedebug
JobFile=/psft900_a/jde.log
JD Edwards EnterpriseOne generates log files in the IFS directory called /psft900_
a.
```

Note: The directory must exist with proper authority granted to the logging job.

2.5 Cleaning Up the Enterprise Server for IBM i

This section provides an overview of enterprise server cleanup for IBM i and discusses how to:

- Clean up the enterpriser server for IBM i.
- Clear the jde.log and jde.debug files for IBM i.

2.5.1 Understanding Enterprise Server Cleanup for IBM i

When JD Edwards EnterpriseOne ends abnormally, you might need to manually perform cleanup tasks on the IBM i enterprise server. Interprocess Communication (IPC) structures might not be cleaned up after an execution of ENDNET, which might cause further problems when you start JDENet. If the IPC structures are not properly removed by ENDNET, you can manually remove them. IPC structures might become locked by an interactive job. For example, you might have to sign off and sign back on to perform a successful cleanup.

The JD Edwards EnterpriseOne IBM i server is shipped with the DSPIPC and CLRIPC commands, which enable you to display the IPC-related information and to remove IPC structures.

If tracing is turned on in addition to IPC, you should clear the jde.log and jdedebug files. This action keeps the files from becoming too large and removes old messages from it.

Note: Clear IPC structures only when you are ready to restart the JDENet process.

2.5.2 Prerequisite

Ensure that the library list is correct before executing the IPC commands. Each of the commands calls the IPCS command for all of the IPC types. Each command has two parameters: from and to. Use these parameters to specify the starting and ending IPC addresses on which you want to operate. The default value for the from parameter is *INI; this is the address specified in the .INI file. The default value for the to parameter is *CALC; this means that the value is calculated based on the value of the from parameter. For example, you could specify 999 more than the from parameter.

Note: IBM Opti-Connect and Opti-Mover products use the IPC shared memory address 9999. Avoid setting the jde.ini file setting IPCStartKey to a starting value that uses the range of 9000 to 9999.

2.5.3 Cleaning Up the Enterprise Server for IBM i

To clean up the enterprise server for IBM i:

From an IBM i command line, enter these IPCS commands:

```
DSPIPC

CLRIPC
```

2.5.4 Clearing the jde.log and jde.debug Files for IBM i

For IBM i:

1. To clear the jde.log stream files, enter this command:

   ```
   DEL `/PSFTrelease/jde_*'
   ```

   Where release is the JD Edwards EnterpriseOne release, such as psft900.

2. To clear the jdedebug log, enter this command:

   ```
   DEL `/PSFTrelease/jdedebug_*'
   ```

   Where release is the JD Edwards EnterpriseOne release, such as psft900.

2.6 Setting Up a Printer for IBM i

This section provides an overview of printer setup for IBM i and discusses how to:

- Create the OUTQ.
- Start the OUTQ.
- Print multiple copies to a remote printer.

2.6.1 Understanding Printer Setup for IBM i

For printing, JD Edwards EnterpriseOne IBM i servers generate PostScript, PCL, or line printer reports. The line printer OUTQ configuration is similar to most typical IBM i OUTQ configurations. This section provides the steps necessary to set up the Postscript and PCL OUTQ configurations.

Unless otherwise specified in the printer definition, the default OUTQ used for printing batch process reports is the same as the default OUTQ of the user submitting the job.

See Also:

- "Defining Print Properties for Reports" in the *JD Edwards EnterpriseOne Tools Development Tools: Report Printing Administration Technologies Guide*.

2.6.2 Creating the OUTQ

To create the OUTQ, enter this command:

```
CRTOUTQ OUTQ(QGPL/outqname) RMTSYS(*INTNETADR) RMTPRTQ(`')
CNNTYPE(*IP) DESTTYPE(*OTHER) TRANSFORM(*NO) INTNETADR(`IP Address of
your printer')
```

Note: Some printers require that you set the parameter RMTPRTQ to something other than `'. See the instruction manual for the printer for additional information. For example, you must set this parameter to PASS for the IBM Network Printer 4317.

2.6.3 Starting the OUTQ

To start the OUTQ:

1. Enter this command:

   ```
   STRRMTWTR outqname
   ```

   For example:

   ```
   STRRMTWTR QGPL/JDE_HP4PSB
   ```

2. If you need to release the outqueue before using it, enter this command:

   ```
   RLSOUTQ outqname
   ```

   For example, enter DEL '/PSFTrelease, where release is the JD Edwards EnterpriseOne release, as in PSFT900.

2.6.4 Printing Multiple Copies to a Remote Printer

This task is necessary only when the output queue does not support printing multiple copies, and it applies to remote output queues only. Only system administrators can print multiple copies to a remote printer.

1. End the remote writer to which the output queue is connected.
2. Use the Change Output Queue (CHGOUTQ) command to change the Display Options (DSPOPT) parameter so that it contains the value XAIX.
3. Restart the remote writer.

   The output queue should now be able to send multiple copies of the documents to the remote printer.

2.7 Administering Batch Processes for IBM i

This section provides an overview of batch process administration for IBM i and discusses how to:

- Monitor batch processes.
- Review batch output files.
- Encode the passwords of users who submit batch jobs.

2.7.1 Understanding Batch Process Administration for IBM i

Administering batch processes involves knowing what processes run when JD Edwards EnterpriseOne starts, where files are placed before and after printing, and how to watch those processes.

Depending on how the software is installed, jobs run under several subsystems on the IBM i. The first subsystem, PSFT900, is created during the installation process and is responsible for running the JD Edwards EnterpriseOne net and kernel processes. QBATCH is the default subsystem in which jobs run, but you can use other subsystems to distribute the workload.

When you send a batch process report to an IBM i server for processing, the network jobs are responsible for accepting and queuing the request, while the QBATCH subsystem is responsible for executing the report. To monitor the batch requests, use the WRKACTJOB command, specifying QBATCH as the subsystem.

A job appears indented underneath the subsystem. A job such as the R0006P job is the actual report that is running at this time. The program PRINTUBE is the job that is responsible for running and printing the request. When the job is finished, it leaves the queue, and the print job is either printed and deleted, or saved in the E900SYS/PRINTQUEUE file.

When users submit batch reports to run on the IBM i, a corresponding Portable Document Format (PDF) file is created on the enterprise server.

The default location for the PDF files is under the PRINTQUEUE folder of the installation directory in IFS, for example, /E900SYS/PRINTQUEUE. Users can access the PDF files directly on the enterprise server, or go to the submitted jobs on the client and view the PDF file.

The naming convention for each member is based on the JD Edwards EnterpriseOne job number, which is a unique number that the system assigns when the report is submitted. This number is a unique print request ID and increments each time a report is submitted to the enterprise server, regardless of whether the job is successful or fails. It is not related to the process ID or job number that the IBM i assigns to the batch job.

If you submit a batch process report to a specific server, the OUTQ for printing dependents on the jde.ini file settings for the workstation. You must change the default OUTQ specified in the jde.ini file of the enterprise server. This setting is in the [Network Queue Settings] section and is called DefaultPrinterOUTQ. This OUTQ is used when no OUTQ is passed to the enterprise server from the workstation, or when the OUTQ name that is passed to the enterprise server is Default.

Two other settings, based on the jde.ini file on the workstation, tell the server whether to print the report immediately upon completion and whether to save the output from the report or delete it. Both of these settings are set in this manner:

```
[NETWORK QUEUE SETTINGS]
SaveOutput=TRUE
PrintImmediate=TRUE
```

Setting SaveOutput to TRUE causes the enterprise server to save the PDF files in E900SYS/PRINTQUEUE until you explicitly delete them. Setting PrintImmediate to TRUE tells the enterprise server to print the job immediately after completing the report.

You should encourage workstation users to use the SaveOutput=FALSE entry in their jde.ini files. If users at workstations decide to save their output, they should periodically delete the entries using the correct JD Edwards EnterpriseOne Job Master Search in the Job Control Master program (P986110B).

Note: To display job numbers, end-users can use the Job Control Master program (P986110B). Similarly, system administrators can use the Work With Servers application (P986116). While both applications perform similar functions, most sites generally use security to restrict access to the Work With Servers application to system administrators. Both programs use the Job Master Search form to display job numbers that correspond to member names. You can use either program to delete .PDF files by deleting appropriate entries.

Finally, if you have the proper authority, you can run batch process reports from the server command line with this command:

```
RUNUBE USER(USER) PASSWORD(PASSWORD) ENVIRON(ENVIRONMENT)
```

```
REPORT(REPORTNAME) VERSION(VERSION)
```

2.7.1.1 Example: Running Reports from the Command Line for IBM i

This example displays a command for executing the Business Unit Report (R0006P):

```
RUNUBE USER(SF5488324) PASSWORD(PASSWORD) ENVIRON(PD900)
REPORT(R0006P) VERSION(XJDE0001)
```

This command begins processing version XJDE0001 of the report in the PD900 environment. After completion, the PostScript spool file resides on the printer_1 OUTQ. The spool file leaves printer_1, and the .PDF file is not deleted.

2.7.1.2 Example: Scheduling Reports from the Command Line for IBM i

You can schedule a report from the command line for processing on a future date. You do this with the SBMJOB (submit job) command. Many options are available for this command, but the general form will be similar to these example:

```
SBMJOB CMD(RUNUBE USER(SF5488324) PASSWORD(PASSWORD) ENVIRON(PD900)
REPORT(R0006P) VERSION(XJDE0001)) SCDDATE(*FRI) SCDTIME(0600)
```

This command schedules the XJDE0001 version of the Business Unit Report (R0006P) to run on the next Friday at 06:00am. This job is submitted in the default job queue for the user who submitted the job. You can specify overrides on the command line or by prompting (F4) for more information.

You can review reports that have been submitted in this method by using the WRKSBMJOB command. This command displays all jobs submitted by the current user for batch processing. Information that this command displays includes the job name, the user who submitted the job, the type of job (BATCH), and the status. Using F11 also displays scheduling information for jobs that have been submitted but not yet run.

2.7.2 Monitoring Batch Processes

To monitor batch processes:

1. Sign on to the IBM i enterprise server using an administrative account.
2. Enter this command, substituting Subsystem with the appropriate subsystem name:

   ```
   WRKACTJOB SBS(Subsystem)
   ```

2.7.3 Reviewing Batch Output Files

To review the PDF output files:

1. From Windows Explorer, use this command to map a drive to the root directory of IFS on the IBM i machine:

   //machinename/root

2. Navigate to the PrintQueue folder in the System directory (for example, the directory might be called /E900SYS/PrintQueue), and view the PDF files.

2.7.4 Encoding the Passwords of Users Who Submit Batch Jobs

On the IBM i, when you want to encode user passwords for batch jobs, you need to change settings in the [SECURITY] section of the JDE.INI file.

Change these setting in the JDE.INI file to False to deactivate encoding:

```
[SECURITY]

ServerPswdFile=TRUE
```

2.8 Running Multiple Instances of JD Edwards EnterpriseOne on the IBM i

This section provides overviews of running multiple instances of JD Edwards EnterpriseOne and database security parameters on the IBM i and discusses how to:

- Copy libraries and directories.
- Apply security to multiple instances of JD Edwards EnterpriseOne on the IBM i.
- Create a JD Edwards EnterpriseOne subsystem on the IBM i.

Server Manager fully supports multiple foundations. This includes the installation and management of multiple instances of JD Edwards EnterpriseOne on a single server.

See Also: ■8.98 Server Manager Guide on My Oracle Support, sections: Register/Install on JD Edwards Enterprise Server for EnterpriseOne..

2.8.1 Understanding Running Multiple Instances of JD Edwards EnterpriseOne

You might want to run multiple instances of JD Edwards EnterpriseOne on an IBM i server for these reasons:

- To test a new service pack.
- To upgrade to a new version of JD Edwards EnterpriseOne.

Note: You cannot use JD Edwards EnterpriseOne Planner to help you set up data for multiple instances of JD Edwards EnterpriseOne. Be prepared to manually copy data and to set up new Object Configuration Manager (OCM) mappings for each new instance.

A JD Edwards EnterpriseOne instance on the IBM i server is uniquely identified by these objects:

- JD Edwards EnterpriseOne system directory (integrated file system, or IFS) and library (QSYS file system).
- Path codes (IFS and QSYS file systems).
- Use of selected .ini file settings.

The JDE.INI settings that you use to uniquely define a JD Edwards EnterpriseOne instance are summarized in this table:

Section in server JDE.INI file	Parameter	Purpose
[INSTALL]	DefaultSystem=	The name of the JD Edwards EnterpriseOne System library. This value must be unique for each JD Edwards EnterpriseOne instance.
[JDEIPC]	StartIPCKeyValue=	The value of the first interprocess communication (IPC) ID of a range of keys, which JDEIPC uses for shared memory. This value, plus the value of the maxNumberofResources parameter, defines the range of IPC IDs that the software uses for an instance of JD Edwards EnterpriseOne.
[JDENET]	ServiceNameListen=	The TCP/IP port number that the server uses for receiving communications packets from workstations and other JD Edwards EnterpriseOne servers.
[JDENET]	ServiceNameConnect=	The TCP/IP port number that the server uses for sending communications packets to workstations or other JD Edwards EnterpriseOne servers.
[DBSYSTEM SETTINGS]	Default Env=	The default environment for an instance of JD Edwards EnterpriseOne.
[DB SYSTEM SETTINGS]	Default PathCode=	The data source for an instance of JD Edwards EnterpriseOne.
[DB SYSTEM SETTINGS]	Library=	The database library that stores the system tables used by JD Edwards EnterpriseOne at startup.

Similarly, to apply JD Edwards EnterpriseOne security throughout multiple instances, you use these items to uniquely identify an instance:

- OCM mappings.
- Database.
- JD Edwards EnterpriseOne user profile (the owner and default user ID under which JD Edwards EnterpriseOne jobs start).
- Selected settings in the JDE.INI file.

The JDE.INI settings that you use to uniquely define a JD Edwards EnterpriseOne instance when you are applying security throughout multiple instances are summarized in this table:

Section in server JDE.INI file	Parameter	Purpose
[DEBUG]	DebugFile	Specifies the location of the jdedebug.log file.
[DEBUG]	JobFile	Specifies the location of the jde.log file.
[DEBUG]	JDTSFile	Specifies the location of the lock manager trace file on the IBM i.
[DB SYSTEM SETTINGS]	Database	Specifies the name of the database in which the system tables reside.
[SECURITY]	DataSource	Specifies the name of the JD Edwards EnterpriseOne data source that contains the security tables and is used for user validation.

To create an instance of JD Edwards EnterpriseOne on the IBM i, complete these tasks:

- Copy needed libraries and directories and modify the values of selected parameters in the .ini library.

  To create an instance of JD Edwards EnterpriseOne on the IBM i, you copy these objects:

  - System library
  - System directory
  - Path code library
  - Path code directory

- Apply security to multiple instances of JD Edwards EnterpriseOne, if you desire to do so.

  If you want to apply security to multiple instances of JD Edwards EnterpriseOne, complete the steps in these task. If you do not want to apply security to multiple instances, proceed to the steps for creating a JD Edwards EnterpriseOne subsystem and starting a JD Edwards EnterpriseOne service.

- Create a new JD Edwards EnterpriseOne subsystem identification.

  On the IBM i platform, a subsystem is a logical process that is used to run system jobs, whether they are JD Edwards EnterpriseOne or other application jobs. JD Edwards EnterpriseOne network and kernel jobs run under the IBM i subsystem, which we ship with a default description. You can use this description without modification when you are running a single instance of JD Edwards EnterpriseOne on the IBM i server.

  If you decide to run multiple instances of JD Edwards EnterpriseOne, you need to create a new subsystem with a unique description for each instance of JD Edwards EnterpriseOne that you create. To create a new JD Edwards EnterpriseOne subsystem description, you use the CRTOWSBS command.

See Also:

- Administering Security JD Edwards EnterpriseOne Database Security for IBM i.
- "Creating a New Environment Using the Director Mode" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.
- Setting Up IBM i Database Security for Multiple JD Edwards EnterpriseOne Instances.
- "Managing JD Edwards EnterpriseOne Subsystems" in the *JD Edwards EnterpriseOne Tools System Administration Guide*.

2.8.2 Understanding IBM i Database Security Parameters

You use the IBM i database security parameters to modify user and administrator profiles, to secure objects, and so on. These parameters appear on the Set Up OneWorld Authority (SETOWAUT) form.

2.8.2.1 Type

Depending on the value that you enter in this field, you can implement a full security setup, modify only the security profiles, or modify only the datapaths authority. A full security setup includes the system library, datapath, pathcode, and user profiles.

- Use *FULL when you initially implement SETOWAUT. This value directs SETOWAUT to perform all of the security routines.
- Use *DTAPATH only when you need to secure one or more datapaths.
- Use *PROF to perform only the user profile routines. SETOWAUT uses the user profile settings in the command to direct the process.
- Use *SYSTEM to perform the System library authority functions. If you are running a single instance of JD Edwards EnterpriseOne, *SYSTEM secures the System library and all of the objects within it with the AUTL OWADMINL. If you are running multiple instances of JD Edwards EnterpriseOne, *SYSTEM secures the library and all the objects contained within it with the administrative authorization list created by the SETOWAUT program for each individual instance of JD Edwards EnterpriseOne. Note that SETOWAUT must be run separately for each instance of JD Edwards EnterpriseOne.

  Additionally, all the *PGM objects with attributes of *CLP, *CLLE, or *CLE will have the program attributes modified for adopt authority. The system library is treated differently to enable the administration of JD Edwards EnterpriseOne.

  You can use this parameter to lock other non-system libraries that contain objects that you can use to administer JD Edwards EnterpriseOne.

2.8.2.2 Additional Profile Work That SETOWAUT Performs When You Use Types *FULL or *PROF

When you enter Type *FULL or *PROF, SETOWAUT does these:

- Creates the ONEWORLD and OWADMINL authorization lists (if they do not already exist) if you are running a single instance of JD Edwards EnterpriseOne. If you are running multiple instances of JD Edwards EnterpriseOne, SETOWAUT creates both authorization lists and uses the names that you specified for each instance of JD Edwards EnterpriseOne.

- Changes the owner of both lists to ONEWORLD if you are running a single instance of JD Edwards EnterpriseOne. If you are running multiple instances of JD Edwards EnterpriseOne, SETOWAUT changes the owner of both lists to the user profile name that you specified for each instance of JD Edwards EnterpriseOne.
- Adds JDE to both lists if you are running a single instance of JD Edwards EnterpriseOne.
- Adds PSFT to both lists if you are running a single instance of JD Edwards EnterpriseOne.
- Changes *PUBLIC entry to *EXCLUDE in both lists.

2.8.2.3 INILIB (INI Library)

This field identifies the library in which the JDE.INI file resides for the security application. The *NONE value enables you to specify that the JDE.INI file is either not needed or not available.

Note: You cannot use the parameter value *NONE if the Type parameter value is *FULL or *SYSTEM.

Use a library name if all of these requirements are true:

- A JD Edwards EnterpriseOne INI library is located on the host system.
- The control files (OCM) are located on the host system.
- The JDE.INI file references the OCM library.

  When the Type field contains the value *FULL or *SYSTEM, the library and all of the objects will be secured with SYSTEM attributes. SETOWAUT uses the JDE.INI file to perform all of the INI retrievals.

When any of the previous requirements are false, use *NONE. This setting requires you to enter actual values in any parameter that allows the value *INI.

2.8.2.4 DTAPATH Datapath (library)

If you set the INI library field to *NONE, you must manually set datapaths in this field.

Type *INI in this field to use the datapaths that are set in the JDE.INI file. You can also type specific datapaths in this field. You can type up to 10 datapaths at a time.

Use *INI when these are true:

- SETOWAUT will modify each library based upon the ALLOBJECTS parameter.
- The INILIB parameter contains the library name in which the JDE.INI file is located (the INILIB value is not *NONE). This parameter tells SETOWAUT to use the JDE.INI file to retrieve the datapath libraries. SETOWAUT retrieves the library name from the JDE.INI value in the [DB SYSTEM SETTINGS] Library and uses this setting to access the Object Configuration Master (F986101) and Data Source Master (F98611) tables. SETOWAUT selects all of the library names (F98611.OMDATB2) that meet these criteria:
  - F986101.OMDATP = F98611.OMDATP
  - OMUGRP = *PUBLIC, OMSTSO = `AV'
  - OMSRVR = the host name

2.8.2.5 Modify System Profile

Values for this field are Y and N.

> **Note:** This field does not appear when you set up authorization for multiple instances of JD Edwards EnterpriseOne and you enter a value other than ONEWORLD in the USRPRF field.

Enter Y when you want to do these:

- Modify or create the system profile that has not yet been modified. For example, you might enter this information to modify a system profile:
  - *NONE in the GRPPRF field.
  - *NONE in the SUPGRPPRF field.
  - *USER in the USRCLS field.
  - *SIGNOFF in the INLMNU field.
  - *NONE in the INLPGM field.
  - *JOBCTL in the SPCAUT field.
- Grant authority to change the profile ONEWORLD to *USE profile QSECOFR.
- Revoke *ALL authority from *PUBLIC.

Enter N only when the system profile has the correct attributes.

2.8.2.6 Modify JDE Profile

Values for this field are Y and N.

> **Note:** This field does not appear when you set up authorization for multiple instances of JD Edwards EnterpriseOne and you enter a value other than ONEWORLD in the USRPRF field.

Enter Y when you want to do these:

- Modify or create the JDE profile that has not been modified. For example, you might enter these to modify a JDE profile:
  - *NONE in the GRPPRF field.
  - *NONE in the SUPGRPPRF field.
  - *USER in the USRCLS field.
  - *NONE in the INLPGM field.
  - *JOBCTL *SAVSYS in the SPCAUT field.
- Revoke *ALL authority from *PUBLIC.

Enter N only when the profile JDE has the correct attributes.

2.8.2.7 Modify Security Profile

You can enter up to 10 security profiles at a time in this field to modify using the SETOWAUT program.

Note: It is recommended that you delete existing security profiles before running SETOWAUT. After running SETOWAUT and creating security profiles, the passwords must be changed to correspond with passwords that were set up using JD Edwards EnterpriseOne User Security. The security user is used as the system user in JD Edwards EnterpriseOne User Security.

SETOWAUT must be run with the PSFT user profile specified as a security profile when using JD Edwards EnterpriseOne. If you enter a security profile that does not already exist, SETOWAUT creates the profile and modifies the profile accordingly. You can do any of these:

- Create or modify the profile by entering information such as these:
  - *USER in the USRCLS field.
  - *SIGNOFF in the INLMNU field.
  - *NONE in the INLPGM field.
  - *NONE in the SPCAUT field.
  - ONEWORLD in the GRPPRF field, if you are running a single instance of JD Edwards EnterpriseOne. If you are running multiple instances of JD Edwards EnterpriseOne, enter in the GRPPRF field the JD Edwards EnterpriseOne User Profile name that you entered in the USRPRF field.
  - JDE in the SUPGRPPRF field, if you are running a single instance of JD Edwards EnterpriseOne. If you are running multiple instances of JD Edwards EnterpriseOne, enter in the SUPGRPPRF field the JD Edwards EnterpriseOne User Profile name that you entered in the USRPRF field.
- Revoke *ALL authority from *PUBLIC.
- Grant profile ONEWORLD *CHANGE authority to the security profile.
- Grant security profile *CHANGE authority to ONEWORLD.

  Sample Results for SETOWAUT in the JD Edwards EnterpriseOne Tools 8.94 Implementation Guide: Server and Workstation Administration.

2.8.2.8 JD Edwards EnterpriseOne DB Admin Profile

When you type *INI in this field, SETOWAUT retrieves the user and password values from the [SECURITY] section of the JDE.INI file. If you type a value that does not exist, SETOWAUT creates a profile with a password that is the same as the profile name. If the profile exists, SETOWAUT modifies the profile to be a JD Edwards EnterpriseOne database administrator.

Enter a profile to be used as a database administrator. This profile will have all rights to all JD Edwards EnterpriseOne objects. These database administrator profiles are allowed to perform certain JD Edwards EnterpriseOne processes (RUNUBE and PORTTEST) that an administrator with normal privileges cannot perform.

If the profile does not exist, the system creates the profile with a password that is the same name as the profile. If the profile does not exist, you should set the password to expire (PWDEXP = *YES). For example, this occurs:

- If BV3C is in library list, SETOWAUT will place this program as the initial program. (This program lists all of the JD Edwards EnterpriseOne occurrences to enable the user to select one occurrence at signon).

- USRCLS is set to *PGMR.
- SPCAUT is set to *NONE.
- GRPPRF is set to ONEWORLD if you are running a single instance of JD Edwards EnterpriseOne. If you are running multiple instances of JD Edwards EnterpriseOne, GRPPRF is set to the JD Edwards EnterpriseOne User Profile name that you entered in the USRPRF parameter field.

This profile revokes *ALL authority from *PUBLIC and grants ONEWORLD *USE rights to the DB ADMIN profile.

2.8.2.9 BSFNLIB (Libs or *INI (Default PathCode))

Type *INI in this field to use the pathcode library and the associated specification file directory that is set in the JDE.INI file. You can also type specific pathcode libraries in this field. You can type up to 10 pathcodes at a time.

> **Note:** If you enter *NONE in the INI library field, you must set pathcodes in this field.

Use *INI when the INILIB parameter contains the library name in which the JDE.INI file is located (INILIB does not contain *NONE). This parameter tells SETOWAUT to use the JDE.INI file to retrieve the application pathcode libraries. SETOWAUT retrieves the library name from the JDE.INI value in [DB SYSTEM SETTINGS] Library and uses this setting to access the Object Configuration Master (F986101) and Data Source Master (F98611) tables. SETOWAUT selects all of the library names (F98611.OMLIB) that meet these criteria:

- F986101.OMDATP = F98611.OMDATP
- OMUGRP = *PUBLIC
- OMSTSO = `AV'
- OMDBNM = F00942

SETOWAUT retrieves EMPATHCD (pathcode) from each record in the Object Path Master File table (F00942) for each library (F98611.OMLIB).

For each pathcode, SETOWAUT modifies the library and associated IFS directory (specifies path) accordingly.

2.8.2.10 Secure Log Path

Y and N are values for this field. The recommended value is N.

Enter N when you do not want to secure JDE log paths.

Enter Y only when you need to secure the log paths. One situation in which you might secure JDE log paths is when logs are being deleted without permission.

Only DB administrators have permission to access the logs in the log path.

2.8.2.11 Secure All Objects

Use this field to secure objects when you are running multiple instances of JD Edwards EnterpriseOne. The parameter appears on the SETOWAUT form only when you configure an instance of JD Edwards EnterpriseOne by entering a value other than ONEWORLD in the USRPRF field.

*NONCOEXIST is the default value for the Secure All Objects parameter, and we recommend that you use this value. This value secures all directories, but not the files in the directories.

Entering COEXIST secures the files as well as the directories. Entering COEXIST can degrade performance because the system must verify authority for every object that the user wants to access. This value is the equivalent of entering *ALLOBJECTS when you run a single instance of JD Edwards EnterpriseOne. The value *COEXIST can only be used for OneWorld Xe, and must never be used for JD Edwards EnterpriseOne.

2.8.3 Prerequisites

Before you complete the tasks in this section:

- Verify that enough space exists on the direct access storage device (DASD) to create a new instance of JD Edwards EnterpriseOne.
- Assess data storage and backup requirements.
- Consider the procedure that you will follow for updating the JD Edwards EnterpriseOne server with new versions of JD Edwards EnterpriseOne.
- Determine the strategy for performing server package builds and updates. This might include, for example, setting up a second deployment server.
- Create a new environment for use with each new JD Edwards EnterpriseOne instance.
- Set up security for multiple instances of JD Edwards EnterpriseOne.

2.8.4 Copying Libraries and Directories

To copy libraries and directories:

1. End JD Edwards EnterpriseOne services, if necessary.
2. Remove JD Edwards EnterpriseOne security, if necessary.
3. From the IBM i main menu, copy the JD Edwards EnterpriseOne system library in the QSYS file system by typing this command:

   ```
   CPYLIB E900SYS E900CST
   ```

   Where E900CST is the name for the system library in the new instance of JD Edwards EnterpriseOne.
4. From the IBM i main menu, copy the JD Edwards EnterpriseOne system directory in the IFS by first the using this command to create a temporary library:

   ```
   CRTLIB TEMPLIB
   ```

5. Create a save file in the temporary library for the system directory by typing this command:

   ```
   CRTSAVF FILE (TEMPLIB/E900SYS)
   ```

6. Save the system directory into the save file by typing this command:

   ```
   SAV DEV ('/QSYS.LIB/TEMPLIB/E900SYS.FILE') OBJ(('/E900SYS)) USEOPTBLK(*NO)
   DTACPR
   (*YES)
   ```

7. Restore the save file for the system directory to a directory with a new name by typing this command:

```
RST DEV('/QSYS.LIB/TEMPLIB/E900SYS.FILE') OBJ(('/E900sys/*' *INCLUDE/E900cst'))
```

Where E900cst is the name of the new system directory.

Note: Throughout the entire copying procedure, the name for the new directories and libraries must match.

---

8. From the IBM i main menu, copy the path code library in the QSYS file system by typing this command:

   ```
   CPYLIB PRD900 CST900
   ```

   Where CST900 is the name for the path code library in the new instance of JD Edwards EnterpriseOne. The name of the library for the new instance cannot exceed eight characters in length.

   Note: The path code directory for any environment that you intend to use for a new instance of JD Edwards EnterpriseOne must be copied to the new directory. You cannot share path code directories between two or more instances of JD Edwards EnterpriseOne because this might corrupt the specification file.

   ---

9. From the IBM i main menu, copy the path code directory in the IFS by first using this command to create a save file in the temporary library:

   ```
   CRTSAVF FILE(TEMPLIB/PRD900)
   ```

   Note: You must follow the procedure for copying the path code directory for each path code that you copy.

   ---

10. Save the path code directory into the save file by typing this command:

    ```
    SAV DEV('QSYS.LIB/TEMPLIB/PRD900.FILE') OBJ(('/prd900/*')) USEOPTBLK(*NO)
    DTACPR
    (*YES)
    ```

11. Restore the save file for the path code directory to a directory with a new name by typing this command:

    ```
    RST DEV('QSYS.LIB/TEMPLIB/PRD900.FILE') OBJ(('/prd900/*' INCLUDE '/cst900'))
    ```

    Where cst900 is the name of the new path code directory.

12. From the IBM i main menu, create a JD Edwards EnterpriseOne subsystem from the system library by typing this command:

    ```
    CRTOWSBS <subsystem name> <system library>
    ```

    Where <subsystem name> is the name you give to the JD Edwards EnterpriseOne subsystem for the new instance of JD Edwards EnterpriseOne, and <system library> is the name of the system library in the QSYS file system for the new instance of JD Edwards EnterpriseOne.

Note: You can use the same subsystem for multiple instances of JD Edwards EnterpriseOne.

13. Modify these parameters in the INI library:

```
[INSTALL]
DefaultSystem=<System Library>

[JDEIPC]
startIPCKeyValue=<Unused start key not within another instance's IPC range>

[JDENET]
serviceNameListen=<Available port>
serviceNameConnect=<Available port>

[DB SYSTEM SETTINGS]
Default Env=<New environment>
Default PathCode=<New path code>
```

2.8.5 Applying Security to Multiple Instances of JD Edwards EnterpriseOne on the IBM i

To apply security to multiple instances of JD Edwards EnterpriseOne on the IBM i:

1. Copy the OCM library.
2. Copy the database libraries, such as SYS900, 900MAP, and so on.
3. Create a new IBM i user profile for each new instance of JD Edwards EnterpriseOne.
4. From the IBM i main menu, create a new log path in the IFS by typing this command:

   ```
   CRTDIR DIR('/900CSTLOG')
   ```

   Where CSTLOG is the name of the new IFS log directory.
5. Modify these parameters in the INI library:

```
[DEBUG]
DebugFile=<new log path>/JDEDEBUG.LOG
JobFile=<new log path?/JDE.LOG
JDETSFile=<new log path>/JDETS.LOG

[DB SYSTEM SETTINGS]
Database=<new OCM library>

[SECURITY]
DataSource=<Location of new F98OWSEC library>
```

Note: The parameter values in the [DEBUG] section must be uppercase.

2.8.6 Creating a JD Edwards EnterpriseOne Subsystem on the IBM i

To create a JD Edwards EnterpriseOne subsystem on the IBM i:

1. Stop JD Edwards EnterpriseOne services.
2. From the IBM i main menu, type this command, and then press Enter or press the F4 key:

   ```
   CRTOWSBS
   ```

3. On the CREATE New JD Edwards EnterpriseOne Subsystem form, enter character values for these parameters, and then press Enter:
   - SUBSYSTEM
   - SYSLIB

   Note: The maximum number of characters allowed for the description of each parameter is 10. Verify that the name of the system library matches the name that you created when you copied the JD Edwards EnterpriseOne system library in the QSYS file system.

   The CRTOWSBS command creates a new subsystem description in the JD Edwards EnterpriseOne system library and updates the STRNET and ENDNET programs with the new subsystem name as the default parameter.
4. To delete the old subsystem description from the system library, type this command, and then press Enter or press the F4 key:

   ```
   WRKOBJ OBJ <SUBSYSTEM NAME>/<SYSTEM LIBRARY NAME>) OBJTYPE(*SBSD)
   ```

   Where SUBSYSTEM NAME is the subsystem description that you want to delete and SYSTEM LIBRARY NAME is the system library where the subsystem description is located.
5. In the Work with Objects form, type 4 for Delete, and then press Enter.
6. From the IBM i main menu, clear IPC memory by typing this command:

   ```
   CLRIPC
   ```

7. From the IBM i main menu, start JD Edwards EnterpriseOne IBM i services by typing this command:

   ```
   STRNET
   ```

2.9 Administering Security JD Edwards EnterpriseOne Database Security for IBM i

This section provides an overview of JD Edwards EnterpriseOne data base security administration and discusses how to:

- Set up IBM i database security for a single JD Edwards EnterpriseOne instance.
- Set up IBM i database security for multiple JD Edwards EnterpriseOne instances.
- Add administrators.
- Remove administrative authority from user profiles.
- Display user profile information.

2.9.1 Understanding JD Edwards EnterpriseOne Database Security Administration

You can secure profiles and objects for JD Edwards EnterpriseOne on the IBM i with the Set Up OneWorld Authority (SETOWAUT) command. When you enter this command, a form appears that enables you to enter specific security information for the system. The authority is implemented only on the IBM i machine on which you execute the command.

Note: If you upgraded to JD Edwards EnterpriseOne from an existing ERP installation and do not intend to rerun SETOWAUT, then you must manually add the PSFT user profile to the existing security profile authorization list. (The default name for authorization list is OneWorld.)

The SETOWAUT command enables you to set up security for a single instance of JD Edwards EnterpriseOne or for multiple instances of JD Edwards EnterpriseOne. If you run multiple instances of JD Edwards EnterpriseOne, you can set up separate user profiles for each instance. The SETOWAUT command sets up the authorities for each JD Edwards EnterpriseOne instance, adds profile names to an authorization list, and sets object ownership for each JD Edwards EnterpriseOne instance.

Two separate authorization lists exist for maintaining security. Values in two parameters of the SETOWAUT program specify the authorization lists.

The USRPRF parameter value specifies the JD Edwards EnterpriseOne user profile. When you run the SETOWAUT program, the program automatically creates a user profile authorization list with the same name. This list secures all JD Edwards EnterpriseOne objects.

The ALLOBJECTS parameter determines how SETOWAUT secures JD Edwards EnterpriseOne objects. The recommended setting for this parameter is *NONCOEXIST. Using this setting, the resulting authorization list secures only the root directories and the libraries. This is true for all libraries except the System library; SETOWAUT secures all of the objects in the system library. The value ALLOBJ secures every object in all JD Edwards EnterpriseOne libraries and directories. This parameter is not recommended because it negatively affects JD Edwards EnterpriseOne performance.

The COEXIST option can be used for OneWorld Xe, but never for JD Edwards EnterpriseOne. COEXIST is not valid with JD Edwards EnterpriseOne.

This release of JD Edwards EnterpriseOne introduces the PSFT user profile. To use JD Edwards EnterpriseOne software, this user profile must have access to objects that are owned by this instance of the software, regardless of whether SETOWAUT is used (that is, the default profile is the ONEWORLD user profile). To provide the PSFT user profile access to objects, you must do these:

- Change PSFT user profile attribute GRPPRF to the name of the JD Edwards EnterpriseOne or multiple instance USRPRF (the default value is ONEWORLD).
- Verify that the PSFT user profile attribute OWNER is set to *GRPPRF. If it is not, manually set this value to *GRPPRF.

The USRAUTL parameter value specifies the administrative authorization list. When you run the SETOWAUT program, the program automatically creates an administrative authorization list that gives specified users administrative access to JD Edwards EnterpriseOne. Any user who will perform basic JD Edwards EnterpriseOne administration tasks, such as Start, End, Clear IPC, and so on, on the IBM i must be added to this list. CRTOWADPRF is a supplied command that adds administrative

users to this list; RMVOWADPRF is a supplied command that removes such users from the list.

Use PROFTYPE(*USER) to perform basic JD Edwards EnterpriseOne administrative tasks. Use PROFTYPE(*ADMIN) for users who need access to all JD Edwards EnterpriseOne objects. (*ADMIN is similar to security officer but can only be used for JD Edwards EnterpriseOne.

Whether you want to set up security for one instance of JD Edwards EnterpriseOne or for multiple instances, the Set Up OneWorld Authority (SETOWAUT) form appears when you run the SETOWAUT command. However, the parameter values that you enter and the parameter fields that appear on the form differ, depending on whether you set up security for one instance or for multiple instances. These parameter differences are explained in these three tables:

Parameters Present in SETOWAUT Form for Both Single and Multiple Instances of JD Edwards EnterpriseOne	Meaning	Value to be Entered for a Single Instance of JD Edwards EnterpriseOne	Value to be Entered for Multiple Instances of JD Edwards EnterpriseOne
USRPRF	JD Edwards EnterpriseOne User Profile	JD Edwards EnterpriseOne	Configurable. Enter a new value for each instance of JD Edwards EnterpriseOne.
USRAUTL	Admin. Authorization List	OWADMINL	Configurable. Enter a new value for each instance of JD Edwards EnterpriseOne.

Parameters Present in SETOWAUT Form for Single Instance of JD Edwards EnterpriseOne Only	Meaning	Value to be Entered for a Single Instance of JD Edwards EnterpriseOne	Value to be Entered for Multiple Instances of JD Edwards EnterpriseOne
OWPRF	Modify ONEWORLD Profile	Y is the default value.	Parameter is not present if you enter a value other than ONEWORLD for the USRPRF parameter.
JDEPRF	Modify JDE Profile	Y is the default value.	Parameter is not present if you enter a value other than ONEWORLD for the USRPRF parameter.

Parameter Present in SETOWAUT Form for Multiple Instances of JD Edwards EnterpriseOne Only	Meaning	Value to be Entered for Multiple Instances of JD Edwards EnterpriseOne	Value to be Entered for Single Instance of JD Edwards EnterpriseOne
OBJOPT	Secure All Objects	N is the default value. Enter **Y** if you want to secure all objects that appear in one or more directories. Because it can degrade system performance, entering Y is not recommended.	Parameter is not present if you enter OneWorld as the value for the USRPRF parameter.

This information provides a summary of the security model when you run a single instance of JD Edwards EnterpriseOne:

Library	Description of Security
JD Edwards EnterpriseOne System Library	SETOWAUT secures all of the objects in the system library. Administrative programs, such as CLRIPC, STRNET, ENDNET, and PORTTEST, are set to adopt the authority of the owner.

You can set up security for a single instance of JD Edwards EnterpriseOne, or you can set up security for separate JD Edwards EnterpriseOne instances. In the latter case, the SETOWAUT program creates a user profile and individual authorization lists for each instance, which establishes object ownership.

You can set up security for separate instances of JD Edwards EnterpriseOne as well. To do so, you enter a value other than ONEWORLD for the User Profile parameter and a value other than OWADMINL for the Admin. Authorization List parameter. You enter different values for these parameters for each instance of JD Edwards EnterpriseOne that you run.

Note: Use caution when you use JD Edwards EnterpriseOne security to lock a library that contains third-party software. We do not support the IBM i JD Edwards EnterpriseOne database security with third-party software.

2.9.1.1 Sample Results for SETOWAUT

You can expect these examples for each of the various commands. Using Client Access, sign onto the IBM i, type each command on the command line, and press F4. For libraries (data sources and pathcodes), the required parameters are object type (*LIB) and the name of the library.

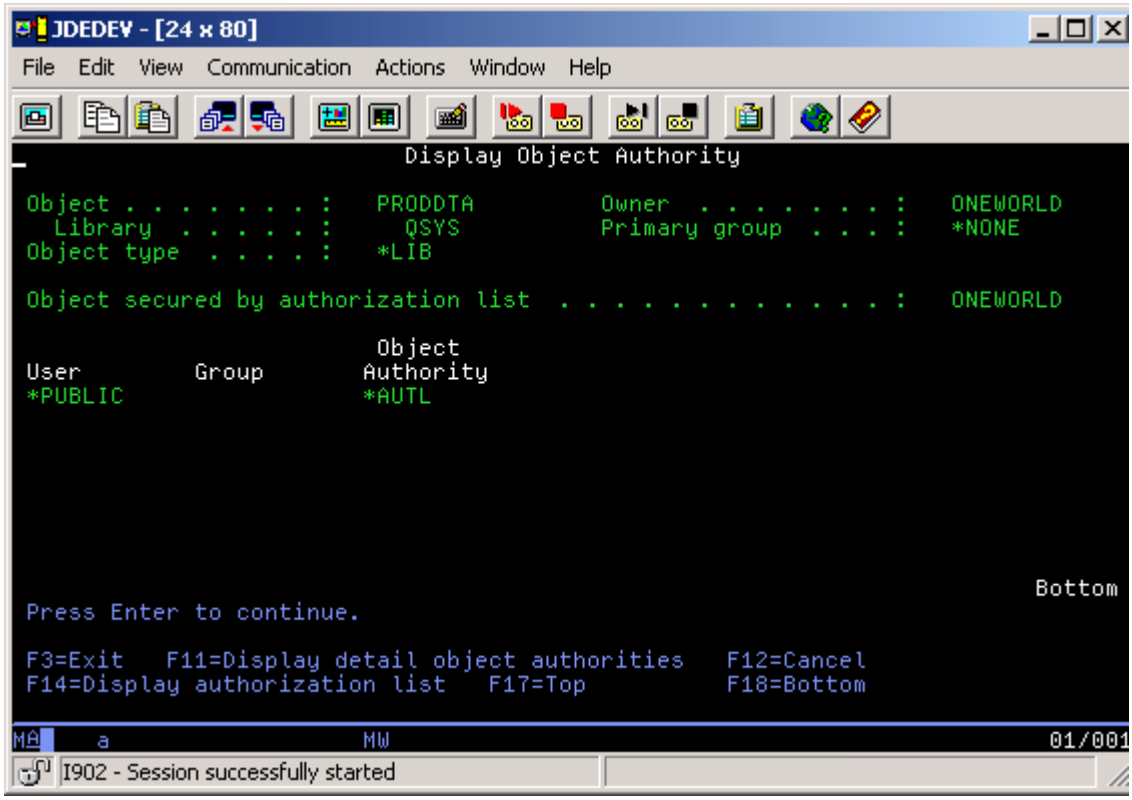
If you set up multiple instances of JD Edwards EnterpriseOne, the owner of each instance is the user profile that you entered in the JD Edwards EnterpriseOne User Profile parameter during the authority setup. If you set up a single instance of JD Edwards EnterpriseOne, the owner is ONEWORLD.

Similarly, if you set up multiple instances of JD Edwards EnterpriseOne and you display object authority, the value that appears is the name of the user profile for all objects except the SYSTEM library. The object authority for the SYSTEM library when

you run multiple instances of JD Edwards EnterpriseOne is the name of the Admin. Authorization List. If you set up a single instance of JD Edwards EnterpriseOne, all objects are secured by the authorization list, except the SYSTEM library, which is secured by the OWADMINL authorization list.

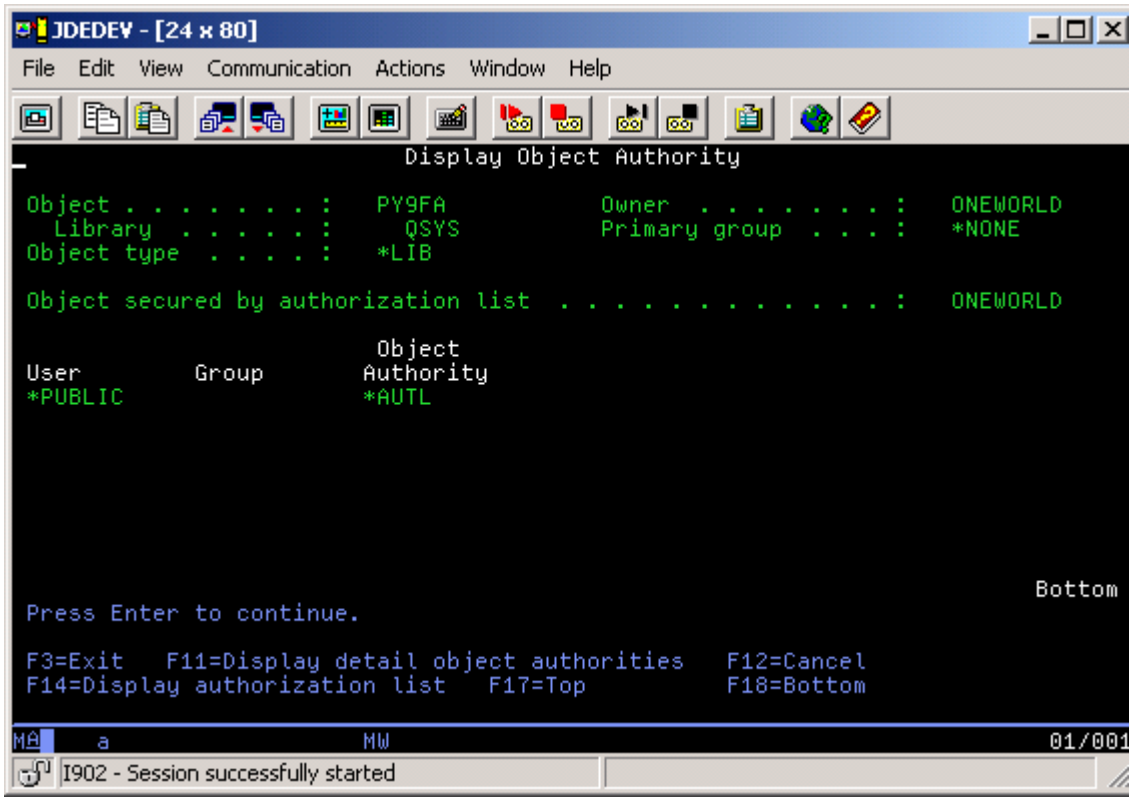
This is an example of the data source DSPOBJAUT:

Figure 2–2 Data Source DSPOBJAUT



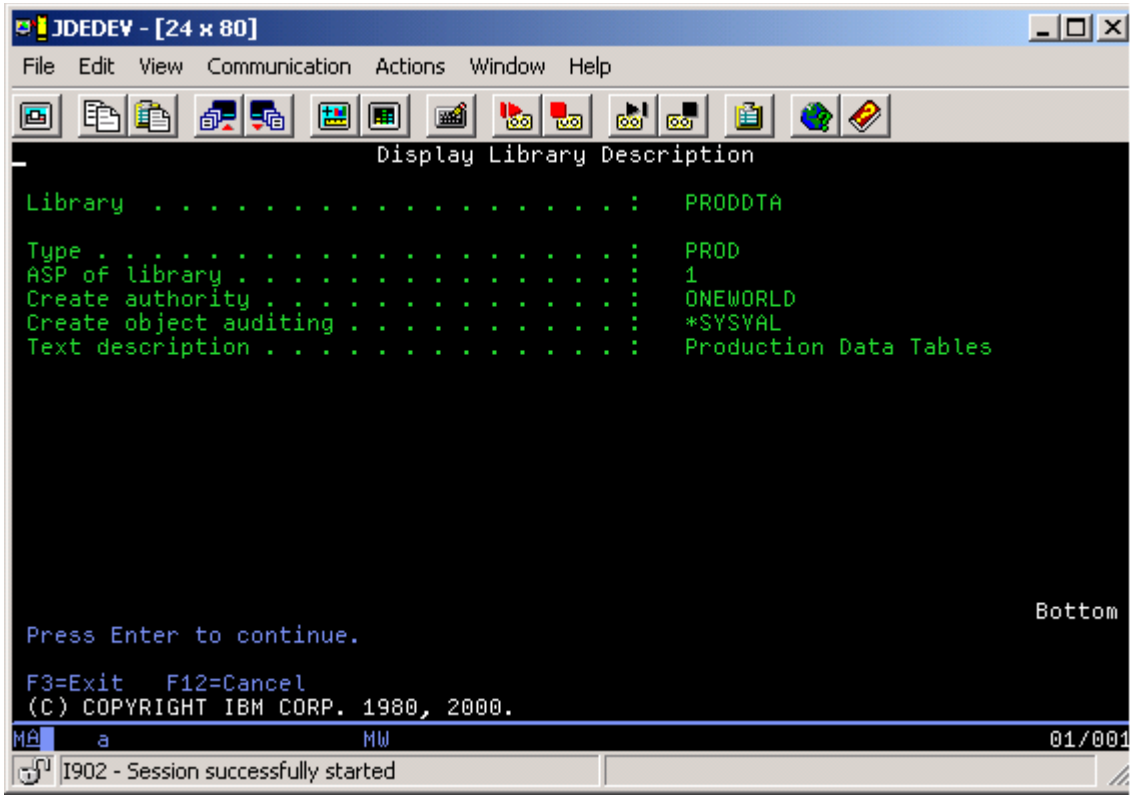
This is an example of the data source DSPOBJAUT:

Figure 2–3 Pathcode DSPOBJAUT



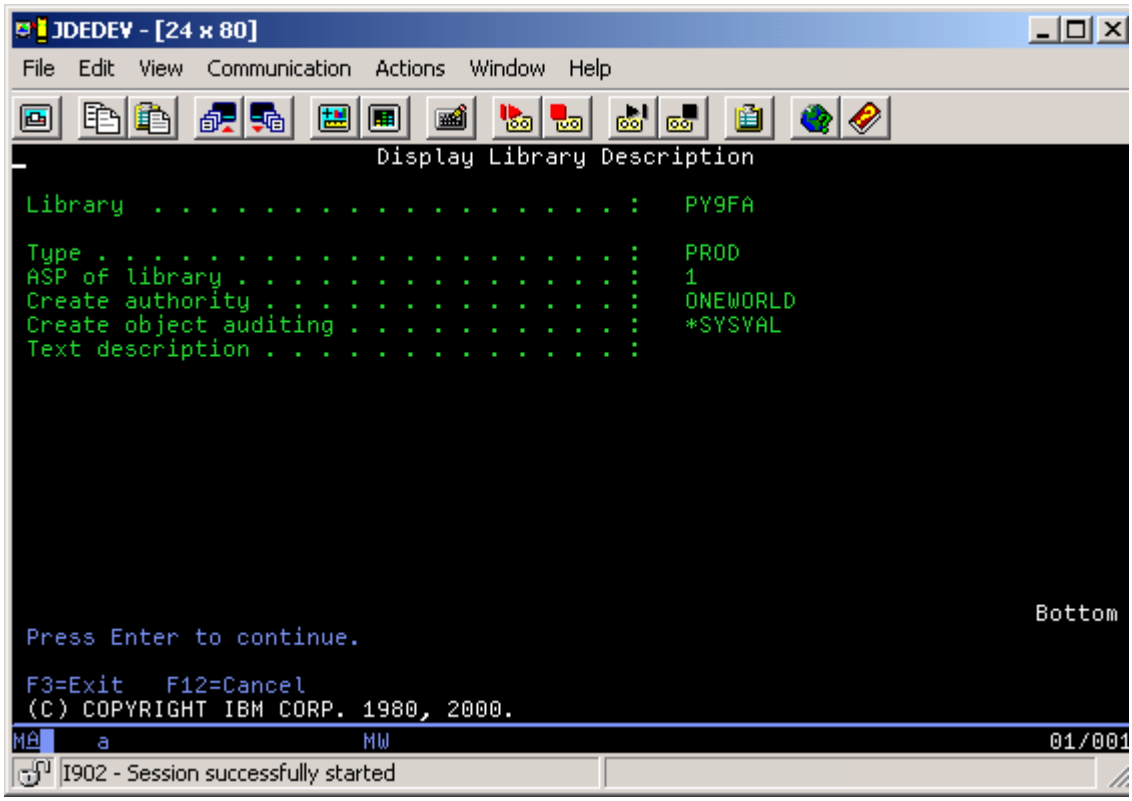
This is an example of the data source DSPLIBD:

Figure 2–4 Data source DSPLIBD



This is an example of the pathcode DISLIBD:

Figure 2–5 Pathcode DSPLIBD



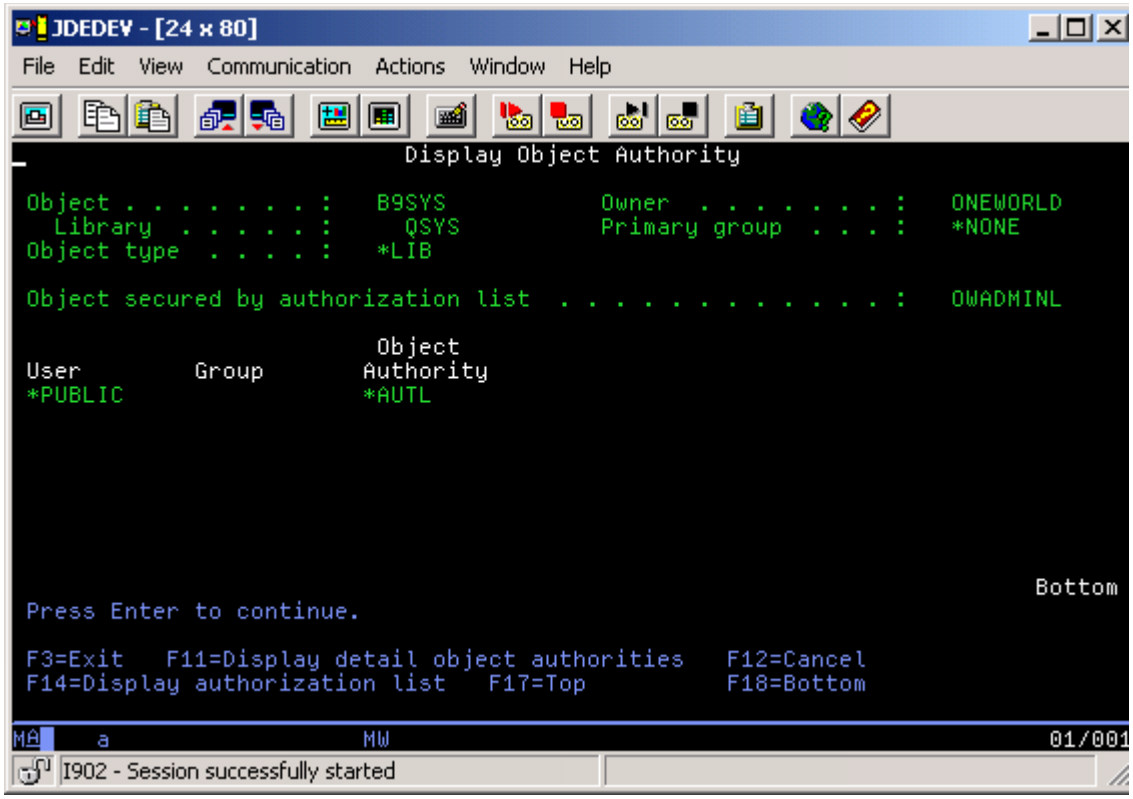
Note: Authority for objects in data sources and pathcodes should remain the same after you run SETOWAUT. You can see this by displaying the authority for an object in each library before and after you run SETOWAUT. The forms should be identical. The required parameters are object name, object type (*FILE or *PGM), and the library name in which the object resides.

Owner, object security, and authority creation differ depending on whether you are running a single instance of JD Edwards EnterpriseOne or multiple instances.

SETOWAUT changes the authority on system libraries. You can view this for both DSPOBJAUT and DSPLIBD on system libraries. The shaded information in these illustrations should correspond to the information that appears on the form. The required parameters are the object name, object type (*PGM), and the name of the library in which these objects reside.

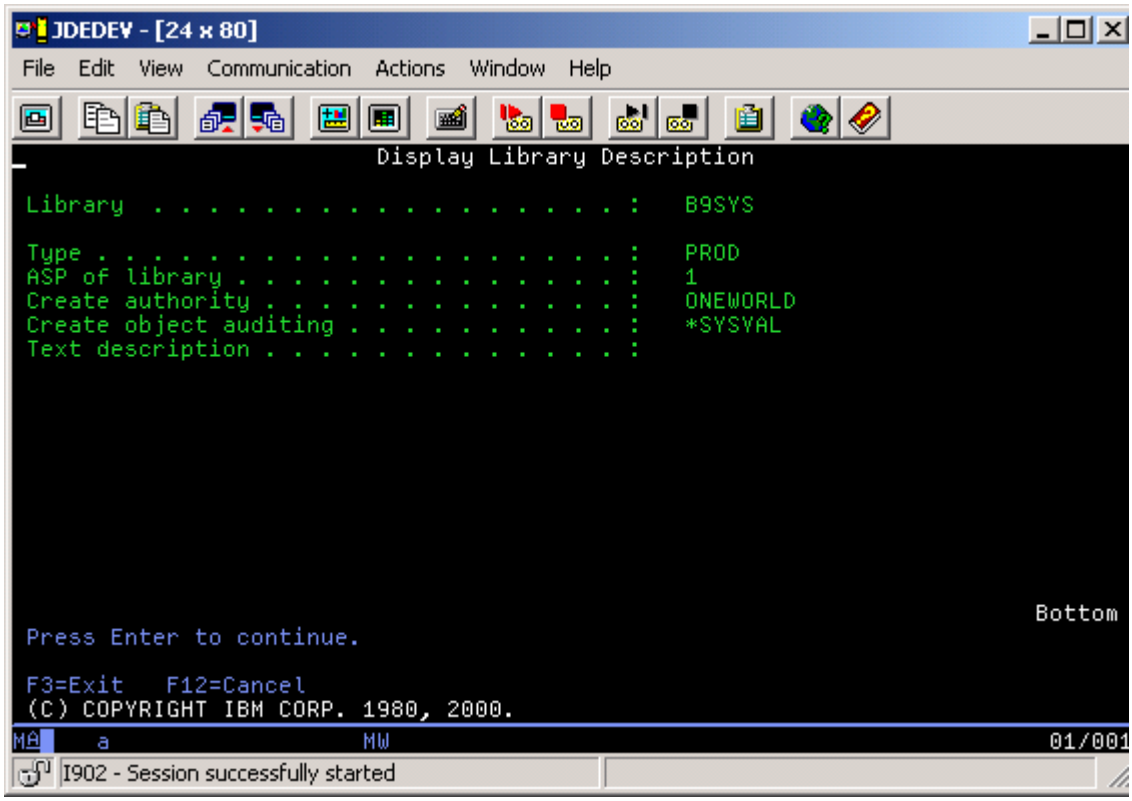
This is an example of the system library DSPOBJAUT:

Figure 2–6 System library DSPOBJAUT



This is an example of the system library DSPLIBD:

Figure 2–7 System library DSPLIBD



The authority changes for objects within system libraries that either contain the attributes CLLE or CLP or that share the same name. You can use commands to review the authority on these objects. The required parameters are object name, object type (*PGM or *CMD), and the name of the library in which these objects reside.

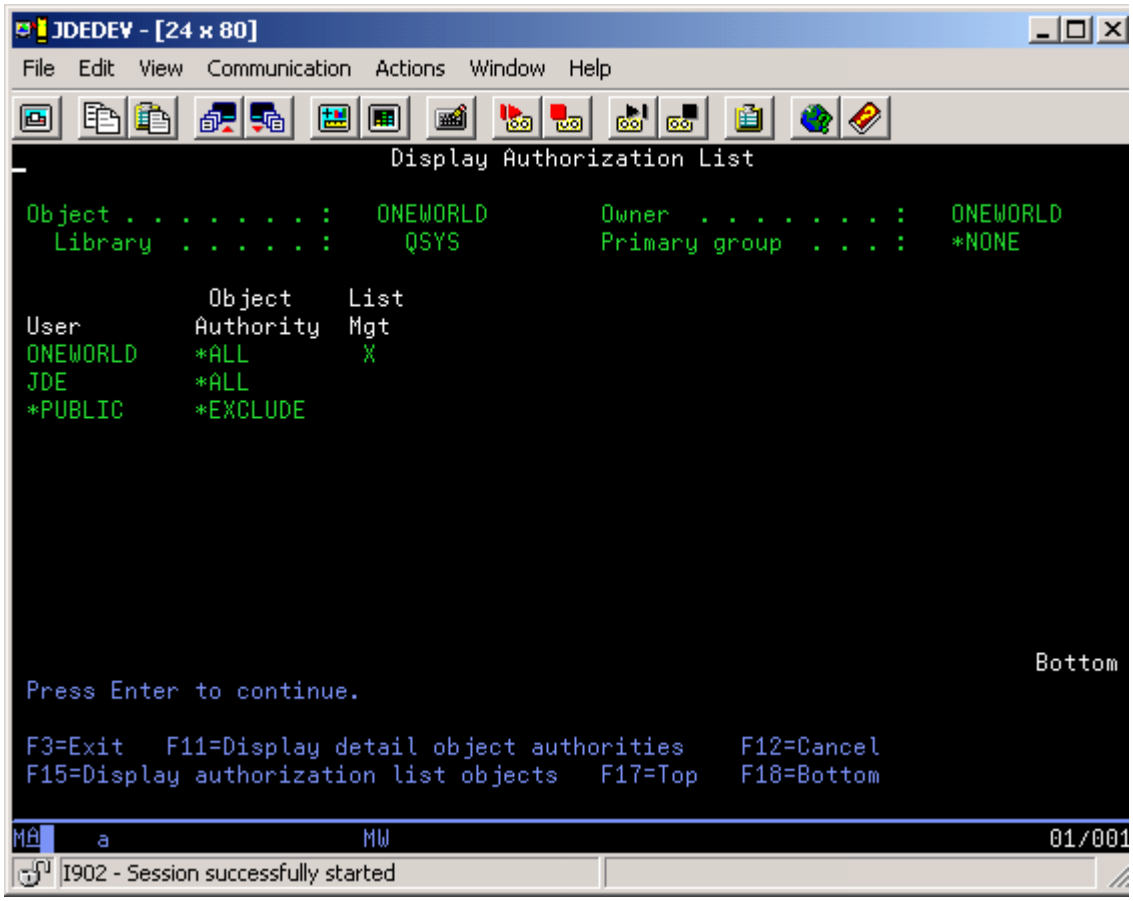
2.9.1.2 Sample Results for Authorization Lists

Use these commands to view the authorization list authorities. The name of the list is the only necessary parameter:

- IFS directories (specification files).
- WRKLNK - option 9 Work with authority.

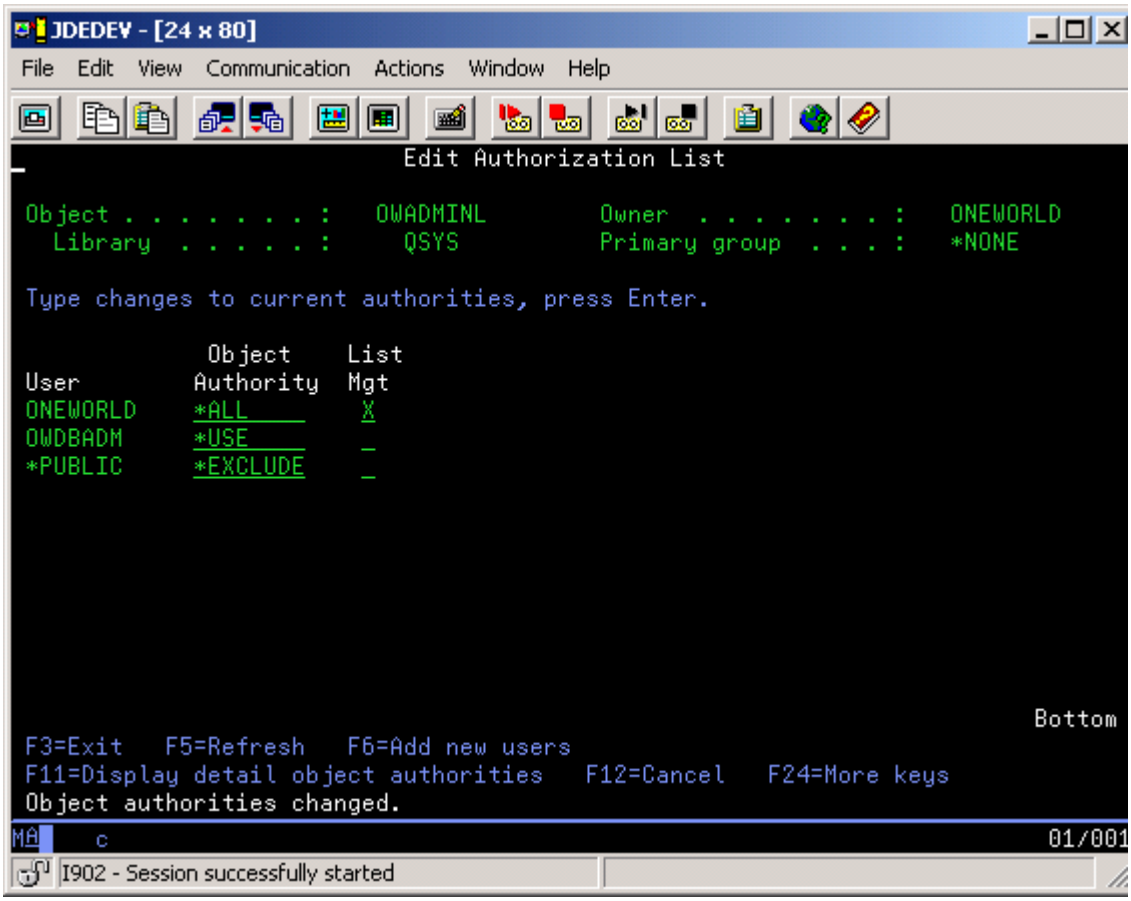
This is an example of DSPAUTL:

Figure 2–8 Display Authorization List



This is an example of DSPAUTL:

Figure 2–9 Edit Authorization List



2.9.2 Prerequisite

Before you enter a value for the USRPRF and USRAUTL parameters, verify that the value is not being used for an authorization list for any other instance of JD Edwards EnterpriseOne. To do so, run the DSPAUTL command. On the Display Authorization form, you can enter the value that you intend to use to make sure that it is unique.

2.9.3 Setting Up IBM i Database Security for a Single JD Edwards EnterpriseOne Instance

To set up IBM i database security for a single JD Edwards EnterpriseOne instance:

1. In the SETOWAUT library, on the command line, type this command, press F4, and then press F11:

   ```
   SETOWAUT
   ```

   Note: Verify that the SETOWAUT library is in the library list. If it is not, run the ADDLIBLE command.

   The Set Up OneWorld Authority (SETOWAUT) form appears.

2. On Set Up OneWorld Authority (SETOWAUT), complete the USPRF field with OneWorld, and then press Enter:

The form displays additional security parameters. You can specify various security settings, including library access.

3. Complete these required fields, and then press Enter:
   - USRAUTL

     Enter **OWADMINL.**
   - TYPE
   - INILIB
4. Complete any additional fields, if necessary.
5. Press Enter.

2.9.4 Setting Up IBM i Database Security for Multiple JD Edwards EnterpriseOne Instances

To set up IBM i database security for multiple JD Edwards EnterpriseOne instances:

1. In the SETOWAUT library, on the command line, type this command and press F4:

   ```
   SETOWAUT
   ```
2. On Set Up OneWorld Authority (SETOWAUT), complete the USRPRF field, and then press Enter:

   The SETOWAUT program uses this name when it creates a user profile authorization list.
3. The form expands to reveal an additional security parameter. The Modify OneWorld Profile (OWPRF) and Modify JDE Profile (JDEPRF) parameters, which appear when you enter OneWorld as the User Profile parameter value, do not appear when you enter a value other than OneWorld.
4. Complete these required fields and press Enter:
   - USRAUTL

     Enter a name that identifies the administrative authorization list.
   - TYPE
   - INILIB
5. Complete any additional fields, if necessary.
6. Press Enter.

2.9.5 Adding Administrators

You can add administrators to the administrative authorization list by running the CRTOWADPRF command. The command also enables you to designate levels of authority to the administrators whom you are adding to the list.

1. On the command line, enter this command and press F4:

   ```
   CRTOWADPRF USRPRF
   ```
2. On Set Up OW User Profile (CRTOWADPRF), in the ADMIN USER Profile field, enter the name of an administrator whom you want to add to the administrative authorization list. You can add up to 10 administrators at a time.

3. In the JD Edwards EnterpriseOne USER Profile field, Type the JD Edwards EnterpriseOne user profile name that you entered in the USRPRF field during setup.
4. In the ADMIN Authorization List, Type the Admin. Authorization List name that you entered in the USRAUTL field during setup.
5. In Profile Type, type ***USER** to give the profiles basic administration capabilities, such as STRNET, ENDNET, CLRIPC, CLRLCK, DSPIPC, DSPSTMF, IPCS, LINKBSFN, and PID2JOB.

   Type ***ADMIN** if the profiles need rights to PORTTEST and RUNUBE, as well as the basic administration capabilities.
6. In Initial program to call, type **BV3C** if you want the system to display a list of environments when the administrators sign on to JD Edwards EnterpriseOne, *SAME to use the current initial program setting, or *NONE to remove the initial program setting.

> **Note:** For JD Edwards EnterpriseOne, the initial program to call by default is BV3C. This program sets the IBM i to provide a choice of environments at signon. A user with an administrator profile who signs on to an environment can then perform JD Edwards EnterpriseOne commands on the IBM i server.

2.9.6 Removing Administrative Authority from User Profiles

To remove a user's administrative authority, you run the RMVOWADPRF command and complete the Remove OW Profile Authority form.

> **Note:** Submit this command to a batch subsystem.

1. On the command line, enter this command and press F4:

   ```
   RMVOWADPRF
   ```
2. On Remove OW Profile Authority (RMVOWADPRF), complete these fields and press Enter:

Field	Description
User Profile	Enter the name of the user from whom you want to remove authority.
Admin. Authorization List	Type the Admin. Authorization List name that you entered in the USRAUTL field during setup.
JD Edwards EnterpriseOne User Profile	Type the JD Edwards EnterpriseOne user profile name that you entered in the USRPRF field during setup.

2.9.7 Displaying User Profile Information

After you run SETOWAUT, you can review user profiles and authorization lists to verify that the information is correct.

1. On the command line, enter this command:

   ```
   DSPUSRPRF
   ```

2. On Display User Profile (DSPUSRPRF), type the name of a user profile in the User Profile field, and then press Enter.

   Information similar to this example appears:

```
User profile . . . . . :  ONEWORLD
Previous sign-on . . . . . :  03/23/04 15:16:53
Sign-on attempts not valid . . . . . :  0
Status . . . . . :  *ENABLED
Date password last changed . . . . . :  02/27/03
Password expiration interval . . .
. . . . :  *NOMAX     Set password to expired . . . . . :  *NO
User class . . . . . :  *USER
Special authority . . . . . :  *JOBCTL
Group profile . . . . . :  *NONE
Owner . . . . . :  *USRPRF
Group authority . . . . . :  *NONE
Group authority type . . . . . :  *PRIVATE
Supplemental groups . . . . . :  *NONE
Assistance level . . . . . :  *SYSVAL
Current library . . . . . :  *CRTDFT
Initial program . . . . . :  *NONE
Library . . . . . :
Initial menu . . . . . :  *SIGNOFF
Library . . . . . :
Limit capabilities . . . . . :  *NO
Text . . . . . :
Display sign-on information . . . . . :  *SYSVAL
Limit device sessions . . . . . :  *SYSVAL
Keyboard buffering . . . . . :  *SYSVAL
Storage information:
   Maximum storage allowed . . . . . :  *NOMAX
   Storage used . . . . . :  286236536
   Storage used on independent ASP . . . . :  *NO
Highest scheduling priority . . . . . :  3
Job description . . . . . :  ONEWORLD
   Library . . . . . :   QGPL
Accounting code . . . . . :
Message queue . . . . . :  ONEWORLD
Library . . . . . :   QUSRSYS
Message queue delivery . . . . . :  *NOTIFY
Message queue severity . . . . . :  00
Output queue . . . . . :  *WRKSTN
Library . . . . . :
Printer device . . . . . :  *WRKSTN
Special environment . . . . . :  *SYSVAL
Attention program . . . . . :  *SYSVAL
Library . . . . . :
Sort sequence . . . . . :  *SYSVAL
Library . . . . . :
Language identifier . . . . . :  *SYSVAL
Country identifier . . . . . :  *SYSVAL
Coded character set identifier . . . . . :  *SYSVAL
Character identifier control . . . . . :  *SYSVAL
Locale job attributes . . . . . :  *SYSVAL

User profile . . . . . :  JDE

Previous sign-on . . . . . :  03/23/04 15:25:53
Sign-on attempts not valid . . . . . :  0
Status . . . . . :  *ENABLED
```

```
Date password last changed . . . . . :  02/27/03
Password expiration interval . . . . . :  *NOMAX
Set password to expired . . . . . :  *NO
User class . . . . . :  *USER
Special authority . . . . . :  *JOBCTL
                                        *SAVSYS
Group profile . . . . . :  *NONE
Owner . . . . . :  *USRPRF
Group authority . . . . . :  *NONE
Group authority type . . . . . :  *PRIVATE
Supplemental groups . . . . . :  *NONE
Assistance level . . . . . :  *SYSVAL
Current library . . . . . :  *CRTDFT
Initial program . . . . . :  J98INIT
Library . . . . . :   JDFOBJ7R2
Initial menu . . . . . :  *MAIN
Library . . . . . :   *LIBL
Limit capabilities . . . . . :  *NO
Text . . . . . :
Display sign-on information . . . . . :  *SYSVAL
Limit device sessions . . . . . :  *SYSVAL
Keyboard buffering . . . . . :  *SYSVAL
Storage information:
Maximum storage allowed . . . . . :  *NOMAX
Storage used . . . . . :  11243168
Storage used on independent ASP . . . . :  *NO
Highest scheduling priority . . . . . :  3
Job description . . . . . :  JDE
Library . . . . . :   QGPL
Accounting code . . . . . :
Message queue . . . . . :  JDE
Library . . . . . :   QUSRSYS
Message queue delivery . . . . . :  *NOTIFY
Message queue severity . . . . . :  00
Output queue . . . . . :  *DEV
Library . . . . . :
Printer device . . . . . :  *WRKSTN
Special environment . . . . . :  *SYSVAL
Attention program . . . . . :  *SYSVAL
Library . . . . . :
Sort sequence . . . . . :  *SYSVAL
Library . . . . . :
Language identifier . . . . . :  *SYSVAL
Country identifier . . . . . :  *SYSVAL
Coded character set identifier . . . . . :  *SYSVAL
Character identifier control . . . . . :  *SYSVAL
Locale job attributes . . . . . :  *SYSVAL

User profile . . . . . :  JDEOW

Previous sign-on . . . . . :  03/23/04 15:28:02
Sign-on attempts not valid . . . . . :  0
Status . . . . . :  *ENABLED
Date password last changed . . . . . :  02/27/03
Password expiration interval . . . . . :  *NOMAX
Set password to expired . . . . . :  *NO
User class . . . . . :  *USER
Special authority . . . . . :  *NONE
Group profile . . . . . :  ONEWORLD
Owner . . . . . :  *GRPPRF
```

```
Group authority . . . . . :  *NONE
Group authority type . . . . . :  *PRIVATE
Supplemental groups . . . . . :  JDE
Assistance level . . . . . :  *SYSVAL
Current library . . . . . :  *CRTDFT
Initial program . . . . . :  *NONE
Library . . . . . :
Initial menu . . . . . :  *SIGNOFF
Library . . . . . :
Limit capabilities . . . . . :  *NO
Text . . . . . :
Display sign-on information . . . . . :  *SYSVAL
Limit device sessions . . . . . :  *SYSVAL
Keyboard buffering . . . . . :  *SYSVAL
Storage information:
Maximum storage allowed . . . . . :  *NOMAX
Storage used . . . . . :  147904
Storage used on independent ASP . . . . :  *NO
Highest scheduling priority . . . . . :  3
Job description . . . . . :  QDFTJOBD
Library . . . . . :   QGPL
Accounting code . . . . . :
Message queue . . . . . :  JDEOW
Library . . . . . :   QUSRSYS
Message queue delivery . . . . . :  *NOTIFY
Message queue severity . . . . . :  00
Output queue . . . . . :  *WRKSTN
Library . . . . . :
Printer device . . . . . :  *WRKSTN
Special environment . . . . . :  *SYSVAL
Attention program . . . . . :  *SYSVAL
Library . . . . . :
Sort sequence . . . . . :  *SYSVAL
Library . . . . . :
Language identifier . . . . . :  *SYSVAL
Country identifier . . . . . :  *SYSVAL
Coded character set identifier . . . . . :  *SYSVAL
Character identifier control . . . . . :  *SYSVAL
Locale job attributes . . . . . :  *SYSVAL

User profile . . . . . :  OWDBADMIN

Previous sign-on . . . . . :  03/23/04 15:30:12
Sign-on attempts not valid . . . . . :  0
Status . . . . . :  *ENABLED
Date password last changed . . . . . :  02/27/03
Password expiration interval . . . . . :  *NOMAX
Set password to expired . . . . . :  *NO
User class . . . . . :  *PGMR
Special authority . . . . . :  *NONE
Group profile . . . . . :  ONEWORLD
Owner . . . . . :  *GRPPRF
Group authority . . . . . :  *NONE
Group authority type . . . . . :  *PRIVATE
Supplemental groups . . . . . :  JDE
Assistance level . . . . . :  *SYSVAL
Current library . . . . . :  *CRTDFT
Initial program . . . . . :  *NONE
Library . . . . . :
Initial menu . . . . . :  MAIN
```

```
Library . . . . . :   *LIBL
Limit capabilities . . . . . :  *NO
Text . . . . . :
Display sign-on information . . . . . :  *SYSVAL
Limit device sessions . . . . . :  *SYSVAL
Keyboard buffering . . . . . :  *SYSVAL
Storage information:
Maximum storage allowed . . . . . :  *NOMAX
Storage used . . . . . :   0
Storage used on independent ASP . . . . :  *NO
Highest scheduling priority . . . . . :  3
Job description . . . . . :  QDFTJOBD
Library . . . . . :   QGPL
Accounting code . . . . . :
Message queue . . . . . :  JDEOW
Library . . . . . :   QUSRSYS
Message queue delivery . . . . . :  *NOTIFY
Message queue severity . . . . . :  00
Output queue . . . . . :  *WRKSTN
Library . . . . . :
Printer device . . . . . :  *WRKSTN
Special environment . . . . . :  *SYSVAL
Attention program . . . . . :  *SYSVAL
Library . . . . . :
Sort sequence . . . . . :  *SYSVAL
Library . . . . . :
Language identifier . . . . . :  *SYSVAL
Country identifier . . . . . :  *SYSVAL
Coded character set identifier . . . . . :  *SYSVAL
Character identifier control . . . . . :  *SYSVAL
Locale job attributes . . . . . :  *SYSVAL
```

3

Administering the UNIX and Linux Servers

This chapter contains the following topics:

3.1 Understanding Server Administration for UNIX and Linux

The JD Edwards company supports Oracle's JD Edwards EnterpriseOne enterprise servers for UNIX operating systems on the Hewlett-Packard HP 9000 (HP-UX), the IBM RS/6000 (AIX), and the Sun Microsystems SPARC (Solaris) platforms. In addition, beginning with JD Edwards EnterpriseOne 8.93, RedHat Enterprise Linux AS on the Intel Architecture is also supported. To operate the UNIX or Linux enterprise server, you need to perform administrative procedures on the server to ensure that JD Edwards EnterpriseOne will run properly.

Note: Some information in this and other guides refers to UNIX generically and includes the supported Linux platforms unless otherwise noted.

This section discusses:

- JD Edwards EnterpriseOne Directory Structure for UNIX and Linux.
- JD Edwards EnterpriseOne Architecture and Process Flow for UNIX and Linux.
- JD Edwards EnterpriseOne Initialization for UNIX and Linux.

3.1.1 JD Edwards EnterpriseOne Directory Structure for UNIX and Linux

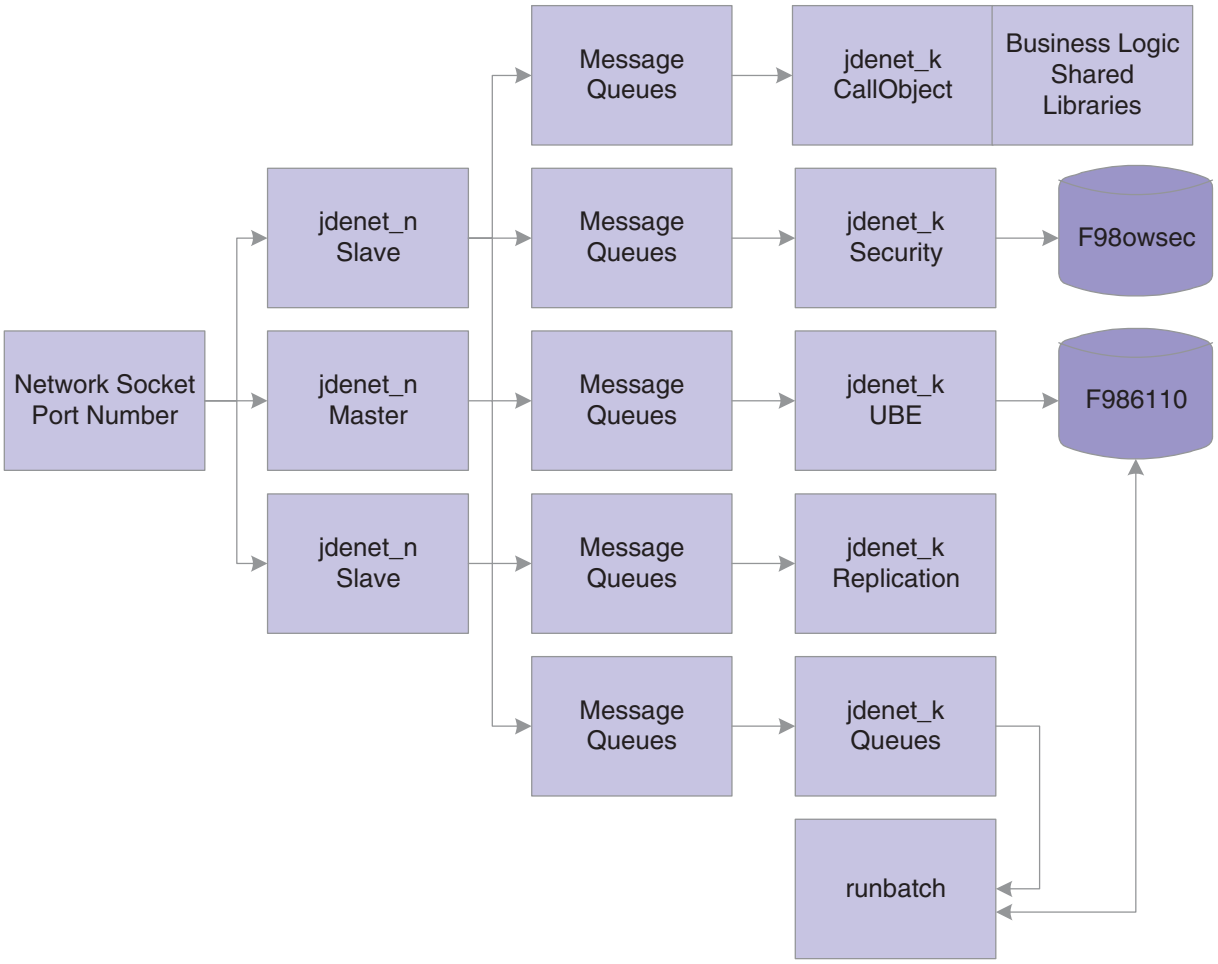
This is a list of directories that are shipped on the UNIX and Linux JD Edwards EnterpriseOne Server Installation CD. They should be installed under the JD Edwards EnterpriseOne base directory; for example, install them in /u01/JDEdwards/E900. Indented names indicate subdirectories of the directories, which are not indented.

Directory	Description
pathcode	The main directory for the business function shared libraries, C header files, object files, source files, and specification (spec or TAM) files. Upon installation, this directory is copied to the correct path codes, such as PD900 and DV900. These subdirectories are included: ■ bin32 - Business function shared libraries. ■ Spec files - Binary data files in a JD Edwards proprietary format.
system	The main directory for the system-level executables, shared libraries, C header files, libraries, and localization files. These subdirectories are included: ■ bin32 - System-level executables and scripts. ■ include - System-level C header files. ■ includev - System-level C header files provided by third-party vendors, such as Vertex. ■ lib - System-level shared libraries and export files. ■ libv32 - System-level shared libraries provided by third-party vendors.
ini	The location of the JDE.INI file.
PrintQueue	The location to which all .PDF file outputs for reports are written.
log	The location to which the jde_xxx.log and jdedbug_xxx.log files are written.
packages	The server package installation base directory. Directories exist here only when a package has been installed. Under the package directory are subdirectories named for each package that has been installed. Located under each package are these directories: ■ bin32 - Business function shared libraries. ■ include - Business function header files. ■ obj - Business function object files. (These are divided among lower-level subdirectories that correspond to each shared library in the bin32 directory.) ■ source - Business function source files. (These are divided among lower-level subdirectories that correspond to each shared library in the bin32 directory.) ■ spec - Specification files. (These binary data files are in a JD Edwards proprietary format.)

3.1.2 JD Edwards EnterpriseOne Architecture and Process Flow for UNIX and Linux

The host server processes in this flowchart perform the indicated actions.

Figure 3–1 UNIX host server process



This information explains the process flow:

1. The jdenet_n Master process spawns jdenet_n Slave and jdenet_k processes (also called kernels) at startup or as they are needed. JD Edwards EnterpriseOne uses a number of different types of kernels to handle different types of processing, even though all of these have the same process name in the operating system (jdenet_k). The definitions for the number of processes to start and what types to start are stored in the JDE.INI file.

2. The queue kernel process spawns the runbatch process whenever a relevant batch process request is placed in the Job Control Status Master table (F986110). The runbatch process completes the job, updates the F986110 table, and then quits. In JD Edwards EnterpriseOne, you use the Job Queue Maintenance program (P986130) to set up and manage the job queues.

   Nearly all jdenet_k processes access various other database tables as needed. The runbatch process, for instance, accesses and modifies any database table that is relevant to the particular business logic it is running.

3. Message queues are a type of interprocess communication (IPC) resource. They are allocated by the jdenet_n processes by calls to the operating system. While the software is running, operating system information about the message queues can be obtained by using the command ipcs.

When message packets are routed to the jdenet_n process from a client or another server, the jdenet_n process places them in the appropriate message queue according to the type of message. For example, when a client submits a batch process, a message is routed to the batch process kernel; when business logic needs to be run on the server, a request is routed to the CallObject kernel; when a user signs on to the system, a request is routed to the security kernel, and so on.

Each message queue has an identifier (IPC key) so that multiple processes can access them. JD Edwards EnterpriseOne uses a configurable IPC key range, which is controlled by the startIPCKeyValue in the JDE.INI file, in case a conflict occurs with other software that is using IPC resources.

3.1.2.1 jdenet_n Operation

The jdenet_n process usually starts when you run the supplied JD Edwards EnterpriseOne startup script: RunOneWorld.sh, which then starts all other processes as needed.

The jdenet_n process listens to the socket (port) as specified in the JDE.INI file by the keywords ServiceNameListen and ServiceNameConnect. These two keywords should be set to the same number; this number must be the same for every client who wishes to connect to the JD Edwards EnterpriseOne server.

The definitions for the particular jdenet_k processes to start are also given in the JDE.INI file. They are listed in the sections headed by [JDENET_KERNEL_DEFx]. Each of these entries lists the type of jdenet_k processes to start and the maximum number of jdenet_k processes of this type to start.

The number of jdenet_n slave processes to start is listed in the JDE.INI file under the keyword maxNetProcesses. The purposes of these slave processes are to provide parallel processing for the job of listening to the socket and to put the associated messages on the message queues for the jdenet_k processes to finish.

3.1.2.2 jdenet_k Operation

jdenet_k processes are referred to as kernel processes. They do the actual work on the enterprise server. When a jdenet_k process starts, it can be any type of kernel process. The jdenet_n process instructs each kernel process to be of a certain type.

The jdenet_k process that becomes a CallObject kernel has the job of calling business function logic on the server. Business function logic is written in C code and compiled into UNIX-shared libraries. The shared libraries are loaded onto the jdenet_k processes and then called directly through a C function call.

The jdenet_k process that becomes a batch process kernel waits for requests to run batch processes from the client. These batch processes are then placed in the Job Control Status Master table (F986110). The processes are then picked up by the queue kernel processes that launch runbatch processes, as required.

Many other types of jdenet_k processes exist. Review the JDE.INI file for a complete list.

3.1.3 JD Edwards EnterpriseOne Initialization for UNIX and Linux

This initialization occurs when you start JD Edwards EnterpriseOne programs, such as the queue kernel, runbatch, and so on:

- The environment name is passed as a command line argument to the program (such as porttest, runbatch, and so on).

- This environment can be translated to a different environment, based on the settings in the [SERVER ENVIRONMENT MAP] section of the JDE.INI file.
- The environment that is used must be a valid entry in the Library List Master File table (F0094). Likewise, it must have a valid corresponding path code in the Environment Detail table (F00941).
- These JDE.INI settings in the [DB SYSTEM SETTINGS] section are used to determine where the JD Edwards EnterpriseOne server startup tables, such as the Data Source Master (F98611) and the Object Configuration Master (F986101), are located:
  - Base Datasource
  - Object Owner
  - Server
  - Database
  - Load Library
  - Type
- Using this information, the F986101 table in the specified database on the server is opened.
- When an override for a given table or the current user exists, that data source (the OMDATP field in the F986101 table) is used for the given object or user and environment. Otherwise, the data source in which OMOBNM=DEFAULT for the given environment is used. Ignore any inactive records (that is, OMSTSO=NA).

  We strongly recommend that you do not have any default records for reports (OMOBNM=DEFAULT and OMFUNO=UBE) on the server. These records might prevent report interconnections (that is, one report calling another report) from starting correctly.
- Each unique data source in the F986101 table should correspond to one entry in the F98611 table.
- The corresponding information in the F98611 table must be correct. In particular, the OMDLLNAME field must display the correct library for the database to which the data source points.
- For an Oracle database, the OMDATB field from the F98611 table maps to an entry in the tnsnames.ora file. This tnsnames.ora file must be set up correctly. (Ask an Oracle database administrator to verify the setup).

3.2 Starting the Enterprise Server for UNIX or Linux

This section provides an overview of the enterprise server startup for UNIX or Linux and discusses how to:

- Start the enterprise server of UNIX or Linux manually.
- Start the enterprise server for HP-UX automatically.
- Start the enterprise server for AIX and Solaris automatically.
- Start the enterprise server for Linux automatically.
- Verify the JD Edwards EnterpriseOne installation.

3.2.1 Understanding Enterprise Server Startup for UNIX or Linux

You can start the enterprise servers either manually at the command line or automatically when the server boots. The manual process is the same for all supported platforms, but the automatic process varies slightly by platform.

> **Note:** If you are running JD Edwards EnterpriseOne on the same server as the Oracle database, you must make sure that Oracle is running before you start JD Edwards EnterpriseOne. In particular, if you are starting JD Edwards EnterpriseOne at system boot time, you must make sure the Oracle startup processes are completed first.

RunOneWorld.sh is the script that starts the JD Edwards EnterpriseOne system on the enterprise server. This script:

- Checks for existing JD Edwards EnterpriseOne processes.

  The script returns an error if it detects that JD Edwards EnterpriseOne is already running.

- Runs the rmics.sh script to clear IPC resources.

  This script ensures no IPC resources conflict with other software.

- Starts jdenet_n, which is the JD Edwards EnterpriseOne network listener that receives requests from JD Edwards EnterpriseOne workstations.

- Runs a program called cleanup that checks for unfinished batch processes from a previous shutdown.

The default database parameters for UNIX might not fully support multiple users. You might reach the maxprocess limit for the database. The initial settings are for a small database, so you should change these parameters to a medium setting to avoid database problems. These settings reside in the init.ora file. These path is an example of where you might typically find this file:

```
/u01/app/oracle/product/8.0.5/dbs/init.ora
```

3.2.2 Starting the Enterprise Server for UNIX or Linux Manually

To start the enterprise server for UNIX or Linux manually:

> **Note:** This procedure is the same for all supported UNIX or Linux operating systems.

1. Sign on to the machine using the appropriate user ID, as set up during the installation process.

   If you used the JD Edwards-recommended user ID, the user ID is jde.

2. Enter these commands:

   – `cd log_directory`

     This command moves the user's current directory to the log directory. The administrator determines the name of this directory.

   – `rm -f jde*log*`

This command deletes the log files in the directory.

Note: Use extreme care when you enter this command. A syntax error in this command can cause severe problems on the system.

– `RunOneWorld.sh`

This script starts the JD Edwards EnterpriseOne system.

3. Sign off the system.

3.2.3 Starting the Enterprise Server for HP-UX Automatically

To start the enterprise server for HP-UX automatically:

1. Create a script named psft in /sbin/init.d with all necessary permissions for execution.

   The script should contain only these:

   ```
   #! /sbin/sh

   /bi n/su - psft_user -c `$SYSTEM/bin32/RunOneWorld.sh'
   ```

   The value psft_user is the name of the user who owns the shell script $SYSTEM/bin32/RunOneWorld.sh. Make sure that no interactive commands appear in the psft_user profile, and that RunOneWorld.sh has all necessary permissions for execution.

2. Using this command, create a soft link named S995psft to the psft script in the directory named /sbin/rc2.d.

   ```
   ln -s /sbin/init.d/psft /sbin/rc2.d/S995psft
   ```

3. Verify that these line is present in the profile of the user who owns RunOneWorld.sh:

   ```
   /usr/local/bin/oraenv
   ```

   Before you execute oraenv, ensure that the Oracle environment variables of ORACLE_BASE, ORACLE_HOME, ORACLE_SID, ORACLE_TERM, and ORAENV_ASK are properly assigned and exported. Also, you must add $ORACLE_HOME/bin to the PATH environment variable.

4. Set ORACLE_TERM to **hp.**

5. Set ORAENV_ASK to **NO.**

6. If this command is in the profile, delete it:

   ```
   unset ORAENV_ASK
   ```

3.2.4 Starting the Enterprise Server for AIX and Solaris Automatically

To start the enterprise server for AIX and Solaris automatically:

1. Create a script named rc.psft in /etc with all necessary permissions for execution.

   The script should contain only these:

   ```
   #! /bin/sh
   ```

```
/bin/su - psft_user -c `$SYSTEM/bin32/RunOneWorld.sh'
```

The value psft_user is the name of the user who owns the shell script $SYSTEM/bin32/RunOneWorld.sh. Make sure there are no interactive commands in the psft_user .profile, and that RunOneWorld.sh has all the necessary permissions for execution.

2. Add this line at the end of the text file named inittab in /etc:

```
psft:2:wait:/etc/rc.psft
```

3. Verify that this line is present in the .profile of the user who owns RunOneWorld.sh.

```
. /usr/bin/oraenv
```

Before you execute oraenv, ensure that the Oracle environment variables of ORACLE_BASE, ORACLE_HOME, ORACLE_SID, ORACLE_TERM, and ORAENV_ASK are properly assigned and exported. Also, you must add $ORACLE_HOME/bin to the PATH environment variable.

4. Set ORACLE_TERM to **hp.**
5. Set ORAENV_ASK to **NO.**

To see a list of values for ORACLE_SID, look at the oratab text file in /etc.

6. If this command is in the .profile, you must delete it:

```
unset ORAENV_ASK
```

3.2.5 Starting the Enterprise Server for Linux Automatically

To start the enterprise server for Linux automatically:

Add this line to the rc.local file in the /etc directory:

```
/bin/su - psft_user -c `$SYSTEM/bin32/RunOneWorld.sh'
```

The value psft_user is the name of the user who owns the shell script $SYSTEM/bin32/RunOneWorld.sh. Make sure there are no interactive commands in the psft_user .profile (or .bash_profile), and that RunOneWorld.sh has all the necessary permissions for execution.

3.2.6 Verifying the JD Edwards EnterpriseOne Installation

To verify the JD Edwards EnterpriseOne installation:

After you start JD Edwards EnterpriseOne, execute these commands:

```
cd $SYSTEM/bin32

porttest userID password environment
```

The porttest program initializes an environment, initializes a user, opens the Account Balances table (F0902), and displays up to 99 rows of data.

Note: The parameters for userID, password, and environment should be any valid JD Edwards EnterpriseOne user ID, password, or environment.

3.2.6.1 Understanding Java Runtime Engine Installation Issue on Unix

With tools release 8.98 the Java Runtime Engine is bundled into the tools code. It will be in the directory .../system/jre. However, the problem is that prior to Apps Release 8.12 the installer will not set all the correct paths for this. So if a customer is using 9.0 with tools 8.98 they will most likely get an error like this from the JDE log:

```
12320/1 MAIN_THREAD                              Tue Nov  8 17:35:05.884717
ipcmisc.c299
        process 12320 </u04/oneworld/e1t_owa_stageing_development/system/bin32
/jdenet_n> registered
in entry 0

12320/1 MAIN_THREAD                              Tue Nov  8 17:35:37.374029
 netstart.c247
        Failed to autostart kernel in range 29


and get a core file when the jdenet_k #29 tries to start.  To fix this they will
 need to add the path .../system/jre/1.4 to the LD_LIBRARY_PATH, LIBPATH, SHLIB_
PATH.
```

3.3 Shutting Down the Enterprise Server for UNIX or Linux

The shutdown process is identical for all supported UNIX or Linux operating systems.

EndOneWorld.sh is the script that stops the JD Edwards EnterpriseOne system on the enterprise server. This script completes these functions:

- Checks for existing runbatch processes.

  If any runbatch (batch process) is running, the user is prompted to make sure that he or she wants to shut down the enterprise server.
- Checks for and ends JD Edwards EnterpriseOne processes other than jdenet_n and jdenet_k.
- Shuts down jdenet_n and jdenet_k processes by running endnet.
- Runs the rmics.sh script to clean up any remaining IPC resources.

3.3.1 Shutting Down the Enterprise Server for UNIX or Linux

To shut down the enterprise server for UNIX or Linux:

1. Sign on using the appropriate user ID that you set up during the installation process.
2. Execute these commands:

   ```
   cd $SYSTEM/bin32

   EndOneWorld.sh
   ```

3.4 Setting Up a Printer for UNIX or Linux

Each supported UNIX system use different processes for setting up printers. HP-UX uses a tool called SAM to help in setting up a printer; AIX uses a tool called SMIT; Solaris uses a tool called Admintool; and RedHat Enterprise Linux AS uses a tool called printgui-conf. Each of these processes requires a privileged account to access the specific setup tasks. Normally, you will need to use the root account of the system. For more information about printer setup, see the appropriate HP-UX, AIX, or Solaris documentation.

See Also:

- "Defining Print Properties for Reports" in the *JD Edwards EnterpriseOne Tools Development Tools: Report Printing Administration Technologies Guide*.

3.5 Administering Batch Processes for UNIX or Linux

This section provides an overview of batch process administration for UNIX or Linux and discusses how to:

- Monitor batch processes.
- List batch output files.
- Run reports form the command line for UNIX or Linux.
- Schedule reports from the command line for UNIX or Linux.

3.5.1 Understanding Batch Process Administration for UNIX or Linux

Administering batch processes involves knowing what processes run when JD Edwards EnterpriseOne starts, where files are placed before and after printing, and how to watch those processes.

Processes running for JD Edwards EnterpriseOne are owned by the user who started the JD Edwards EnterpriseOne software. The user ID for this user is set up during the installation of the software, and is site dependent. When JD Edwards EnterpriseOne starts, these processes start and run under the environment and security of the user who started them:

Process	Description
jdenet_n	The network listener that listens for connection requests.
jdenet_k	The jdenet_n process starts the jdenet_k processes, which control JD Edwards EnterpriseOne components, such as the security server, the transaction monitor, and data replication.

Use the jdejobs command to monitor current batch processes. This example is a sample output:

```
pfst900 (EnterpriseOne Admin,,,):
Semaphores: 1  Shmem Segs: 5  Msg.Queues: 13
```

```
Jobs on ent-1:
6137 ttyp6   0:43 jdenet_n
6163 ttyp6   0:44 jdenet_k
6188 ttyp6   0:44 jdenet_k
7213 ttyp6   2:12 jdenet_n
7241 ttyp6   0:47 jdenet_k
9008 ttyp6   1:36 jdenet_n
9009 ttyp6   0:45 jdenet_k
11042 ttyp6   0:09 runbatch
```

In the output, jdenet_n jobs are listening for requests, and four jdenet_k jobs are handling various JD Edwards EnterpriseOne kernel functions. A runbatch job is processing a report.

The first column of the output displays the UNIX process ID that is associated with each process. For more information about a particular process, look for the files in the log directory that have the same process ID as part of the file name.

All output from each report, regardless of whether it is a preview, is placed in the PrintQueue directory under the installation directory of JD Edwards EnterpriseOne before printing. Depending on the JDE.INI settings for the workstation, the job might not be deleted after printing.

Jobs are printed to the location specified in the JDE.INI file unless a JD Edwards EnterpriseOne program overwrites them. Use the Printers program to specify default printers.

Two settings in the JDE.INI file for the workstation tell the server whether to print the report immediately upon completion, and whether to save the output from the report or delete it. These settings are as follows:

```
 [NETWORK QUEUE SETTINGS]

SaveOutput=TRUE

PrintImmediate=TRUE
```

Setting SaveOutput to TRUE causes the JDE.INI to hold the jobs within the PrintQueue directory until the user explicitly deletes them. Setting PrintImmediate to TRUE tells the JDE.INI file to print the job immediately after completion of the report.

You can list output files. The returned data looks similar to this:

```
R014021_XJDE0001_4554_PDF
R014021_XJDE0001_4554_PDF.jde.log
R014021_XJDE0001_4554_PDF.jdedebug.log
R31515_XJDE0001_4566_PDF
R31515_XJDE0001_4566_PDF.jde.log
R31515_XJDE0001_4566_PDF.jdedebug.log
R94NM08_XJDE0008_4568_PDF
R94NM08_XJDE0008_4568_PDF.jde.log
R94NM08_XJDE0008_4568_PDF.jdedebug.log
R94NM10_XJDE0016_4526_PDF
R94NM10_XJDE0016_4526_PDF.jde.log
R94NM10_XJDE0016_4526_PDF.jdedebug.log
R94NM10_XJDE0016_4526_PDF.ps
R94NM10_XJDE0016_4527_PDF
R94NM10_XJDE0016_4527_PDF.jde.log
R94NM10_XJDE0016_4527_PDF.jdedebug.log
R94NM10_XJDE0016_4527_PDF.pcl
```

The file names in this example are the actual reports that were generated when the job was executed. The file naming conventions are as follows:

Segment	Description
R31515	The report name
XJDE00001	The report version executed
1914	The request number assigned by JD Edwards EnterpriseOne
PDF	A PDF file, meant for viewing on the workstation
.jde.log	The file extension that indicates the log file for the report
.jdedebug.log	The file extension that indicates the debug log for the report
.ps	The file extension that indicates a file formatted for postscript printing
.pcl	The file extension that indicates a file formatted for pcl printing

You should encourage workstation users to use the SaveOutput=FALSE entry in their jde.ini file. If users at workstations decide to save their output, they should periodically delete the entries through JD Edwards EnterpriseOne. When you delete .PDF files from the operating system, the corresponding JD Edwards EnterpriseOne print job entries in the Job Control Status Master table (F986110) are not deleted. You must manually delete these entries from JD Edwards EnterpriseOne using the Work with Servers program (P986116).

3.5.2 Monitoring Batch Processes

To monitor batch processes:

From the operating system prompt, enter this command, replacing userid with the user ID of the user who started JD Edwards EnterpriseOne:

```
jdejobs <userid>
```

If you omit the user ID, the current user is assumed.

jdejobs is a script in the JD Edwards EnterpriseOne $SYSTEM/bin32 directory that uses the UNIX ps command to display job information.

3.5.3 Listing Batch Output Files

To list batch output files:

1. From the operating system prompt, enter this command:

   ```
   cd $EVRHOME/PrintQueue
   ```

   This command changes the directory to the PrintQueue directory. The environment variable EVRHOME should be set to the JD Edwards EnterpriseOne installation directory.

2. Enter this command to list the files:

   ```
   ls
   ```

3.5.4 Running Reports from the Command Line for UNIX or Linux

You can initiate batch process reports from the server command line by issuing this command (you must have the proper authority and the path equal to the description in the installation instructions) :

```
runube  <[-p|-P] [-f|-F|-d|-D passfile] [user password]>
   <Environment>
   <Role>
   <ReportName>
   <VersionName>
   <JobQueue>
   <"Interactive"|"Batch">
   <"Print"|"Hold">
   <"Save"|"Delete">
    [Printer]
```

The format for the passfile parameter is:

```
[enterpriseoneusername
 password]
```

Note: The [user password] parameter has been deprecated.

For the command parameters, only the first character of the parameter name is required. The vertical bar symbol (|) indicates that you must specify one of the parameters on either side of the vertical bar. The brackets indicate an optional parameter. These options apply to the runube command:

Parameter	Description
–p	Prompts for user/password information.
–P	Prompts for user/password information.
–f	Reads user/password information from the plain text file that is specified in passfile (FilePath).
–F	Reads user/password information from the plain text file that is specified in passfile (FilePath).
–d	Reads user/password information from the plain text file and indicates the automatic removal of the file after the job has read the credentials from it.
–D	Reads user/password information from the plain text file and indicates the automatic removal of the file after the job has read the credentials from it.
Interactive	The system holds the current terminal session until the entire report is processed.
Batch	The runube command submits the job to a UBE kernel, which in turn will send the job submission to Queue kernel, then returns control of the terminal to the user.
Print	After the batch process completes generating its output, the output is printed to the specified printer queue. If you do not specify a printer on the runube command line, the system uses the EnterpriseOne user's default printer specified in the Printers program (P98616).

Parameter	Description
Hold	The batch output is not printed immediately after the job completes, but may be printed later from 'Work with Submitted jobs'
Save	The system retains the report's output and all records of its execution.
Delete	The system removes any output from the PrintQueue directory, from Report definition output, and also removes the records of the jobs execution from F986110 after the report prints.

3.5.4.1 Example: Running Reports from the Command Line for UNIX or Linux

This example displays a command for executing a batch process report:

```
runube KL5952 mypass PROD *ALL R0006P XJDE0001 QBATCH I P D printer_1
```

3.5.5 Scheduling Reports from the Command Line for UNIX or Linux

You can schedule a report from the command line for processing on a future date, daily, or even a recurring day of the week. This task can be accomplished by using the operating system utilities called at, batch, and cron. The batch and at utilities are used to schedule single occurrence jobs; cron can be used to schedule recurring jobs. Use the at command or the batch command to schedule a job at a later time. The command line structure of these commands is identical, but you use them differently.

The batch command is intended to run a job immediately in the background, providing that the system load is low enough to handle the request. If the system load is not low enough, the job is held until system activity is low enough to handle the new request load.

The at command also runs jobs in the background, but enables you to schedule the job to run at a future time. You can use this utility to run the batch job during off-peak hours.

The command format for the batch command is as follows:

```
batch command
```

The command format for the at command is as follows:

```
at -t CCYYMMDDHHMMSS command
```

The -t switch is used to schedule the time. This table describes the CCYYMMDDHHMMSS variable:

Segment	Description
CC	Century (first two digits of the year).
YY	Year (last two digits of the year).
MM	Two-digit value of the month (such as 02 for February).
DD	The day of the month (01 - 31).
HH	The hour to start the job (00 - 23).
MM	The minute to start the job (00 - 59).
SS	The second to start the job (00 - 59).

Segment	Description
command	The command to run at the specified time. To schedule a report, use the runube command.

You can use the cron UNIX utility to run jobs at a scheduled time. You can specify variable times, such as once a year or once every hour. The operation of this utility is controlled by a table of events based upon each user.

Enter this command to modify the cron schedule and edit the cron table for the current user:

```
crontab -e
```

The format of the cron table is as follows:

```
mm HH DD MM W command
```

This table describes the variables for this command:

Segment	Description
mm	The minute to run the job (00 - 59, or * for any minute).
HH	The hour to run the job (00 - 23, or * for any hour).
DD	The day of the month to run the job (0 - 31, or * for any day).
MM	The month to run the job (1 - 12, or * for any month).
W	The day of the week to run the job (0 - 6, with 0 being Sunday).
command	The command to run at the specified time.

After exiting the editor, the operating system should respond with a message stating that the crontab has been modified.

3.5.5.1 Example: Scheduling Single-Occurrence Reports from the UNIX or Linux Command Line

This example displays a command line used to schedule a report to run at 06:00 on February 26, 2005:

```
at -t20050226060000 runube KL595218 mypass PROD *ALL R0006P XJDE0001
QBATCH Interactive Print Delete printer_1
```

3.5.5.2 Example: Scheduling Recurring Reports from the UNIX or Linux Command Line

This example displays a command line used to schedule a report to run at 06:00, any Sunday in the month of February (by the use of * for the day of the month and 0 for the day of the week).

```
00 06 * 02 0 runube KL5952 mypass PROD *ALL R0006P XJDE0001 QBATCH
Interactive Print Delete printer_1
```

3.6 Maintaining File Security for UNIX and Linux

This section provides an overview of file Security and discusses how to:

- Set specification file security.
- Set business function file security.
- Set executables security.
- Set jde.ini file security.

3.6.1 Understanding File Security Maintenance for UNIX and Linux

Overall, only two accounts ever need operating system access to the JD Edwards EnterpriseOne environment files and version executables: the account that starts and stops JD Edwards EnterpriseOne, and the account that builds the environment SPEC and BSFN files. Normally, these accounts are the same.

Specification (SPEC) files are the first part of the environment files. You access these files by the JD Edwards EnterpriseOne kernel processes. These files should never be accessed directly by an operating system user. Because of this, security on these files should be read/write for the user and role. They are not executables, so no reason exists for setting the executable option for any user, or role.

Business function security should be similar to SPEC file security. This enables the business function code to be viewed, but not modified directly on the server. In general, both business function changes and SPEC file changes are controlled by the deployment server.

You should prevent access to the JD Edwards EnterpriseOne executable files to prevent other users from attempting to start JD Edwards EnterpriseOne. Running the same version of JD Edwards EnterpriseOne on the same system and using the same JDE.INI settings can cause unpredictable results. In most cases, the second startup will fail, but giving users access to the shutdown procedures can enable them to shut down JD Edwards EnterpriseOne.

You must keep the jde.ini file as secure as possible. This file contains a database user name and password that enables JD Edwards EnterpriseOne security to function. This database account is given read authority to the JD Edwards EnterpriseOne Security table (F98OWSEC), which controls JD Edwards EnterpriseOne access.

Access to the F98OWSEC table, which contains privileged database user names and passwords, could give a user the ability to manipulate any data in the database, regardless of its sensitivity or security. Because of this, you should restrict access to the jde.ini file as much as possible.

3.6.2 Setting Specification File Security

To set specification file security:

Add this line to the .profile:

```
umask 022
```

This command sets the default file security for files that get created on the server. When a package build completes, SPEC files and business functions should be created with read permission for everyone, and with write permission for only the file owner. In general, both business function changes and SPEC file changes are controlled by the deployment server.

The security for the SPEC files should look similar to these example:

```
-rw-r--r-- psft psft jdeblc.xdb

-rw-r--r-- psft psft jdeblc.ddb
```

3.6.3 Setting Business Function File Security

To set business function file security:

1. Enter this command in the BSFN Source directory:

   ```
   chmod 644 *.c
   ```

2. Enter this command in the BSFN Include directory:

   ```
   chmod 644 *.h
   ```

   The security for the BSFN files should look similar to these example:

   ```
   -rw-r--r-- psft psft b4200100.c

   -rw-r--r-- psft psft b4200100.h
   ```

3.6.4 Setting Executables Security

To set executables security:

UNRECOGNIZED STYLE ->class=singlestep>Enter this command:

```
cd $SYSTEM/bin32

chmod 540 *..sh
```

The access granted by this command gives all users in the JD Edwards EnterpriseOne role read-only permission to the files, but does not grant them execute privilege. You can omit read access if desired.

The security for the JD Edwards EnterpriseOne executables should look similar to these example:

```
-r-xr----- psft psft RunOneWorld.sh

-r-xr----- psft psft EndOneWorld.sh
```

3.6.5 Setting jde.ini File Security

Warning: Implementing JDE.INI file security will prevent Server Manager from modifying configuration settings.

To set JDE.INI file security:

1. Enter this command:

   ```
   cd $JDE_BASE

   chmod 600 JDE.INI
   ```

   This command sets maximum security for the JDE.INI file. The JDE_BASE environment variable is set to the directory that contains the JDE.INI file.

   Note: The file name is case-sensitive.

The security for the JDE.INI file should look similar to this:

```
-rw------- psft psft JDE.INI
```

Denying write access to the user psft is not strictly necessary, but can prevent accidental modification of JDE.INI settings, which could adversely affect the operation of JD Edwards EnterpriseOne.

2. If you want to deny the user write access, enter this command:

```
chmod 400 JDE.INI
```

Because it is important to keep access to the JDE.INI file as secure as possible, you should also limit the amount of access to the user psft (or the user account that starts and stops JD Edwards EnterpriseOne) to a minimum. Users with access to this account might obtain the user names and passwords in the F98OWSEC table, and, thus, gain privileged access to the database.

Note: Denying write access may prevent Server Manager from being able to change the configuration items within the JDE.INI file.

3.7 Working with HP-UX and Solaris Kernel Parameter Settings

Beginning in Solaris 10, Sun made a major change in the way IPC parameters are handled. These parameters are no longer changed through the /etc/system file - some have been eliminated, and some must now be changed through the Solaris resource controls facility. A discussion of Solaris resource controls is beyond the scope of this document; however, we will provide one example here that may be required before starting JD Edwards EnterpriseOne for the first time. When starting the JD Edwards EnterpriseOne software on the server, the initial jdenet_n process tries to create a semaphore array containing the number of elements indicated by the "maxNumberOfSemaphores" parameter in the enterprise JDE.INI file. By default, Solaris 10 will allow a semaphore array with a maximum of 512 elements. If the setting in the JDE.INI file is greater than 512, the system default will have to be changed. In Solaris 10, the following command is the simplest way to change the default:

```
projmod -K 'process.max-sem-nsems=(privileged,2048,deny)' default
```

This command changes the default project to allow semaphore arrays with up to 2048 elements. This is most likely the only resource control that you will have to change in Solaris 10 to be able to start JD Edwards EnterpriseOne. To see the resource control limits for a given user, sign on to that user and run the following command:

```
prctl $$
```

See the Solaris documentation "System Administration Guide: Solaris Containers-Resource Management and Solaris Zones" at http://docs.sun.com..

The kernels for HP-UX and Solaris (prior to Solaris 10) include a long list of configurable parameters. These parameters control the quantity of various resources available within the HP-UX and Solaris kernels. Also, the JD Edwards EnterpriseOne server software, specifically the IPC facilities, is sensitive to numerous kernel parameters for operation. These parameters differ among the various vendor implementations of UNIX. To change the values of kernel parameters for HP-UX, you must use the System Administration Management (SAM) tool to modify the parameters, which might require rebooting the system. For Solaris, you must reboot the system after you modify kernel parameters in the /etc/system file. The proper

values of these parameters depend on various criteria, such as number of users on the system, active applications, and the resource requirements for the active applications.

For HP-UX, you set kernel parameters with the SAM tool. To modify these parameters for Solaris, open the /etc/system file with the a text editor. You can set any given parameter to either a simple numerical value or an expression, based on the values of other parameters. The system administrator must set the kernel parameters. UNIX security refers to users with access to administrative functions as superusers.

When you first set up an HP-UX or a Solaris machine for JD Edwards EnterpriseOne, you should run SAM for HP-UX or an editor for Solaris, and change the kernel parameters. On an HP-UX system, you can see the current values of kernel parameters by running the kmtune command, or by running SAM. On a Solaris system, type the command sysdef -i to see the current kernel settings.

JD Edwards EnterpriseOne is not the only software to use the resources that the kernel parameters control. Therefore, for each parameter, the requirements for JD Edwards EnterpriseOne are either the minimum defaults provided with HP-UX and Solaris, in addition to the defaults provided with HP-UX and Solaris, or the requirements of other software installed on the system.

Note: The number of JD Edwards EnterpriseOne users that a machine serves, the number of instances of JD Edwards EnterpriseOne server software running on a machine, and the size of any databases on the machine are primary factors that affect the settings for HP-UX and Solaris kernel parameters. The number of jdenet_n, jdenet_k, and runbatch or runube processes running should reflect this information.

This list provides the definitions of terms essential to the understanding of HP-UX and Solaris kernel parameters:

Parameter	Definition
jdenet_n	The maximum number of jdenet_n (net) processes that can be created for an instance JD Edwards EnterpriseOne server software running on the system. This is controlled by the maxNetProcesses parameter in the JDENET section of the JDE.INI file for each instance of JD Edwards EnterpriseOne.
jdenet_k	The maximum number of jdenet_k (kernel) processes that can be created for an instance of JD Edwards EnterpriseOne server software running on the system. This is controlled by the maxKernelProcesses parameter in the JDENET section of the JDE.INI file for each instance of JD Edwards EnterpriseOne. Note that the maxNumberOfProcesses parameters in the JDENET_Kernel_Def sections do not matter here.

This screen capture provides an example of a Solaris editor that displays information for shared memory segments. The parameter name appears at the end of each line in the editor, such as shmmax at the end of this line: set shmys shminfo_ shmmax=4294967295:

3.7.1 Message Queues

Generally, the system clears queues quickly. If a problem arises, you can revise values for these parameters to rectify the situation:

Parameter	Description
mesg	This value must be 1. System-V style message queues are valid.
msgmni	The value of msgmni represents the number of message queue identifiers. These identifiers determine the number of message queues that can exist throughout the system. In addition to the system default value and the requirements of other software, calculate what is needed for the JD Edwards EnterpriseOne installation (per JD Edwards EnterpriseOne instance). You can use these equation to estimate the number of message queues necessary for JD Edwards EnterpriseOne: 1 + jdenet_n + 2 x jdenet_k + (max number of concurrent runbatch, runube, and runprint processes)
msgtql	The value of msgtql represents the number of message headers. This number determines the total number of messages that can be in all the message queues at the same time. In addition to the requirements of other software, allow a value equal to 10 x msgmni for the requirements of JD Edwards EnterpriseOne.
msgmap	The value for msgmap represents the number of entries in the map of free message segments. The default value of msgtql + 2 should be used. If the value of msgmap is less than the value of msgtql + 2, attempts to create a message queue or to send a message might fail. **Note:** This parameter is no longer used in Solaris 8.
msgmnb	The value of msgmnb represents the maximum number of bytes that can reside on a single message queue at the same time. You should set the value for msgmnb at only a fraction of msgseg x msgssz. For JD Edwards EnterpriseOne, a value of 32768 is reasonable. You can set a larger value as long as the product of msgseg x msgssz is large enough. The minimum value is 8192. Additional requirements of this parameter might increase the value of msgmnb.
msgmax	The value of msgmax represents the maximum size, in bytes, of a single message. Do not set msgmax with a larger value than the value of msgmnb. The recommended setting is msgmax = msgmnb. The minimum value is 1024. Additional requirements of this parameter might increase the value of msgmax.

Inside the HP-UX and Solaris kernels (prior to Solaris 8), messages in message queues reside in message segments. These parameters, which do not apply to Solaris 8, determine the size and number of segments available throughout the system:

Parameter	Description
msgssz	The value of msgssz represents the size of each message segment in bytes. For JD Edwards EnterpriseOne, a value of 64 is adequate for most situations.
msgseg	The value of msgseg represents the number of message segments throughout the system. In addition to the requirements of other software, allow a value equal to 50 x the msgmni requirement for JD Edwards EnterpriseOne, or approximately 4096 per instance.

3.7.2 Semaphores

These definitions apply to semaphores:

Parameter	Description
sema	This value must be 1. System-V style message queues are valid.

Parameter	Description
semmni	The value of semmni represents the maximum number of semaphore identifiers that can exist throughout the system. For JD Edwards EnterpriseOne, two identifiers exist for each instance of JD Edwards EnterpriseOne, so the default value supplied with the HP-UX and Solaris systems should suffice.
semmap	The value of semmap represents the number of entries in the map of free semaphores. The default value of semmni + 2 should suffice. If you decrease the value of semmap, attempts to create a semaphore set, which occurs during JDEIPC initialization, might fail. **Note:** This parameter is not used in Solaris 8.
semmns	The value of semmns represents the maximum number of semaphores that can exist throughout the system. Each instance of JD Edwards EnterpriseOne allocates 1000 semaphores by default. However, you can customize this value in the JDE.INI file. In the [JDEIPC] section, modify the parameter maxNumberOfSemaphores to customize the number of semaphores that an instance of JD Edwards EnterpriseOne allocates. For all releases of JD Edwards EnterpriseOne, the JD Edwards EnterpriseOne requirement is in addition to the requirements of other software. A good starting point for a typical JD Edwards EnterpriseOne installation (single instance) with Oracle should be 2000.
semmnu	The value of semmnu represents the maximum number of semaphore undo structures for the entire system. Effectively, this value is the maximum number of semaphores that the system can lock at the same time. For JD Edwards EnterpriseOne, enable one for each JD Edwards EnterpriseOne process that can exist for all installations of JD Edwards EnterpriseOne on the system. Use these equation to determine this value: 1 + jdenet_n + jdenet_k + maximum number of runbatch processes + maximum number of runprint processes + maximum number of runube processes **Note:** This equation is similar to the equation used to calculate the value for msgmni. If you will be running a large number of batch queues or print jobs, you might need to increase the value of this parameter. The number of outstanding print requests at a given time, whether printing or waiting for a printer, determines the number of jdeprint processes. A reasonable estimate for the upper limit of this value is 10. However, this estimate is application-dependent. For example, a large warehouse that constantly prints pick slips might have more requests. The number of batch processes that run directly on the server, not from a client, determine the number of runube processes. This value depends on the use of the system. Theoretically, this value has no limit.
semume	The value of semume represents the maximum number of semaphore undo structures per process. Effectively, this value is the maximum number of semaphores that a given process can lock at the same time. JD Edwards EnterpriseOne requires a minimum value of 4 for semume. This minimum value is not in addition to the system default and the requirements of other software. This value is a simple minimum. The default value provided with the system should suffice.
semmsl	The value for semmsl, which applies to Solaris and newer versions of HP-UX, represents the maximum number of semaphores per unique identifier. For JD Edwards EnterpriseOne, this must be set equal to or higher than the maxNumberOfSemaphores setting in the JDE.INI file. For the default installation, you should set this parameter to 1000.

3.7.3 Shared Memory

These definitions apply to shared memory:

Parameter	Description
shmem	The shmem value must be 1 to enable shared memory.
shmmax	The value of shmmax represents the maximum size, in bytes, of a single shared memory segment. The default value provided with the system should suffice. Other software packages, such as Oracle, might require an increase in this value.
shmmni	The value of shmmni represents the maximum number of shared memory segments throughout the system. For JD Edwards EnterpriseOne, enable 20 per instance of the JD Edwards EnterpriseOne server software running on the system. This requirement is in addition to the system default value and the requirements of other software.
shmseg	The value of shmseg represents the maximum number of shared memory segments to which any one process can attach at a given moment. The default value provided with the system should suffice.

3.7.4 File Descriptors

These definitions apply to file descriptors:

Parameter	Description
nfile	The value of nfile represents the maximum number of open files, or sockets, throughout the system. The default value should be enough to handle most JD Edwards EnterpriseOne needs. However, you must make explicit allowance for the maximum number of sockets that jdenet_n processes can create to communicate with clients. This number is the sum of all sockets for all instances of JD Edwards EnterpriseOne server software that runs on the system. The maxNetConnections parameter in the [JDENET] section of each JDE.INI file indicates this sum. This requirement is in addition to the system default value and the requirements of other software.
maxfiles	(rlim_fd_cur in the Solaris /etc/system file) The value of maxfiles represents the default soft limit on the number of file descriptors that any given process can have. A system call can raise the soft limit of a process as high as maxfiles_lim. For JD Edwards EnterpriseOne, the minimum value for maxfiles should equal at least the largest of all the maxNetConnections values in all the JDE.INI files in use + 10. This requirement is a minimum value, not a value in addition to the system default value and the requirements of other software. **Note:** If this parameter is too small, JD Edwards EnterpriseOne might not open the log file to generate an error message.
maxfiles_lim	(rlim_fd_max in the Solaris /etc/system file) The value of maxfiles_lim represents the hard limit of file descriptors that any given process can have. For JD Edwards EnterpriseOne, the minimum value for maxfiles should equal at least the largest of all the maxNetConnections values in all of the JDE.INI files in use + 10. This requirement is a minimum value, not a value in addition to the system default value and the requirements of other software.

3.7.5 Processes

This definition applies to processes:

Parameter	Description
maxuprc	The value of maxuprc represents the maximum number of processes that can run under a single user ID. This number is of particular concern on systems with either a very large JD Edwards EnterpriseOne installation or multiple instances running under the same user ID. You must allow for the total number of JD Edwards EnterpriseOne processes that might run at one time, plus other system processes that the JD Edwards EnterpriseOne user might be running.

3.8 Working with Linux Kernel Parameter Settings

This section provides an overview of Linux kernel parameter settings.

3.8.1 Understanding Linux Kernel Parameter Settings

The Linux operating system uses many of the same kernel parameters as Solaris, but they are managed in a slightly different way. In the Linux 2.4 kernel, IPC parameters are defined and maintained in the /proc file system, in the directory /proc/sys/kernel. They can be modified dynamically by editing the appropriate file, but for enterprise applications, you should override the default parameters at boot time. In RedHat Enterprise Linux, the default parameters can be overridden at boot time by adding entries to the /etc/sysctl.conf file. Use the command ipcs -l to view the current values for IPC resource limits.

3.8.1.1 IPC Resources

These five entries in the /etc/sysctl.conf file affect JD Edwards EnterpriseOne IPC resources:

Parameter	Description
kernel.sem	This setting controls these four different semaphore limits: ■ Maximum number of semaphores per array (semmsl on Solaris). ■ Maximum number of semaphores in the system (semmns). ■ Maximum operations per semop call (semopm). ■ Maximum number of semaphore arrays (semmni). For JD Edwards EnterpriseOne, you might need to increase the first value, semaphores per array, particularly if you increase the value of maxNumberOfSemaphores in the jde.ini file. Some database products also require that the fourth value, number of semaphore arrays, be increased from the default value.
kernel.shmmax	The default value for this parameter might be sufficient for JD Edwards EnterpriseOne, but some database products recommend that this be set to 256 Mb, or 90 percent of total memory, whichever is greater.
kernel.msgmax	This parameter defines the maximum size of a message. The recommendation for JD Edwards EnterpriseOne is 65535.
kernel.msgmnb	This parameter defines the maximum number of bytes on a message queue. The recommendation for JD Edwards EnterpriseOne is 65535.
kernel.msgmni	This parameter defines the maximum number of message queues (identifiers) in the system. You can use these equation to estimate the number of message queues that are necessary for JD Edwards EnterpriseOne: 1 + jdenet_n + 2 x jdenet_k + (max number of concurrent runbatch, runube, and runprint processes)

3.8.1.2 File Limits

In addition to the IPC resource limits, WebSphere and the JD Edwards EnterpriseOne HTML Server can require a large number of open files. To see the current values, review the file /proc/sys/fs/file-nr. This read-only file contains these three values:

- Total allocated file handles
- Currently used file handles
- Maximum file handles

The first value represents a peak, so when this value approaches the maximum value, consider raising the limit. If the peak value reaches the limit, you will get unpredictable results because processes will not be able to open files. To change the maximum file handle limit, use the fs.file-max setting. This setting controls the maximum number of files that can be simultaneously open throughout the entire system. The recommendation for JD Edwards EnterpriseOne is 32768, and this number might need to be increased to 65536 for larger installations.

3.8.1.3 Example: /etc/sysctl.conf

These lines are from a typical sysctl.conf file that are used to set kernel parameters based on the previous information:

```
fs.file-max = 32768

kernel.shmmax = 268435456

kernel.sem = 500 32000 32 1024

kernel.msgmax = 65535

kernel.msgmnb = 65535

kernel.msgmni = 1024
```

3.9 Working with AIX Kernel Parameter Settings for JD Edwards EnterpriseOne

This section provides an overview of AIX kernel parameter settings for JD Edwards EnterpriseOne and discusses how to:

- Set the value of maxuproc.
- View the system parameters.
- Set tune parameters.

3.9.1 Understanding AIX Kernel Parameter Settings for JD Edwards EnterpriseOne

AIX contains a set of kernel parameters (system parameters) that determine functionality and a separate set of performance parameters (tune parameters) that determine performance.

Setting the kernel parameters requires you to run the system management tool (SMIT). AIX has few configurable parameters that influence JD Edwards EnterpriseOne software; of those that influence JD Edwards EnterpriseOne, just one can cause the software to become inoperable. This parameter is maxuproc. The maxuproc parameter controls the number of processes that a single user can run simultaneously.

3.9.2 Setting the Value of maxuproc

To set the value of maxuproc:

1. Sign on as the root user.
2. On the command line, enter this command:

   ```
   smit
   ```

3. In SMIT, select the System Environments item and then select the Change/Show Characteristics of Operating System item.
4. Change the value of Maximum number of processes to enable all JD Edwards EnterpriseOne processes that might run at one time, plus any other system processes the JD Edwards EnterpriseOne user might be running.

   Accept the default values for all other system parameters. This table lists these system parameters for general reference:

Parameter	Description
maxbuf	Max pages in block I/O buffer cache.
maxmbu	Max real memory for MBUFS.
autorestart	Automatically reboot after crash.
iostat	Continuously maintain disk I/O history.
maxpout	High water mark for pending write I/O per file.
minpout	Low water mark for pending write I/O per file.
keylock	State of system keylock at boot time.
fullcore	Enable full core dump.
pre43core	Use pre-430 style core dump (AIX 4.3 only).
logfilesize	Error log file size.
memscrub	Enable memory scrubbing.
dcache	Size of data cache in bytes.
icache	Size of instruction cache in bytes.
realmem	Size of usable physical memory.
primary	Primary dump device.
conslogin	System console login.

3.9.3 Viewing the System Parameters

To view the system parameters:

Enter this command:

```
lsattr-E-lsys0
```

To change a system parameter, you must navigate to the correct SMIT menu option.

3.9.4 Setting Tune Parameters

Setting the tune parameters requires you to run these commands:

- For network parameters: no

- For device parameters: chdev
- For nfs parameters: chnfs
- For general tuning parameters: vmtune

Tune parameters can also be kept at their default values. Changes to tune parameters are generally needed only for performance reasons. Proper settings for optimal performance might vary with changes in the underlying database, hardware configuration, and JD Edwards EnterpriseOne configuration.

Performance tuning for AIX running JD Edwards EnterpriseOne or Oracle involves setting parameters that control virtual memory for paging, Raid, disk system types, and CPU scheduling.

3.9.4.1 Example: Disk Striping

Disk striping is the technique of spreading sequential data across multiple disk drives so data can be accessed in parallel from several drives at once. If striping is used, then these tune parameters are set:

Parameter	Value
stripe size	64KB
max_coalesce	64KB
minpgahead	2
maxpgahead	16 x number of disk drives
maxfree	minfree + maxpgahead

3.10 Running Multiple Instances of the JD Edwards EnterpriseOne Enterprise Server

This section provides an overview of running multiple instances of the JD Edwards EnterpriseOne enterprise server and discusses how to do so.

3.10.1 Understanding Running Multiple Instances of the JD Edwards EnterpriseOne Enterprise Server

Common reasons for running multiple instances of the JD Edwards EnterpriseOne enterprise server are to test a new service pack or to upgrade to a new version of JD Edwards EnterpriseOne. You can run multiple instances of the JD Edwards EnterpriseOne server on the same machine by following a few simple guidelines.

Note: These steps do not create a new database or any new database tables. Therefore, you will be using the same data tables that are used by the original instance of JD Edwards EnterpriseOne that was installed. If you want to create a completely separate set of database tables, follow the instructions for setting up a new environment.

After you make all of the changes described in this chapter, you can start and stop the new JD Edwards EnterpriseOne instance independently of the original instance.

All existing JD Edwards EnterpriseOne environments will be valid for the new instance, provided that you have copied the corresponding path code directory for a

given environment. All current logical data sources and OCM mappings will be recognized by the new instance.

Server Manager fully supports multiple foundations. This includes the installation and management of multiple instances of JD Edwards EnterpriseOne on a single server.

See Also:

- *Server Manager Guide* on My Oracle Support, sections: Register/Install on JD Edwards Enterprise Server for EnterpriseOne.
- "Adding an Environment" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

3.10.2 Prerequisite

Verify that you have enough disk space to create copies of the current JD Edwards EnterpriseOne system directory and at least one path code directory.

3.10.3 Running Multiple Instances of the JD Edwards EnterpriseOne Enterprise Server

To run multiple instances of the JD Edwards EnterpriseOne enterprise server:

1. The system administrator should create a new user ID that owns the new JD Edwards EnterpriseOne instance.

   Create the user ID using the appropriate administration tool, such as smit, SAM, admintool, or useradd.

   Note: Although you can run multiple instances of the JD Edwards EnterpriseOne server using the same UNIX or Linuxuser ID, it is not recommended. The software depends on certain environment variables to function correctly, and these variables are easier to manage under different user IDs.

2. Sign on using the new user ID.
3. Copy the .profile and .psft files from the home directory of the original user ID to the home directory of the new user ID.
4. Change the .profile file for the new user ID, if appropriate.
5. Change the .psft file for the new user ID to reference the new directory path in which you will create the new JD Edwards EnterpriseOne instance.

   For example:

   Original .psft file:

   ```
   export EVRHOME=/u01/JDEdwards/E900
   ```

   New .psft file:

   ```
   export EVRHOME=/u02/JDEdwards/E900
   ```

6. Create the directory in which the new JD Edwards EnterpriseOne instance will reside.

   For example, type these:

   ```
   mkdir -p /u02/JDEdwards/E900
   ```

7. Copy the system directory, the ini directory, and at least one path code directory from the original instance of JD Edwards EnterpriseOne to the new directory path.

   These sample commands accomplish this:

   ```
   cp -R /u01/JDEdwards/E900/system /u02/JDEdwards/E900
   cp -R /u01/JDEdwards/E900/ini /u02/JDEdwards/E900
   cp -R /u01/JDEdwards/E900/DV900 /u02/JDEdwards/E900
   ```

   Note: The path code directories for any environments that you intend to use for this second instance of JD Edwards EnterpriseOne must be copied to the new directory. You cannot share path code directories between two or more instances of JD Edwards EnterpriseOne, as this sharing might corrupt specification files.

8. Create an empty log directory under the new path using a command such as this:

   ```
   mkdir -p /u02/JDEdwards/E900/log
   ```

9. In the new JDE.INI file, change all references to the original directory name to the new directory name, including the [INSTALL], [DEBUG], and [BSFN BUILD] sections.

   For example:

   ```
   [DEBUG]
   DebugFile=/u02/JDEdwards/E900/log/ jdedebug.log
   JobFile=/u02/JDEdwards/E900/log/jde.log

   [INSTALL]
   B9=/u02/JDEdwards/E900

   [BSFN BUILD]
   BuildArea=/u02/JDEdwards/E900/packages
   ```

10. Change the new JDE.INI file to reference a port number and starting IPC key that are different from the original JD Edwards EnterpriseOne instance.

    The values are defined by these parameters; but the numbers are only examples:

    ```
    [JDENET]
    serviceNameListen=6009
    serviceNameConnect=6009

    [JDEIPC]
    startIPCKeyValue=9000
    ```

11. From the client workstation JDE.INI file, change the serviceName parameters to match those of the server JDE.INI file.

4

Administering the Windows Server

This chapter contains the following topics:

4.1 Understanding Server Administration for Windows

The JD Edwards company supports Oracle's JD Edwards EnterpriseOne enterprise servers that run the Microsoft Windows Server. You can operate the enterprise server for Microsoft Windows in a logic or database server environment. You need to perform certain administration procedures on the enterprise server to ensure that the software runs properly.

This section discusses:

- JD Edwards EnterpriseOne directory structure for Microsoft Windows.
- JD Edwards EnterpriseOne architecture and process flow for Microsoft Windows.
- JD Edwards EnterpriseOne Initialization for Microsoft Windows.
- JDE.INI settings for starting batch queues on Microsoft Windows.
- Active Directory.

4.1.1 JD Edwards EnterpriseOne Directory Structure for Windows

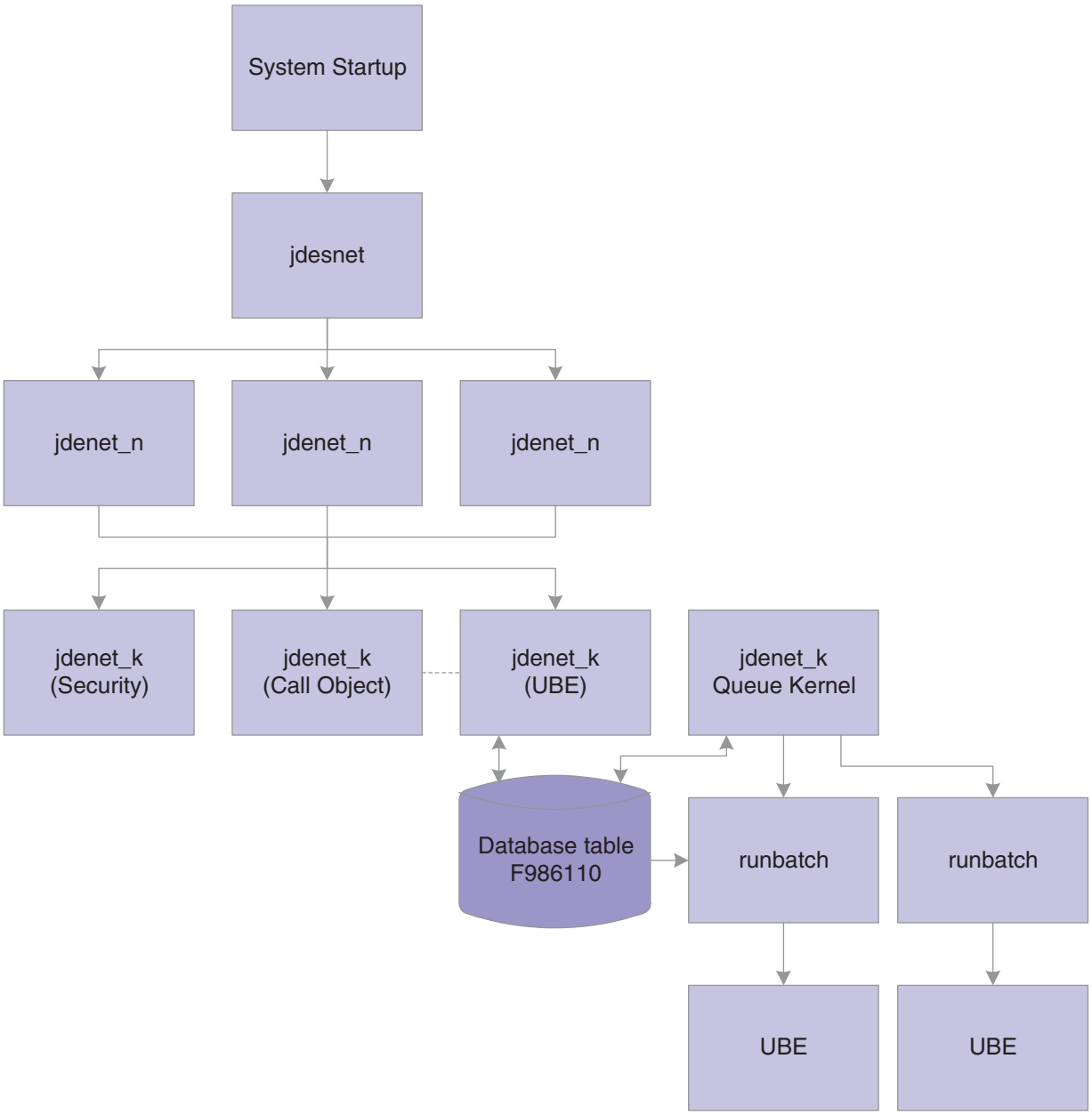
This table lists the directories that are copied to the Windows enterprise server when Oracle's JD Edwards EnterpriseOne is installed. They should be installed under the JD Edwards EnterpriseOne base directory (such as z:\JDEdwards\E900\ddp). Indented names indicate subdirectories of the directories.

Directory	Description
pathcode	The main directory for the business function shared libraries, C header files, object files, source files, and specification (spec or TAM) files. Upon installation, this directory will be copied to the correct path codes, such as PD900 and DV900. These subdirectories are included: ■ bin32, which includes business function shared libraries. ■ spec, which includes specification files. These binary data files are in a JD Edwards proprietary format.
system	The main directory for the system-level executables, shared libraries, C header files, libraries, and localization files. These subdirectories are included: ■ bin32, which includes system-level executables and shared libraries. ■ include, which includes system-level C header files. ■ includev, which includes system-level C header files provided by third-party vendors such as Vertex. ■ lib, which includes system-level shared libraries and export files. ■ libv32, which includes system-level shared libraries provided by third-party vendors.
PrintQueue	The directory to which all .PDF file output for reports is written.
log	The directory to which jde_xxx.log and jdedbug_ xxx.log files are written.
packages	The server package installation base directory. Directories exist here only if a package has been installed. Under the package directory are subdirectories named for each package that has been installed. Located under each package are these subdirectories: ■ bin32, which includes business function shared libraries. ■ include, which includes business function header files. ■ obj, which includes business function object files. These are divided among lower-level subdirectories that correspond to each DLL in the bin32 directory. ■ source, which includes business function source files. These are divided among lower-level subdirectories that correspond to each DLL in the bin32 directory. ■ spec, which includes specification files. These binary data files are in a JD Edwards proprietary format.

4.1.2 JD Edwards EnterpriseOne Architecture and Process Flow for Windows

These host server processes perform the indicated actions:

Figure 4–1 Windows server processes



All communications between the client and the host server occur using sockets. The communications between jdenet_n and jdenet_k occur with shared memory. jdenet_n and queue kernel communicate using the Job Control Status Master database table (F986110).

This text explains the process flow:

- During Windows system startup, jdesnet runs automatically, provided that it is installed to start automatically. Otherwise, it must be started manually.
- This information applies to the JD Edwards network service:
  - The program is system\bin32\jdesnet.exe.
  - Each time that a new server or workstation connects to this server, jdesnet might start another jdenet_n until the number of jdesnet and jdenet_n jobs

equals the value in the maxNetProcesses field in the [JDENET] section of the JDE.INI file.

- Each time that a new request, such as a batch application or CallObj is submitted, jdesnet (and any jdenet_n processes) might start another jdenet_k process until the number of jdenet_k jobs equals value in the maxKernelProcesses field in the [JDENET] section of the JDE.INI file.
- Jdenet_n can be run manually by running system\bin32\jdenet_n.
- This information applies to the JD Edwards queue service:
  - The program is system\bin32\jdesque.exe.
  - The service runs the number of instances of queue kernels specified in the UBEQueues, PackageQueues, and SpecInstallQueues fields in the [NETWORK QUEUE SETTINGS] section of the JDE.INI.
- When a user submits a batch application, jdesnet or jdenet_n (as part of the host server) communicates with the client as follows:
  - The host server programs are system\bin32\jdesnet.exe and system\bin32\jdenet_n.exe.
  - The client environment is initialized.
  - The client tells the host server (using a socket) to initialize its environment.
  - The host server (for example, jdenet_n) initializes its environment and gets environment and user handles.
  - The host server passes the environment and user handles to the client (using a socket).
  - The client launches the batch application and then sends data to the host server (using a socket).
  - If the maximum number of kernel (for example, jdenet_k; the k stands for kernel) processes has not been met, jdesnet or jdenet_n might start a new jdenet_k process.
  - If the maximum number of jdenet_k processes has been met, jdesnet or jdenet_n puts the message in a queue for a jdenet_k process.
  - The client frees the user environment.
  - The client tells the host server (using a socket) to free the user environment for the server.
  - The host server frees its user environment.
  - The client tells the host server (using a socket) to free the environment for the server.
  - The host server frees its environment.
- When the UBE Jdenet_k (the kernel) writes to the database (batch application only), this occurs:
  a. The program is system\bin32\jdenet_k.exe.
  b. Jdenet_k adds a record in the F986110 database table. The record has a status of W (Waiting).
- The Queue Kernel periodically checks the contents of table F986110 and launches a runbatch process.

- When runbatch processes the batch application, this occurs:
  - The program is system\bin32\runbatch.exe.
  - The system changes the status stored in table F986110 to P (Processing).
  - The system starts the batch application.
  - If the batch application completes successfully, it changes the status in table F986110 to D (Done).
  - If the batch application does not complete successfully, it changes the status in table F986110 to E (Error).
- Unlike the many processes that execute when a batch application is submitted, jdenet_k performs the processing when a user submits a CallObject and these actions occur:
  - Cannot start the service name service on the enterprise server.
  - Error 1069: The service did not start due to a logon failure.

4.1.3 JD Edwards EnterpriseOne Initialization for Windows

This initialization occurs when you start JD Edwards EnterpriseOne programs such as queue kernel, runbatch, and so on:

- The environment is passed as a command line argument to the program (such as porttest, queue kernel) or retrieved by jdenet_k from the QEnv key in the [NETWORK QUEUE SETTINGS] section of the JDE.INI file.
- This environment might be translated to a different environment, based on the settings in the [SERVER ENVIRONMENT MAP] section of the JDE.INI file.
- The environment that is used must be a valid entry in the Library ListMaster File table (F0094) and must have a valid corresponding path code in the Environment Detail - OneWorld table (F00941).
- These JDE.INI settings in the [DB SYSTEM SETTINGS] section specify where the JD Edwards EnterpriseOne server startup tables, such as the Data Source Master (F98611) and Object Configuration Master (F986101) tables, are located:
  - Base Datasource
  - Object Owner
  - Server
  - Database
  - Load Library
  - Type
- Using this information, the F986101 table opens in the specified database on the server.
- When an override exists for a given table, BSFN, or the current user, that data source (OMDATP field in the F986101 table) is used for the given object or user and environment. Otherwise, the data source in which OMOBNM=DEFAULT for the given environment is used. Ignore any inactive records (that is, OMSTSO=NA). We strongly recommend that you do not have any default records (OMOBNM=DEFAULT) for batch applications (OMFUNO=UBE). These records might prevent report interconnections (such as one report calling another report) from starting correctly.

- Each unique data source in the F986101 table should correspond to one entry in the F98611 table.
- The corresponding information in the F98611 table must be correct. In particular, the OMDLLNAME field must display the correct DLL for the database to which the data source points.
- For an Oracle database, the OMDATB field from the F98611 table maps to an entry in the tnsnames.ora file. This tnsnames.ora file must be set up correctly (check with an Oracle database administrator).
- For an IBM DB2 for LUW (Linux, UNIX, Windows) database, the OMDATB field from the F98611 table maps to an entry in the ODBC data source. This datasource must be set up correctly (check with a IBM DB2 for LUW (Linux, UNIX, Windows) database administrator).
- For a Microsoft SQL Server, Microsoft Access, or Client Access database, the OMDATB field from the F98611 table maps to a data source specified in the ODBC Data Source Administrator applet in the Windows Control Panel. This data source must be set up correctly. If multiple users plan to sign on to this Windows platform and run JD Edwards EnterpriseOne or PORTTEST, the data sources must be defined on the System DSN tab. Otherwise, User Data Sources can be used.

  If you are using Microsoft Windows 2000 to open the ODBC Data Source Administrator, from the Start menu, select Programs, then Administrative Tools, and then Data Sources (ODBC).
- This information pertains to the setup of SQL Server ODBC drivers, using the ODBC Data Source Administrator applet:
  - The data source name must match the name in the F98611 table.
  - The description can be anything that you want.
  - The server is the name of the database server.
  - The network address includes the database server name, a comma, and a port in which the database user listens.
  - Network Library should be set to Default.
  - Click the Options button for more settings.
  - The database name is usually set to JDE. You can set it to Default.
  - The language name should be set to Default.
  - The Generate Stored Procedure for Prepared Statement option should be turned off.
  - The Use ANSI Quoted Identifiers option should be turned on.
  - The Use ANSI Nulls, Padding and Warnings option should be turned on.
  - The Convert OEM to ANSI characters option should be turned off.
- This information pertains to the setup of Client Access ODBC drivers, using the ODBC Data Source Administrator applet:
  - On the General tab the data source name must match the name in the F98611 table. The system is the name of the database server.
  - On the Server tab, the default libraries should be the IBM i library, and the commit mode should be Commit immediate (*NONE).

- On the Format tab, the naming convention should be System naming convention (*SYS).
- On the Other tab, if the data that you are transferring using this data source contains a Binary Large Object (BLOB), translation should be set to Do not translate CCSID 65535. If the data that you are transferring using this data source does not contain a BLOB, translation should be set to Translate CCSID 65535.

4.1.4 JDE.INI Settings for Starting Batch Queues on Windows

These JDE.INI settings are used to start batch queues on the Windows enterprise server:

```
[NETWORK QUEUE SETTINGS]
UBEQueues=number of batch queues
UBEQueue1=batch queue name
UBEQueue2=batch queue name
PackageQueues=number of package queues
PkgQueue1=package queue name
PkgQueue2=package queue name
SpecInstallQueues=number of spec install queues
SpcQueue1=spec install queue name
QEnv=queue environment
QUser=queue user
QPassword=queue user password
```

This table describes each setting:

Setting	Description
number of batch queues	Identifies the number of batch queues available. If you do not specify a number of batch queues that matches the number specified here, JD Edwards EnterpriseOne uses QBATCH when a missing queue is called.
batch queue name	Identifies the name of the batch queue. For example, for UBEQueue2, you might specify the queue as QBATCH2. You should specify a number of batch queue names that is equal to the value that you specify for the number of batch queues.
number of package queues	Identifies the number of package queues that are available. If you do not specify a number of package queues that matches the number specified here, JD Edwards EnterpriseOne uses QBATCH when a missing queue is called.
package queue name	Identifies the name of the package queue. For example, for PkgQueue2, you might specify the queue as XBATCH2. You should specify a number of package queue names that is equal to the value that you specify for the number of package queues.
number of spec install queues	Identifies the number of specification install queues available. If you do not specify a number of specification install queues that matches the number specified here, JD Edwards EnterpriseOne uses QBATCH when a missing queue is called.

Setting	Description
spec install queue name	Identifies the name of the specification install queue. For example, for PkgQueue2, you might specify the queue as XBATCH2. You should specify a number of specification install queue names equal to the value that you specify for the number of specification install queues.
queue environment	Identifies the JD Edwards EnterpriseOne environment under which the Windows operating system starts the queues.
queue user	Identifies a valid JD Edwards EnterpriseOne user.
queue user password	Identifies the password for the queue user.

4.1.5 Active Directory

Windows Active Directory is Microsoft's implementation of a hierarchical, object-based directory service for managing system resources, including developers, end users, and groups. If you publish JD Edwards EnterpriseOne server information in Active Directory, client workstations use this information to locate and connect to the server dynamically. If JD Edwards EnterpriseOne service changes from one server to another, workstations can still connect to the server by referencing published server information in Active Directory.

Note: Active Directory is a Windows feature, and its use with JD Edwards EnterpriseOne is platform-specific and optional. If you are running JD Edwards EnterpriseOne enterprise servers on Unix or IBM i platforms, client workstations still reference their jde.ini files to connect to the server.

4.1.5.1 SCP Object in Active Directory

JD Edwards EnterpriseOne NT service installation creates a Service Connection Point (SCP) object in Active Directory. The SCP object specifies the server name and port number.

Starting JD Edwards EnterpriseOne service on a server automatically updates the SCP object with the server name and port number, and establishes the SCP object status as Running. When service stops, the status of the SCP object automatically changes to Stopped.

Note: JD Edwards EnterpriseOne Windows service installation creates the SCP object in Active Directory only if you have added an [Active Directory] section to the jde.ini file on the server before installation.

When a user signs on to JD Edwards EnterpriseOne, JD Edwards EnterpriseOne searches Active Directory for an SCP object with a service name that matches the parameter value in the [Active Directory] section of the workstation jde.ini file. JD Edwards EnterpriseOne selects an SCP object that has a status of Running and retrieves the server name and port number, which enables the workstation to make a connection to the server.

4.1.5.2 Additions to the Server JDE.INI file

For each server that you publish in Active Directory, you must add an [Active Directory] section in the JDE.INI file on the server. In the [Active Directory] section, you include the SCPToPublish entry, which identifies the SCP object in the Active Directory.

The value of the SCPToPublish parameter should be unique for each object, and you should consistently adhere to a naming convention for ease of administration. For example, the value of each SCPToPublish parameter might represent a version of JD Edwards EnterpriseOne.

This is a sample entry in the [Active Directory] section of the server JDE.INI file.

```
SCPToPublish     JDEDWARDS_ENTERPRISEONE_900_SP1
```

If you move JD Edwards EnterpriseOne service from one server to another or change the service port number, no changes to the workstation JDE.INI file are needed, so long as the name of the SCP object in Active Directory and the parameter values of the [Active Directory] section of the workstation JDE.INI file match.

> **Note:** Although users can automatically connect to a new server when a change in service is made, batch processes and business functions are not automatically mapped to the new server. Therefore, you typically need to change OCM mappings for the users so that they use the new data source.

4.1.5.3 Additions to the Workstation JDE.INI File

You also add an [Active Directory] section to the workstation JDE.INI file that specifies the name of the SCP object that contains port number and server name information.

These parameters are included in the [ActiveDirectory] section of the workstation JDE.INI file:

- JdenetSCP (the connection port).
- SecurityServerSCP (the security server).
- LockManagerSCP (the Lock Manager).
- UnifiedLogonServerSCP (unified logon server).

For each of these parameters, you assign as the value the name of the SCP object in the Active Directory file. For example, enter JDEDWARDS_ ENTERPRISEONE_900_SP1.

This table presents an example of the parameters that you add to the [Active Directory] section of the workstation JDE.INI file. The value of each parameter is the SCP object name in Active Directory.

Parameter of [Active Directory] Section of Workstation JDE.INI File	Meaning	Parameter Value: name of SCP Object in Active Directory
JdenetSCP	Connection port	JDEDWARDS_ ENTERPRISEONE_900_SP1
SecurityServerSCP	Security server	JDEDWARDS_ ENTERPRISEONE_900_SP1
LockManagerSCP	Lock manager	JDEDWARDS_ ENTERPRISEONE_900_SP1

Parameter of [Active Directory] Section of Workstation JDE.INI File	Meaning	Parameter Value: name of SCP Object in Active Directory
UnifiedLogonServerSCP	Unified logon server	JDEDWARDS_ ENTERPRISEONE_900_SP1

4.2 Setting Up a Printer for Windows

This section provides an overview of Printer Setup for Windows and Windows Services, Accounts, and Permissions and discusses how to:

- Add a printer.
- Determine or change printer ownership.
- Set up user accounts on an enterprise server.
- Change the domain.
- Add a local account.
- Add a user to the administrators group.

4.2.1 Understanding Printer Setup for Windows

Setting up a printer for a Microsoft Windows enterprise server involves setting up accounts under which JD Edwards EnterpriseOne runs, establishing printer ownership, and defining the printer. The default printer used for printing reports will be the system default printer.

4.2.2 Understanding Windows Services, Accounts, and Permissions

Before you can successfully set up a printer for Windows, you should understand the relationship of JD Edwards EnterpriseOne to Windows services, accounts, and permissions, which involves these:

- Assigning permissions to the accounts under which JD Edwards EnterpriseOne services run.
- Making printers accessible from the service programs.
- Assigning ownership for accounts to enable access to printers.

Every Windows printer is associated with one network account called the printer's owner. When JD Edwards EnterpriseOne runs a batch report, service programs must be able to access a printer. You can define this printer to be locally accessible only by the enterprise server or remotely accessible by other network resources (for example, it might be attached to a print server). You can specify a printer that is connected directly to an enterprise server as a local or network printer, depending on how you added the printer from the Control Panel.

When you create a Windows user account, you must associate that account with one of these two domains:

- Local. This domain is associated with a particular Windows machine. For example, each Windows machine has a local administrator account. Local accounts cannot access network resources, such as network printers. Any account names that do not begin with a domain name are considered to belong to the local domain.
- Network. This domain is spread across a Windows network. Users in the network domain can access network resources, such as printers and disk drives, on other

servers. Account names that are assigned to the network domain must begin with a domain name, such as domain1\john_doe.

In this table, you must define two types of service accounts and printer ownerships for the two types of printers:

Printer Type	Account and Owner
Local	The service account type can be local or network. The printer owner account can be local or network.
Network	The service account type must be network. The printer owner account must be network.

Windows services enable programs to run on a Windows platform even when no user is signed on to the machine. For the JD Edwards EnterpriseOne enterprise server, you must run these two service programs:

- Network: This program provides the network connection between the JD Edwards EnterpriseOne workstation and the JD Edwards EnterpriseOne enterprise server.
- Queue: This program starts jobs (either batch reports or server package installations) on the enterprise server.

The accounts under which Windows services run must have permissions to start and stop services on the local machine. You must specify permissions for one of these:

- Individual users, such as administrator and guest accounts.
- Groups of users, such as administrators (note the plural; administrators are different than an individual administrator).

The accounts that automatically have permissions to start and stop services include:

- The Administrator user.
- Users specifically designated by the Administrator user.
- Users who belong to the Administrators group (which is different from an individual administrator).
- Users that belong to the Power Users group.

Note: We strongly recommend that you use an account for a user who belongs to the local Administrators group.

You must add a printer in Microsoft Windows before you can use it in JD Edwards EnterpriseOne.

4.2.3 Adding a Printer

To add a printer:

1. Click the Windows Start button.
2. Select Settings, and then select Printers.
3. Select Add Printer.
4. On Add Printer Wizard, follow the system-guided steps.

For a local printer, these steps include selecting the port to which the printer is attached, specifying the type of printer that you are installing, specifying a name for the printer, and indicating where the drivers are located, if needed.

For a network printer, these steps involve selecting a print server and printer and indicating whether the printer is the default printer for the enterprise server.

Note: When you are defining a printer, do not use a space character in the name. If you do, JD Edwards EnterpriseOne will not be able to correctly read or access the physical printer.

4.2.4 Determining or Changing Printer Ownership

To determine or change printer ownership:

1. From Control Panel, select Printers.
2. Select a printer, right-click, and select Properties.
3. Click the Security tab.
4. Click the Ownership button.

   The Owner dialog box displays the current owner of the printer.
5. On owner, to make the account that you are currently signed onto the owner of the printer, select Take Ownership, and then click OK.

4.2.5 Setting Up User Accounts on an Enterprise Server

You can set up local users to add local and network accounts to groups.

To set up user accounts on an enterprise server:

1. On the enterprise server, under Windows, select Start,Settings,Control Panel,Administrative Tools, then Computer Management.
2. On the Tree tab, select Local Users and Groups, and then click the Users folder.

4.2.6 Changing the Domain

To change the domain:

1. From the main menu of User Manager, select User.
2. Select User Domain.

   The Select Domain form displays all domains. The local domain is named the same as the enterprise server and does not appear in the list. However, you can still type the name of the enterprise server in the Domain field.

   In this example, the name of the local machine is the same as the domain: DEVS5. That name is appears in the title bar as \\DEVS5. Although that syntax might typically indicate a network machine, in this case it represents a local machine name because the name of the machine and the domain are the same.
3. Click OK.

   The User Manager form displays all of the accounts for the domain that you chose. If you select a network domain, all listed names represent network accounts. Likewise, if you select the local domain, all listed names represent local accounts.

4.2.7 Adding a Local Account

If you are using a local printer, you can use either a local or network account to run the JD Edwards EnterpriseOne services.

1. Sign onto Windows as a user with administrative privileges in the local domain.
2. From Computer Management, select System Tools, and then select Local User and Groups.
3. From the Action menu, select New User.
4. On New User, complete these fields:
   - User name
   - Full Name
   - Description
   - Password
   - Confirm Password
5. Complete these options, as appropriate for the installation:
   - User must change password at next logon.
   - User cannot change password.
   - Password never expires.
   - Account disabled.
6. Click Create.
7. Click Cancel.

4.2.8 Adding a User to the Administrators Group

To add an existing account (either local or network), you must use the local domain.

1. From the User Manager main window, double-click the Administrators group.

   The user Administrator belongs to the Administrators group. Local accounts are not preceded by a domain name, and network accounts are preceded by a domain name. For example, the domain member with a name JDE is a local account, and a member with the name JDEMD1\AY5600427 is a network account.
2. On Administrators Properties, click Add.

   A list displays all users in the selected domain.
3. On Select Users or Groups, select the domain of the user whom you want to add to the Administrators group.
4. Select the user whom you want to add to the Administrators group.
5. Click Add to add the user to the group, and then click OK.

4.3 Working with Network Services

This section provides an overview of network services and discusses how to:

- Set up the network service.
- Start the network service.

- Stop the network services.
- Clean up the enterpriser server for Windows.
- Uninstall the network service.
- Start the enterprise server for Windows manually.
- Verify the JD Edwards EnterpriseOne installation.

4.3.1 Understanding Network Services

JD Edwards EnterpriseOne uses the Network service on the enterprise server. This service is installed during the installation process using the jdesnet -i service from the system\bin32 directory.

When you install this service, the system adds these entries to the Windows registry:

- The name of the service that appears on the Services form (used when controlling the services).
- The location of the JD Edwards EnterpriseOne executable files.

During a new installation, or after you have renamed or moved the directory tree for an existing installation, you should reinstall the services.

After the initial installation, you will need to reinstall the Network service only when it has been uninstalled. You will need to uninstall this service only when the JD Edwards EnterpriseOne directory tree is renamed, moved, or deleted. The process to uninstall this service removes these entries from the Windows registry:

- The names that appear for the service on the Services form.
- The location of the JD Edwards EnterpriseOne executable files.

After the Network service is installed, you must set up the service under a network account, if you are using a network printer, or a local account, if you are using a local printer. If you are using a network account, it must be in either the Administrators or Power Users group.

Note: We strongly recommend that you use a user who belongs to the local Administrators group.

After you have installed and set up the Network service, you must start the service before JD Edwards EnterpriseOne can use it. Later, if you need to stop services, you must do so in the proper order.

After JD Edwards EnterpriseOne is shut down, you can determine whether any processes completed abnormally. If so, you need to clean up the enterprise server. Unforeseen circumstances can cause JD Edwards EnterpriseOne processes to terminate abnormally. Processes that terminate abnormally are called runaway processes. After shutting down JD Edwards EnterpriseOne, look for any runaway processes and, if any exist, manually terminate them.

4.3.2 Setting Up the Network Service

To set up the network service:

1. From the Start menu, select Programs, Administrative Tools, and then Services.
2. Select the JD Edwards EnterpriseOne Network service.

The name of the service is in the form JDE release Network, where release is the current JD Edwards EnterpriseOne release. For example, the Network services name for Release E900 is JDE 900 Network.

3. Click Action, then click Properties.
4. On the General tab, if you want JD Edwards EnterpriseOne to start automatically when the enterprise server boots, click the Automatic option under Startup Type.
5. On the Log On tab, click the This Account option.
6. Enter the account name under which the JD Edwards EnterpriseOne Network service will run.
7. Enter the password for the account and a confirmation of the password.
8. Click OK.

4.3.3 Starting the Network Service

To start the Network service:

1. From the Services window, select the JD Edwards EnterpriseOne Network service.

   The name of the service is in the form JDE release Network, where release is the current JD Edwards EnterpriseOne release. For example, the Network services name for E900 is JDE 900 Network.
2. From the Action menu, click Start.
3. Use the Windows Task Manager to ensure that these processes are running:
   - jdesnet.exe.
   - jdenet_k.exe processes. (None, one, or more might exist.)

4.3.4 Stopping the Network Services

When you stop the Network service, follow the steps in the proper sequence.

To stop the Network service:

1. From the Services window, select the Network service.

   The name of the JD Edwards EnterpriseOne Network service is in the form JDE release Network. For example, the Network services name for JD Edwards EnterpriseOne 8.10 is JDE 812 Network.
2. Use the Windows Task Manager to ensure that all JD Edwards EnterpriseOne processes are terminated.

   This might take several minutes. These processes should be terminated and, therefore, should not appear in the list of processes in Task Manager:
   - jdesnet.exe
   - jdenet_n.exe
   - jdenet_k.exe
   - runbatch.exe
   - ipcsrv.exe

4.3.5 Cleaning Up the Enterprise Server for Windows

To clean up the enterprise server for Windows:

1. In the Processes tab of Task Manager, search for any JD Edwards EnterpriseOne Host Server processes, such as jdesnet, jdenet_n, jdenet_k, and runbatch.

   Wait until all the JD Edwards EnterpriseOne Host Server processes are terminated. If all processes terminate, you do not need to perform the remaining steps in this task. Otherwise, continue with the next step.

2. Select a process in Task Manager.
3. Click End Process.
4. If the runaway process does not terminate, continue with the next step.
5. In Task Manager, right-click the process and select debug.
6. When the Visual C++ main window appears, select the Stop debugging option from the Debug menu.
7. Exit from Visual C++, and then repeat these steps for each runaway process.
8. If none of the previous steps stops the runaway process, reboot the enterprise server.

4.3.6 Uninstalling the Network Service

To uninstall the Network services:

Run this program from the \system\bin32 directory:

```
jdesnet -u
```

4.3.7 Starting the Enterprise Server for Windows Manually

If JD Edwards EnterpriseOne does not run through the Control Panel Services applet, you can run Network manually.

Note: If you start JD Edwards EnterpriseOne manually, you must stop the JD Edwards EnterpriseOne processes using the Windows Task Manager.

To start the enterprise server for Windows manually:

1. On the enterprise server for Windows, sign on with administrator privileges.

   If you used the user ID that we recommend, the value is **PSFT.**

2. On the Windows toolbar, from the Start menu, select Run, and then enter these commands:

   ```
   drive: installpath\system\bin32\jdenet_n
   ```

   Where installpath is the path to the JD Edwards EnterpriseOne installation.

   This command launches an executable program that starts the JD Edwards EnterpriseOne network (JDENet) internal processes.

   If you run jdenet_n from a command prompt, ensure that the working directory is the subdirectory \system\bin32.

4.3.8 Verifying the JD Edwards EnterpriseOne Installation

You can verify the JD Edwards EnterpriseOne installation with the PORTTEST program.

Note: When you run PORTTEST, make sure that one of this is true:

If the network service, such as jdesnet.exe, is running, make sure that you are signed on to Windows under the same user account as the net service is running. You can then run PORTTEST from a command prompt.

If the network process, such as jdenet_n.exe, is run from the command prompt, you can run PORTTEST from the command prompt.

To verify the JD Edwards EnterpriseOne installation:

In the command line, enter these commands:

```
cd \JDEdwards\E900\ddp\system\bin32

porttest <userid> <password> <environment>
```

The program initializes an environment, initializes a user, opens the Account Balances table (F0902), and displays up to 99 rows of data. The number of rows of data that the program displays depends on the data in the table. If you run the program before anyone enters data into the table, you will not see any data on the screen. In this case, the lack of data does not indicate an error. Review the messages on the form and the corresponding jde.log file to determine the results of the program.

4.4 Administering Batch Processes for Windows

This section provides an overview of batch process administration for Windows and discusses how to:

- Monitor batch processes.
- Review batch output files.
- Run reports from the command line for Windows.
- Schedule reports from the command line for Windows.

4.4.1 Understanding Batch Process Administration for Windows

Administering batch processes involves knowing the processes that run when JD Edwards EnterpriseOne starts, where files are placed before and after printing, and how to watch those processes.

The user who started the JD Edwards EnterpriseOne software owns the processes that are running for JD Edwards EnterpriseOne; Windows Task Manager cannot track this information. When the software starts, a number of processes start and run under the environment and security of the user who started them. These processes are as follows:

Process	Description
jdesnet.exe	The network listener that listens for connection requests.
jdenet_n.exe	A network listener that listens for connection requests. Depending on the jde.ini setting, zero, one, or more of these processes can run simultaneously.
jdenet_k.exe	The job responsible for coordination between the net and queues. It is not started until the first batch job is submitted to the server.
runbatch.exe	The job responsible for executing the submitted reports.
ipcsrv.exe	The process responsible for passing Binary Large Objects (BLOBs) between other processes.

4.4.2 Monitoring Batch Processes

You can use the Task Manager to continuously monitor the performance of each job, the amount of CPU time it is consuming, and the amount of memory it is using. By default, the display refreshes every second.

4.4.3 Reviewing Batch Output Files

All output from each report, regardless of whether it is a preview, is placed in the PrintQueue directory under the JD Edwards EnterpriseOne installation directory before it is printed. Depending on the JDE.INI settings of the workstation that submitted the job, the job might or might not be deleted after being printed. Unless the submitter identified a printer, jobs are printed to the default printer that you specified for the enterprise server.

Two settings, based upon the workstation's JDE.INI file, tell the server whether to print the report immediately upon completion and whether to save the output from the report or delete it. Here are examples of both of these workstation settings:

```
[NETWORK QUEUE SETTINGS]
SaveOutput=TRUE
PrintImmediate=TRUE
```

Setting SaveOutput to TRUE causes the enterprise server to hold the jobs within the PrintQueue directory until the user explicitly deletes them. Setting PrintImmediate to TRUE tells the enterprise server to print the job immediately after completion of the report.

Users should be strongly encouraged to use the SaveOutput=FALSE entry in their JDE.INI file. When users decide to save their output, they should periodically delete the entries through JD Edwards EnterpriseOne. Deleting the output files from the operating system will not delete the corresponding JD Edwards EnterpriseOne print job entries (for example, entries might still exist in the database). These print job entries still have to be deleted manually.

To list all files in the PrintQueue directory, use Windows Explorer to change the working directory to the PrintQueue directory.

These file names are the actual reports that were generated when the job was executed. The file names follow these conventions:

Segment	Description
S_	Identifies the first part of a file name. Indicates that a specification installation was performed by the workstation. The system omits this prefix when no specification installation was performed.
R0006P	Identifies the report name.
XJDE0001	Identifies the report version.
UBE	Identifies the type of request.
216	Identifies the request number assigned by JD Edwards EnterpriseOne.
PS	Indicates a PostScript file.
PDF	Indicates a PDF (Portable Document Format) file. This file can be viewed on the workstation using Adobe Acrobat.

4.4.4 Running Reports from the Command Line for Windows

If you are a user with the proper authority and path (equal to that described in the installation instructions), you can run batch report processes from the server command line by first changing to the JD Edwards EnterpriseOne system directory (system\bin32) and then entering these commands:

```
runube  <[-p|-P] [-f|-F|-d|-D passfile] [user password]>
   <Environment>
   <Role>
   <ReportName>
   <VersionName>
   <JobQueue>
   <"Interactive"|"Batch">
   <"Print"|"Hold">
   <"Save"|"Delete">
    [Printer]
```

The format for the passfile parameter is:

```
[enterpriseoneusername
 password]
```

Note: The [user password] parameter has been deprecated.

For the command parameters, only the first character of the parameter name is required. The vertical bar symbol (|) indicates that you must specify one of the parameters on either side of the vertical bar. The brackets indicate an optional parameter. These options apply to the runube command:

Parameter	Description
–p	Prompts for user/password information.
–P	Prompts for user/password information.
–f	Reads user/password information from the plain text file that is specified in passfile (FilePath).
–F	Reads user/password information from the plain text file that is specified in passfile (FilePath).

Parameter	Description
–d	Reads user/password information from the plain text file and indicates the automatic removal of the file after the job has read the credentials from it.
–D	Reads user/password information from the plain text file and indicates the automatic removal of the file after the job has read the credentials from it.
Interactive	The system holds the current terminal session until the entire report is processed.
Batch	The runube command the job to a UBE kernel, which in turn will send the job submission to Queue kernel, and then returns control of the terminal to the user.
Print	After the batch process completes generating its output, the output is printed to the specified printer queue. If you do not specify a printer on the runube command line, the system uses the EnterpriseOne user's default printer specified in the Printers program (P98616).
Hold	The batch output is not printed immediately after the job completes, but may be printed later from 'Work with Submitted jobs'
Save	The system retains the report's output and all records of its execution.
Delete	The system removes any output from the PrintQueue directory, from Report definition output, and also removes the records of the jobs execution from F986110 after the report prints.
OutQ	Optional. This is the printer name on which the given report is printed. If this option is not specified, the report will be printed on the enterprise server default printer.

4.4.4.1 Example: Running Reports from the Command Line for Windows

This example lists commands for executing a batch process report:

```
cd \JDEdwards\E900\ddp\system\bin32

runube KL5595218 KL5595218 PROD R0006P XJDE0001 QBATCH Interactive
Print Delete printer_1
```

4.4.5 Scheduling Reports from the Command Line for Windows

You can schedule a report from the command line for processing on a future date, daily, or even on a recurring day of the week. To schedule one-time only reports, use the at command.

When you issue jobs with the at command, they run in the background. However, the at command enables you to schedule a future time of execution. You can use this command to run a batch job during off-peak hours.

> **Note:** Use of the at command depends on how security is configured on the Windows enterprise server. You should limit the amount of access that users have to submit jobs on the server. If possible, only an administrator should do this type of scheduling.

The command format for the at command is as follows:

```
at [\\computername\ time [/INTERACTIVE] [/EVERY:date[,...] |
/NEXT:date[,...]] command
```

Where these options apply:

Parameter	Description
\\computername	Identifies the computer on which to run the program. If you do not specify a value, the default is the local machine.
time	Specifies the time to run the job, such as 08:00.
/Windows INTERACTIVE	enables the program to interact with the Windows operating system desktop.
/EVERY:date	Specifies the days on which to run the job. Values are M, T, W, Th, F, S, and Su.
/NEXT:date	Specifies the next date for the first execution. If you do not specify a value, the default value is today's date.
command	Specifies the command to run. To run batch jobs here, use the runube command with any of its parameters.

4.4.5.1 Example: Scheduling Reports from the Command Line for Windows

This example lists a sample command that you can use to schedule a JD Edwards EnterpriseOne batch report to run on the DEPLOY machine at 06:00 every Sunday:

```
at \\DEPLOY 06:00 /EVERY:Su z:\b731\system\bin32\runube KL5595218 KL5595218
PROD R0006P XJDE0001 QBATCH Interactive Print Delete printer_1
```

4.5 Maintaining File Security for Windows

You should be aware of the security that is set up for the files on a JD Edwards EnterpriseOne enterprise server. System-wide, only these two accounts will ever need operating system access to the JD Edwards EnterpriseOne environment files and version executables:

- The account that starts and stops JD Edwards EnterpriseOne.
- The account that builds the environment specification (SPEC) and business function (BSFN) files (if this account is separate from the startup and shutdown account).

> **Note:** The Server Manager managed home agent service must operate using the same account that is used to start and stop JD Edwards EnterpriseOne.

4.5.1 Specification File Security

Specification files are the first part of the environment files. You access these files using the JD Edwards EnterpriseOne kernel processes. These files should never be accessed directly by an operating system user; therefore, security for these files should be read/write for the user and group. These files are not executables, so you do not need to set the executable option for any user, group, or other.

4.5.2 Business Function File Security

You should keep business functions secure. In an environment in which development takes place, you must have a strict form of version control on source and object files. If the business function files change without the knowledge of the JD Edwards EnterpriseOne administrators, rebuilding them might produce unknown or undesired results. Most likely, a developer is working to correct a problem, but the problem could become worse.

You should set a high level of security on the source, include, and object files.

4.5.3 JD Edwards EnterpriseOne Executables Security

You should prevent access to JD Edwards EnterpriseOne executable files to prevent other users from attempting to start up JD Edwards EnterpriseOne. Running the same version of JD Edwards EnterpriseOne on the same system, using the same JDE.INI settings, can cause unpredictable results. In most cases, the second startup will fail, but giving users access to the shutdown procedures enables them to shut down JD Edwards EnterpriseOne.

4.5.4 JDE.INI File (Enterprise Server) Security

Warning: Implementing JDE.INI file security will prevent Server Manager from modifying configuration settings.

You must keep the JDE.INI file on the Windows enterprise server as secure as possible. This file contains a database user name and password that enables JD Edwards EnterpriseOne security to function. This database account is given read authority to the OneWorld Security table (F98OWSEC), which controls JD Edwards EnterpriseOne access.

Note: The F98OWSEC table contains privileged database user names and passwords, which could give a user the ability to manipulate any data in the database, regardless of its sensitivity or security. Therefore, access to the enterprise server JDE.INI file should be minimized.

Denying written access to JD Edwards EnterpriseOne is not necessary, but prevents accidental modification of JDE.INI settings that could adversely affect the operation of JD Edwards EnterpriseOne.

Because of the importance of limiting access to the JDE.INI file for security reasons, you also should limit access to the JD Edwards EnterpriseOne account (or the user account that starts and stops JD Edwards EnterpriseOne). Users with access to this account can easily obtain the F98OWSEC user names and passwords, and gain privileged access to the database.

4.6 Running Multiple Instances of JD Edwards EnterpriseOne on Windows

This section provides an overview of running multiple instances of JD Edwards EnterpriseOne on Windows and discusses how to:

- Run Multiple Instances of JD Edwards EnterpriseOne on Windows.
- Generate a unique identifier.
- Modify the server jde.ini files.

- Modify the workstation jde.ini file.
- Uninstall JD Edwards EnterpriseOne services.
- Move or change a JD Edwards EnterpriseOne directory tree.

Server Manager fully supports multiple foundations. This includes the installation and management of multiple instances of JD Edwards EnterpriseOne on a single server.

See Also:

- 8.98 Server Manager Guide on My Oracle Support, sections: Register/Install on JD Edwards Enterprise Server for EnterpriseOne..

4.6.1 Prerequisites

Before you complete the tasks in this section:

- Verify that you have enough disk space to create copies of the current JD Edwards EnterpriseOne system directory and at least one path code directory.
- Verify that you install each new instance of JD Edwards EnterpriseOne in a separate directory tree and that the version-level directories are different. For example, JD Edwards EnterpriseOne version 1 might be installed in the z:\JDEdwards\b9 directory tree, while JD Edwards EnterpriseOne version 2 might be installed in the z:\JDEdwards\E900 directory tree.

4.6.2 Running Multiple Instances of JD Edwards EnterpriseOne on Windows

You can run multiple instances of JD Edwards EnterpriseOne on a Windows 2000 server. You might do so to test a new service or to upgrade to a new version of JD Edwards EnterpriseOne. You do not need to install a separate machine to run multiple instances of JD Edwards EnterpriseOne, so long as you follow a series of recommended steps.

Each instance of JD Edwards EnterpriseOne must have a unique identifier. You set the value of this identifier in the CLSID parameter of the server JDE.INI file. To generate the identifier, you run the uuidgen program.

For each new instance of JD Edwards EnterpriseOne, you modify the values of parameters in the JDE.INI file on the server. Each value for each JD Edwards EnterpriseOne instance must be unique. This table presents the server jde.ini file parameters that require modification, the purpose of each, and example values for each:

Section of server JDE.INI file	Parameter	Purpose	Example Value
[DEBUG]	DebugFile=	Name of the log file that contains debugging data.	z:\JDEdwards\E900_2\log\jdedebug.log
[DEBUG]	JobFile=	Name of the log file that contains log data.	z:\JDEdwards\E900_2\log\jde.log
[INSTALL]	StartServicePrefix=	Prefix that is used for names of the JD Edwards EnterpriseOne network and queue services.	Instance 2

Section of server JDE.INI file	Parameter	Purpose	Example Value
[INSTALL]	B9=	Base directory of the JD Edwards EnterpriseOne installation.	z:\JDEdwards\E900_2
[JDEIPC]	StartIPCKeyValue=	Integer that indicates an arbitrary starting point in memory for interprocess communications. For multiple instances of JD Edwards EnterpriseOne, differences between the values of the parameter must be at least 1000.	6000
[JDEIPC]	CLSID=	Unique string generated by the NT guidgen program. The string identifies each instance of JD Edwards EnterpriseOne.	1E0CF350-AF81-11D0-BD7B-0000F6540786
[JDENET]	serviceNameListen=	The TCP/IP port number used by the server to receive communication packets from workstations.	6005
[JDENET]	serviceNameConnect=	The TCP/IP port number used by the server to send communications packets to servers.	6005

You are not required to install network and queue services for an existing JD Edwards EnterpriseOne instance unless you change the location of the system\bin32 directory for the new instance. For example, you might decide to put the directory on a new disk.

To move or rename a directory for EnterpriseOne instance after you install its services, you must uninstall the network service and uninstall the IPC Automation Server (ipcserv.exe). You can then move or rename the JD Edwards EnterpriseOne directory and reinstall the network service. The IPC Automation Server automatically reinstalls itself when it is first used.

After you have installed services for each JD Edwards EnterpriseOne installation, you must modify the workstation JDE.INI file so that the values of these parameters match those that you set up in the server JDE.INI file:

- serviceNameListen=
- serviceNameConnect=

See Also:

- Setting Up the Network Service.

4.6.3 Generating a Unique Identifier

To generate a unique identifier:

1. From the Start menu on the Windows taskbar, select Run, and then enter this command:

   ```
   uuidgen-oFILENAME
   ```

   Where FILENAME is the name of the file that will contain the new identifier.

   Note: For help about the options for the uuidgen program, run the uuidgent-? command:

   The uuidgen program creates a unique identifier and stores it in the file that you specified.

2. Copy the identifier.
3. Open the server JDE.INI file and paste the identifier into the CSLID parameter under the [JDEIPC] section of the file.

4.6.4 Modifying the Server JDE.INI Files

To modify the server JDE.INI file:

1. In the system\bin32 subdirectory for each new JD Edwards EnterpriseOne instance, open the server JDE.INI file.
2. In the [DEBUG] section of the JDE.INI file, in the DebugFile= parameter, type the name of the log file that will contain debugging information.
3. In the [DEBUG] section, in the JobFile= parameter, type the name of the file that will contain log information.
4. In the [INSTALL] section, in the StartServicePrefix= parameter, type the value to be used for the names of the JD Edwards EnterpriseOne network and queue services. The names are listed in the Services window under Control Panel.

   The default value is JDE followed by the current version number, such as 900. The default value produces the service names JDE 900 Network and JDE 900 Queue.
5. In the [INSTALL] section, in the B9= parameter, type the name of the base directory of the JD Edwards EnterpriseOne installation. The JD Edwards EnterpriseOne server uses this value to determine the location of the executables and DLLs used to run JD Edwards EnterpriseOne programs.
6. In the [JDEIPC] section of the JDE.INI file, modify the values of these parameters:

Parameter	Value
StartIPCKeyValue	Type a number for the starting point in memory for interprocess communications. For multiple instances of JD Edwards EnterpriseOne, verify that the difference between starting point values for each instance is at least 1000. The default value is 5000. **Note:** To ensure that the difference between starting point values is at least 1000, review the maxNumberofResources parameter in the [JDEIPC] section of the JDE.INI file. If the parameter value is less than 1000, change the value.
CLSID=	Type the unique string that is generated by the NT guidgen program.

7. In the [JDENET] section of the JDE.INI file, modify the values of these parameters:

Parameter	Value
serviceNameListen=	Type the port number for the TCP/IP port used by the server to receive communications packets from the workstations. Each instance of JD Edwards EnterpriseOne must communicate with workstations through a different port. The default value is jde_server.
serviceNameConnect=	Type the port number for the TCP/IP port used by the server to send communications packets to the workstations. Each instance of JD Edwards EnterpriseOne must communicate with workstations through a different port. The default value is jde_server.

4.6.5 Modifying the Workstation JDE.INI File

To modify the workstation JDE.INI file:

1. In the Windows directory on the workstation, locate and open the jde.ini file.

   Examples of the windows directory include c:\winnt and c:\windows.

2. Modify the values of these parameters to match the values in the server jde.ini file:
   - serviceNameListen=
   - serviceNameConnect=

4.6.6 Uninstalling JD Edwards EnterpriseOne Services

To delete an instance of JD Edwards EnterpriseOne after you install its services, you must uninstall the services for that instance before you delete the JD Edwards EnterpriseOne directory tree.

To uninstall JD Edwards EnterpriseOne services:

1. From a command line prompt, change directories to the system\bin32 directory of the JD Edwards EnterpriseOne instance.

2. For example, enter this command:

   ```
   C:\> d:\E900\system\bin32
   ```

3. To uninstall network services, enter this command:

   ```
   jdesnet -u
   ```

   This command removes some settings in the Windows registry that were created when you installed JD Edwards EnterpriseOne services.

4.6.7 Moving or Changing a JD Edwards EnterpriseOne Directory Tree

To move or change a JD Edwards EnterpriseOne directory tree:

1. From a command line prompt, change directories to the system\bin32 directory of the JD Edwards EnterpriseOne instance.

   For example, enter this command:

   ```
   C: \> d:\E900\system\bin32
   ```

2. To uninstall network services, enter this command:

   ```
   jdesnet -u
   ```

   Note: You do not need to reregister ipcsrv.exe in the new directory because the executable is automatically registered when a binary large object is first transferred using interprocess communications.

3. Move or change the directory tree.
4. Reinstall JD Edwards EnterpriseOne Services.

5

Monitoring System Web Services Gateway (WSG) Servers from the Web

This chapter contains the following topics:

- Section 5.1, "Understanding Monitoring System Web Services Gateway (WSG) Servers from the Web"
- Section 5.2, "Monitoring System Web Services Gateway (WSG) Servers from the Web"

5.1 Understanding Monitoring System Web Services Gateway (WSG) Servers from the Web

The System Web Services Gateway Server Monitor enables you to use the web to monitor a WSG Broker Server. The server-monitoring APIs for ActiveWorks are available in Java code, and JD Edwards EnterpriseOne uses them to provide the WSG Server Monitor.

5.2 Monitoring System Web Services Gateway (WSG) Servers from the Web

To monitor an WSG server, you must enter a valid host name and port number in the WSG Server Monitor workspace. From the dialog box in the workspace, you can select from five views:

- Broker Summary
- Event Types
- Client Groups
- Client States
- Broker Logs

5.2.1 Broker Summary View

The Broker Summary view enables you to monitor information about one or more brokers running on the Broker Server. The broker provides essential system services, such as receiving, sending, and queuing events. Events are messages sent to and received by resources in the system, including client workstations and other servers.

To view broker summary information, enter a host name and port number in the WSG Server Monitor workspace, and then select Broker Summary from the dialog box and click the Monitor button.

The Broker Summary view appears.

The Broker Summary view contains a table with various parameters, the values of which provide information about one or more brokers running on the server. This table identifies the parameters and offers a brief explanation of each one:

Parameter in Table of Broker Summary View	Parameter Meaning
Number	The number of the broker.
Broker Name	The name of the broker. The default name is Broker # 1.
Broker Host	The Broker Server on which the broker is running.
Territory	A set of brokers that share information about event types and client groups.
Description	A full description of the broker, provided when the broker is installed on the server.
Event Types	Types of messages received and sent by the broker.
Client States	Information about a client maintained by the broker. Clients connect to brokers.
Client Groups	A list of all client groups on the broker. A client group is a set of properties shared by broker clients.

The Event Types, Client States, and Client Groups parameter columns contain magnifying glass buttons you can click to view more detailed information about each one. Each of these parameters has its own view, which you can access from the dialog box in the Broker Summary view or in the WSG Server Monitor workspace.

The Broker Summary view also displays:

- Number of non-SSL (Secure Sockets Layer) connections.
- Highest number of non-SSL connections.
- Number of SSL connections.
- Highest number of SSL connections.
- Server disk space, in megabytes.

You can view details about the server on which the broker is running by clicking the Server Statistics URL.

5.2.2 Event Types View

An event type defines the properties of an event, including the data fields that the event carries, the event's unique name, and a storage type, which defines how the broker stores the event.

The word events, when it is used in discussing the WSG Server Monitor, corresponds to the word messages, as that word is used in discussing the Enterprise Server Monitor. Events are messages exchanged by resources in the system. For example, an

event might be processing a purchase order. To process a purchase order, the client and broker communicate.

When you select Event Types from the WSG Server Monitor workspace and click the Monitor button, the Event Types view appears.

The Event Types view contains a table with various parameters, the values of which provide information about the event types handled by the broker. This table identifies the parameters and offers a brief explanation of each one:

Parameter in Table of Event Types View	Parameter Meaning
Event Name	The unique name of the event.
Description	A description of the function of each event.
Storage Type	An event attribute that determines how the event is stored in the broker. Storage types are: ■ Guaranteed, which means that events are stored on disk using a logged commit. ■ Persistent, which means that events are stored on disk using operating asynchronous input/output. ■ Volatile, which means that events are stored in memory.
Time to Live	The amount of time that an event type can exist in the broker.
Fields	The names and types of each data field within the event type.

You can view information about the data fields for each event type by clicking the magnifying glass button in the Fields column. The Event Types Fields view appears; it identifies each field name for the event type and the field type, such as string.

5.2.3 Client Groups View

A client group is a set of broker clients with particular properties. For example, a client group defines the server on which clients access the broker.

When you select Client Groups from the WSG Server Monitor workspace and click the Monitor button, the Client Groups view appears.

The Client Groups view contains a table with various parameters, the values of which provide information about the client groups that provide control of client access to the broker. This table identifies the parameters and offers a brief explanation of each one:

Parameter in the Client Groups View	Meaning
Client Group Name	The name of the client group. Each group has a specific set of properties defined using the ActiveWorks Manager.
ACL	The Access Control List, which is a list of SSL certificates that define the entities with permission to access the broker or create a client within a client group.
Can Publish	The event types that a client group can publish.
Can Subscribe	The event types that a client group can subscribe to.
Statistics	Additional statistics on the client group, such as how long the group has been on the server and how many events have been published by clients that belong to the group.

Click the magnifying glass in the ACL, Can Publish, Can Subscribe, and Statistics columns to view additional information about each of these parameters.

5.2.4 Client States View

A client state is information maintained by the broker about a client connected to the server. For example, the client group to which a client belongs is a client state.

When you select Client States from the WSG Server Monitor workspace and click the Monitor button, the Client States view appears.

The Client States view contains a table with various parameters, the values of which provide information about the clients connected to the server. This table identifies the parameters and offers a brief explanation of each one:

Parameter in the Client States View	Meaning
Identifier	A unique identifier for the client connected to the server.
Client Group	The client group to which the client belongs.
App. Name	The name of the application that describes the client's connection to the server.
Access Label	Indicates, if appropriate, the value of the access label required for a client to connect to the client group to which it belongs.
Authenticator	The name of the certification authority that issued the certificate if SSL is enabled for the client.
Can Share	Indicates whether state sharing is enabled. If state sharing is enabled, the number of sessions allowed for a client can be set.
High Seq.	The highest published sequence number used by the client.
Max Shared Connections	The maximum number of server connections that the client can share.
Owner Name	The client owner's user name.
Sessions	Information about a client session. Click the magnifying glass button in the Sessions column to view detailed session information.

5.2.5 Broker Log View

You can use the WSG Server Monitor to view a broker log containing information about events that have run on the server during a specified time. You can view all log messages, or you can limit the view to warnings, alerts, or information messages.

When you select Broker Log from the WSG Server Monitor workspace and click the Monitor button, the Broker Log view appears.

From the Broker Log, you can specify:

- The starting date from which you want to view log files.
- The maximum number of entries you want to view.
- The type of file you want to view.

Click the View Log button to see the log file entries.

6

Working with JD Edwards EnterpriseOne on Windows Terminal Server

This chapter contains the following topics:

6.1 Understanding Windows Terminal Server

Windows Terminal Server (WTS) provides an excellent solution for Oracle's JD Edwards EnterpriseOne in a WAN environment. WTS enables you to set up multiple terminal server client machines that need only contain the WTS client software. You can use less powerful machines to function as terminal server clients. These clients connect to a machine set up with TSE software. Multiple users can simultaneously connect to the same terminal server to run JD Edwards EnterpriseOne.

Windows Terminal Server (WTS) is a multi-user extension to the Microsoft Windows family of operating systems. WTS enables users to share an application that resides on the terminal server. The terminal server performs all the processing for an application, and then sends a picture of the screen to the client terminal. Only keystrokes and mouse movement occur at the terminal. These movement commands travel through the network to the server, which returns the modified screen to the terminal.

This list provides an example of the WTS process flow:

- Step 1: JD Edwards EnterpriseOne client applications execute on terminal server.
- Step 2: The terminal server sends the video for the user interface across any connection.
- Step 3: The terminal server client displays the user interface.
- Step 4: The terminal server client sends keystrokes, mouse clicks and screen shots back to the terminal server where processing occurs.

By sending only the information necessary to recreate the screen and convey mouse and keyboard events, TSE provides LAN-like performance over WAN and dialed connections.

TSE enables you to set up multiple users to work with a single client installation of JD Edwards EnterpriseOne. By sharing a single copy of JD Edwards EnterpriseOne on the terminal server, you reduce the costs of deployment and administration.

Note: Sun-Solaris can be added to the list of enterprise servers that can run in a JD Edwards EnterpriseOne configuration with a terminal server.

6.1.1 Incorporating Citrix MetaFrame with WTS

WTS provides multi-user technology that uses a presentation protocol called Remote Desktop Protocol (RDP). RDP, based on the International Telecommunications Union T.120 protocol, is a viable option if you plan to use only Win16/Win32 clients in an uncomplicated configuration.

If the network comprises multiple platforms and requires optimum performance, Citrix offers a product called Citrix MetaFrame with multi-user technology that provides additional functionality to WTS, such as load-balancing and the support of more client platforms. MetaFrame uses the Independent Computing Architecture (ICA) presentation protocol on which Citrix based WinFrame, a successful multi-user product for Windows 3.51.

Note: If you start JD Edwards EnterpriseOne as a specified application through ICA, you cannot view the jde.ini, jde.log, and jdedebug.log files.

This table lists the capabilities of WTS RDP and MetaFrame:

Capability	TSE RDP	MetaFrame
Client Platforms	WTS client software runs on these platforms: ■ Windows 16-bit ■ Windows 32-bit ■ Some RDP-equipped Windows terminals ■ Windows CE	MetaFrame runs on these platforms: ■ DOS ■ Windows 16-bit ■ Windows 32-bit ■ X-Term ■ MacIntosh ■ Solaris ■ Windows CE ■ Some ICA-equipped network computers ■ Some internet browsers as a plug-in
Network Topologies	WTS RDP supports the TCP/IP standard.	MetaFrame supports these standards: ■ IPX ■ SPX ■ PPP ■ NetBIOS
Load-balancing	Windows supports load balancing.	You can purchase an option for MetaFrame that provides load-balancing capabilities.
Encryption	N/A	You can purchase an option for MetaFrame that provides the encryption of ICA traffic.

Capability	TSE RDP	MetaFrame
CCPDD:cut/copy/paste/drag/drop	Windows 2000 supports cut, copy, paste, drag, and drop.	MetaFrame enables CCPDD between the session window and the underlying Windows desktop.
Device Mapping	WTS RDP enables you to map local devices for printing through a work-around.	MetaFrame enables you to map devices local to the WTS client from the terminal server. For example, you can locally map hard drives, fax modems, and printers.
Session Shadowing	Windows supports remote control.	With MetaFrame loaded, WTS supports an administration tool called session shadowing. Session shadowing helps administrators audit remote sessions. You might also use session shadowing for video conferencing and in a support desk role.

6.1.2 WTS Restrictions in Multi-user Mode

JD Edwards EnterpriseOne is WTS-aware. WTS-aware means that when you deploy JD Edwards EnterpriseOne as a client on a terminal server, the software automatically recognizes the terminal server and configures itself to run in multi-user mode. Because of the configuration required by multi-user mode, these restrictions apply to terminal server users:

- Disabled development with Form Design Aid, Report Design Aid, and Table Design Aid, including Object Management Workbench check-in and check-out capabilities.

  Note: The restriction on development does not prevent new versions of existing applications, but only the modification of current applications and the creation of new applications. Also, a complete set of development specifications (500+ MB) defeats the purpose of a thin client.

- Disabled local processing for batch applications.

  All batch applications process on a separate batch server to avoid an impact to performance on the terminal server.

- Disabled Just-In-Time-Installation (JITI).

  Because you deploy a full client package to the terminal server, JITI is not recommended.

- Disabled intensive specification file access.

  Applications that intensively access specification files, such as the Universal Table Browser application, are not active due to the strain put on the terminal server when the application retrieves data. Also, file level locking could prevent access to data in the specification files for other users.

A standard JD Edwards EnterpriseOne configuration and a terminal server configuration are not mutually exclusive within an enterprise. You can mix a standard configuration with a terminal server configuration to maximize the overall performance. For example, you can use a standard JD Edwards EnterpriseOne configuration over a LAN, and use a terminal server configuration to support remote sites across a WAN.

6.1.3 Network Considerations

The terminal server must reside on the same local area network (LAN) as the enterprise server or database server, or both. Include one normal JD Edwards EnterpriseOne client on the LAN to verify performance and function. Normal JD Edwards EnterpriseOne LAN requirements apply.

For a wide area network (WAN), you must use a 56KB, or faster, line.

6.1.4 Performance Considerations

When you add any ICA session, change the Window Colors display properties to 16 color mode or to the lowest setting that the software and hardware allow. You can access display properties from the Control Panel.

Also, for the Citrix setup, you need to select the Compress data stream and Cache bitmaps to disk options. See the appropriate Citrix documentation for more information about how to modify these settings.

6.2 Setting Up JD Edwards EnterpriseOne on the Terminal Server

Because JD Edwards EnterpriseOne is WTS-aware, running on a terminal server is almost identical to running a standard client.

When performing certain processes, such as creating log files and running UBEs, JD Edwards EnterpriseOne checks whether it resides on a terminal server or standard client. If it detects a terminal server, the software automatically switches to multi-user mode.

In multi-user mode, JD Edwards EnterpriseOne processes data while simultaneously protecting data integrity and maintaining performance on the terminal server. Multi-user mode also masks any of the multi-user activity from a user so that a terminal server session of JD Edwards EnterpriseOne looks no different than a standard JD Edwards EnterpriseOne client session.

See Also:

- *JD Edwards EnterpriseOne Hardware and Software Requirements Guide*.

6.2.1 Setting Up JD Edwards EnterpriseOne on the Terminal Server

To set up JD Edwards EnterpriseOne on the terminal server:

1. Install Windows Terminal Server on the machine that you will use as the terminal server.

   Refer to the Microsoft documentation for information about how to set up Windows Terminal Server software.

2. Install Microsoft Terminal Server Client software onto the machines that you will use as terminal server clients.

   You can connect to the terminal server from the Terminal Server Client option on the Programs menu. This file also resides in the Terminal Server Client subdirectory in the Program Files directory. Refer to Microsoft documentation for information about how to set up Terminal Server Client software.

3. Depending on the type of database you use, you might need to install software so that the client can properly connect with the server where the database resides.

> **Note:** You must use Add/Remove Programs on the Control Panel to install applications on the terminal server. During installation, make sure that you select the All users begin with common application settings option on the Change User Option dialog box.

Install these software packages on the terminal server as necessary:

- Oracle for Windows
- SQL Server client
- Client Access
- DB2 for IBM i Connect

4. Install a full package of JD Edwards EnterpriseOne. You need to install a full package because JD Edwards EnterpriseOne on the terminal server is multi-user. If you install a partial package, multiple users will simultaneously experience Just-In-Time-Installation, which will negatively affect performance on the terminal server.

> **Note:** Install JD Edwards EnterpriseOne from the WTS console or use a third-party remote administration software. Do not install through a remote desktop connection. Use Add/Remove Programs on the Control Panel to install JD Edwards EnterpriseOne on the terminal server. During installation, make sure that you select the "All users begin with common application settings" option on the Change User Option dialog box.

After you perform these steps, you should be able to successfully run JD Edwards EnterpriseOne from terminal server client machines.

6.3 Troubleshooting JD Edwards EnterpriseOne on Windows Terminal Server

This section discusses how to:

- Troubleshoot UBE output security on WTS.
- Submit a UBE locally and run it on the WTS.
- Troubleshoot import/export with Microsoft Excel.
- Troubleshoot specification files are locked.
- Reduce JITI frequency.
- Troubleshoot user cannot restart JD Edwards EnterpriseOne.
- Troubleshoot logging off versus disconnecting.
- Troubleshoot shortcuts do not work in email messages.
- Troubleshoot data selection and sequencing criteria lost.
- Troubleshoot run-time error occurs during server connection test.
- Troubleshoot JD Edwards EnterpriseOne development tools are disabled.
- Troubleshoot users experience problems accessing JD Edwards EnterpriseOne.

- Troubleshoot log path is incorrect.
- Troubleshoot shortcut path is incorrect.
- Troubleshoot only one user can log in to JD Edwards EnterpriseOne.

6.3.1 Troubleshooting UBE Output Security on WTS

A JD Edwards EnterpriseOne WTS user sends PDF files by default from the enterprise server to the local \E900\PrintQueue directories using the Work With Servers (P986116) application. Users select View PDF from the Row menu of the Submitted Job Search form. Because the files are saved to a user's local PrintQueue directory, another user can view the PDF file in Windows Explorer or in Adobe Acrobat.

You can relocate the PrintQueue directory by adding this section to the jde.ini file:

```
[NETWORK SETTINGS]

OutputDirectory=C:\WTSRV\Profiles\USERNAME\Windows
```

Server administrators need to make the jde.ini file modification of each user on each WTS so that the PDF output points to each individual user profile directory. With the PrintQueue directory located in the user profiles directory, the PDF files are protected by Windows security. Only server and system administrators have access to the files.

6.3.2 Submitting a UBE Locally and Running it on the WTS

You cannot locally submit UBEs to run on the WTS because of resource constraints such as CPU power. You can run UBEs on a dedicated WTS or during hours when no other JD Edwards EnterpriseOne users are using the machine. Complete the task to submit a UBE locally and run it on the WTS.

To submit a UBE locally and run it on the WTS:

1. From the System Administration Tools menu (GH9011), select Logical Data Sources (P986115).
2. On Logical Data Sources, select the WTS machine name with System as the data source and click Select.
3. On Work with Data Sources, click Add.
4. On Data Source Revisions, complete these fields:
   - Data Source User

     Enter DB to specify a Local Data Source.
   - Data Source Name

     Enter WTS Local.
   - Data Source Type Enter N to specify MSDE/ODBC.

     Enter E to spicify OEE.
   - DLL Name

     Enter JDBODBC.DLL.
   - Database Name

     Enter System Local.
   - Server Name

Enter LOCAL.

– Platform

Enter LOCAL.

5. When you run the UBE on the WTS, on the Work with Batch Versions from, select the report you want to run and click Select.
6. On Version Prompting, select Advanced from the Form menu.
7. On Advanced Version Prompting, select the Override Location option and click OK.
8. On Version Prompting, select the Data Selection option and click Submit.
9. On JDE Data Sources, select WTS Local as the data source and click Select.

6.3.3 Troubleshooting: Import/Export with Microsoft Excel

Importing a Microsoft Excel spreadsheet into a JD Edwards EnterpriseOne grid intermittently fails when users are running WTS.

Please check the MTR (minimum technical requirements) to be sure you are running the latest supported versions of Microsoft Office. You install a single of Microsoft Office 2000 on the WTS. Multiple users then connect to the server and run Microsoft office from the server.

6.3.4 Troubleshooting: Specification Files are Locked

In these circumstances, JD Edwards EnterpriseOne users get a message box to indicate that a specification file is currently unavailable:

- When another JD Edwards EnterpriseOne session on the same WTS machine performs a data dictionary Just-In-Time-Installation (JITI).
- When another JD Edwards EnterpriseOne session receives a WTS run-time error dialog box. Generally, this type of error occurs when a memory violation occurs.

In both cases, the specification file or files are locked. The specification files will be unlocked when either the WTS completes the JITI process or a user closes the WTS run-time error dialog box.

To prevent the specification files from being locked when JD Edwards EnterpriseOne performs a JITI, reduce the frequency that JD Edwards EnterpriseOne performs JITIs. Complete this task:

6.3.5 Reducing JITI Frequency

To reduce JITI frequency (B73.3.1, B73.3.2, and B73.3.3):

1. Run the Generate global table spec (R98CRTGL) batch application on a non-WTS JD Edwards EnterpriseOne client to generate full GLBLTBL specification files.
2. Copy the full glbltbl.ddb and glbltbl.xdb to the WTS machines.

6.3.6 Troubleshooting: User Cannot Restart JD Edwards EnterpriseOne

Occasionally, when a memory violation occurs in a JD Edwards EnterpriseOne WTS session, the terminal server prevents the user from restarting JD Edwards EnterpriseOne. The administrator must then sign onto the terminal server and end the

OEXPLORER.exe process from the Task Manager. After the administrator ends the process, the user can sign on to JD Edwards EnterpriseOne again.

When a run-time exception occurs, JD Edwards EnterpriseOne should immediately exit. To instruct JD Edwards EnterpriseOne to immediately exit in this situation, change the EXCEPTION_Enabled setting in the jde.ini to False:

```
 [INTERACTIVE RUNTIME]

EXCEPTION_Enabled=False
```

6.3.7 Troubleshooting: Logging Off Versus Disconnecting

Users should always log off their WTS session rather than disconnecting. Logging off shuts down all processes completely for the user.

6.3.8 Troubleshooting: Shortcuts Do Not Work in Email Messages

Workflow provides the ability to send shortcuts to JD Edwards EnterpriseOne applications using email messages. This function does not work when the email application, such as Microsoft Outlook, is not currently active on the terminal server. When the email application invokes the shortcut, the operating system attempts to launch the shortcut on the local machine and not on the terminal server.

Run the email application and JD Edwards EnterpriseOne on the same terminal server machine.

6.3.9 Troubleshooting: Data Selection and Sequencing Criteria Lost

This situation occurs when two or more users are signed on to the same terminal server using the same pathcode.

The first user submits a batch application from Batch Versions, changes the data selection criteria, and then stops at the printer screen. The second user then goes into Batch Versions to submit the same version of the batch application, changes the data selection criteria, and then stops at the printer screen. When the users click OK to send the batch application to the enterprise server for processing, the data selection criteria for the second user overrides the selection criteria for the first user.

A modification to batch processing in JD Edwards EnterpriseOne now saves data selection and sequencing criteria in memory rather than in specification files.

6.3.10 Troubleshooting: Run-Time Error Occurs During Server Connection Test

The Server Administration Workbench (SAW) application receives a run-time error when it performs a server connection test. This situation occurs when the user who performs the connection test does not possess the authority to access the pinging mechanism on the target machine.

6.3.11 Troubleshooting: JD Edwards EnterpriseOne Development Tools Are Disabled

JD Edwards EnterpriseOne development tools are disabled on the terminal server. Currently, we instruct customers to perform all development on non-WTS machines.

6.3.12 Troubleshooting: Users Experience Problems Accessing JD Edwards EnterpriseOne

Only administrators can run JD Edwards EnterpriseOne. This situation is a result of the way JD Edwards EnterpriseOne was installed on the terminal server.

As the Administrator, you should use the Add/Remove Programs application on the Control Panel to install JD Edwards EnterpriseOne on the terminal server. During installation, make sure that you select the All users begin with common application settings option on the Change User Option dialog box. This option ensures that the terminal server maintains JD Edwards EnterpriseOne specific files, such as the jde.ini file, across user profiles.

6.3.13 Troubleshooting: Log Path is Incorrect

The log path in the jde.ini for individual users is incorrect.

The JD Edwards EnterpriseOne installation program sets the WTSLogs setting to False. Users should change this setting to True after the installation and before any users run JD Edwards EnterpriseOne.

When the WTSLogs setting is True, the output log directories for each user point to the home directory of the user rather than to the root directory of the drive. The output log directories settings are also defined in the jde.ini file.

6.3.14 Troubleshooting: Only One User Can Sign in to JD Edwards EnterpriseOne

The main JD Edwards EnterpriseOne window fails to appear after entering the password for all other users.

For B73.3.2 with Service Pack 10 or greater, place the JD Edwards EnterpriseOne command line switch /NoLogo, located in the JD Edwards EnterpriseOne shortcuts, on the desktop and on the Start menu if not already present. Separated by a space, append the text to the end of the line in the Target edit box of the shortcut properties window. Do not include the quotation marks. This will prevent the display of the splash screen.

7

Administering JD Edwards EnterpriseOne on a Unix Cluster

This chapter contains the following topics:

7.1 Understanding Clustering

High availability clusters provide redundancy of software and hardware so that a single point of failure will not interrupt service. If a failure occurs, the clustering software automatically detects the problem and shifts to an alternate machine without ending processes and interrupting the enterprise.

Clustering enables Oracle's JD Edwards EnterpriseOne processes running on a machine that fails to continue running without interruption on a second machine. The second machine has a setup that supports the given processes. Essentially, JD Edwards EnterpriseOne moves to the alternate machine without requiring you to restart a process that was active on the machine that failed.

Note: Each node in the cluster must have the appropriate software and hardware to ensure that processing moves successfully from server to server.

Make sure you understand the clustering software and the tasks necessary to implement the software on a given platform.

7.1.1 Hp-UX Clustering

Hewlett-Packard provides two mutually exclusive software products to manage high availability clusters:

- Hewlett-Packard Multi-Computer/ServiceGuard (MC/ServiceGuard)
- Hewlett-Packard Multi-Computer/LockManager (MC/LockManager)

  You must use MC/LockManager when you use Oracle Parallel Server (OPS).

You will set up only one of these products for the HP-UX cluster.

JD Edwards EnterpriseOne requires a named IP address for workstations to connect with a server. With the Hewlett-Packard clustering software, you can assign a floating IP address that can move from node to node within the cluster. You should enter this IP address into the WINS or DNS database so that workstations can access the address. If the enterprise servers are not using DNS to resolve host names, you must also add the floating IP address to the /etc/hosts file on each node in the cluster where JD Edwards EnterpriseOne might run.

Note: If you do not assign a floating IP address, then whenever JD Edwards EnterpriseOne moves to another node in the cluster, the workstations will be unable to connect with the servers.

7.2 Maintaining Multiple Instances of JD Edwards EnterpriseOne in a Clustered Environment

When you run multiple instances of JD Edwards EnterpriseOne in a clustered environment, you must consider several factors. Even though each instance might begin on a separate node, a situation might arise for which multiple instances need to run on the same node. When this happens, communication to each JD Edwards EnterpriseOne instance must occur on a different port number, or service name, and each instance must use a different range of IPC keys. These parameters in the jde.ini file control these settings:

```
[JDENET]
serviceNameListen=Service Name or Port Number
serviceNameConnect=Service Name or Port Number
[JDEIPC]
startIPCKeyValue=Numeric Value
```

where

- Service Name or Port Number is the actual port number or the name of a service that you enter into the /etc/services file.
- Numeric Value is the IPC key value of the JD Edwards EnterpriseOne instance.

  The IPC key values should differ by at least 1000 between any two JD Edwards EnterpriseOne instances.

7.3 Setting Up Clustering

This section discusses how to:

- Configure Oracle Parallel Server (OPS).
- Set up an Oracle package for MC/ServiceGuard.
- Set up a JD Edwards EnterpriseOne package.

7.3.1 Configuring Oracle Parallel Server (OPS)

Oracle Parallel Server (OPS) enables concurrent database access from multiple nodes in a cluster. If you use OPS, you must install MC/LockManager, not MC/ServiceGuard.

This task describes how to set up OPS and MC/LockManager. For more information, refer to the Hewlett Packard documentation on setting up OPS and MC/LockManager.

To set up Oracle Parallel Server and MC/LockManager:

1. Install Oracle client software on each node in the cluster where it might run. The JD Edwards EnterpriseOne database should be created on shared disks, or on a machine outside the cluster, so that it can be accessed from multiple nodes.
2. Create a package in MC/ServiceGuard with no services.

   This package should be set up with an associated IP address so that it can be reached from any node where it might run. This package should also specify the shared volume group on which the JD Edwards EnterpriseOne database will reside.
3. Edit the package control script to add the Oracle startup and shutdown commands. This code sample provides an example of the customer_defined_run_cmds function from a package control script:

```
 function customer_defined_run_cmds
{
# ADD customer defined run commands.
export ORACLE_HOME=/u01/app/oracle/product/8.0.5
export ORACLE_SID=jde1
export ORAENV_ASK=NO
. $ORACLE_HOME/bin/oraenv
su oracle -c '$ORACLE_HOME/bin/lsnrctl start'
su oracle -c '$ORACLE_HOME/bin/svrmgrl' <<EOF1
connect internal
startup
exit
EOF1
test_return 52 }
```

4. You can use these same steps to enter the Oracle shutdown commands into the customer_defined_halt_commands section of the package control script.

7.3.2 Setting Up an Oracle Package for MC/ServiceGuard

If you use MC/ServiceGuard, you should set up a package for Oracle. An Oracle package enables the Oracle processes to move from one node to another when a node fails or during scheduled maintenance.

You do not need to perform this step if you use MC/LockManager.

To set up an Oracle package for MC/ServiceGuard:

1. Install Oracle on each node in the cluster.

   Create the JD Edwards EnterpriseOne database on shareable disks so that multiple nodes can access the database.
2. Create a package in MC/ServiceGuard with no services.

You should set up this package with an IP address so that any node on the cluster can access and run the package. This package should also specify the shared volume group where the database will reside.

3. Edit the package control script to add the Oracle startup and shutdown commands.

   This code sample provides an example of the customer_defined_run_cmds function from a package control script:

```
Function customer_defined_run_cmds

{
# ADD cusomter defined run commands.
Export ORRACLE_HOME=/u01/app/oracle/product/8.0.5
Export ORACLE_SID=jde1
Export ORAENV_ASK=NO
.$ORACLE_HOME/bin/oraenv
su oracle '$ORACLE_HOME/bin/svrmgr1'<<EOF1
connect internal
startup
exit
EAOF1
Test return 52
}
```

   You can use the same steps to enter the Oracle shutdown commands into the customer_defined_halt_commands section of the package control script.

7.3.3 Setting Up a JD Edwards EnterpriseOne Package

The standard JD Edwards EnterpriseOne enterprise server software requires minimal modifications to function in a cluster. These modifications include these items:

- Enterprise server jde.ini
- owenv script file in the $SYSTEM/bin32 directory
- Package control script

To set up a JD Edwards EnterpriseOne package for a cluster:

1. In the server jde.ini file, locate the [CLUSTER] section, then change these setting:

```
[CLUSTER]

PrimaryNode=Package IP name
```

   where Package IP name represents name given to the IP address that is associated with the JD Edwards EnterpriseOne package that you are creating.

2. Change the owenv script file in the $SYSTEM/bin32 directory. The owenv script file contains the settings for various UNIX environment variables required by JD Edwards EnterpriseOne.

3. Create a package using SAM.

Note: You can also use the command line to create a package. See HP documentation for details.

For JD Edwards EnterpriseOne, set up a package with an associated floating IP address, but with no services. This setup is necessary because the cluster manager needs to start services without environment variables under the root user. The software will not run properly unless you set environment variables. This setup also enables you to utilize the installation defaults and the start and end scripts provided by the JD Edwards company.

Depending on the needs of the enterprise, you might want to install JD Edwards EnterpriseOne on a shared volume group. This setup enables multiple nodes in a cluster to access a single version of JD Edwards EnterpriseOne, but only one node at a time. This setup also enables you to easily update JD Edwards EnterpriseOne through server package installations.

4. Using SAM, modify the package control script on each node to start and stop JD Edwards EnterpriseOne.

> **Note:** You might need to vary the control scripts for different nodes in the cluster to configure different volume group names or path names. In this case, you must edit the scripts on each node individually instead of using SAM.

This code sample provides an example of the customer_defined_run_cmds function from a package control script:

```
function customer_defined_run_cmds
{
# ADD customer defined run commands
# wait 60 seconds for Oracle to come up
sleep 60. /home/jde/owenvsu jde
<< EOF1mv $OWHOME/log/jde*.log
$OWHOME/log/oldlogscd
$SYSTEM/bin32RunOneWorld.sh
EOF1
test_return 51
}
```

This code sample provides an example of the customer_defined_halt_cmds function from a package control script:

```
function customer_defined_halt_cmds
{
# ADD customer defined halt commands
.. /home/jde/owenv
su jde << EOF2
cd $SYSTEM/bin32
EndOneWorld.sh
sleep 15
rmics.sh
EOF2
test_return 52
}
```

The following list provides explanations for these functions:

```
sleep 60
```

The "run" function first waits 60 seconds for Oracle processes to start. HP states that you should set the PKG_SWITCHING_ENABLED parameter to NO for applications that access OPS. This setting prevents these applications from starting

before OPS is active. If you use the sleep command in the script, you do not need to modify this setting. You can remove the sleep command from the script when you use the control script for a backup node with OPS running.

```
. /home/jde/owenv
```

This line runs the owenv script to set up UNIX environment variables. The owenv script resides in the $SYSTEM/bin32 directory. You must edit this script to ensure that the correct setup exists for all necessary environment variables for JD Edwards EnterpriseOne and Oracle. In these examples, the script has been moved to the home directory of the jde user. The script might need to move to the home directory if you use a different SID to access Oracle from different nodes.

```
su jde
```

This line switches to the user ID that owns the JD Edwards EnterpriseOne processes. If you omit this line, the root user owns the JD Edwards EnterpriseOne processes.

```
mv $OWHOME/log/jde*.log $OWHOME/log/oldlogs
```

This line moves any logs in the log directory to a backup log directory, which you create. This command is particularly important if the JD Edwards EnterpriseOne instance resides on a shared disk where a "failed over" instance of JD Edwards EnterpriseOne will use the same physical disk space as the failed instance. You might consider adding the command rm $OWHOME/log/oldlogs/* before this line to clean out any older versions of logs.

```
RunOneWorld.sh; EndOneWorld.sh
```

These are the standard start and stop scripts that we provide for the UNIX enterprise server.

The directory that contains the package control script also contains the control.sh.log file, which contains the results of starting and stopping a package. This file is the first place to check if problems arise when you start or stop a package. In particular, it will contain any output or error messages from the customer-defined commands you might enter.

7.4 Setting up HACMP for AIX Clustering

This section provides an overview of HACMP for AIX clustering and discusses how to create group and user accounts.

7.4.1 Understanding HACMP for AIX Clustering

If a failure occurs, HACMP provides a transparent recovery for critical applications. You can configure a cluster using any RS/6000 processor and a variety of network adapters and disk subsystems to satisfy the LAN, disk capacity, and performance requirements.

Be careful when you delete or write to shared files. You might want to move old log files rather than delete them. If you move a package running on a shared file system from one node to another, the new instance of JD Edwards EnterpriseOne references the logs and files from the old instance.

7.4.1.1 How HACMP Works

HACMP for AIX (Version 4.2) enables customers to automatically detect system failures and recover users, applications, and data on backup systems, minimizing downtime to minutes or seconds. In addition, using HACMP for AIX virtually eliminates planned outages, since users, applications and data can be moved to backup systems during scheduled system maintenance. HACMP Version 4.2 adds new features such as the Cluster Single Point of Control (CSPOC) and Dynamic Reconfig, which enable the system administrator to add users, files, and security functions without stopping mission-critical jobs.

HACMP provides several configuration options, including these:

- Idle standby for up to seven processors being backed up by a single processor.
- Rotating standby for up to seven processors backed up by a standby processor in a predefined or contention takeover sequence.
- Mutual takeover for up to eight processors backing each other up by sharing the application workloads.
- Concurrent access for up to eight processors working on the same jobs and sharing the same data.

The configuration flexibility of HACMP enables customers to select the cluster topology and database manager that best suits the requirements of their computing environment. IBM states that HACMP can support both concurrent and parallel data access within a common cluster. HACMP also operates with the new Parallel Database Products, such as IBM DB2 for IBM i Parallel Edition and Oracle 8 Parallel Server.

Several components make up the HACMP environment, including these:

- Nodes

  Nodes are the core of an HACMP cluster. A node is a processor that runs the AIX operating system, HACMP, and the mission-critical software. Software execution can be spread over several nodes for system load balancing. In the event of a failover, HACMP executes customer-defined scripts that will establish environments and start specific software packages on a standby node.

- Shared external disk

  Shared external disks are disks that are physically connected to multiple nodes. The shared disks store mission-critical data, which is shared among processes running on separate nodes.

- Networks

  Networks are the independent components of HACMP. TCP/IP is the protocol with which HACMP was designed to function. It has been tested with ethernet, token ring, and Fiber Distributed Data Interface (FDDI) topology.

- Network adapters
- Clients

7.4.1.2 Installation Considerations

Installing the HACMP cluster requires that you create logon accounts and use the Oracle Standard Enterprise Database Management System (DBMS).

A major consideration when setting up HACMP and JD Edwards EnterpriseOne is user accounts. When these accounts are created, they are given unique user IDs and unique role IDs. When a node fails over to another node, these unique IDs are matched

to names in the /etc/passwd and /etc/role files. If no matches occur, the unique user and role values are then used as IDs, which can create problems with access and security.

To avoid a problem, before starting the JD Edwards EnterpriseOne installation or configuration, create all user accounts and roles on all nodes that will be used in the cluster environment. Use the same unique number for all users and the same unique number for all roles. The easiest way to do this is to use the add user/role function found in the HACMP extension of SMIT.

If JD Edwards EnterpriseOne is already installed, use the existing user and role ID numbers to create accounts on the nodes that are defined in the resource group.

Oracle Standard Enterprise DBMS is used in the HACMP configuration explained in this chapter. The resource control scripts are coded to start and stop the database using standard Oracle program calls. These scripts can be easily modified to allow for changes in database start and stop procedures as well as the introduction of commands for Oracle Parallel Server. To minimize installation and configuration problems, have the database administrator review the commands in the control scripts to ensure that they are correct for the installation.

7.4.2 Creating Group and User Accounts

By performing this operation using the add group/user option in the HACMP component of SMIT, all user accounts and group assignments are synchronized across all nodes. This ensures that when the resource disk volume groups remount on the failover system, the user and group IDs match.

To create group and user accounts:

1. Verify that HACMP is running on all nodes within the resource group.
2. Select a unique ID number that can be assigned to the new group and user that you want to create.
3. Verify the selection by searching all password and group files on the node where the new user and group will be created.
4. Enter this command on the command line:

   ```
   # smit hacmp
   ```

5. From the menu, select Cluster System Management, then Cluster Users & Groups, then Groups, and then Add a Group to the Cluster.
6. Select the resource group to which you want to add the new group. The resource group identifies the nodes that will need to be updated. Next, add a group called JD Edwards EnterpriseOne and assign it the unique ID number that you chose.
7. Press Enter.
8. From the Cluster Users & Groups panel, select Users and then Add a User to the Cluster.
9. Select the same resource group that you chose for adding a group.
10. Add a user psft, assign it the pre-selected unique ID number, and select the JD Edwards EnterpriseOne group.
11. Repeat these steps for the Oracle sign on, creating the group dba.

7.5 Setting Up JD Edwards EnterpriseOne for HACMP

The standard JD Edwards EnterpriseOne enterprise server software requires minimal modifications to function in a cluster, including editing the enterprise server jde.ini file, the owenv script, the start resource control script, and the stop resource control script.

This section discusses how to:

- Edit the owenv script.
- Edit the Start Resource Control script.
- Edit the Stop Resource Control script.

7.5.1 Editing the owenv Script

To edit the owenv script:

1. In Windows Explorer, go to the /$SYSTEM/bin32 directory and open the owenv file.
2. Edit the emphasized lines:

```
#! /bin/ksh
## set OWHOME to point to the base install path for JD Edwards EnterpriseOne
export OWHOME=/ow2/JDEdwards/E900
## set ENVIRON to the path code from which you want to run business functions
export ENVIRON=MSTR
## set up the path to your JD Edwards EnterpriseOne system and path code
export SYSTEM=$OWHOME/system
export APPDEV=$OWHOME/$ENVIRON
## set JDE_BASE to the location of your JDE.INI file
export JDE_BASE=$SYSTEM/ini/aix
## set up the Oracle environment
export ORACLE_HOME=/u01/app/oracle/product/8.0.5
export ORACLELIB=$ORACLE_HOME/lib
## the remaining variables point to libraries and executables
export SHLIB_PATH=$SYSTEM/lib:$APPDEV/bin32:$ORACLELIB:$SYSTEM/libv32
export LD_LIBRARY_PATH=$SHLIB_PATHexport PATH=$PATH:$SYSTEM/bin32
```

3. Save and close the file.

7.5.2 Editing the Start Resource Control Script

To edit the start resource control script:

1. In Windows Explorer, go to the /$SYSTEM/bin32 directory and open the StartResource.sh file.
2. Edit the emphasized lines:

```
#_____________________
#Global Variables
#_____________________
#
#export PATH=$PATH
#Set environment variables
#
./usr/sbin/cluster/scripts/owenv
#
loop=0
StartUpError=false
```

```
ORACLEPROCESS=oracle
OWStartupDir=$SYSTEM/bin32
StartUpLog=$OWStartupDir/OWStartup.log
ORACLE_UID=oracle
APP_UID=psft
LOGFILES=$OWStartupDir/jde*.log
```

The first emphasized line executes the script that sets various required JD Edwards EnterpriseOne environment variables.

Note: This script, as well as others, will be relocated into the scripts directory as described in the Control Scripts section under Creating an Application Server in the JD Edwards EnterpriseOne Tools 8.94 Implementation Guide: Server and Workstation Administration.

ORACLE_UID and APP_UID are the login ID names for Oracle and JD Edwards EnterpriseOne. These are used in the script so that the respective applications are started with the proper application ownership.

Note: This script is delivered with these IDs undefined. If the script is executed, an error message will be generated.

---

3. Save and close the file.

7.5.3 Editing the Stop Resource Control Script

To edit the stop resource control script:

1. In Windows Explorer, go to the /$SYSTEM/bin32 directory and open the StartResource.sh file.
2. Edit the bold lines:

```
#_____________________
#Global Variables
#_____________________
export PATH=$PATH:.
#
# Set environment variables
#
./usr/sbin/cluster/scripts/owenv
LogicalVolumn=/ow2
ShutdownDir=$SYSTEM"/bin32"
StartUpLog=$OWStartupDir"/OWStartup.log"
APP_UID=psft
```

Similar to the script modification described in the previous procedure, the APP_ID needs to have the login ID of the application owner. In this example, psft owns the application.

3. Save and close the file.

7.6 Creating an Application Server

The application server is a method that invokes predetermined actions of applications. The server is called in the cluster startup or shutdown sequence and executes

predefined scripts depending on what activity is occurring. As part of the cluster installation and configuration process, an application server must be created and the access path to the control scripts defined.

This section discusses how to:

- Move the control scripts.
- Define an application server.
- Define cluster resources.

7.6.1 Prerequisite

The control scripts that are included with JD Edwards EnterpriseOne are located in the /$SYSTEM/bin32 directory. These scripts are generic and can be modified as needed to meet the requirements. Before you can create an application server, you must move the scripts to a non-shared disk directory.

7.6.2 Moving the Control Scripts

To move the control scripts:

1. Log in as root and enter these commands:

   ```
   #export SYSTEM=< JD Edwards EnterpriseOne system directory path>
   ```

   Where JD Edwards EnterpriseOne system directory path is the path to the system directory. An example of the path is /ow2/JDEdwards/900/system.

   ```
   # cd /usr/sbin/cluster

   # mkdir scripts

   # cd scripts
   ```

   Be sure to include the periods, preceded by a single space, in these commands:

   ```
   # cp $SYSTEM/bin32/StartResource.sh .

   # cp $SYSTEM/bin32/owenv .

   # chmod 755 *
   ```

2. Repeat this step on all failover nodes and FTP over the modified script files.

7.6.3 Defining an Application Sever

To define an application server:

1. From the command line, enter this command:

   ```
   # smit hacmp
   ```

2. From the menu, select Cluster Configuration, then Cluster Resources, then Define Application Servers, and then Add an Application Server.
3. In the Server Name field, enter **OneWorldSrv**. This adds a label to the resource server that controls the starting and stopping of JD Edwards EnterpriseOne.
4. In the Start Script field, enter the path of the StartResource.sh script - for example, `/usr/sbin/cluster/scripts/StartResource.sh`.

5. In the Stop Script field, enter the path of the StopResource.sh script - for example, `/usr/sbin/cluster/scripts/StopResource.sh`.
6. Press Enter.

7.6.4 Defining Cluster Resources

This procedure explains how to let HACMP know that you have defined an application server. This is so that HACMP will know to use the server during a cluster-related event. Within the cluster parameters display is a field in which this resource is defined.

To define cluster resources:

1. From the command line, enter this command:

   ```
   # smit hacmp
   ```

2. From the menu, select Cluster Configuration, Cluster Resources, Change/Show Resource for a Resource Group.
3. Select the resource group - for example, JD Edwards EnterpriseOne.
4. On Configure Resources for a Resource Group, in the Application Server field, enter the name of the application server that you defined in the previous procedure.

7.7 Setting Up Sun Solaris Clustering

This section provides an overview of Sun Solaris clustering and discusses how to:

- Modify the SunStartResource.sh script.
- Modify the SunStopResource.sh script.
- Modify the owenv script.
- Modify the SunOracleMgr.sh script.
- Register JD Edwards EnterpriseOne with SUNClustering.

7.7.1 Understanding Sun Solaris Clustering

This software provides higher availability for the applications because it enables you to recover almost instantaneously from a power failure or hardware problem. It also enables applications to be available during scheduled downtime.

This documentation assumes that you have successfully installed Oracle and the SUNClustering software. If you are having trouble with either of these products, you should contact Oracle or Sun as needed.

You must have a disk that is accessible to all machines in the cluster, and this disk must be large enough to accommodate the JD Edwards EnterpriseOne installation. If you wish to place the database on the same cluster as well, the database file must also be placed on a shared disk accessible to all machines in the cluster (although not necessarily on the same shared disk as the one that the JD Edwards EnterpriseOne server is on).

The SUNClustering 2.2 or greater API is recommended.

Four cluster-specific scripts are delivered with JD Edwards EnterpriseOne:

- SunStartResource.sh

- SunStopResource.sh
- owenv
- SunOracleMgr.sh

These scripts can be found in the system/bin32 directory under the base JD Edwards EnterpriseOne installation directory.

The SunStartResource.sh script runs whenever a node in the cluster starts the JD Edwards EnterpriseOne service. It must be registered with the SUNClustering software and should handle everything that needs to happen when the JD Edwards EnterpriseOne service starts or is switched from one node to another.

The SunStopResource.sh script runs whenever a node in the cluster stops the JD Edwards EnterpriseOne service. It is registered with the SUNClustering software, and handles shutting down various processes and any cleanup that needs to happen when the JD Edwards EnterpriseOne service is stopped.

The owenv script sets various UNIX environment variables that are needed by JD Edwards EnterpriseOne. The script is called from within the SunStartResource.sh and SunStopResource.sh scripts.

The SunOracleMgr.sh script is necessary if you are running the database on the same cluster as the JD Edwards EnterpriseOne server. If you are not running the database on the same cluster, you can ignore this section.

See Also:

- Registering JD Edwards EnterpriseOne with SUNClustering.

7.7.2 Modifying the SunStartResource.sh Script

To modify the SunStartResource.sh script:

1. Under Global Variables, there is a call to /suncldata/JDEdwards/E900_sp1/system/bin32/owenv. Change /suncldata/JDEdwards/E900_sp1/ to the same path that OWHOME was set to in the owenv script.
2. Set the APP_UID to the user name that JD Edwards EnterpriseOne is to be run under.
3. Set ORACLE_UID to the user that is to run Oracle if the database is on the same cluster as JD Edwards EnterpriseOne.
4. If you are using Oracle, uncomment the section under Check for ORACLE running and Check to see if ORACLE started/running.

7.7.3 Modifying the SunStopResource.sh Script

To modify the SunStopResource.sh script:

1. Under Global Variables, change the call to /suncldata/JDEdwards/E900_sp1/system/bin32/owenv to the same thing that is in the SunStartResource.sh script.
2. Set APP_UID and ORACLE_UID to the users running JD Edwards EnterpriseOne and Oracle, respectively. These values will be the same as in SunStartResource.sh.
3. Under the Shutdown JD Edwards EnterpriseOne section, set LOGDIR to be the location where all the log files are located.

7.7.4 Modifying the owenv Script

To modify the owenv script:

1. Set OWHOME to be the base directory of JD Edwards EnterpriseOne, for example, /suncldata/JDEdwards/E900_sp1/.
2. Set ENVIRON to the pathcode that you are using, for example, PROD or CRP.
3. Once OWHOME and ENVIRON are set, SYSTEM, APPDEV, AND JDE_BASE should be correct.
4. Set ORACLE_HOME to be the location of the ORACLE installation on the machine, for example, /suncldata/app/oracle/product/8.0.5.
5. Set ORACLE_SID as needed.

7.7.5 Modifying the SunOracleMgr.sh Script

To modify the SunOracleMgr.sh script:

1. In the Setup Global Variables section of the script, set ORACLE_UID to the user ID that is used to start Oracle.
2. Set ORACLE_HOME to the appropriate value for the Oracle installation.

7.7.6 Registering JD Edwards EnterpriseOne with SUNClustering

You must register JD Edwards EnterpriseOne with the SUNClustering software.

To register JD Edwards EnterpriseOne with SUNClustering:

1. To register JD Edwards EnterpriseOne with SUNClustering, log in as the root user and enter this command:

   ```
   /opt/SUNWcluster/bin/hareg -r[service name] -m start_net=[absolute path of
   SunStartResource.sh] -mstop_net=[absolute path of SunStopResource.sh]
   ```

   Where service name can be anything you want, but you may want to make it something easy to remember like JD Edwards since you will need to use that name when modifying JD Edwards EnterpriseOne registry with SUNClustering.

2. Enter this command with no options:

   ```
   /opt/SUNWcluster/bin/hareg
   ```

   If the line containing the service name you assigned to JD Edwards EnterpriseOne contains off, then enter this command:

   ```
   /opt/SUNWcluster/bin/hareg -y [service name]
   ```

   This sets the data service to "on," which enables the data service to switch physical hosts when appropriate.

3. To test whether the cfluster switches, enter this command:

   ```
   /opt/SUNWcluster/bin/haswitch
   ```

   Also, try pulling the plug on the current active host.

   Note: You should not use reboot or shutdown on the primary node as a test as doing so will result in an error and not in a switchover.

7.8 Troubleshooting HP-UX Clustering

These sections address specific problems with HP-UX clustering.

7.8.1 Problems with Oracle Parallel Server (OPS)

Complete these steps if you experience problems with OPS:

- Verify that the cluster software is operational. OPS requires the cluster software to start before OPS can start.
- Verify that DLM is enabled in the DLM configuration. Oracle Group Management Services (OGMS) will not start if DLM is disabled.

7.8.2 JD Edwards EnterpriseOne Does Not Start

Complete these steps if JD Edwards EnterpriseOne fails to start:

- When you start JD Edwards EnterpriseOne using the package control script, first check the control script log for errors. Look for errors in the script that occur before the RunOneWorld.sh command.
- Check the log directory for log files. If none reside, verify that the JD Edwards EnterpriseOne processes exist in the proper directory and that you correctly set the $SYSTEM environment variable.
- If the log file names are in all capital letters, the $JDE_BASE environment variable might be set incorrectly. If you incorrectly set this environment variable, the process will be unable to locate the jde.ini file.
- Verify whether an entry exists in the /etc/hosts table for the floating IP address. If no entry exists, jdenet_n will start, but all other processes will return this message in the log: 239-gethostbyname returned Connection refused.
- If no entry for the floating IP address exists that the workstation can reference, it will fail to connect and return this message in the log: 11001-gethostbyname returned 11001 (WSAHOST_NOT_FOUND): The host was not found.

7.8.3 Problem with Workstation Connection to a JD Edwards EnterpriseOne Server; Endnet Works Improperly on the Server

You must associate an IP address with the JD Edwards EnterpriseOne package. The package must be operational for the IP address to be active. Otherwise, workstations will not connect to the server and endnet will not work properly on the server.

7.8.4 JD Edwards EnterpriseOne Does Not Work From the Package Control Script

Oracle must be operational and the owenv must reference the proper SID for JD Edwards EnterpriseOne to work from within the package control script.

7.8.5 Package Does Not Switch to the Backup Node upon Failure or Removal from the Cluster

You must enable automatic switching in the package failover options. If you do not enable this setting, the package will not switch to the backup node when the node fails or you remove the node from the cluster. If you do not want the package to switch, for example, you might want to stop JD Edwards EnterpriseOne, you can disable this flag and then halt the package.

7.8.6 Package Halt Fails

If JD Edwards EnterpriseOne does not end cleanly during a package halt, the package halt might fail. This could occur if Oracle is not operational or if JD Edwards EnterpriseOne cannot access the database. You might need to change the test condition in the package control script, or add commands to search for remaining JD Edwards processes and end them.

7.8.7 Placement of the owenv File

Generally, the owenv file should not reside on the shared disk. Different environment settings, particularly ORACLE settings, might exist depending on which node you run a package. If you placed the JD Edwards EnterpriseOne bin32 directory on a shared disk, move the owenv file to another directory.

8

Administering JD Edwards EnterpriseOne on a Windows Server Cluster

Oracle's JD Edwards EnterpriseOne supports Microsoft Cluster services as a server cluster. This chapter does not cover Network Load Balancing Clusters.

Configuration of Microsoft Server Cluster has changed with Oracle's JD Edwards EnterpriseOne 8.9. This information is valid beginning with JD Edwards EnterpriseOne 8.9 and should not be used with previous releases. For more information about how to install and set up a Microsoft Server Cluster, see the appropriate Microsoft documentation.

This chapter contains the following topics:

8.1 Prerequisites

Before you complete the tasks in this chapter:

- Partition the disk array to logically divide the software components. Typically, you need to set up these partitions:
  - A partition that contains the cluster software.
  - A partition that contains the JD Edwards EnterpriseOne software.
  - A partition that contains the database management system (DBMS) software and database if these reside on the cluster system.
- Configure the network, which includes setting up connections among servers, workstations, and printers.
- Set up the cluster according to the Microsoft documentation for the Window Server version you are using.
- JD Edwards EnterpriseOne requires a static IP address assigned to a virtual server name. The virtual server name is used as the JD Edwards EnterpriseOne enterprise server name or application server name in the cluster.

Although you need only one network card in each node, you should use two cards to ensure redundancy. One network card communicates with the public network, and the second card connects between nodes. This setup enables the cluster to remain active when the primary node loses the network connection. If you use only one network card, when a node loses the network connection, that node also loses the connection to other nodes in the cluster.

- If the database and Oracle's JD Edwards EnterpriseOne are both running on the cluster, they can be configured to run on separate nodes. To accomplish this, separate JD Edwards EnterpriseOne resources and database resources into different groups. Also, be sure that JD Edwards EnterpriseOne and the database do not share disk resources. JD Edwards EnterpriseOne resources need to be in the same group as the cluster network name and cluster IP address. This can be the cluster group.

  When the JD Edwards EnterpriseOne and database groups are in separate groups, the database group must be online before bringing the JD Edwards EnterpriseOne resources online.

  If you do not require JD Edwards EnterpriseOne and database resources to run on separate nodes, place all database and JD Edwards EnterpriseOne resources in the cluster group.

  See Also:

  - Starting the Windows Enterprise Server in the *JD Edwards EnterpriseOne Installation Guide*.
  - "Managing Server Jobs" in the *JD Edwards EnterpriseOne Tools System Administration Guide* .

8.2 Upgrading JD Edwards EnterpriseOne in a Microsoft Windows Server Cluster Environment

If you are already running JD Edwards EnterpriseOne in a cluster but are upgrading to JD Edwards EnterpriseOne 8.9 or later, you must change the cluster configuration for the cluster to operate properly. JD Edwards EnterpriseOne 8.9 and later releases require use of an IP address and a network name separate from the cluster IP address and network name.

Note: In previous releases, the cluster network name and IP address were used for the JD Edwards EnterpriseOne enterprise server name.

A Microsoft Windows Server 2008 Server Cluster requires an IP address and network name separate from the cluster IP address and network name for JD Edwards EnterpriseOne. Because the cluster name exists in JD Edwards EnterpriseOne ini files and tables, the recommended solution is to rename the cluster and create new resources for the JD Edwards EnterpriseOne group using the old cluster name and a different IP address. This requires changing the static IP address associated with the old cluster name to avoid an IP address conflict. Refer to the Microsoft documentation about renaming a cluster. After the cluster is renamed, continue with these instructions.

8.3 Setting Up JD Edwards EnterpriseOne on a Microsoft Windows 2008 Server Cluster

This section explains how to set up JD Edwards EnterpriseOne on a Microsoft Windows 2008 Server Cluster. The example in this section pertains to a two node cluster configuration.

Note: JD Edwards EnterpriseOne executables (such as runube) running from a command line on a Microsoft Windows Cluster server node on Microsoft Windows 2008 and later releases, return the node name instead of the cluster name for the enterprise server name. This happens because the executable is not a defined resource in Microsoft Windows Cluster Services.

This creates an issue when trying to retrieve the job in JD Edwards EnterpriseOne. To resolve this problem, add the following setting to the jde.ini file of the enterprise server. You can add this setting to any location, but it is recommended that you add it after the [DEBUG] section:

[CLUSTER]

PrimaryNode=Cluster Name

To set up JD Edwards EnterpriseOne on a Microsoft Windows 2008 Server Cluster:

1. Consult the JD Edwards EnterpriseOne MTRs to determine the appropriate Microsoft Windows OS level. Install the OS on each node. Do not use domain controllers for a JD Edwards EnterpriseOne cluster configuration.
2. Follow the appropriate Microsoft documentation for specific instructions on activating Microsoft Server Cluster Software.
3. Install JD Edwards EnterpriseOne on a shared disk on the cluster. All nodes will share a single copy of JD Edwards EnterpriseOne. The name used for the JD Edwards EnterpriseOne enterprise server will be the virtual name created in the Prerequisites section.
4. Open Failover Cluster Management from Administrative Tools.
5. Right-click the cluster name and select "Configure a Service or Application." If the Before you Begin page appears, click ='Next. On the "Select Service or Application" screen, select Generic Service from the list.
6. On the Select Service screen, select JD Edwards EnterpriseOne network service, and then click Next.
7. On the Client Access Point screen, change the Name field to the JD Edwards EnterpriseOne virtual name created earlier.
8. On the Select Storage screen, select the drive shown, which is the disk where JD Edwards EnterpriseOne is installed.
9. On the Replicate Registry Settings screen, click Next.
10. The next screen is the Confirmation screen. Review your configuration for errors and warnings. Make any necessary changes or if everything looks good, select Next.
11. When the Summary screen appears and indicates the configuration was a success, click Finish.

12. Using Failover Cluster Manager, bring the JD Edwards EnterpriseOne Service online. The JD Edwards EnterpriseOne service has three parts the Server Name, the Disk Drives, and Other Resources, which is the JD Edwards EnterpriseOne Network service. All three parts must be online.
13. Run porttest. If porttest is successful, then you can install the service on secondary nodes.
14. Using Failover Cluster Manager, bring only the JD Edwards EnterpriseOne Network service offline; leave the Server Name and Disk Drives online. Move the JD Edwards EnterpriseOne service to the secondary node.
15. On the secondary node, open a command prompt. Change the directory to the JD Edwards EnterpriseOne system\bin32 directory. Install the JD Edwards EnterpriseOne service by typing jdesnet –i in the command prompt. This will install the JD Edwards EnterpriseOne service.
16. Using Failover Cluster Manager, bring the JD Edwards EnterpriseOne Network service online.
17. Run porttest.
18. Repeat steps 14-17 for each additional node in the cluster where JD Edwards EnterpriseOne will run.

8.4 Setting up EnterpriseOne on a Microsoft Windows Server 2008 R2 Failover Cluster

This section explains how to set up JD Edwards EnterpriseOne on a Microsoft Windows Server 2008 R2 Failover Cluster. The example in this section describes a two node cluster configuration.

Note: JD Edwards EnterpriseOne executables (such as runube) running from a command line on a Microsoft Windows Cluster server node on Microsoft Windows 2008 and later releases, return the node name instead of the cluster name for the enterprise server name. This happens because the executable is not a defined resource in Microsoft Windows Cluster Services. This creates an issue when trying to retrieve the job in JD Edwards EnterpriseOne.

To resolve this problem, add the following setting to the jde.ini file of the enterprise server. You can add this setting to any location, but it is recommended that you add it after the [DEBUG] section:

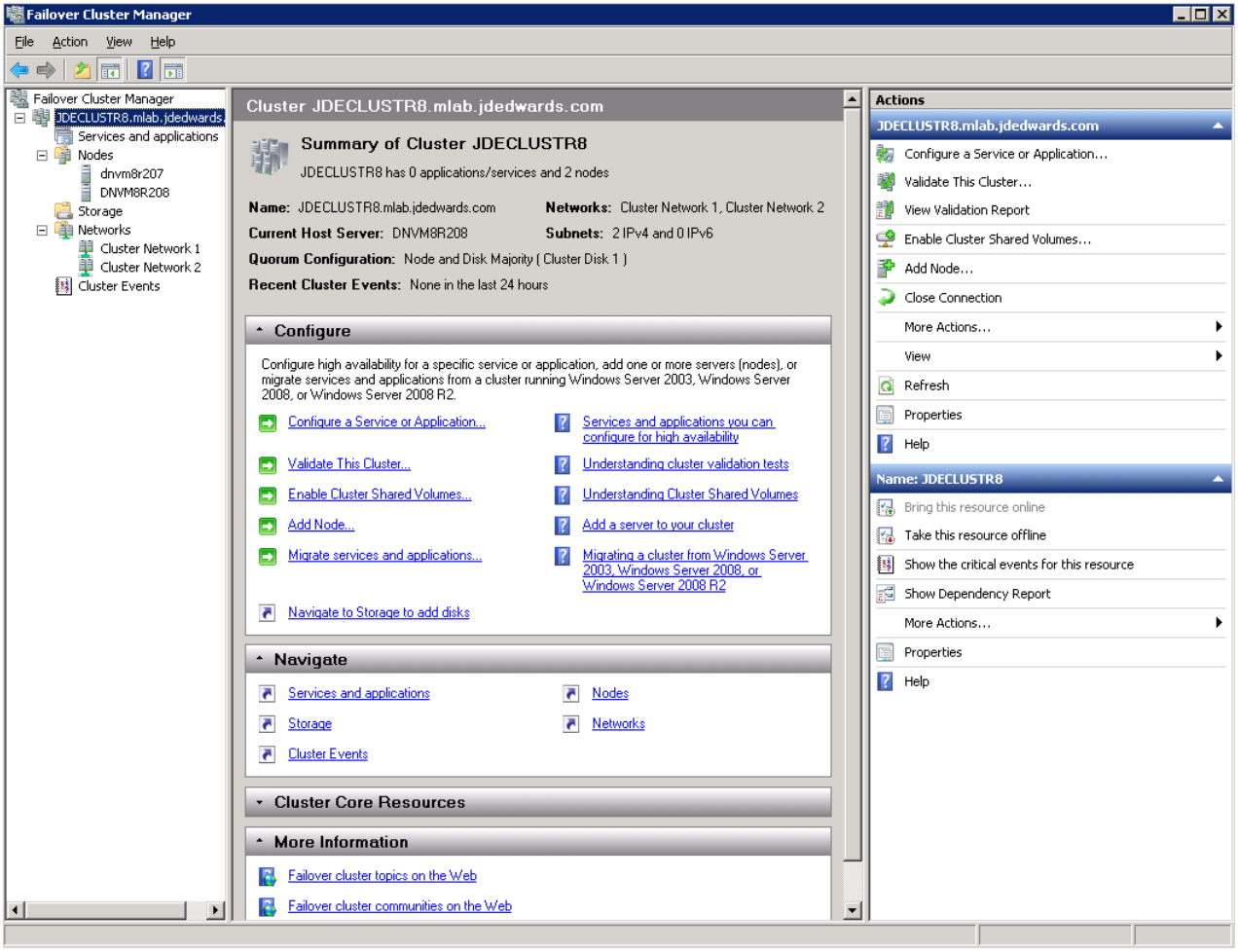
```
[CLUSTER]
PrimaryNode=Cluster Name
```

To set up JD Edwards EnterpriseOne on a Microsoft Windows 2008 R2 Server Failover Cluster:

1. Consult the JD Edwards EnterpriseOne MTRs to determine the appropriate Microsoft Windows OS level. Install the OS on each node. Do not use domain controllers for a JD Edwards EnterpriseOne cluster configuration.
2. Follow the appropriate Microsoft documentation for specific instructions on activating Microsoft Server Cluster Software.
3. Install JD Edwards EnterpriseOne on a shared disk on one node in the cluster. All nodes will share a single copy of JD Edwards EnterpriseOne. The name used for

the JD Edwards EnterpriseOne enterprise server will be the virtual name created in the Prerequisites section.

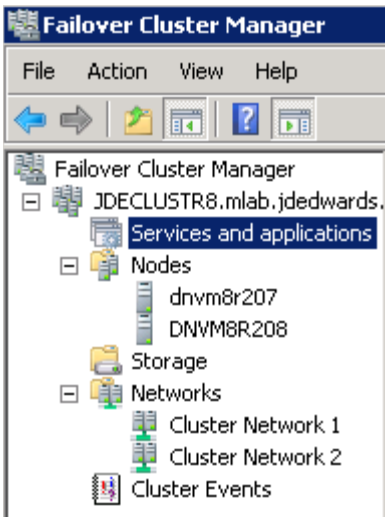
4. Open Failover Cluster Management from Administrative Tools. Click Start /Administrative Tools / Failover Cluster Management. Start the Failover Cluster Management on the node that owns the shared disk used by EnterpriseOne Enterprise Server.



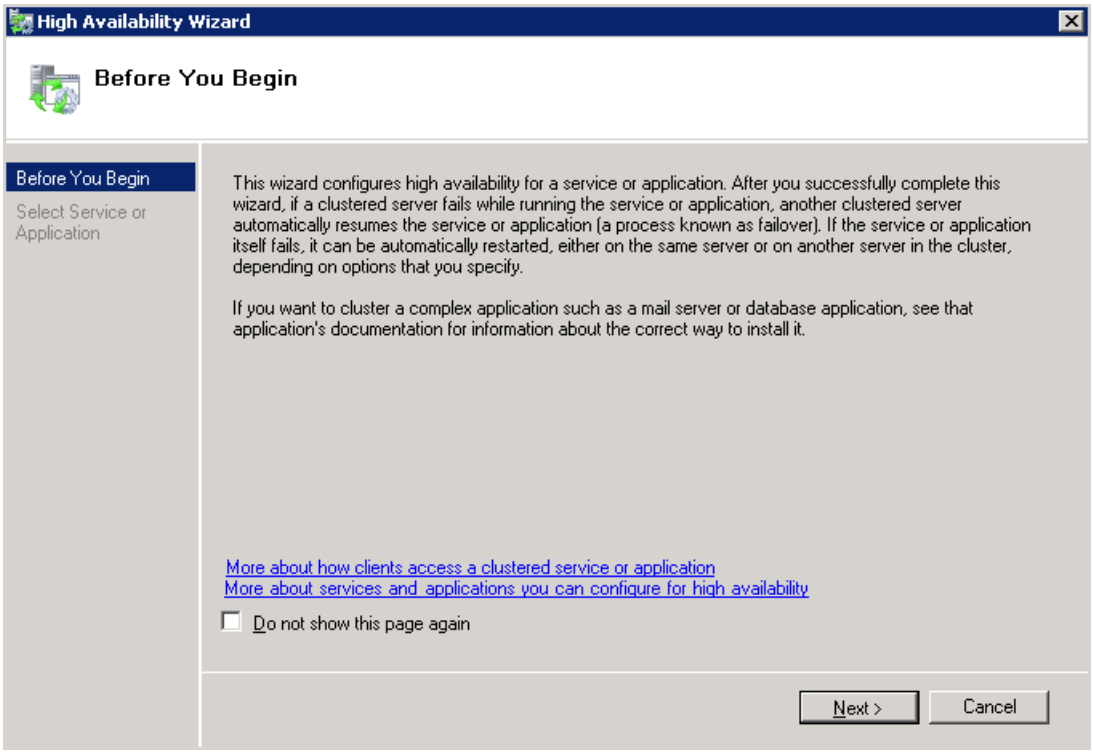
In the example above, the cluster JDECLUSTR8 has been created with two nodes: DNVM8R208 and DNVM8R207.

8.4.1 Installing EnterpriseOne 8.98 JDENET service

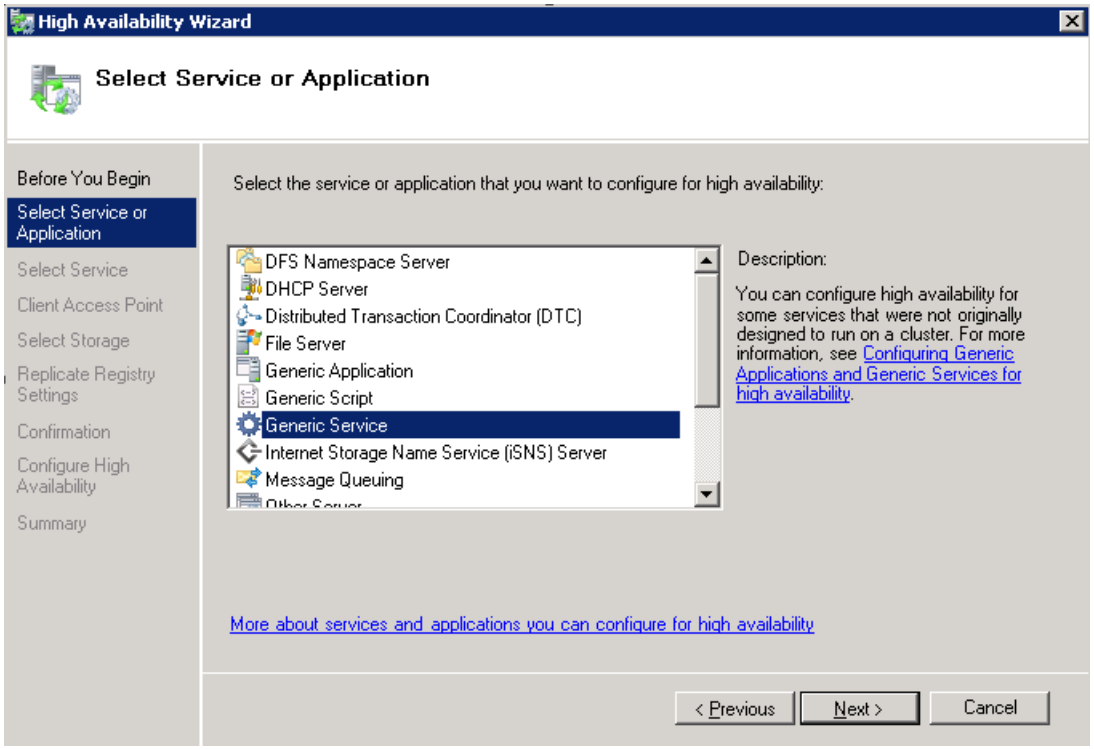
To install an EnterpriseOne 8.98 JDENET service:



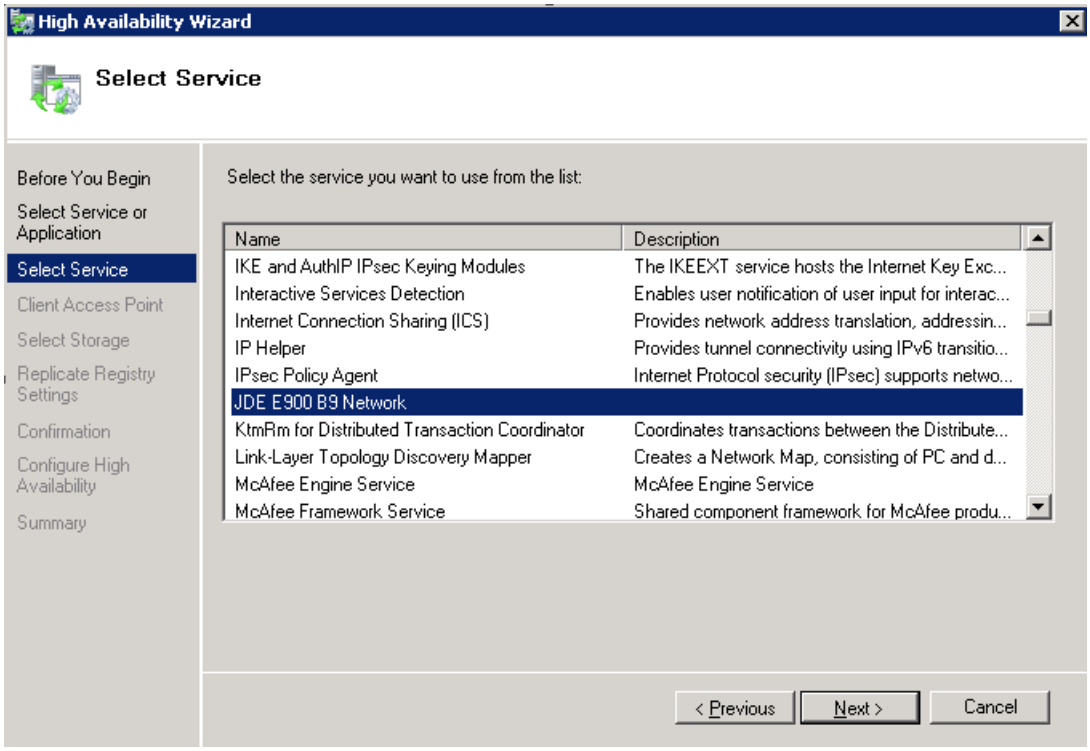
1. Highlight Services and Applications under the cluster name, right click and select Configure a Service or Application.



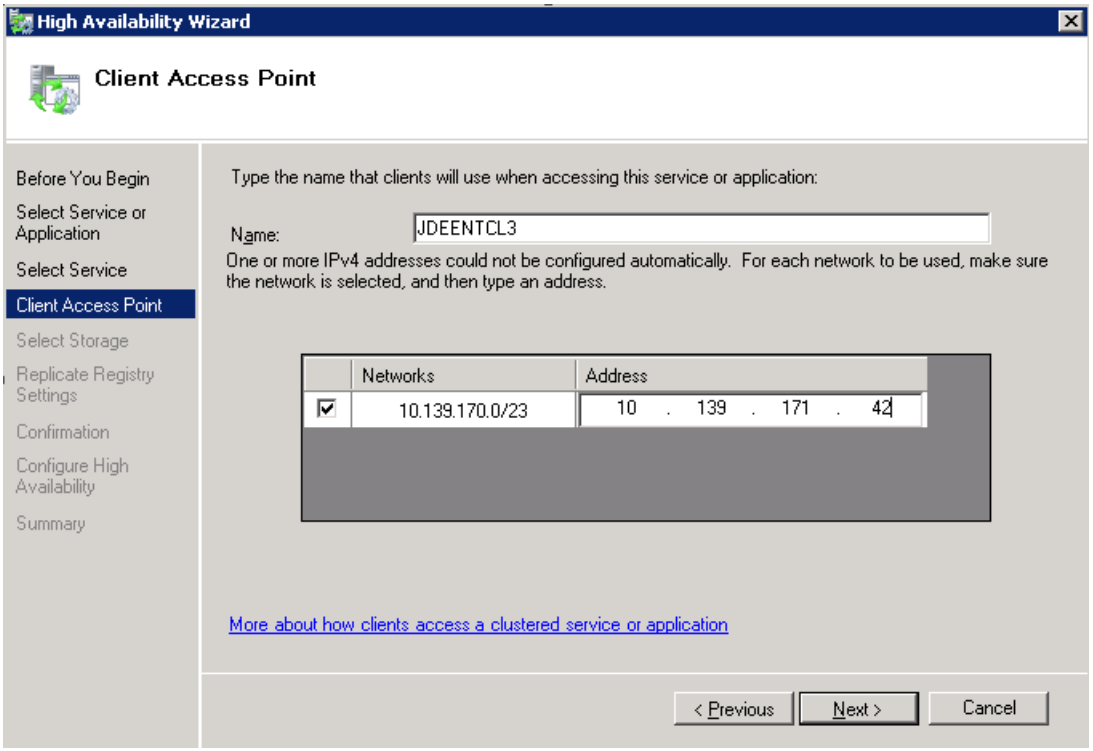
2. Click Next.



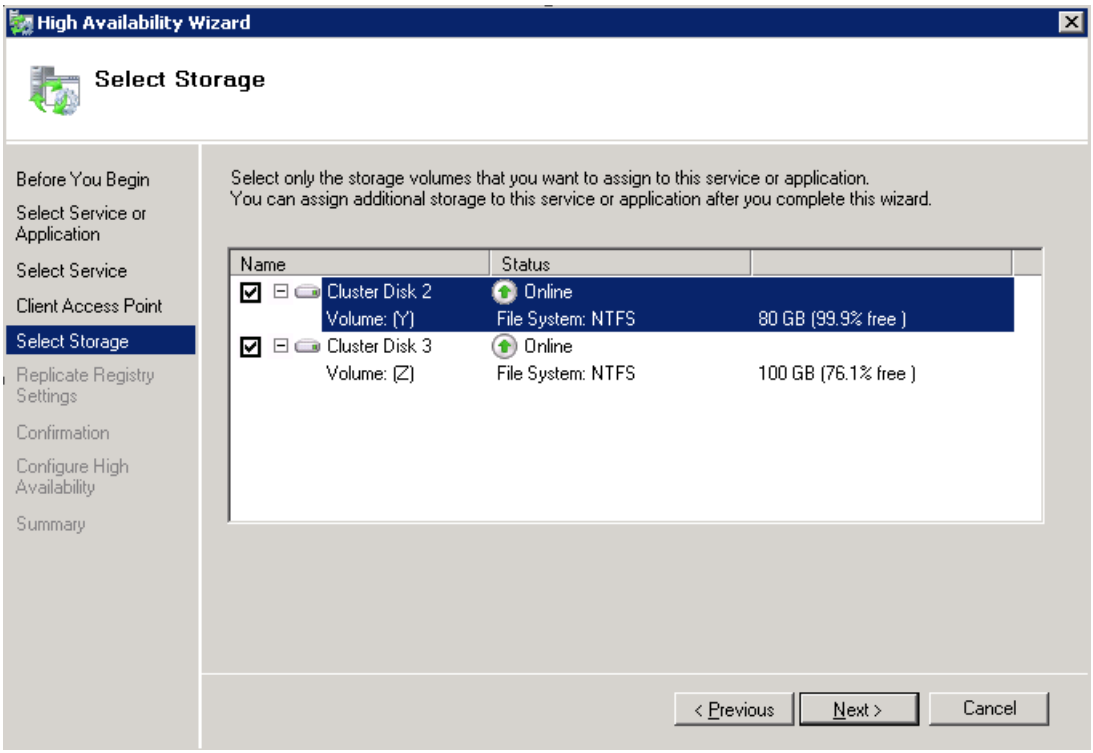
3. Select Generic Service. Click Next.



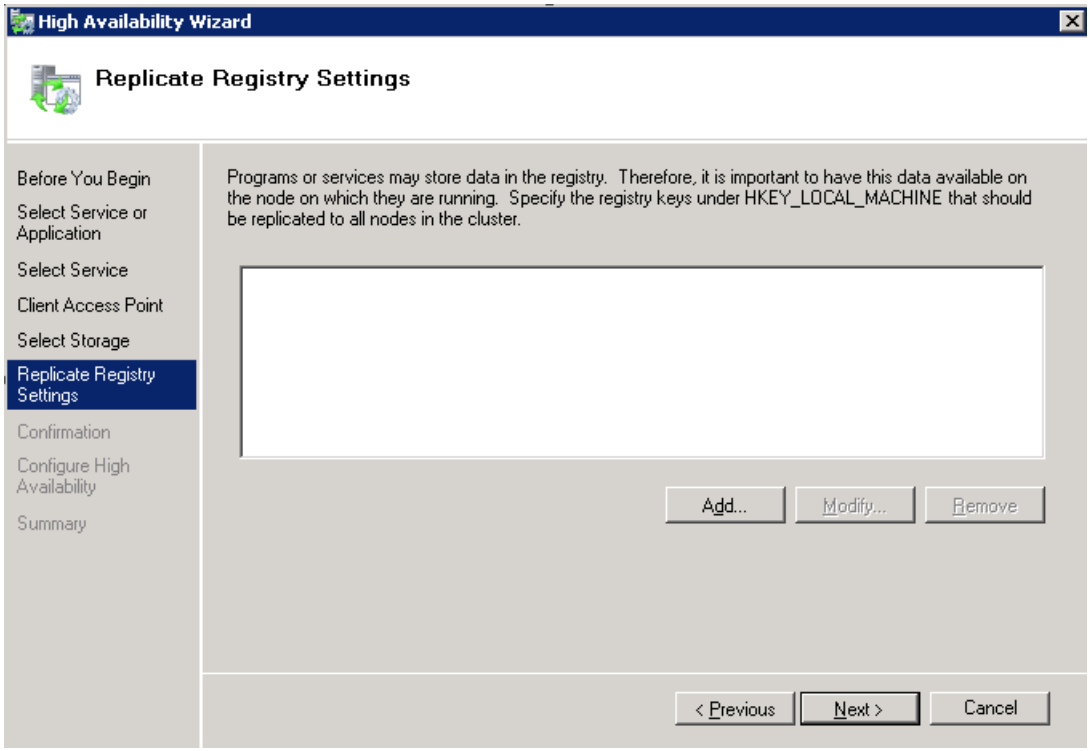
4. Scroll down and select the JDE Network Service. Click Next.



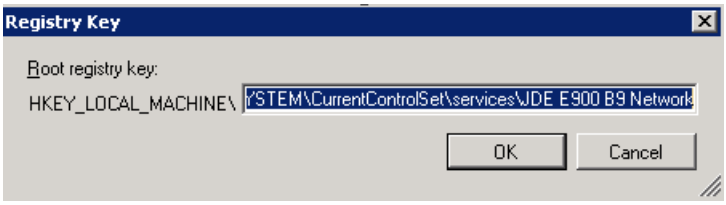
5. Change the Name to the virtual name you have created for EnterpriseOne and change the IP address so that it reflects the IP address of the virtual name you are using. Select Next.



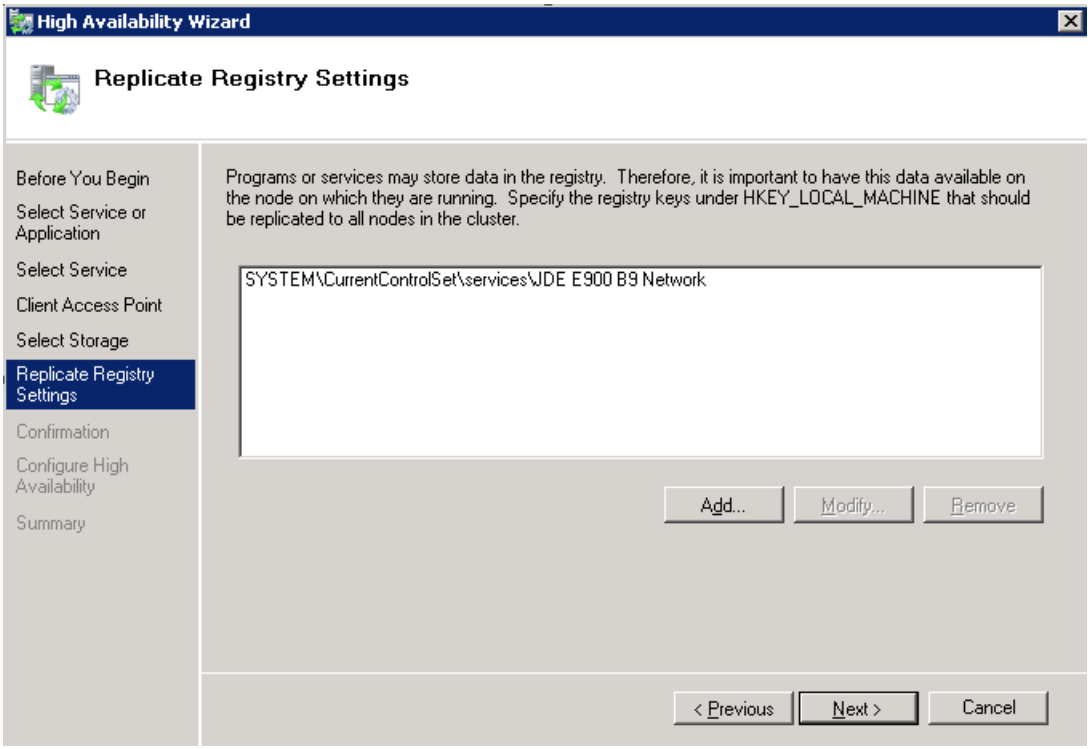
6. Select the disk where EntepriseOne is installed, and then select Next.



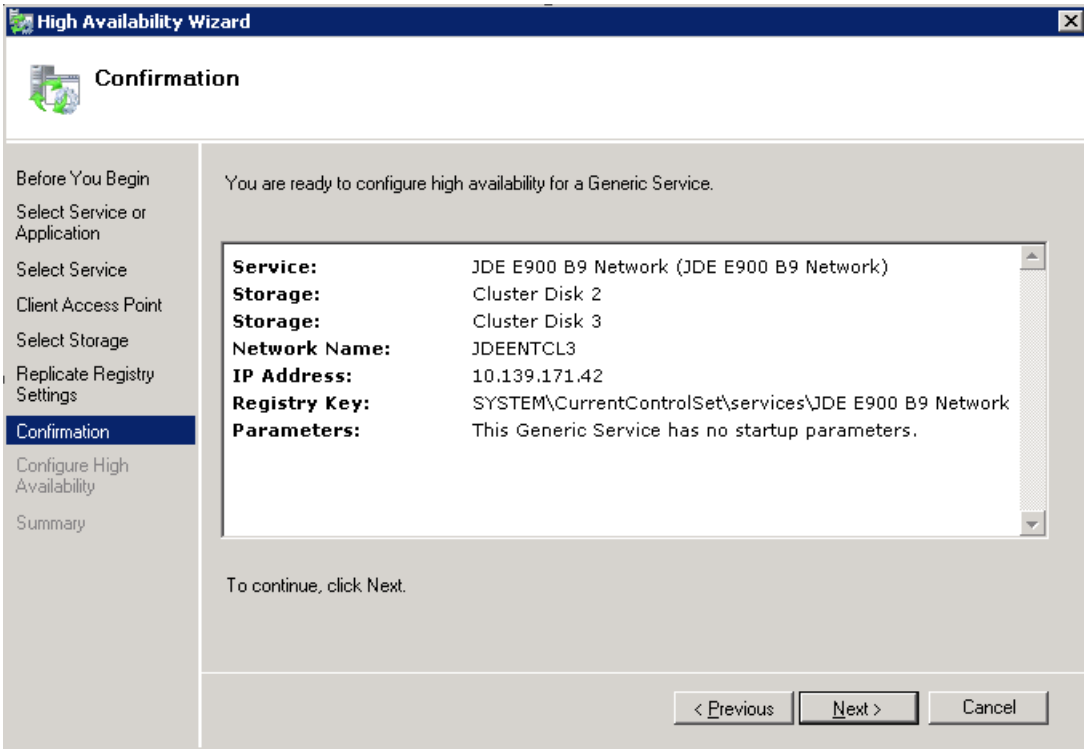
7. Select Add.



8. Add the registry service entry for JDENET (HKEY_LOCAL_MACHINE).
9. Click OK.

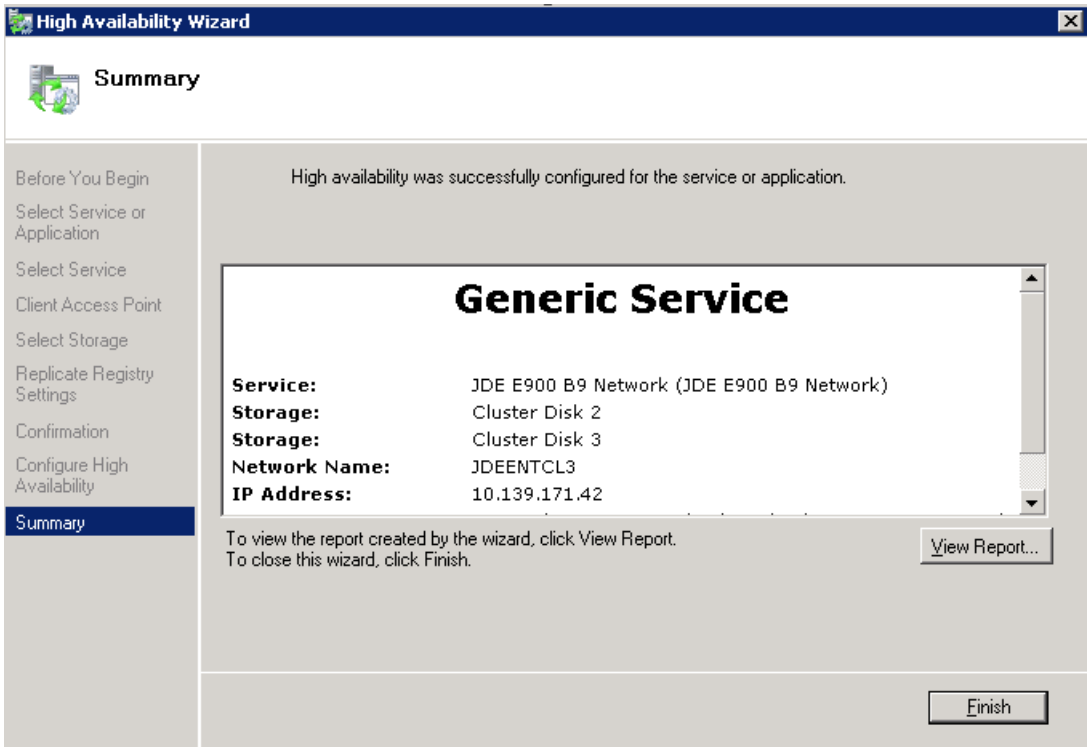


10. Click Next.

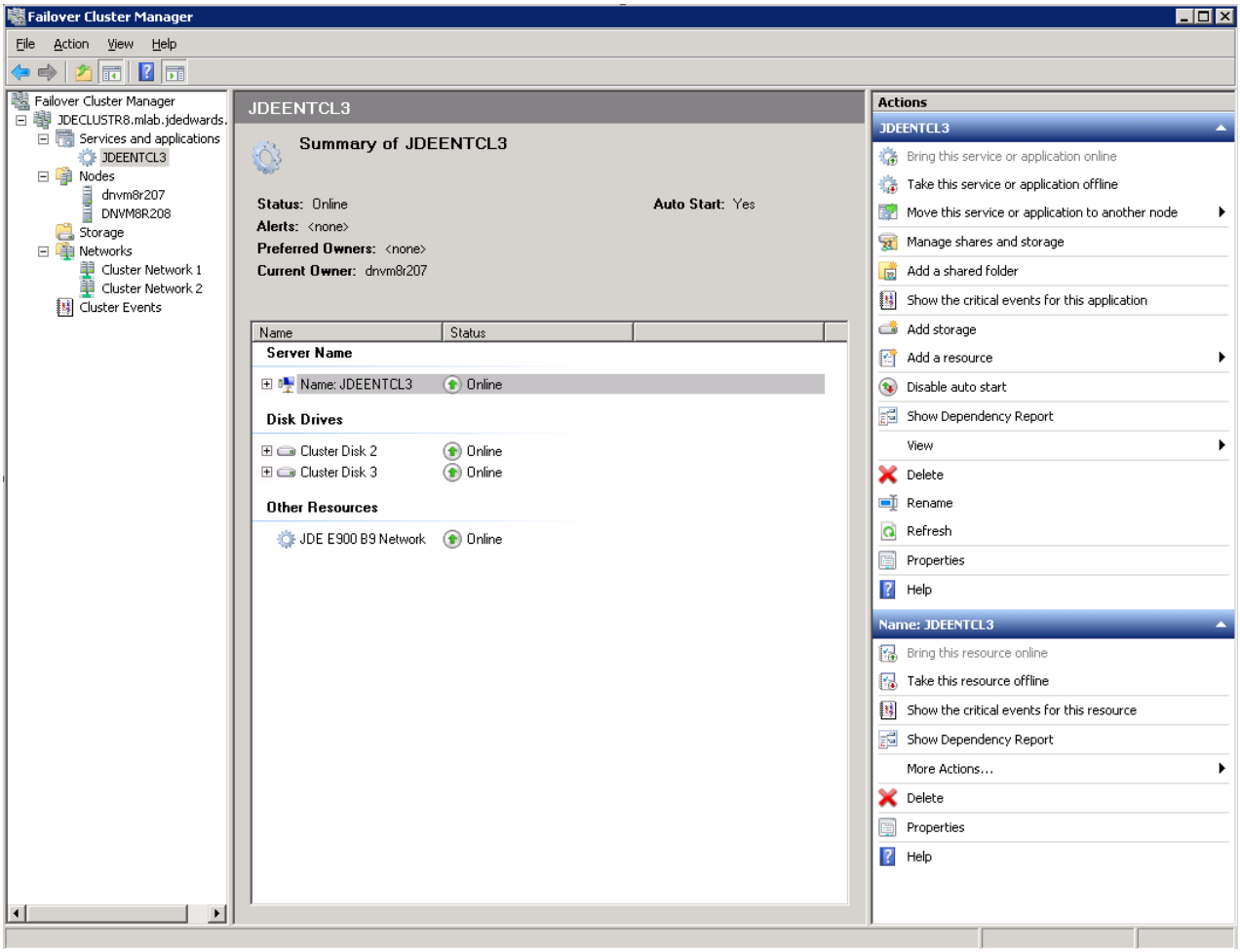


11. Click Next.

12. Configure the High Availability.



13. Click Finish.

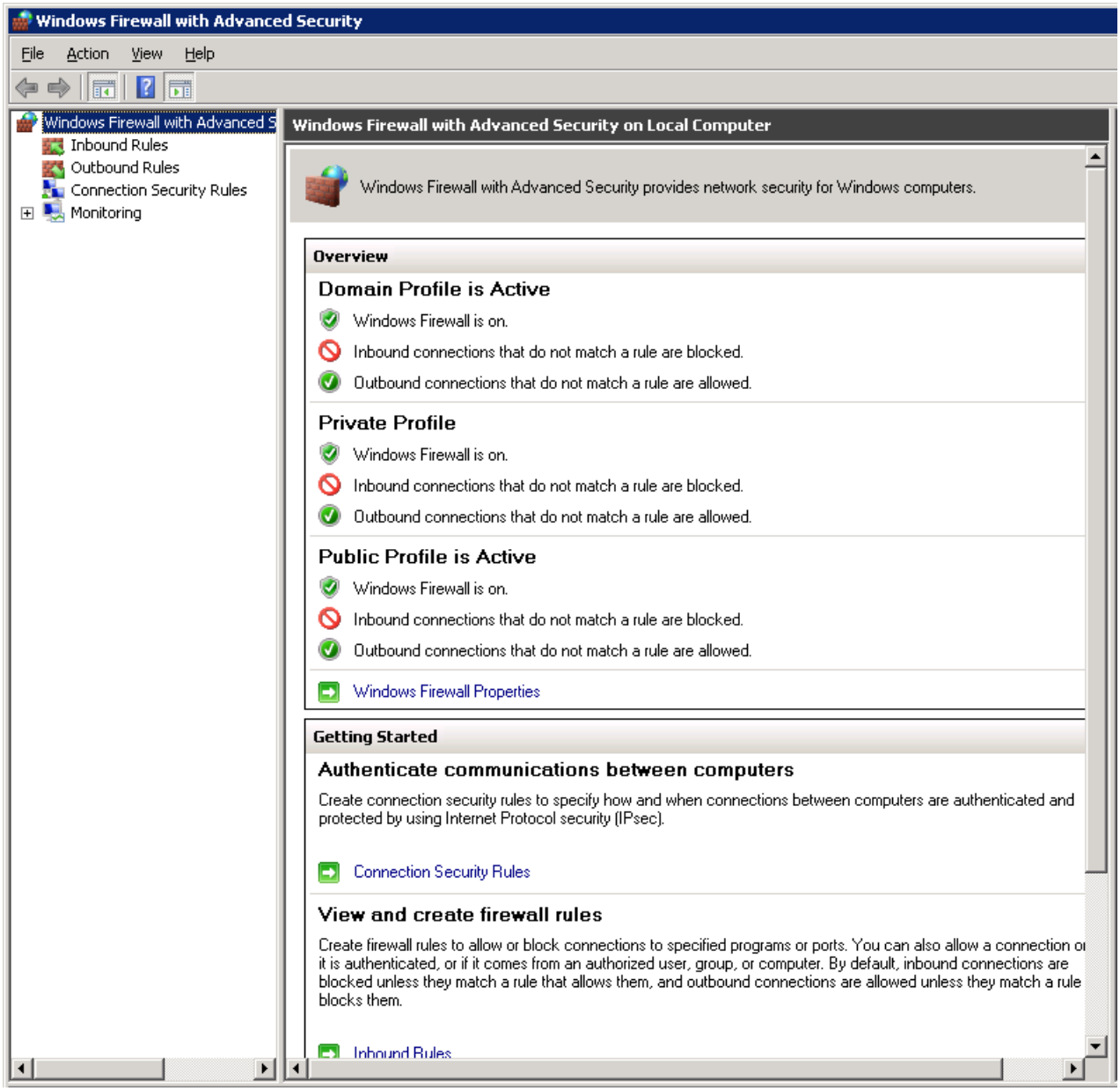


14. The configuration steps will follow, when prompted click Finish.

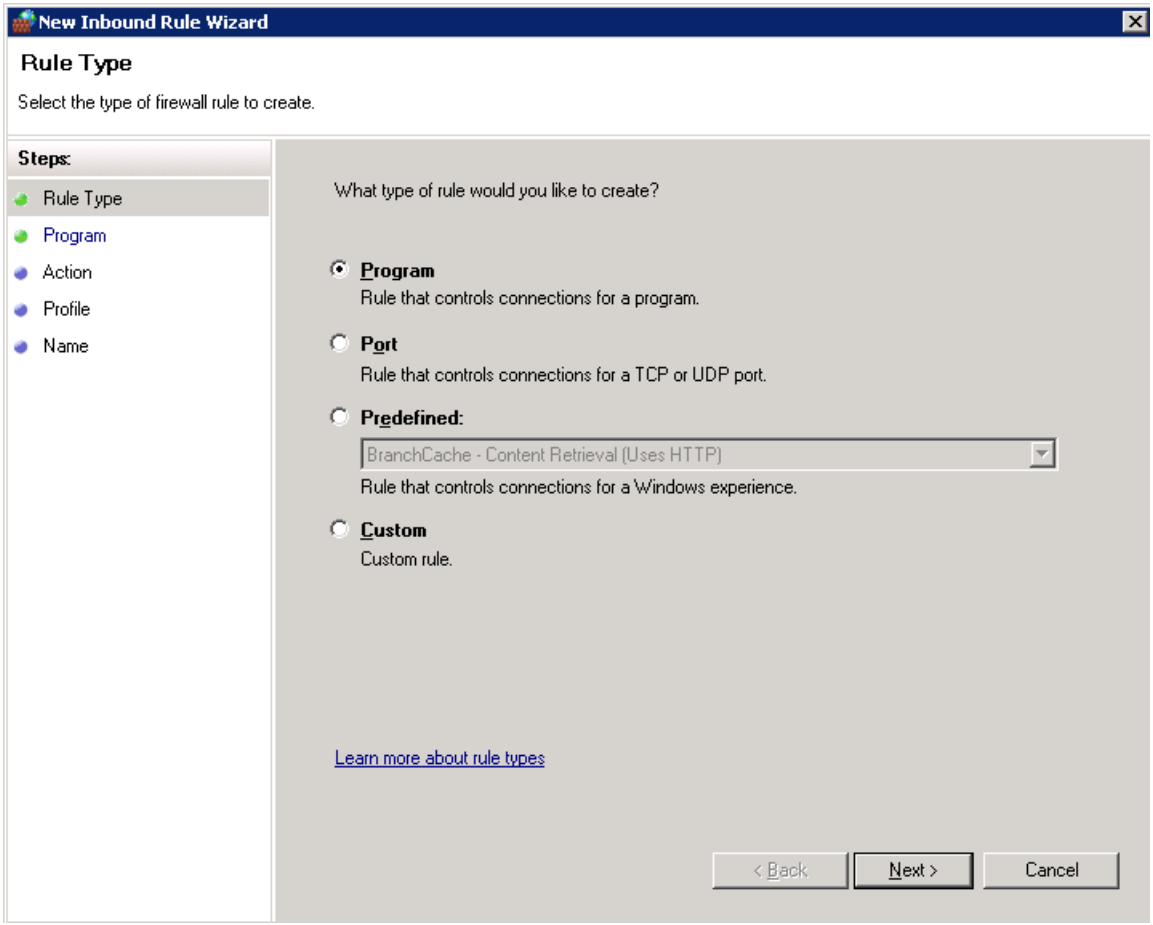
The JDE net service should be online.

Firewall Requirements

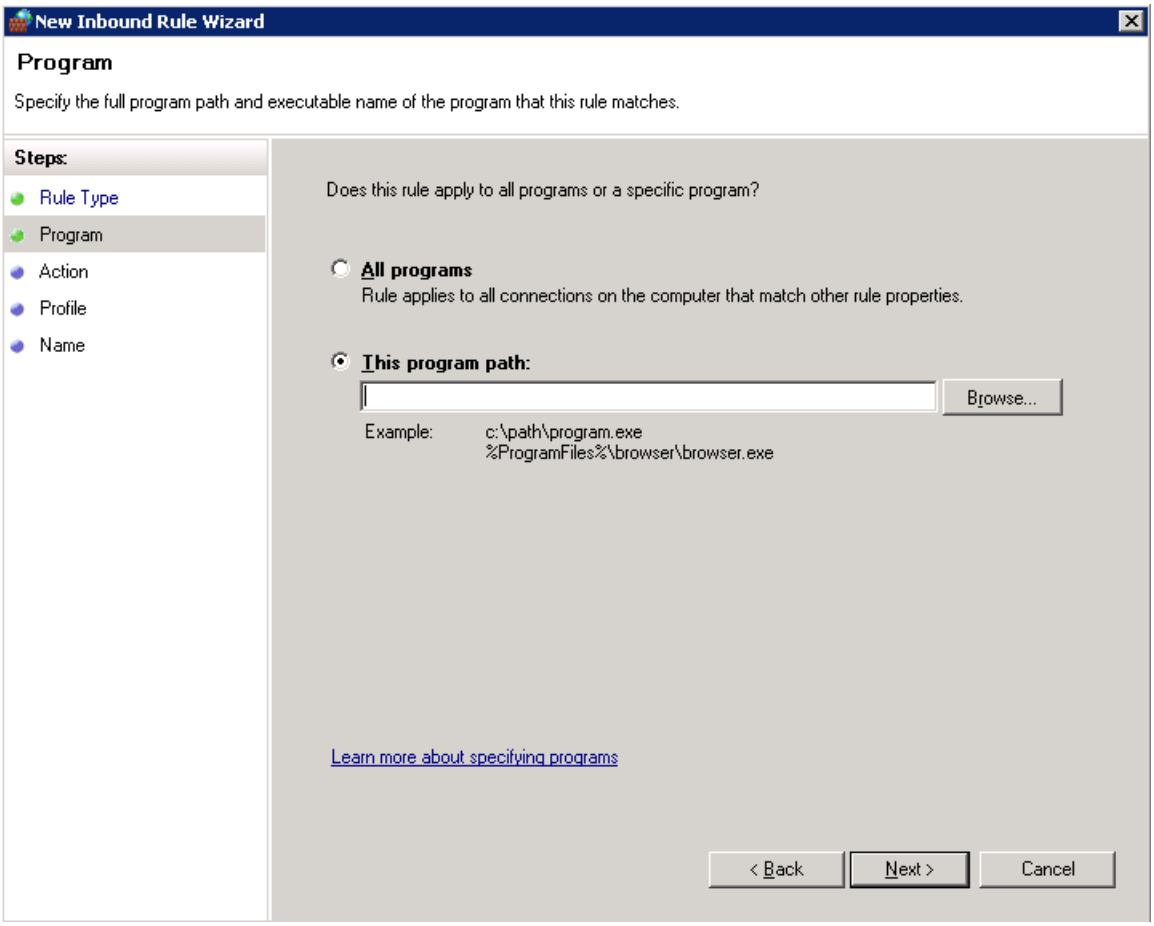
If Windows Firewall is turned on for Domain Profiles you will need to add jdesnet.exe and jdenet_n.exe executables to the Windows Firewall on all nodes. On the node that is active with EnterpriseOne running, go to Start / All Programs / Windows Firewall with Advanced Security.



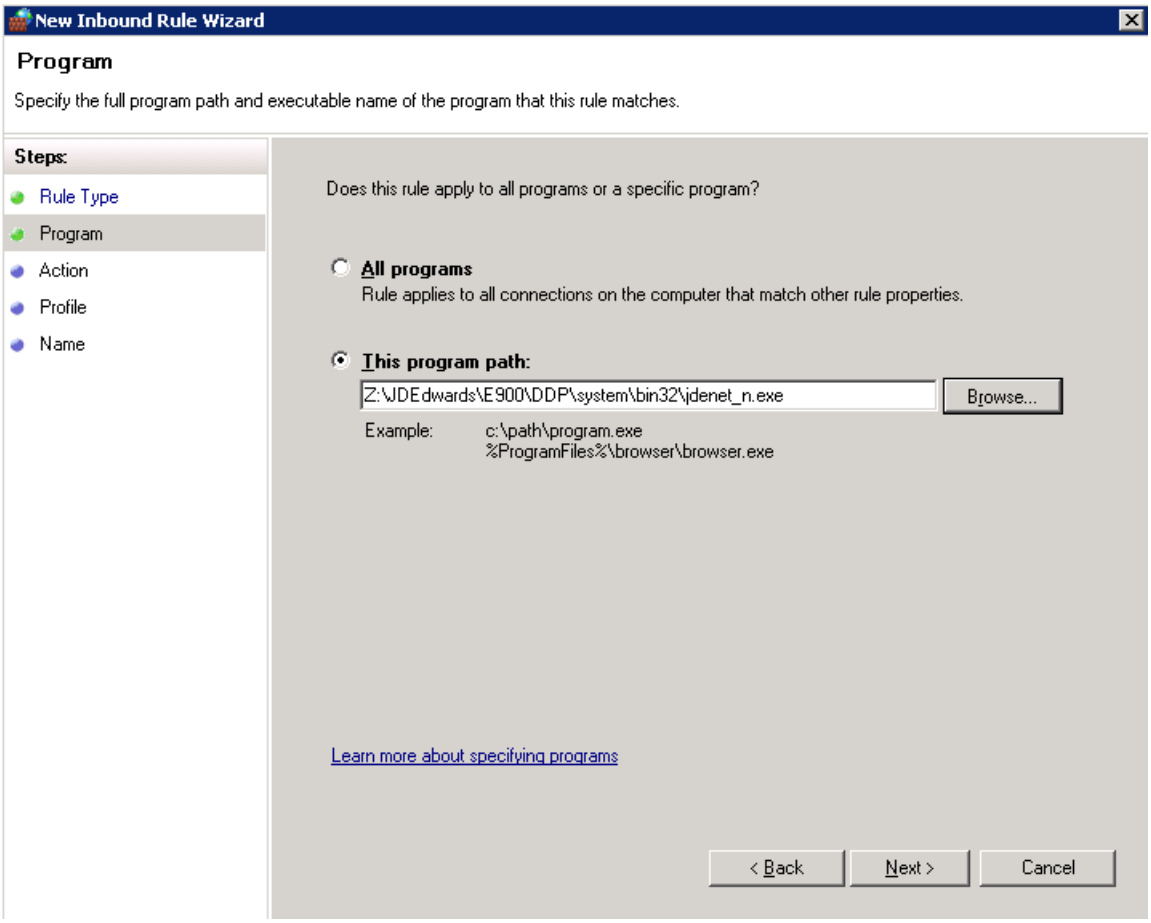
1. Select InBound rules. Right click and select New Rule.



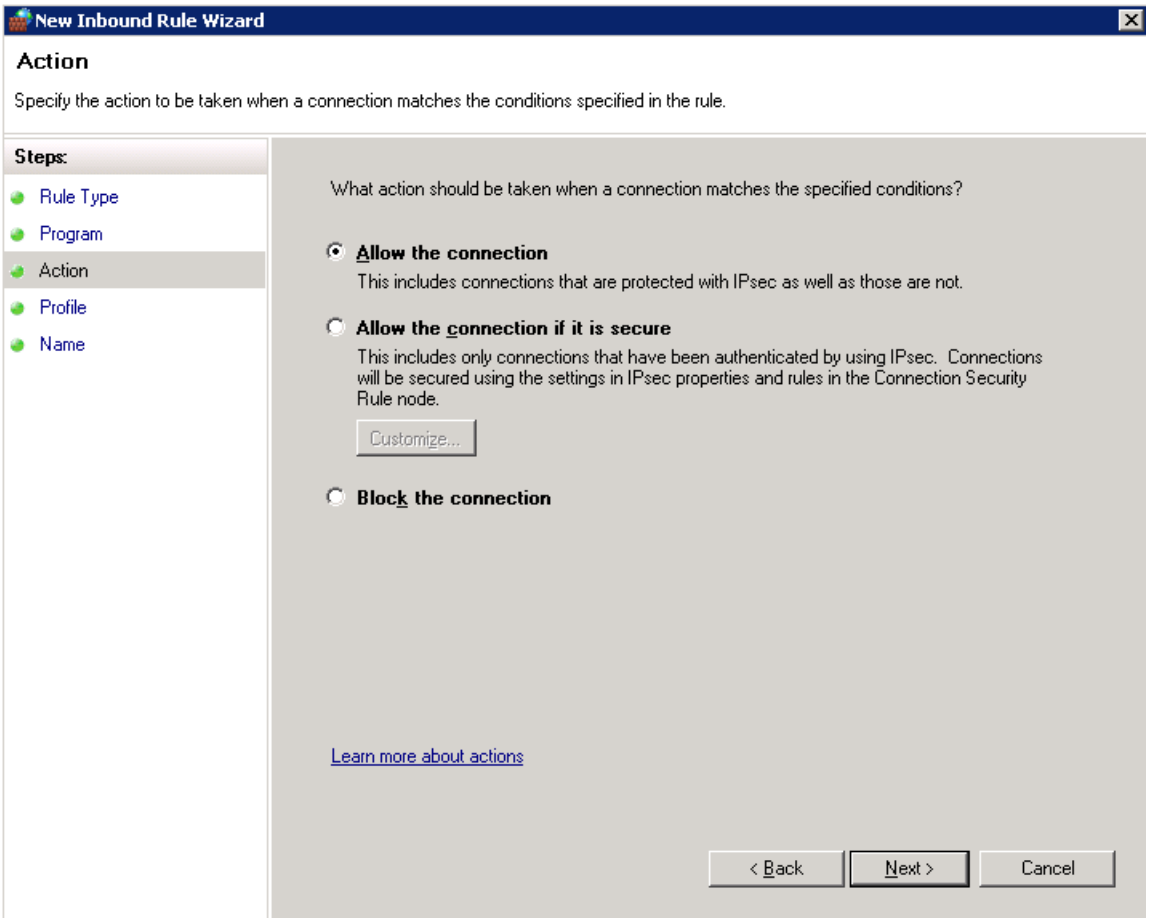
2. Select the Program radio button.
3. Select Next.



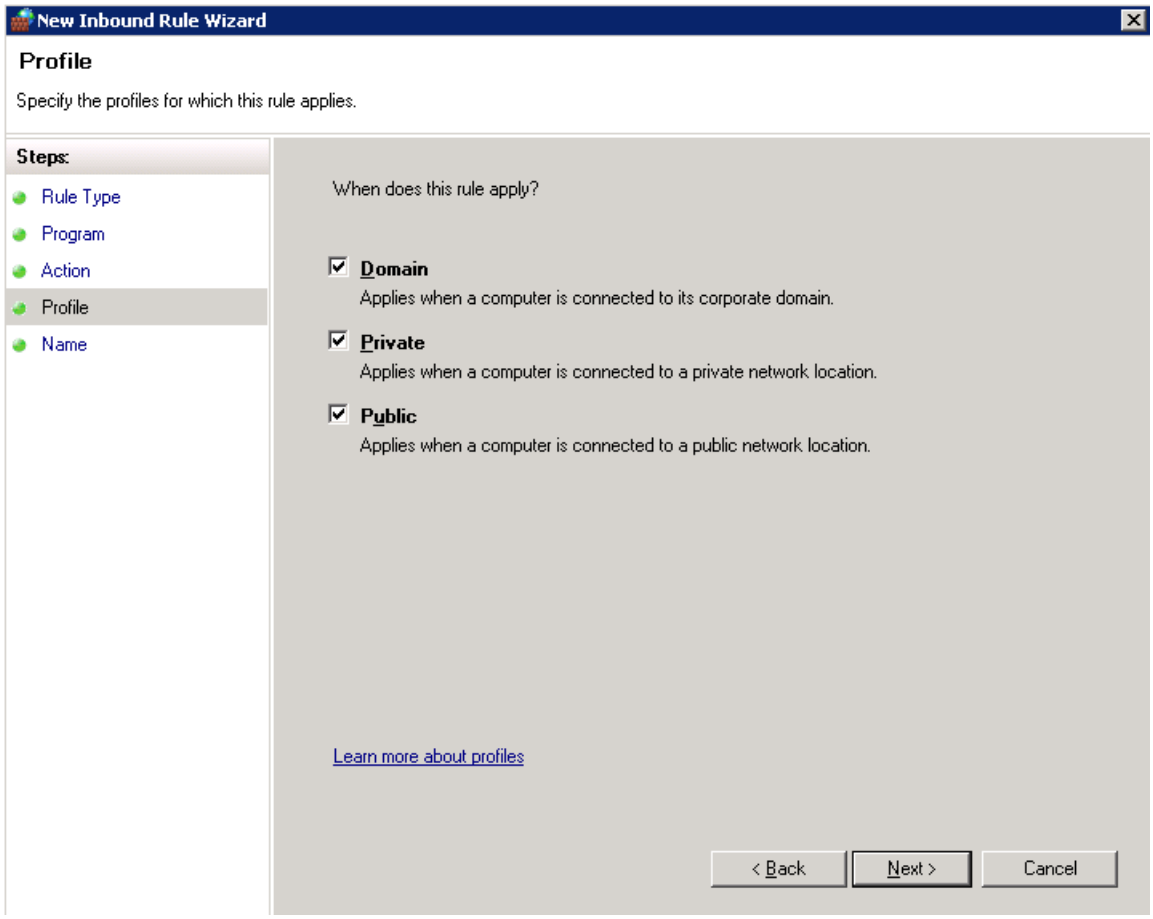
4. Select This Program Path. Drill down to the JDEDWARDS/E900/DDP/System/bin directory and select jdenet_n.exe.



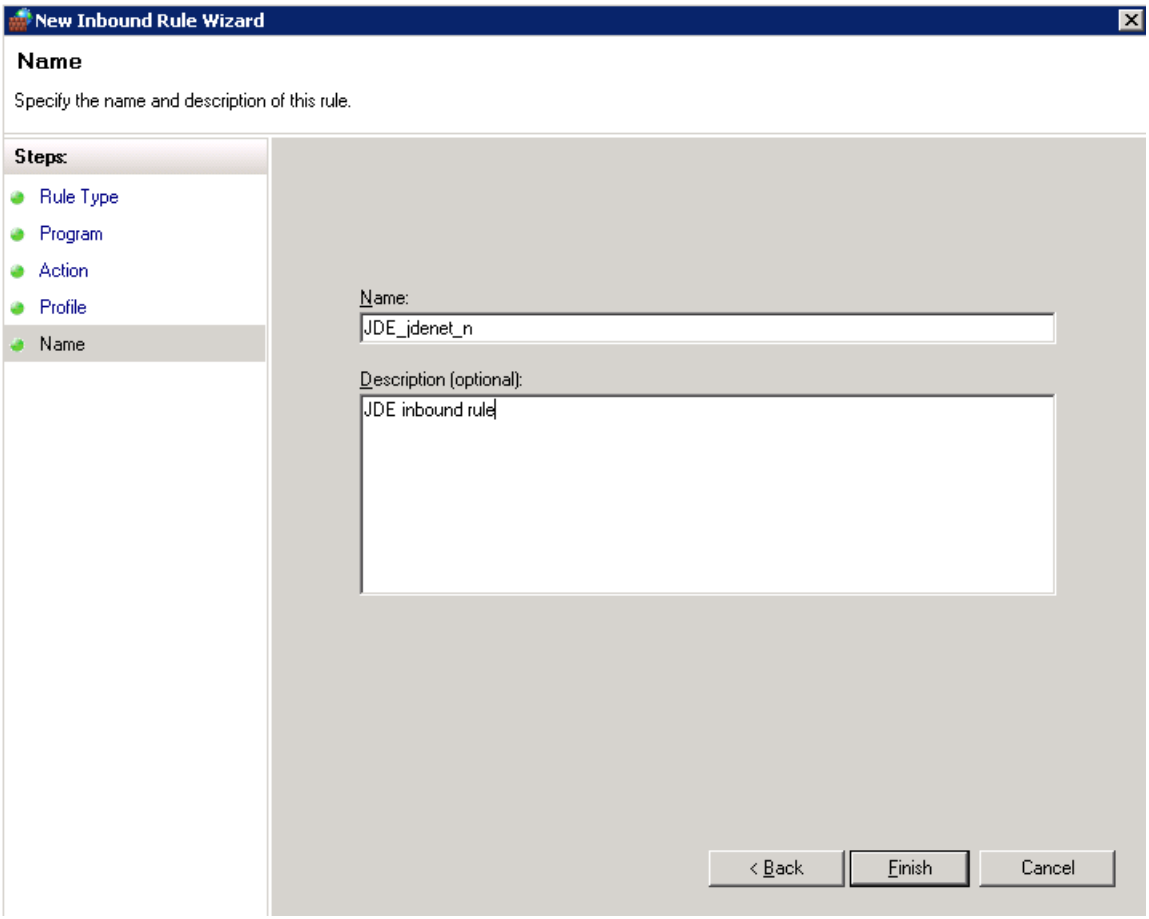
5. Select Next.



6. Select the Allow the connection radio button.
7. Select Next.



8. All the radio buttons should be selected, Domain, Private and Public. If this does meet your company's security requirements you may have to make adjustments.
9. Select Next.



10. Give your rule a name, in the example JDE_jdenet_n is used, add a description.
11. Select Finish.
12. Repeat steps 1 through 11 and select jdesnet.exe instead of jdenet_n.exe.
13. Repeat steps 1 through 12 for each node in your cluster. The node you are working on must be the active node for EnterpriseOne in order to access the shared disk where the EnterpriseOne software resides.

Troubleshooting

1. If you are logged on to an EnterpriseOne web or fat client at the time of a cluster failover it may be necessary to back out of the application or even log off and log on to the EnterpriseOne client to reconnect.
2. Do Not use EnterpriseOne Server Manager to bring up or down the EnterpriseOne jdenet service, use the Failover Cluster Manager.

9

Administering JD Edwards EnterpriseOne on an IBM i Cluster

This chapter contains the following topics:

9.1 Understanding IBM i Clustering

IBM i clustering is a platform-specific software solution that provides users with continuous access to Oracle's JD Edwards EnterpriseOne even when the primary server becomes unavailable. You can switchover from a primary server to a backup server either automatically or manually.

An IBM i cluster consists of more than one node, although not necessarily more than one physical machine. For example, you can use logical partitioning to represent several nodes on a single machine, or you can maintain more than one IBM i machine, each of which represents a cluster node.

A cluster name is associated with a floating or takeover IP address. Each node in the cluster has an IP address associated with a TCP/IP interface. At any time, only one node in the cluster has the interface activated. That node is the primary node, on which JD Edwards EnterpriseOne services are running. All other nodes in the cluster are designated as backup nodes, and the TCP/IP interface is inactive.

IBM i nodes participating in a single cluster use the cluster software to do these tasks:

- Replicate object and data changes from the primary node to backup nodes so that any backup node can assume the primary server role when an interruption in service occurs.

- Read a specifier file that identifies the objects and data that must be replicated and the locations of those specified objects and data.
- Use a backup node to monitor the primary node for availability.
- If the primary node becomes unavailable, use an exit program to activate the floating IF address on the first backup node associated with the cluster name.
- Restart JD Edwards EnterpriseOne on the first backup node, thus making it the primary node.
- Queue changes to objects and data so that an original primary node can be updated once it becomes available for service.

You can use JD Edwards EnterpriseOne and IBM i clustering software to support two-tier configurations (IBM i functioning as data server only), virtual three-tier configurations (IBM i functioning as both logic server and data server), and three-tier server configurations (separate IBM i machines functioning as a logic server and a data server).

IBM recognizes three partners who market the IBM i clustering solution:

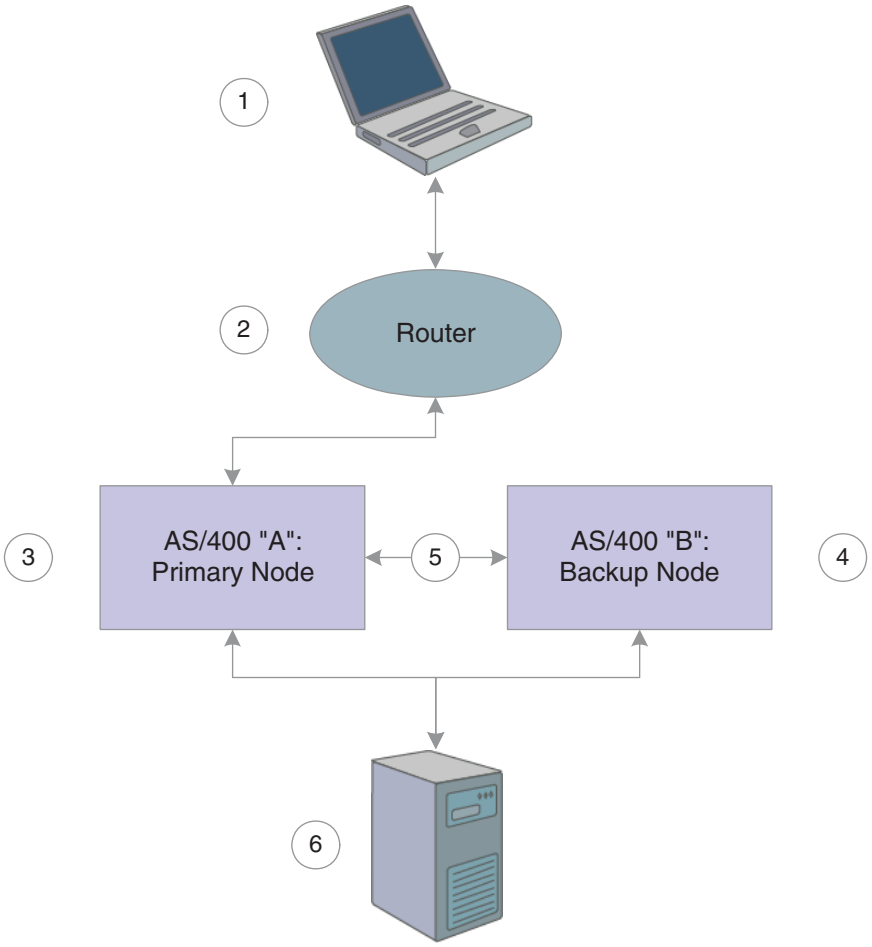
- DataMirror Corporation
- Lakeview Technologies
- Vision Solutions

For more details on these vendor solutions, consult the CNC specialist.

9.1.1 IBM i - JD Edwards EnterpriseOne Architecture with Clustering

This graphic illustrates the main components of a virtual three-tier clustering setup:

Figure 9–1 IBM i Clustering Architecture — normal operation

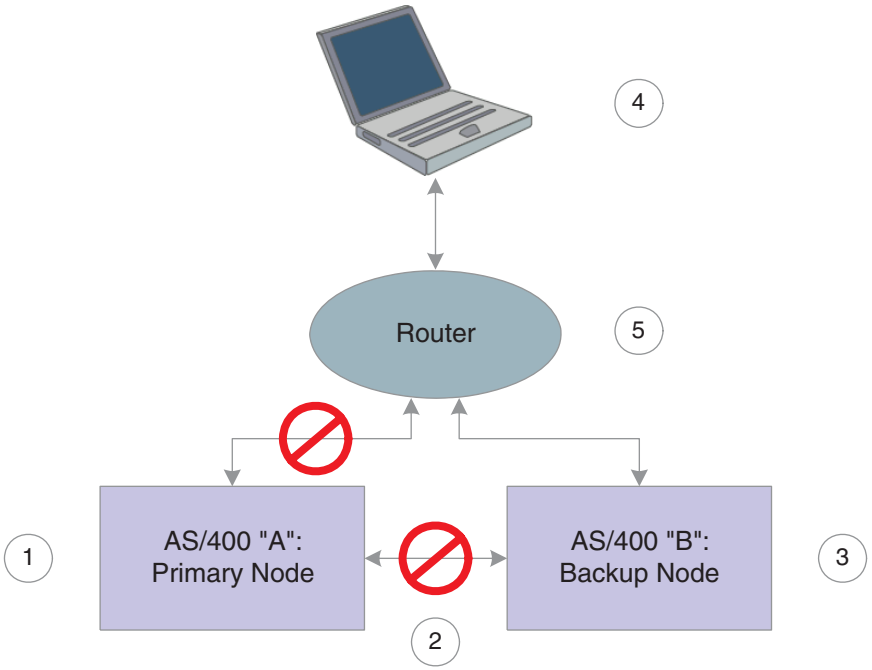


The roles of each component in the cluster are as follows:

- Workstation (1) attempts to access the primary node.
- Router (2) directs workstation access request to the primary node using the cluster name and the associated floating IP address.
- Primary node (3) provides services to the workstation and sends changes to objects specified in the specifier file to the backup node.
- Backup node (4) monitors primary node and applies changes to data and replicated objects using static IP address (5).
- Deployment server (6) updates both primary and backup nodes with changes to application objects using package deployment.

This graphic illustrates the role of each component in the clustering environment when a failover scenario occurs:

Figure 9–2 IBM i Clustering Architecture — failover scenario

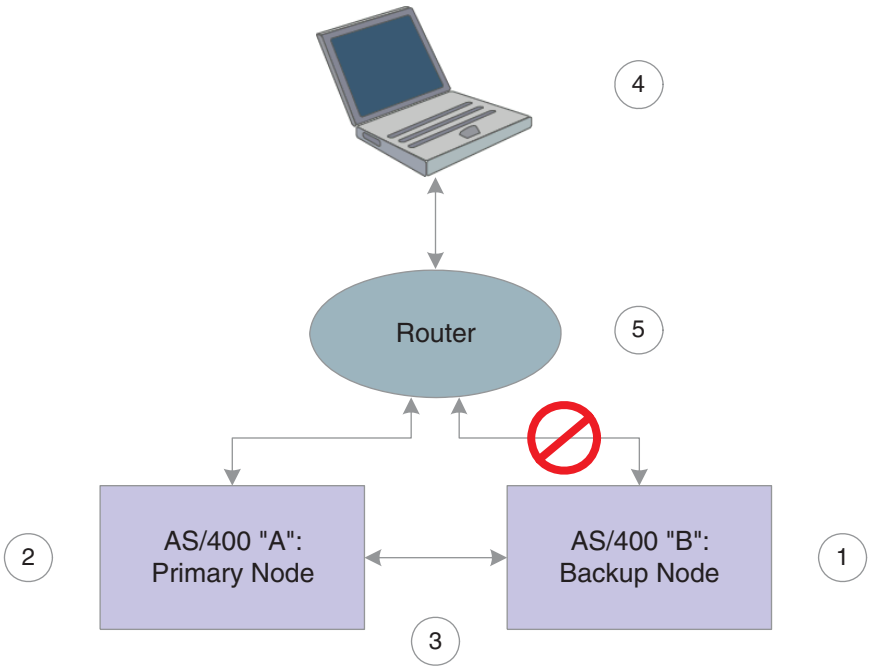


The roles of each component in the cluster in the failover scenario are as follows:

- Ex-primary node (1) ends communication with the workstation and the backup node.
- Object and data changes using static IP address end, as does monitoring of nodes(2).
- Clustering software on IBM i B detects failure of IBM i A, activates failover TCP /IP interface, applies queued changes to objects listed in clustering specifier file, and makes JD Edwards EnterpriseOne services available on IBM i B (3).
- Workstation (4) requests server services and waits for a response for a length of time specified in its jde.ini file. After JD Edwards EnterpriseOne notifies the workstation of a lost server connection, the workstation attempts to reconnect.
- Router (5) receives reconnection attempt from workstation and directs request to the new primary node using the cluster name and the associated floating IP address.

After the failover, the ex-primary node should eventually become available once again for service. This graphic illustrates the role of each component in the clustering environment when the return to normal operations occurs:

Figure 9–3 IBM i Clustering Architecture — return to normal



The roles of each component in the cluster in the return to normal operations are as follows:

- The clustering software stops the failover interface on IBM i B (1), ending communication with the workstation.
- The clustering software starts the failover interface on IBM i A (2) and applies any queued changes to objects listed in the clustering specifier file.
- Replication and monitoring services restart (3).
- Workstation (4) requests server services, waits for a response for a length of time specified in its jde.ini file. After JD Edwards EnterpriseOne notifies the workstation of a lost server connection, the workstation attempts to reconnect.
- Router (5) receives the reconnection attempt from workstation and directs the request to the primary node using the cluster name and the associated floating IP address.

For a full discussion of platform, hardware, and LAN and WAN configurations, consult IBM clustering documentation as well as documentation provided by each of IBM's approved clustering software partners.

9.1.2 JD Edwards EnterpriseOne Objects Used with IBM i Clustering

The JD Edwards company requires you to download the latest IBM i Clustering objects from the Update Center. This topic discusses each of the JD Edwards EnterpriseOne objects required for use with IBM i clustering software:

- Specifier file
- SETOWCLST command
- Application data areas
- Exit program

Consult the CNC specialist if you need to modify any of these files.

Specifier File

The specifier file identifies all JD Edwards EnterpriseOne objects and data that must be replicated from the primary server node to the backup node. Replication ensures that you have a backup of essential business information on the backup node should the primary node fail.

The information in the specifier file enables the creation of application and data cluster resource groups (CRGs). Application CRGs identify nodes in a cluster that can be used to run a particular program or group of programs on the enterprise server. Data CRGs represent the locations of data and objects in the cluster. With this information established, the clustering software is able to replicate the specific set of objects and data to a specific server node.

Note: Do not attempt to modify the specifier file directly. To customize the file, consult the CNC specialist.

This table summarizes the items that must be replicated for JD Edwards EnterpriseOne-IBM i clustering:

Replication Item	Files and Directories for Replication
All data used for an environment	Examples: PRODDTA/*PF PRODCTL/*PF
Object Librarian	OL900/*PF
Central Objects	COPD900/*PF
Data dictionary	DD900/*PF
PrintQueue Directory	/E900SYS/printqueue
JD Edwards EnterpriseOne jobs table (F986110)	SVM900/F986110

Note: Clustering vendor solutions shouldn't replicate JD Edwards EnterpriseOne specification files in the integrated file system (IFS).

SETOWCIST Command

You can use the SETOWCIST command to access the specifier file. This command is necessary to update the specifier file if you add, remove, or modify the name or location of a JD Edwards EnterpriseOne IBM i library or object.

Application Data Areas

A data area is an object used to communicate data such as variable values between programs within a job and between jobs. Application data areas contain information about resilient resources in the JD Edwards EnterpriseOne clustering setup. Resilient resources are objects and other information located on more than one cluster node. Important resilient resources include data and programming objects critical to running JD Edwards EnterpriseOne.

The clustering software uses JD Edwards EnterpriseOne information to create and keep track of CRGs, which identify nodes in a cluster and the types and locations of JD Edwards EnterpriseOne resilient resources.

There are two application data areas in the JD Edwards EnterpriseOne-IBM i clustering environment:

- Input application data area.
- Output application data area.

Input Application Data Area

The input data area is used to communicate information about the JD Edwards EnterpriseOne application to the IBM cluster middleware business partner. The cluster middleware uses this information to create CRGs. The input application data area will contain information about JD Edwards EnterpriseOne, its resilience information, and information about required data.

Output Application Data Area

The output data area is used by the IBM cluster middleware business partner software to track the use of the CRG it created for use in JD Edwards EnterpriseOne. The CRG identifies the nodes in a cluster used to run programs on the enterprise server and the locations of data and objects in the cluster.

9.1.3 Cluster Exit Program

The IBM i clustering software invokes the exit program, which is called CLSTR_EXIT, when a failure on the primary node requires a failover to the backup node. The program stops JD Edwards EnterpriseOne services running on the backup node until replication of all JD Edwards EnterpriseOne objects and data identified in the specifier file has been completed. After replication has completed, JD Edwards EnterpriseOne services restart.

9.1.4 Technical Considerations

To aid in the implementation of IBM i clustering software with JD Edwards EnterpriseOne, we make these suggestions:

- Represent IBM i nodes either by a logical partition or by an entire machine, depending on the system.
- Use separate pipes (LAN cards) to minimize the effects of clustering software functions on end user activities. For example, dedicate one pipe to running applications and one pipe to object/data replication and node monitoring.
- Consult IBM documentation on IBM i clustering solutions for further details on handling LAN card setup.
- Use the JD Edwards EnterpriseOne deployment server to deliver JD Edwards EnterpriseOne packages to server cluster nodes. Do not attempt to use the clustering replication processes to deploy JD Edwards EnterpriseOne packages between cluster nodes.
- Use the clustering software to replicate object/data changes made by JD Edwards EnterpriseOne users.
- Define to the specifier file on each node any changes you make to JD Edwards EnterpriseOne library names or the locations of JD Edwards EnterpriseOne-supplied objects, if those libraries and objects were listed in the specifier file provided by the JD Edwards company.

- Evaluate each JD Edwards EnterpriseOne cumulative release, electronic software update (ESU), service pack, or program temporary Fix (PTF) to determine if any changed objects are listed in the specifier file. If any are, be sure to define the changes to the file.

The setup and configuration of JD Edwards EnterpriseOne discussed in this chapter assumes a virtual three-tier setup for a two-node server cluster. The setup should be used as a general reference guide only. Many variations on the setup are possible and will be determined by the specific requirements of the organization. Consult the Cluster Middleware Vendor or CNC specialist for additional details on how to configure the IBM i system to use clustering.

9.1.5 Minimum Setup Requirements for IBM i Server Nodes

Each node in a virtual three-tier cluster configuration must be set up with these components:

- A host table and domain names server (DNS) entry with the cluster name that is associated with a floating IP address.
- An IP interface with the same floating IP address as was entered in the host table. The floating IP address is used to find the backup node when the primary node is unavailable.
- A second IP interface with a unique address. This address is used for object and data replication.
- Identical operating system releases (such as IBM i V5R2 or higher) Identical IBM i PTF levels.
- Identical versions of JD Edwards EnterpriseOne at identical service pack levels.
- Identical clustering exit programs, object specifier files, and data areas for JD Edwards EnterpriseOne.
- Identical copies of the business data objects listed in the clustering object specifier file.
- IBM i clustering software installed and configured. Download it from the Update Center.
- A server jde.ini file with a [CLUSTER] section that defines the cluster name.

Note: This requirements list applies only to a virtual three-tier configuration. With additional nodes, you can have different clustering exit programs, object specifier files, data areas, and so on.

9.2 Running the SETOWCLST Command

To run the SETOWCLST Command:

1. Sign onto the IBM i system and add the JD Edwards EnterpriseOne system library to the library list.
2. Enter this command:

   ```
   setowclst
   ```

3. Enter one of these commands and press Enter:

   *ADD

*REMOVE

*CHANGE

*CLEARALL

Note: The *CLEARALL command removes all replicated objects. If you select this command, no further steps are required.

4. In the IFS Object field, enter one of these:
   - **Y** if the clustering object is an IFS object
   - •**N** if the clustering object is a QSYS object
   - If the clustering object is an IFS object, enter the name and the path of the IFS clustering object.

     Note: Replicate all objects in a directory by typing * and the path, for example, **/E900SYS/PRINTQUEUE/***.

   - If the clustering object is a QSYS object, enter the name and the library of the QSYS clustering object.

     Note: Special values of *ALL and *PF (physical file) can also be used as valid objects.

9.3 Identifying the Cluster Name

In an IBM i clustering environment, JD Edwards EnterpriseOne uses a logical cluster name to make services available to clients and to manage connections between nodes. You must define the name on the DNS and in the IBM i host names table. The cluster name is associated with a floating IP address.

For example, you might set up two IBM i machines that you name DEINS3 and DENIS4. The cluster name you set up in the DNS might be DENISZ. DENISZ is the cluster name associated with a floating IP address that is defined as a TCP /IP interface and listed in the host table of each node in the cluster.

The client references the cluster name when it requests JD Edwards EnterpriseOne services from the enterprise server. Therefore, services are not tied to a single physical machine. The node with the active TCP /IP interface associated with the cluster name provides the services to the client that requests them.

9.4 Setting up the Enterprise Servers

You must identify the cluster to each node in the cluster by adding a [CLUSTER] section to the server jde.ini file. If the cluster name is DENISZ, you add this entry to the jde.ini file:

```
[CLUSTER]
PrimaryNode=DENISZ
```

Note: If you referenced the logical cluster name or virtual host name (for example, DENISZ) in the installation plan, you only need to modify the enterprise server's jde.ini file as described previously. You don't need to complete any of these setups since all the data sources and ODBC setting are already pointing to the logical host name.

9.5 Setting up the Client for Clustering

You must set up the client jde.ini file for it to connect to the cluster name you identified. To set up the client jde.ini file, make changes to these parameters:

```
[DB system settings]
Server=DENISZ
 [SECURITY]
SecurityServer=DENISZ
```

where DENISZ is the logical cluster name you specified in the DNS and IBM i host names table.

9.6 Setting up the Deployment Server

You must set up the deployment server to connect to each designated enterprise server node in the cluster. You configure the [DB SYSTEM SETTINGS] section and [SECURITY] section of the deployment server jde.ini so that it can connect to the logical data source of the enterprise server.

You should review these Deployment server jde.ini section entries:

```
[DB SYSTEM SETTINGS]
Base Datasource=System - 900
Server=DENIS3
Database=System - 900
Load Library=JDBODBC.DLL
Decimal Shift=Y
Julian Dates=Y
Use Owner=N
Secured=Y
Type=I
DatabaseName2=SY900

[SECURITY]
DataSource=System - 900
```

where System - 900 is the logical data source used to establish initial Object Configuration Manager (OCM) settings, and DEINS3 is the name of an enterprise server node in the cluster.

After you configure the deployment server jde.ini file for a server node in the cluster, you log onto JD Edwards EnterpriseOne on the deployment server as an administrative user and set up logical data sources, database data sources, and OCM for the node. You then configure the deployment server jde.ini for another cluster node. You repeat the sequence of tasks for each node in the cluster.

9.7 Setting Up Logical Data Sources

After you configure the deployment server jde.ini file for clustering, you must set up the logical data sources on each enterprise server node for both system map and server map.

Note: System map configures the client connection to logical data on the enterprise server. Server map configures the server connection to logical data on the enterprise server.

This topic discusses the steps required to complete these tasks:

- Setting up the logical data source for the system map.
- Setting up the logical data source for the server map.

See Also:

- "Setting Up Data Sources" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

9.7.1 Setting up the logical data source for the system map

To set up the logical data source for the system map:

1. On the Systems Administration Tools menu (GH9011), select Logical Data Sources (P986115).
2. Select the logical data source you identified in the Base Datasource= parameter of the [DB SYSTEM SETTINGS] section of the deployment server jde.ini file and click Select.

   The default is System – 900
3. In the Work With Data Sources form, click Add.

   The Data Source Revisions form appears.
4. Complete these fields to create the logical data source, and then click OK:

Field	Description
Data Source Name	Enter the name of the cluster.
Data Source Use	Type **SVR**
Platform	Type **DB2 for IBM i**
Logical Server Name	Enter the name of the cluster. Make sure the name matches the logical cluster name you created in the DNS or in the IBM i host names table.
Server Map Data Source Name	Type <Server Name> - 900 Server Map

5. JD Edwards EnterpriseOne launches a form prompting you to create a new ODBC data source. Because you are configuring a logical ODBC data source for the cluster, not a database data source, click Cancel.

9.7.2 Setting up the logical data source for the server map

To set up the logical data source for the server map:

1. On the Systems Administration Tools menu (GH9011), select Logical Data Sources (P986115).
2. Select the logical data source for the enterprise server node server map and click Select.

   The default is <Server Name> - 900 Server Map.
3. In the Work With Data Sources form, click Add.
4. Complete these fields to complete the logical data source for the server, and then click OK:

Field	Description
Data Source Name	Enter the name of the cluster.
Data Source Use	Type **SVR**
Platform	Type **DB2 for IBM i**
Logical Server Name	Enter the name of the cluster. Make sure the name matches the logical cluster name you created in the DNS or in the IBM i host names table.
Server Map Data Source Name	Type <Server Name> - 900 Server Map

For a full description of the fields in the Data Source Revisions form, see Adding or Modifying a Data Source in the JD Edwards EnterpriseOne Tools 8.94 Implementation Guide: Configurable Network Computing Implementation.

9.8 Setting Up Database Data Sources

After you set up the logical data sources for the cluster, you must set up the database data sources for the enterprise server node. A database data source identifies to JD Edwards EnterpriseOne the database information JD Edwards EnterpriseOne needs to identify and connect to a database, including the type and location of the data.

See Also:

- "Setting Up Data Sources" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

9.8.1 Setting Up the Server Map Database Data Sources

To set up the server map database data sources:

1. On the Systems Administration Tools menu (GH9011), select Database Data Sources (P986115).
2. On the Machine Search and Select form, select the logical data source for the enterprise server node server map and click Select.

   The default is <Server Name> - 900 Server Map.
3. On the Work With Data Sources form, click Find.
4. Select a data source, such as Business Data - PROD, for the enterprise server node and click Select.
5. In the Data Source Revisions form, change the value in the Database Server Name field to the name of the logical cluster.

6. Repeat steps 4 and 5 for each data source.

9.9 Setting Up Object Configuration Manager for Clustering

After you have set up logical data sources and database data sources, use OCM to map objects in each clustering environment.

> **Note:** Complete OCM configuration requires that you complete the steps discussed in this topic for each clustering environment.

This topic discusses the tasks you complete to set up OCM for clustering:

- Configuring OCM for logical data sources for the server map.
- Configuring OCM for logical data sources for the system map.
- Configuring OCM for database data sources.
- Configuring ODBC connections.

9.9.1 Configuring OCM for Logical Data Sources for the Server Map

Access the Machine Search and Select form. You can access this form in the Microsoft Windows client or the web client by entering **P986110** in the Fast Path.

1. On the Machine Search and Select form, select the logical data source for the enterprise server node server map and click Select.

   The default is <Server Name> - 900 Server Map.

2. On the Work With Object Mappings form, click Add.
3. Complete these fields and click OK:

Field	Description
Environment Name	Enter the name of an environment that will use the cluster.
Object Name	Enter **DEFAULT**.
Primary Data Source	Enter the name of the logical cluster.
System Role	Type ***PUBLIC**
Object Type	Enter **BSFN** for business functions.
Data Source Mode	Type **P**

4. Click OK.
5. On the Work With Object Mappings form, make sure that the business function DEFAULT OCM mapping for the environment is active.
6. Click Close.
7. Repeat this task for each clustering environment.

9.9.2 Configuring OCM for Logical Data Sources for the System Map

Access the Machine Search and Select form. You can access this form in the Microsoft Windows client or the web client by entering **P986110** in the Fast Path.

1. On the Machine Search and Select form, select the logical data source you identified in the Base Datasource parameter of the [DB SYSTEM SETTINGS] section of the deployment server jde.ini file and click Select.

   The default is System - 900.

2. On the Work With Object Mappings form, click Add.

3. Complete these fields and click OK:

Field	Description
Environment Name	Enter the name of an environment that will use the cluster.
Object Name	Enter **DEFAULT**
Primary Data Source	Enter the name of the logical cluster.
System Role	Type ***PUBLIC**
Object Type	Type **BSFN**
Data Source Mode	Type **P**

Note: If you want to run UBEs on the server by DEFAULT, then you must set up another logical data source for UBEs and use the logical cluster name.

4. Determine the business functions that should be run on client workstations, not on the enterprise server.

Note: You can run the batch application Create Server Business Function OCM Records (R986140) to accomplish this task. Be sure to create a new version in proof mode. To do so, select enter a value of O to Processing Option 1 when you submit the report.

5. For each business function that runs locally, launch the Work With Object Mappings form, complete these fields and click OK:

Field	Description
Environment Name	Enter the name of an environment that will use the cluster.
Object Name	Enter the name of the business function that you want to run on the client workstation.
Primary Data Source	Type **LOCAL**
System Role	Type a system role
Object Type	Type **BSFN**

6. On the Work With Object Mappings form, make sure that the business function DEFAULT and LOCAL OCM mappings for the environment are active. If they are not, change the status to active

7. Click Close.

8. Repeat this task for each clustering environment.

9.9.3 Configuring OCM for database data sources

To configure OCM for database data sources:

1. On the Systems Administration Tools menu (GH9011), select Database Data Sources (P986115).
2. On the Machine Search and Select form, select the logical data source for the system map and click Select.
3. On the Work With Data Sources form, click Find.
4. Select a data source used by me clustering environment and click Select.
5. On the Data Source Revisions form, enter the name of the logical cluster in the Server Name field and click OK.
6. Repeat steps 3 though 5 for each data source used by the clustering environment.
7. Repeat this task for each clustering environment.

Note: If more than one environment shares a data source, verify that you set up all the environments for clustering. If an environment sharing a data source is not a clustering environment, you might need to set up independent data sources for the environments.

9.9.4 Configuring ODBC connections

To configure ODBC connections:

1. On the ODBC Data Source Administrator form, select the System DSN tab.
2. Select a data source used by the environment that will use clustering and click Configure.

   The Client Access Express ODBC Setup (32-bit) form appears.
3. Select the General tab.
4. From the IBM i combo box, select the name of the logical cluster and click OK.
5. Repeat these steps for each data source used in the cluster.

9.10 Distributing the ODBC Setup from the Deployment Server

After you have completed the configuration of all enterprise server nodes in the cluster, you must distribute the ODBC configuration you set up on the deployment server to all clients (workstations and servers) that will connect to enterprise server nodes in the cluster. Servers that might connect to enterprise server cluster nodes include Windows Terminal Servers and web servers. The deployment server handles the distribution of the ODBC configuration information.

To accomplish the ODBC setup distribution task, you create a .reg file on the deployment server. The .reg file is an executable that contains the ODBC registry settings you set up on the deployment server. After you create the .reg file, client machines must run it to get the saved ODBC settings and set up their ODBC connections.

Administrators can deploy the .reg file to clients that connect to enterprise servers in the cluster.

To distribute the ODBC setup from the deployment server:

1. From the Windows Start menu, run regedit.exe.
2. In the Registry Editor, browse to HKEY_LOCAL_MACHINE\SOFTWARE\ ODBC\ ODBC.INI.
3. Click Registry and select Export Registry File.
4. Save the .reg file.

9.11 Identifying the Cluster Name on the Deployment Server

To complete setup for clustering, you identify the cluster name in the deployment server jde.ini file. Doing so enables the deployment server to look for the cluster when it attempts to connect to an enterprise server node. The active node makes the connection.

Note: Remember to identify the cluster name in the jde.ini files of any servers, such as web servers or Windows Terminal Servers that need to connect to enterprise servers in the cluster.

To identify the cluster name on the deployment server:

1. Open the jde.ini file of the deployment server.
2. Replace any reference to the server (for example, Server=DENIS3) with the cluster name (for example, Server=DENISZ).

10

Backing Up JD Edwards EnterpriseOne Tables

This chapter contains the following topics:

10.1 Understanding Backup Requirements for Servers

A well-planned backup strategy is essential to protect the enterprise information assets. Rigorously following the backup strategy will provide insurance against data lost by acts of nature, hardware or software failure, or human error. The backup strategy must balance the level of protection you need against the physical constraints of the system, such as information storage capacity.

We recommend that the backup strategy include these:

- Perform a full system backup whenever data is at risk, such as when you are installing or upgrading software. In this circumstance, at least back up the database completely.
- Each night, back up changed objects, such as tables and JD Edwards EnterpriseOne objects.
- Each week, back up the deployment server, enterprise servers, and the full database.

When you perform a backup on a server, you can back up either the entire server or only the changed objects and data. You do not need to perform a complete backup of the server nightly. Only directories that change daily require daily backups.

Note: You should outline and implement the backup strategy before you begin the Prototype phase of implementation.

10.1.1 Backing Up a Deployment Server

JD Edwards EnterpriseOne on the deployment server includes these items:

- JD Edwards EnterpriseOne directory (all subdirectories and contents).
- jde.ini file on c:\winnt.
- Services file on c:\winnt\system32\drivers\etc.
- Registry export file.

- JD Edwards EnterpriseOne files in the root directory (c:\):
- jdeapp.ddp
- jdeapp.xdp
- jdeauth.dda
- jdeauth.xda
- jdemod.ddm
- jdemod.xdm
- jdesec.dds
- jdesec.xds
- jdecode.ddm
- jdecode.xdm

If you modify objects, build new packages, or update the Access database delivered during a workstation installation, create backups of the PD900, DV900, and PY900 directories. If you modify help files, create a backup of the HELPS directory. If the media objects reside on the deployment server, create a backup of the MEDIA OBJ directory.

If important data, such as system data, resides on the deployment server, create nightly backups of the JD Edwards EnterpriseOne data sources (Oracle or SQL Server). For example, if the central objects or Object Management Workbench resides on the deployment server, create a nightly backup.

10.1.2 Backing Up an Enterprise Server

JD Edwards EnterpriseOne on the enterprise server runs on the IBM i, UNIX, or Windows operating systems. You back up key libraries on the IBM i and key files on the UNIX and Windows operating systems.

IBM i

These JD Edwards EnterpriseOne IBM i libraries should be backed up:

> **Note:** Shut down the database before you create any backups.

- All JD Edwards EnterpriseOne system libraries.
  - JDEOW
  - SYS900
  - E900SYS
  - SVM900
  - JD Edwards EnterpriseOne data dictionary library: DD900.
  - JD Edwards EnterpriseOne Object Management Workbench library: OL900.
- All JD Edwards EnterpriseOne production libraries (This example is for pristine and production):
  - PD900
  - PY900

  - PRODDTA
  - PRSTDTA
- All JD Edwards EnterpriseOne business data libraries:
  - PRODDTA
  - CRPDTA
  - PRSTDTA
  - TESTDTA
- All JD Edwards EnterpriseOne control libraries:
  - PRODCTL
  - CRPCTL
  - TESTCTL
  - PRSTCTL
- All JD Edwards EnterpriseOne versions libraries:
  - PD900DNT
  - PY900DNT
  - PS900DNT
  - DV900DNT
- IFS (Integrated File System) libraries:
  - PD900
  - PY900
  - PS900
  - TS900
  - DV900
- IBM libraries that require backups:
  - OCPA
  - OGPL
  - Central objects on the deployment server in Oracle or Microsoft SQL Server database.

UNIX

On a JD Edwards EnterpriseOne UNIX system, backup these database files:

> **Note:** Shut down the database before you create any backups using Backup Manager. If you export or import using Data Manager, you do not need to shut down the database.

- System files

  Create backups of all host files under the JDEdwards/E900 directory. For example, /u03/JDEdwards/E900/*.
- Database files

Create backups of all data files that reside in the JD Edwards EnterpriseOne tablespaces.

Use the Oracle Data Manager Tool on the deployment server to make a .dmp file of the desired database, and then back up the .dmp file on tape or hard disk.

Windows

On a JD Edwards EnterpriseOne Windows system, back up these database files:

Note: Shut down the database before you create any backups.

- System files.

  JDEdwards\ddp\E900 directory.

- Oracle database files.

  Create backup files for all data files that reside in the JD Edwards EnterpriseOne tablespaces

  Use the Oracle Data Manager Tool on the deployment server to make a .dmp file of the desired database, and then back up the .dmp file on tape or hard disk.

- Microsoft SQL Server database files.

  Create backup files for all tables that reside in the JD Edwards EnterpriseOne databases.

  Use the SQL Server Database/Object Transfer tool on the enterprise server to copy the desired tables or database (for example, PSFT900) to a backup database.

Note: We recommend that you use the backup tool provided by the RDBMS vendor.

10.1.3 JD Edwards EnterpriseOne Tables and Object Owner IDs

These tables list JD Edwards EnterpriseOne tables by type and with the associated object owner IDs.

Note: If any of the control table merges fail or if the specification merge fails, you might need to restore the tables to a pre-merge condition and run the merge again. Follow the restore instructions for the database.

System Tables

The Object Owner for System tables is sys900.

- F00053
- F000531
- F000532
- F0092
- F00921
- F00924

- F0093
- F0094
- F00941
- F00942
- F00945
- F00946
- F00948
- F00950
- F00960
- F99001
- F986101
- F98611
- F986115
- F986116
- F98613
- F986150
- F986151
- F986152
- F98616
- F986161
- F986162
- F986163
- F986164
- F986165
- F98701
- F98800D
- F98900D
- F9882
- F98825
- F9883
- F9885
- F9886
- F9887
- F9888
- F98881
- F98882
- F98885

- F98887
- F9889
- F98891
- F98892
- F98980
- F98CONST
- F98DRENV
- F98DRLOG
- F98DRPCN
- F98DRPUB
- F98DRSUB
- F98EVDTL
- F98EVHDR
- F98MOQUE
- F98OWSEC
- F98TMPL
- F98VAR

Object Management Workbench (OMW) Tables

The Object Owner for OMW tables is obj900.

- F00165
- F9860
- F9861
- F9862
- F9863
- F9865

Data Dictionary Tables

The Object Owner for the Data Dictionary tables is dd900.

- F00165
- F9200
- F9202
- F9203
- F9207
- F9210
- F9211

Server Map Tables

The Object Owner for Server Map tables is svm900.

- F986101

- F98611
- F986110
- F986111
- F986113
- F98DRPCN
- F98DRLOG

Control Tables

The Object Owners for the Control Tables are:

- Control Tables - PROD: prodctl
- Control Tables - CRP: crpctl
- Control Tables - TEST: testctl
- Control Tables - PS900: prstctl

The Control Tables are listed:

- F0002
- F00021
- F0004
- F0004D
- F0005
- F0005D
- F0082
- F00821
- F00825
- F00826
- F0083
- F0084

Versions Tables

The Object Owners of the Versions tables are:

- Versions - PD900: PD900
- Versions - PY900: PY900
- Versions - DV900: DV900
- Versions - PS900: PS900

The Versions tables are listed:

- F983051
- F98306

Central Objects

The Object Owners of the Central Objects tables are:

- Central Objects - PD900: pd900

- Central Objects - PY900: py900
- Central Objects - DV900: dv900
- Central Objects - PS900: PS900

The Central Objects tables are listed:

- F980011
- F980021
- F983051
- F98306
- F98710
- F98711
- F98712
- F98713
- F98720
- F98740
- F98741
- F98743
- F98745
- F98750
- F98751
- F98752
- F98753
- F98760
- F98761
- F98762
- F98950

Business Data

The Object Owners of the Business Data tables are:

- Business Data - PROD: proddta
- Business Data - CRP: crpdta
- Business Data - TEST: testdta
- Business Data - PS900: prstdta

10.2 Backing Up JD Edwards EnterpriseOne Tables on Servers

This section discusses how to:

- Create a backup for IBM i.
- Creating a backup for Oracle on UNIX or Windows.
- Creating a backup for SQL Server.

- Restoring a backup file for Oracle on UNIX or Windows.
- Restoring a backup file for IBM i.
- Restoring a backup file for SQL Server.
- Restoring a backup file for SQL Server on Windows.

10.2.1 Prerequisites

Before you complete the tasks in this section:

- If you are using SQL Server or Oracle, verify that you have enough disk space for the backup copy before you begin the backup.
- If you are using SQL Server, verify that the Select Into/Bulk Copy option on the Options form is turned on for the database into which you will transfer objects. Double-click the database in the tree structure to access the Options form.

10.2.2 Creating a Backup for IBM i

To create a backup for IBM i:

1. On a tape drive, back up these libraries, depending on which path codes you have installed:

Library name	Description
SYS900	System library
SVM900	Server Map
OL900	Object Librarian
DD900	Data Dictionary
COPY900	Central Objects - Prototype
COPS900	Central Objects - PS900
COPD900	Central Objects - PROD
CODV900	Central Objects - DEV
PRODDTA	Production Business Data
PRODCTL	Production Control Tables
CRPDTA	Prototype Business Data
CRPCTL	Prototype Control Tables
TESTDTA	Test Business Data
TESTCTL	Test Control Tables
PRSTDTA	Pristine Business Data
PRSTCTL	Pristine Control Tables
PY900DNT	Versions for CRP
PD900DNT	Versions for PROD
PS900DNT	Versions for PRST
DV900DNT	Versions for DEV
E900SYS	Server system library
JDEOW	JD Edwards Installation

Library name	Description
PY900	Server modules - Prototype
PY900FA	Package Library - Prototype
PS900	Server modules - PS900
PS900FA	Package Library - PS900
PD900	Server modules - PROD
PD900FA	Package Library - PROD
DV900	Server modules - DEV
DV900FA	Package Library - DEV

2. Back up these IFS structure with the subdirectories:

Library name	Description
PSFT900	Logging directory
E900SYS	Kernel spec and XML
PY900	Spec files for Prototype
PS900	Spec files for PSFT
PD900	Spec files for PROD
DV900	Spec files for DEV
JD Edwards	Contains the spec files for each path code. \JDEdwards\PACKAGES\PY900FA\SPEC*.* \JDEdwards\PACKAGES\PS900FA\SPEC*.* \JDEdwards\PACKAGES\PD900FA\SPEC*.* \JDEdwards\PACKAGES\DV900FA\SPEC*.*

10.2.3 Creating a Backup for Oracle on UNIX or Windows

To create a backup for Oracle on UNIX or Windows:

1. From the Oracle Enterprise Manager Tool, open Data Manager and from the Data menu, select Export.
2. Type the name for the export utility .dmp file.

   Click the Browse button to select the directory where the .dmp file will reside.
3. Click Next.
4. On the Object Selection form, select the objects you want to back up, and then click Next.

   Note: Objects selected in the tree on the Data Manager form appear in the Selected Objects form. You can move objects between forms using the arrow buttons or by dragging and dropping.

   To export objects, expand the Available Objects tree and select the item to export. Use the arrows to move objects to and from the Selected Objects form.

5. On the Tuning form, select generate a log file, if needed.
6. Click Next.

   Note: Select the Generate Log File option and enter a log file name or use Browse to select a log file.

7. On the Advanced Options form, take the default values or select the desired options, and click Next.
8. On the Summary form, verify that all of the chosen objects and options are correct.
9. Click Finish to begin exporting objects.

   A message window opens that displays information about the progress of the export process.

   When the export process is completed, you will receive these message: "Export terminated successfully without warnings."
10. If errors or warnings exist, check the log file to review the export process.

10.2.4 Creating a Backup for SQL Server

To create a backup for SQL Server:

1. From SQL Enterprise Manager, select Database/Object Transfer from the Tools menu.
2. On the Database/Object Transfer form, select a destination server and database on which to create backup copies of the tables.

   Note: The source server and the destination server can be the same, but the database must be different.

3. Keep all default settings and then click the Start Transfer button.

   The Database/Object Transfer tool moves the objects.
4. Perform either of these tasks to verify whether the backup was successful:
   - When the process completes the transfer, click the View Logs button to review the transfer process.
   - Run a SELECT statement to verify that the backup tables transferred to the new database with data.

10.2.5 Restoring a Backup File for Oracle on UNIX or Windows

To restore a backup file for Oracle on UNIX or Windows:

1. From the Oracle Enterprise Manager Tool, open Data Manager and from the Data menu, select Import.
2. Type the name of the import utility .dmp file.
3. Click Next.
4. On the Object Selection form, select the objects you want to restore and click Next.

The Importable Objects tree contains the objects that are importable in the file you specified. To move the object to the Selected Objects tree, select an object in the tree and click the down arrow.

Note: When the .dmp file is on a remote machine, Data Manager uses the Console job and event system to retrieve the file before displaying the data through the Import Wizard. The Remote Import page of the Import Wizard has a status line at the top of the page that displays the progress of data retrieval. The Oracle Enterprise Manager Console must be running.

Three conditions can be displayed: Job Submitted, Job Started, and Job Completed.

Note: Data retrieval must complete successfully before beginning the import operation.

The Selected Objects/Available Objects tree contains the objects to be imported. To remove an object from the list, select the object and use the up arrow or drag and drop.

5. Click Next.
6. On the Associated Objects form, accept the defaults and click Next.
7. On the Tuning form, you can generate a log file, if needed.
8. Click Next.

Note: Select the Generate Log File options and enter a log file name or use Browse to select a log file.

---

9. On the Advanced Options form, select the Increment Type. If you followed the instructions to create a backup, select None for Increment Type and click Next.
10. On the Summary form, verify that all selected objects and options are correct.

Note: You must drop the existing objects in the database that you want to restore or the import process will fail.

---

11. Click Finish to begin importing objects.
12. When the import process is completed, you will receive these message: "Process terminated successfully with no warnings."

    If errors or warnings exist, check the log file to review the export process.
13. Perform a SELECT statement to verify that the backup tables are populated with data.

10.2.6 Restoring a Backup File for IBM i

To restore a backup file for IBM i:

Restore the libraries and IFS directories that you backed up from tape.

10.2.7 Restoring a Backup File for SQL Server

To restore a backup file for SQL Server:

1. Verify that the Choose Into/Bulk Copy option on the Options form is turned on for the database into which you will transfer objects.

   Double-click the database in the tree structure to access the Options form.

2. From SQL Enterprise Manager, select Database/Object Transfer from the Tools menu.

3. On the Database/Object Transfer form, select a destination server and database from which to transfer backup copies of the tables.

   Note: The source server and the destination server can be the same, but the database must be different.

4. Deselect the Transfer All Objects option, but keep all of the other default settings.

5. Click the Choose Objects button, select the objects that you want to transfer, and then click OK to return to the Database/Object Transfer form.

6. Click the Start Transfer button.

   The Database/Object Transfer tool moves the objects.

7. Perform either of these to verify whether the backup was successful:

   – When the process completes the transfer, click the View Logs button to review the transfer process.

   – Run a SELECT statement to verify that the backup tables transferred to the new database with data.

10.2.8 Restoring a Backup File for SQL Server on Windows

To restore a backup file for SQL Server on Windows:

1. Verify that the Select Into/Bulk Copy option on the Options form is turned on for the database into which you will transfer objects.

   Double-click the database in the SQL Enterprise Manager tree structure to access the Options form.

2. Generate scripts for the tables you want to restore and then drop the tables.

3. Use SQL to recreate the scripts for the tables.

4. From the command line, type this command:

   ```
   bcp [[database_name.]owner.] table_name(in|out) datafile /n /u /p /s
   ```

5. Perform a SELECT statement to verify that data populates the backup tables.

11

Generating Serialized Objects for the JD Edwards EnterpriseOne Web Server

This chapter contains the following topic:

- Section 11.1, "Understanding Serialized Object Generation"

11.1 Understanding Serialized Object Generation

To run the web server, the server must have access to a set of serialized Oracle's JD Edwards EnterpriseOne JAS objects. These objects can be generated directly from Oracle's JD Edwards EnterpriseOne objects using the appropriate set of JD Edwards EnterpriseOne specifications.

Generating JD Edwards EnterpriseOne serialized objects requires a specific machine configuration. While it is possible to configure a web server to generate JD Edwards EnterpriseOne serialized objects, the recommended method is to dedicate a separate generation machine for this process.

Note: The eGenerator software can change with each Tools Release.

See the Generating JD Edwards EnterpriseOne Serialized Objects appendix in the*JD Edwards EnterpriseOne Tools 8.98 HTML Web Server Installation Guide*.

There are eight versions of this guide which depend on platform and application server.

12

Understanding Executable Files on the Workstation

This chapter contains the following topics:

12.1 JD Edwards EnterpriseOne Linked Executable Files

This section provides a list of linked executable files (executables) that are in the workstation system/bin32 directory.

Linked executables:

- Are called by other Oracle's JD Edwards EnterpriseOne programs.
- Are called by the JD Edwards EnterpriseOne kernel.
- Have no value if they are run independently of JD Edwards EnterpriseOne.
- Will not run unless they are called by JD Edwards EnterpriseOne.

  This table includes descriptions and instructions for running the linked executable files:

Executable	Description	Call Details
Ap22.exe	JD Edwards EnterpriseOne uses this program to display spreadsheets in a dialog box. This executable is obsolete and has no function in SP10 and beyond.	Obsolete.
BLC2Text.exe	JD Edwards EnterpriseOne uses this program to read workstation JDEBLC spec files and generate a text file with details about each business function source file that is listed in the spec file.	Called by an internal business function build program that is not shipped to customers.
Dir2txt.exe	This program takes a path and a text file name as arguments and places the directory name of the highest branch in the path into the text file.	Called from makefiles that are generated by BusBuild.

Executable	Description	Call Details
Drilldwn.exe	JD Edwards EnterpriseOne uses this utility when generating Balance Auditor functions in Tabular reports.	Called by the JD Edwards EnterpriseOne UBE kernel.
DSArguments.exe	JD Edwards EnterpriseOne uses this utility program to create a CID argument when attempting to connect to a JD Edwards EnterpriseOne Data Source.	Called by the JD Edwards EnterpriseOne kernel.
GBLib.exe	The JD Edwards EnterpriseOne process BusBuild uses this program to determine whether the object files exist.	Called from makefiles generated by BusBuild.
Guimole.exe	JD Edwards EnterpriseOne uses this program to create a bridge between the workstation and the IBM i server to enable green screens to be displayed through JD Edwards EnterpriseOne.	Called by the JD Edwards EnterpriseOne kernel.
InstMon.exe	JD Edwards EnterpriseOne uses this program during sign-in to install update packages if an update package is selected.	Called by the JD Edwards EnterpriseOne kernel.
JDEGenEx.exe	This program generates a list of exports for each dll.	Called from makefiles that are generated by BusBuild.
Rtt.exe	We ship this program for use by business partners only. JD Edwards EnterpriseOne uses this program to build resource files for language translation. The risk is that the existing resource files could be confused with the newly generated files. The user would have to intentionally continue through multiple screens for this to happen.	Do not use this program.
Ubemon.exe	This program monitored long-running UBEs and reported their completion. This program is obsolete and was disabled in SP10.	Obsolete.
Vdt.exe	This Business View Design Tool creates business views when called from Object Librarian or Object Management Workbench (OMW).	Called from Object Librarian or Object Management Workbench (OMW).

Executable	Description	Call Details
owptrl_cli.exe	owptrl_cli is the communication bridge between BMC Patrol Monitoring Tool and JD Edwards EnterpriseOne enterprise and web servers. owptrl_cli converts SAW data to BMC data so that the Agent will understand. The Agent sends a request to the owptrl_cli (for example, give me the list of all processes running on HP9000B port 6012); the owptrl_cli returns detailed information about all the processes running on the server in a format known by the Agent. In this way, BMC can monitor JD Edwards EnterpriseOne servers.	Called by BMC Patrol Agent with a predefined argument list.
DbidCapture.exe	This utility program is called by Autopilot or EventCapture to capture database IDs that are necessary for Autopilot to access JD Edwards EnterpriseOne tables. It is not an end-user program and has no purpose apart from Autopilot or EventCapture.	Called by Autopilot or EventCapture.
ubeprint.exe	ubeprint.exe is not for direct customer use, although the JD Edwards EnterpriseOne product suite does use it.	Called by the JD Edwards EnterpriseOne kernel.
genver.exe	genver creates the win32 version information for the build process.	Called by the JD Edwards EnterpriseOne kernel.
poda.exe	Processing Option Design Aid. All design tools are client side only. OMW passes the executable a set of parameters that are similar to RDA, TDA, and BDA.	Called from OMW when you design a processing option.
RDA.exe	Report Design Aid. All design tools are client side only. OMW passes the executable a set of parameters similar to PODA, TDA, and BDA.	Called from OMW when you design a batch application. RDA can also be opened without command line parameters.
guimole.exe	A secondary executable that is called to pass parameters into the WorldVision session.	Called by WorldVision.
FDA.exe	Form Design Aid (FDA) is used to create interactive applications. FDA is currently configured to run on a fat client.	Called from the design window in OMW for an application.
JdeCabExtract.exe	JdeCabExtract creates self-extracting.exe files.	JD Edwards internal tool.

Executable	Description	Call Details
JdeCompress.exe	JdeCompress creates JD Edwards-compatible cabinet files.	JD Edwards internal tool.
pssg.exe	An obsolete file that is not called by any JD Edwards EnterpriseOne applications.	Obsolete.
GLBUILD.exe	GLBUILD was replaced by Busbuild.exe. It was used to build the business functions.	Obsolete.
krnlspec.exe	This program is used to generate jdekrnl.xdb and jdekrnl.ddb specs from the pristine database.	JD Edwards internal tool.

12.2 JD Edwards EnterpriseOne Standalone Executable Files

You can run standalone executable files directly from either the command line or through Windows Explorer.

This table includes descriptions and instructions for running the standalone executable files:

Executable	Description	Run Instructions
JDECOMConnector2.exe	This program sets up COM connections to the server using the COM Connector product and only works in that context. Contact Customer Support for full documentation.	Run from the command line with a -regserver option.
LogViewer.exe	This program employs a user friendly interface to view and modify plain ASCII JD Edwards EnterpriseOne files such as: ■ jdedebug.log ■ jde.log ■ olt.log ■ jde.ini	Double-click the executable.
MOConv.exe	This utility converts all records to use a period as the decimal separator. Use this utility when records are entered into a single table using both commas and periods as decimal separators. This utility is driven by the MOConv.ini file.	Quit JD Edwards EnterpriseOne and then double-click the executable.

Executable	Description	Run Instructions
Nettest.exe	This utility tests basic JDENET connectivity using the "netecho" function against an enterprise server. Enter the name of the enterprise server in the Host Name box and press Send. The returned data indicates success or failure.	Double-click. Required argument: Enterprise Server Name
Regdlls.exe	This program adds these dynamic link libraries (DLLs) to the registry: jdetapitest.dll jdetapicomtek.dll Register these DLLs before using RunTAPI.exe.	Run from a DOS window in the JD Edwards EnterpriseOne system/bin32 directory.
RunTAPI.exe	This program controls interoperability between JD Edwards EnterpriseOne and telephone switching systems. It is a snap-in (harness) to ComTech CTI Server objects. It requires jde.ini file changes and Regdlls.exe before it can be run. Contact Customer Support for full documentation.	Double-click.
SABridge.exe	This Object Export Facility displays the names of the JD Edwards EnterpriseOne objects along with their descriptions and corresponding product codes.	Double-click the executable.
SnapShot.exe	This program manages multiple workstation installations on the same PC. You can install a new instance of JD Edwards EnterpriseOne by clicking Save to store the current workstation installation in a newly named location. Click Restore to toggle between the current and saved versions.	Exit JD Edwards EnterpriseOne and then double-click the executable.
VerifyOCM.exe	This program reads the OCM tables from the database and verifies that the mappings in OCM are correct.	Run from a DOS window. Required arguments are: ■ JD Edwards EnterpriseOne user ■ JD Edwards EnterpriseOne password ■ JD Edwards EnterpriseOne Environment

Executable	Description	Run Instructions
Vercheck.exe	This program displays, on one screen, the properties of all the files in a directory. The properties are the same as those that are displayed when you right-click a file and select Properties.	Open a DOS window, change the directory to the desired target, and double-click the executable.
GenCOM.exe	This program generates COM wrappers for the business functions that are specified in the script.	Run GenCom.exe from the command line with the name of the script file.
GenCORBA.exe	Creates CORBA wrappers around JD Edwards EnterpriseOne business functions. This is a command line utility that requires a script file as an input. GenCORBA generates CORBA interfaces for JD Edwards EnterpriseOne business functions.	Syntax: GenCORBA[options] [libraries] For example: GenCORBA /Cat /UserID Devuser1 /Password Denuser1 /Environment ADEVHPO2 CAEC
GenJava.exe	GenJava provides access to JD Edwards EnterpriseOne business functions by generating pure Java interfaces to them.	Run GenJava. Syntax: GenJava [options] [libraries] For example: GenJava /Cat /UserID Devuser1 /Password Denuser1 /Environment ADEVHPO2 CAEC
LaunchUBE.exe	LaunchUBE.exe is used to launch the UBE job stand-alone (not going through JD Edwards EnterpriseOne). It replaces the User Interface of UBEPrint.exe.	Double-click the executable or start using the command line.
Autopilot.exe	Autopilot is the centerpiece of all automated testing tools. Using Autopilot, a person can script JD Edwards EnterpriseOne applications to run automatically and save the scripts to run many times. Autopilot is used throughout the company and by many customers for a wide variety of purposes.	Normally started from a desktop button or from the Start menu without command line options.

Executable	Description	Run Instructions
EventCapture.exe	EventCapture is a small program that can be activated alongside JD Edwards EnterpriseOne (in lieu of Autopilot) to capture performance and debugging information. EventCapture is often used instead of Autopilot because it is simpler and quicker than creating an Autopilot script for a single use. With EventCapture, the user drives JD Edwards EnterpriseOne; with Autopilot, the script drives JD Edwards EnterpriseOne.	Normally started from a desktop button or the Start Menu without command line options.
APTestMgr.exe	Autopilot Test Manager is used to run multiple Autopilot scripts in a batch and to manage batches for repeated execution. It has some ability to summarize results, and it is frequently used for regression testing.	Normally started from a desktop button or from the Start Menu without command line options.
VSMerge.exe	JD Edwards ER Compare tool is used to compare and merge Event Rules (ER) for JD Edwards EnterpriseOne Applications, Reports, Table Conversions, NERs (Named Event Rules), and TERs (Table Event Rules). It also can be used to compare and merge C Business functions.	You can launch JD Edwards ER Compare tool from OMW or from the command line.
VSMEditor.exe	VSM Editor is a rarely used GUI tool that creates .VSM files. VSM files are super scripts that name one or more virtual Autopilot scripts to be run in succession by VAPPlayer.	Normally run by double-clicking vsmeditor.exe in bin32.
VirtualRunner.exe	VirtualRunner is a GUI tool for controlling multiple VAPPlayer processes on a single workstation.	Run the tool from a shortcut on the desktop or in the Start Menu. This tool does not use command line arguments.

Executable	Description	Run Instructions
vapplayer.exe	Virtual Autopilot Player enables you to simulate multiple concurrent JD Edwards EnterpriseOne users on a single workstation. It is used primarily for concurrency testing during development and for performance and scalability testing of JD Edwards EnterpriseOne applications. VAPPlayer requires a proper vap.ini (initialization) file. VAPPlayer has many command line arguments, which are optional if vap.ini is fully utilized. See documentation for details. VAPPlayer has no user interface. It produces output in log files.	VAPPlayer can be run from a command line, from the VirtualRunner graphical user interface, or from the Mercury LoadRunner (third-party) software console.
Analyzer.exe	Analyzer.exe is better known as JD Edwards EnterpriseOne Analyzer. It is a powerful instrument that is used to analyze performance data and other debugging information that is generated by a JD Edwards EnterpriseOne application that is run under Autopilot or EventCapture.	Create a desktop button or Start Menu button. No command line arguments are used.
UTBrowse.exe	UTB is a tool that is used for viewing the records in tables. We also use it to view local JD Edwards EnterpriseOne object specifications. UTBrowse.exe uses these two libraries in the bin32 directory: ■ datautils.dll ■ envtool.dll.	Type **UTB** in the EnterpriseOne Fast Path field or click the executable.
tda.exe	Use TDA to modify JD Edwards EnterpriseOne tables.	On the Command Line, type tda.exe -idtablename, where tablename is the name of the table that you want to modify. For example, F0101. You do not need to run JD Edwards EnterpriseOne before running tda.
tc.exe	tc.exe opens the JD Edwards EnterpriseOne Table Conversion Design Tool. This tool is used to design JD Edwards EnterpriseOne Table Conversion batch applications.	Double-click the executable, or run it from the command line using the optional parameter idXXXX, where XXXX is the name of an existing Table Conversion object.

Executable	Description	Run Instructions
Tamvrfy.exe	tamvrfy checks the integrity of all the tam files that are listed in the tamvrfy.lst.	Double-click the executable.
tamtool.exe	tamtool can perform these functions: ■ Recreate a tam file. ■ Copy a tam file. ■ Print index information. ■ Print the index key. ■ Verify the tam file.	Run from the command line.
tampack.exe	tampack.exe is a backup utility in case tamftp.exe does not work for the customer. tampack.exe has about half the functionality of tamftp.exe. tampack.exe is included with the workstation and the deployment server. tampack.exe creates a translated copy of TAM files (RDASPEC.DDB, GBRSPEC.DDB, and so on) on the PC. The translated copies are known as pack files. When the program is finished, the user can manually run ftp.exe to transfer them to a remote enterprise server. When the pack files are on a remote enterprise server, the user can unpack them on the enterprise server.	You must run tampack.exe from a DOS shell and pass in parameters.
tamftp.exe	tamftp.exe comes with the workstation and the deployment server. tamftp.exe transfers TAM files (RDASPEC.DDB, GBRSPEC.DDB, and so on) from the PC to a remote enterprise server that is operational.	You must run the program from a DOS shell and pass it parameters.
pdf2pdl.exe	pdf2pdl is an MFC application that converts PDF files into files containing the printer-specific protocol language for a selected printer. This application is intended only for development to troubleshoot problems with a customer's JD Edwards EnterpriseOne output. The tool can help solve configuration problems.	Double-click the executable.

Executable	Description	Run Instructions
pdfcompare.exe	Displays the objects in the PDF document as a list and compares them.	Double-click the executable.

13

Troubleshooting the Workstation

This chapter contains the following topics:

13.1 Understanding Error Messages

Use this section as a general guide for basic troubleshooting techniques on theOracle JD Edwards EnterpriseOne workstation. To troubleshoot problems, you will need a thorough understanding of the interactive error messages, Oracle's EnterpriseOne Work Center, logging process, and associated log files.

This section provides solutions to these problems that you might encounter on the workstation:

13.1.1 Report Batch Process

You might encounter these issues when running a batch process:

- Report displays no data.

  It displays only the report headers and the text No Data Selected.
- Batch process displays errors on the report.
- Batch process gives unexpected data on the report.

13.1.2 Environment Issues

You might encounter these environment issues:

- Works when the batch process or business function runs locally but not when it runs on the enterprise server.
- For store-and-forward operation, data entered to the local database is not moved to the server as expected.
- Tables are missing.

13.1.3 Data Source Setup Problems

You might encounter these problems with the data source setup:

- Unable to connect to the enterprise server environment.
- Data is displayed incorrectly on the interactive form or batch report.

See Also:

- "Running the Object Configuration Copy Report" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

13.1.4 Error Message Details

When you encounter an error, right-click the error message in the error message window and select Detail to provide additional information about the error. This information provides the source file and the source line that caused the error. If you try to set up an Item/Branch record in P41026 with an invalid item number, you will receive error number 0267 (Item Number Invalid).

When indicating the source file that generated the error, the system provides the entire path of the source location. In this example, the source file is c:\E900\MSTR900\X4101.C, and business function X4101 created the error. The other pieces of the path are directory names. The important information in this example is the file with the .C extension (X4101.C).

If the detail for the error includes the name of the source file, you can identify the process that the file performs to determine what might occur to cause an error. For example, the name of the source file might include system code that indicates the process performed by the file. The process might attempt to run in a module that is not fully functional. The cause of the error might be a constant set to perform a function that is currently unavailable. When you disable the constant, you avoid the error.

Note: If you see a source file description that begins with c:\E900\SYSTEM, the error did not occur through a business function. Possibly, the error occurred through an event rule or the tool, while automatically triggering a data dictionary edit.

Look for conditional statements that determine whether to activate the error message. Look for table names to determine whether the program attempts to retrieve data. Look for other programs that the program might call. Also, read the programmer comments that are included in the source, which might provide a literal explanation for why the code issues an error.

Also look at the data item that caused the error. The data item represents a control on the form. If you get a Blanks Invalid error without an indication of what field you left blank, look at the data item in this error detail box to see which control triggered the error. The field that contains the error might be a hidden field. For example, if you process a transaction that requires a supplier number determined by an Item/Branch combination decided by JD Edwards EnterpriseOne (not by a value that you define on the form) but no supplier number exists for the Item/Branch combination, the software returns the Blanks Invalid error. The field for supplier number does not appear on the form, so the cause of the error is not readily apparent. The data item might alert you to the hidden field and help you resolve the error.

13.1.5 Error Messages Generated by Applications

These error messages are maintained in the data dictionary and are intentionally set to inform the user of a problem. The error message might indicate that the setup is

incorrect or that the user is attempting an invalid action. Examples of these kinds of error messages include Record Invalid and Blanks Invalid. Some generic errors lack applicable descriptions; techniques for troubleshooting these errors are discussed in this chapter.

13.1.6 Frequent Generic Error Messages

Some error messages are too generic to immediately explain an error. Examples are Null Pointer and File Can Not Be Accessed. The full descriptions of these error messages do not provide much information as to how to resolve the problem.

To troubleshoot generic errors, retrace the exact steps that led to the original error. The goal is to reproduce the error. If you cannot duplicate an error condition, then the application is accessing different lines of code than it did when the error occurred. Also look at the information in the error detail box, such as the source file, the source line, and the data item.

13.1.7 Memory Violations

Memory violations occur when you encounter memory leaks in an application. A memory leak is a bug that prevents a program from freeing memory that the program no longer needs. The program continues to consume more memory until no memory remains, and the program crashes. JD Edwards EnterpriseOne applications set aside memory while they run. When the application no longer needs that memory, the application should free the memory for other applications to use. When an application does not properly free memory or when an application attempts to use invalid memory, you receive a memory violation.

Use these techniques to troubleshoot these errors:

- Look at the jdedebug.log to find information about the processing that occurred at the time of the error, such as programs called and tables accessed.
- Follow the exact steps that led to the error to reproduce the memory violation.

  If you cannot duplicate the violation, then the application is accessing different lines of code than it did when the violation occurred. Also look at the information in the error detail box, such as the source file, the source line, and the data item. For UBEs, if the UBE uses a business function that causes memory violations, the UBE will simply stop. In this case, the ube.log is the only way to find out what failed.

13.1.8 Form and Grid Add Failures

The two error messages that follow indicate that an attempt to add a new record to the database failed. The first message indicates that an add within a fix/inspect form failed. The second message indicates that an add within a grid failed. If you receive these errors, you could be attempting to add a duplicate record.

- Attempt to add form record failed.
- Attempt to add grid record failed.

The jde.log is a helpful reference when these errors occur. In general, it includes detailed information about the table into which the user attempted to add a duplicate record.

13.1.9 Communication Failure

When submitting batch processes to a server, you might receive an error telling you that a communication failure has occurred.

When you submit a batch job to a server, you are first asked whether you would like to install the specifications. If the job is submitted successfully, JD Edwards EnterpriseOne reverts to the initial form.

13.2 Troubleshooting the Production Workstation

This section provides an overview of production workstation troubleshooting and discusses how to:

- Perform preliminary troubleshooting.
- Troubleshoot interactive application problems.
- Troubleshoot batch processes resulting in no data.
- Troubleshoot batch processes displaying errors on the report.
- Troubleshoot batch processes displaying unexpected data on the report.
- Troubleshoot batch processes ending in an error when submitted on the server.
- Troubleshoot local data-availability problems.
- Troubleshoot .DLL problems on a production workstation.
- Troubleshoot data source setup problems.

13.2.1 Understanding Production Workstation Troubleshooting

The troubleshooting procedures that you use for a workstation depend on whether the workstation is a production or development machine. Production machines contain only JD Edwards EnterpriseOne applications, so the scope of problems that can occur is limited. In addition to containing prebuilt applications, development machines are equipped with JD Edwards EnterpriseOne and third-party tools. These tools enable developers to create, modify, compile, generate, and troubleshoot JD Edwards EnterpriseOne applications.

As a system administrator, you can perform preliminary troubleshooting on the production workstation to verify the nature of the problem. You will also want to isolate problems to a user's particular workstation and environment.

In general, when you are running an interactive application, the system displays errors at the bottom of a form. The system highlights the fields with errors in red. You can select Details on an error message to see information about where the error was set. For example, if the error resulted from within a business function, the system displays the business function and line number where the error was set.

If the errors cannot be resolved through the error messages that are received in the application, check the error messages in the log files for additional information.

If an application has stopped running, you might need to create or retrieve a new set of specifications for that application. You can overwrite a single application by building a partial package and deploying that package.

A user might encounter several problems when attempting to run a batch process. For example, the output might display only the report headings or it might print a message such as "No Data Selected." If the result of a batch process is no data, several factors could be causing the problem.

Some batch processes will give error messages directly on the report. These messages should include both the short description and error message number. You can view the full description of the error by opening the message in Data Dictionary Design.

If errors are received when you are attempting to sign in to a JD Edwards EnterpriseOne environment, a possible cause is an incorrect data source setup on the workstation. Some indications of incorrect setup are:

- A form continues to request a user ID, a password, and a data source even after valid ones are entered.
- Data is displayed incorrectly on an interactive application.
- Messages in the logs refer to problems connecting to data sources or concerning incorrect passwords.

13.2.1.1 Troubleshooting a Standalone Installation of JD Edwards EnterpriseOne

If you find that you cannot perform a force checkout on a PC running a standalone installation of JD Edwards EnterpriseOne, it is because the software cannot determine the system name for a standalone installation.

The solution is to disable the DNS name in Microsoft Windows.

13.2.1.2 Troubleshooting Enterprise Server Data-Availability Problems

If the workstation is running a report against any enterprise server database, such as Oracle, SQL Server, or DB2 for IBM i, you need to check the database to see whether the SQL statement can find data in the tables. With the help of a database administrator, you can run the same SQL statement against the server database to verify that the expected data exists in the tables.

As an alternative or in addition to these procedures, you can also use the Universal Table Browser to verify table structure and data availability.

If you do not find any data in the tables for the environment against which you are running, then the SQL statement might be incorrect or the table is empty. Check the data selection and processing options, and verify that they are selecting data that is in the tables. If you do not have data in the tables to match what you are searching, then you will get unexpected results or no data on the report.

For example, if you leave the processing options blank (even though that may be a valid entry for a JD Edwards EnterpriseOne batch process), the process might be searching for blank values or for all values. If the data selection is selecting on a company that does not have any records, then the report batch process does not find any records.

13.2.1.3 Troubleshooting Printing Problems

Most printing errors are written to the batch process log. However, some errors might appear on reports or be visible in another form. For example, the report prints an error message, prints in the wrong font, or prints landscape instead of portrait.

These printing problems can occur:

- The batch application produces error messages on the report, for example, Invalid Company Number.
- The report batch process displays the wrong font on the report.

  Check the report properties of the version that you just ran. Also, for the section that is not printing the correct font, check the section properties for the font. If the font is correct, then try printing to a different printer. Otherwise, try using another

workstation to see whether the font that is being sent to the printer is not interpreted correctly.

- The report batch process prints portrait instead of landscape or landscape instead of portrait.

  Check the report properties of the version that you just ran and verify that the properties are correct.

  See Also:

  - "Understanding the Package Build Process" in the *JD Edwards EnterpriseOne Tools Package Management Guide*.
  - *JD Edwards EnterpriseOne Tools Workflow Tools Guide*.
  - "Using the Universal Table Browser" in the *JD Edwards EnterpriseOne Tools System Administration Guide*.
  - "Running the Object Configuration Copy Report" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

13.2.2 Performing Preliminary Troubleshooting

To perform preliminary troubleshooting:

1. Determine whether you can consistently duplicate the problem.
2. If you can duplicate the problem, restart the current application.
3. If the error recurs, restart JD Edwards EnterpriseOne.
4. If the error recurs, reboot the workstation.

   These steps clear any memory or caching problems with the workstation.

13.2.3 Troubleshooting Interactive Application Problems

To troubleshoot interactive application problems:

1. Select one of these to see the text of the message:
   - Display Errors from the Help menu selection.
   - Display button on the toolbar.
   - F8.
2. To see the full description of an error message, right-click and select Full Description.

   The system displays a full glossary of the error and includes information for resolving the issue.

13.2.4 Troubleshooting Batch Processes Resulting in No Data

This task provides a solution to previously discussed problems.

To troubleshoot batch processes resulting in no data:

1. Verify that the data selection on the batch process is appropriate and that data should result.

   Data selection on an item that has no data, such as an inactive company, or an incorrect value will result in a batch process with no data.

2. Check the Work Center to see whether the batch process resulted in an error.

   Most error messages are not printed on the report but are sent as an email message to the user who submitted the report.

   These messages will give the user an example of why the batch process ended without producing the desired results. For example, when the system runs a GL post that ends in error, the report will print only the report headings. All error messages are sent to the Work Center.

   Upon quitting the Work Center, the user receives error messages and a glossary description that indicate why the batch process resulted in no data. Some error messages include hot links that link the user directly to the appropriate interactive application to correct the error.

3. If checking the data selection and correcting any errors does not resolve the issue, activate the applicable logs and continue with these steps.

4. Run the batch process and locate the batch process log for the report that you ran.

   JD Edwards EnterpriseOne names this log with these conventions:

   ```
   report name_version_other identifiers.log
   ```

   This log is located in the local directory under \E900\PrintQueue. If you ran report R04431, it would appear in the local directory, E900\PrintQueue, like this:

   ```
   R04431_XJDE0001_D960823_T104512.log
   ```

5. View the log file using the JD Edwards EnterpriseOne Log Viewer or an ASCII editor such as Notepad or Wordpad.

   Inspect the log for errors or failures of any kind. Also examine the SQL statements that were created by the batch process and verify that they should result in data on the report. The batch process log is the main source for debugging batch processes. However, you can look in the jde.log and jdedebug.log for errors or failures of any kind.

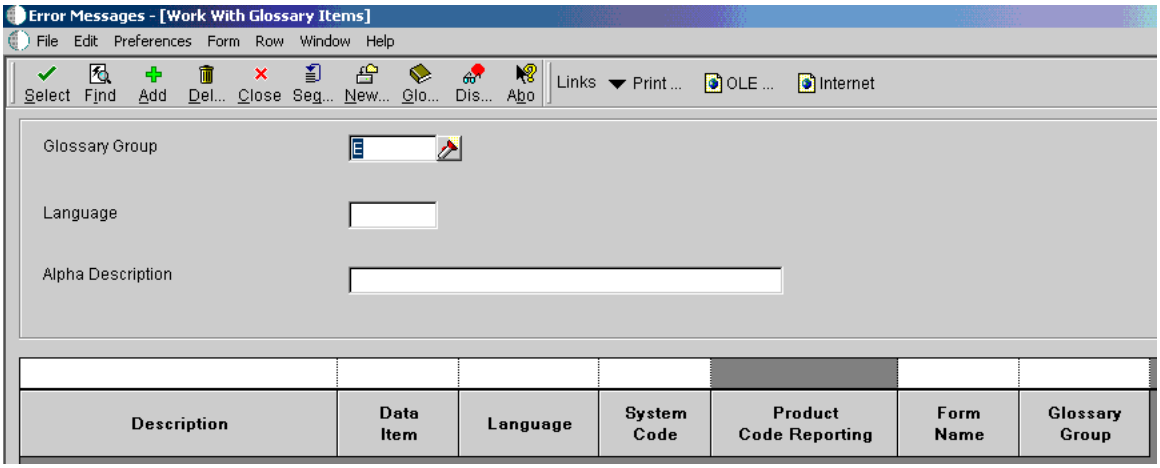
6. Verify that data exists in the tables for the database that you are accessing.

   Use the Universal Table Browser tool to view the database table.

13.2.5 Troubleshooting Batch Processes Displaying Errors on the Report

Access Error Messages from the Data Dictionary Design menu (GH951).

Figure 13–1 Error Messages form



To troubleshoot batch processes displaying errors on the report:

1. On Error Messages, complete the Glossary Group field.
2. Complete these optional fields:
   - Language
   - Alpha Description
3. On the grid, enter values in the Data Item field and click Find to narrow the search to the parti

   cular error code.

   For example, enter 0002 to search for the data item that is associated with the Record Invalid error condition.
4. To see an extended description of the error, select Glossary from the Row menu.

13.2.6 Troubleshooting Batch Processes Displaying Unexpected Data on the Report

To troubleshoot batch processes displaying unexpected data on the report:

1. Verify that the data selection on the batch process is correct and should result in the data output that is expected.
2. Activate the batch process log and run the report.
3. Examine the report process flow and SQL statements to see why the data output on the report is selected.

13.2.7 Troubleshooting Batch Processes Ending in an Error When Submitted on the Server

The default processing location for batch jobs is the server. If a job gives incorrect results or ends in error when run on the server, the problem could lie with the batch process or with the server. When you troubleshoot batch processes ending in an error when submitted on the server

1. Rerun the report, but override the location to run on the workstation rather than the server.

You should be aware that if this is a very large report, the processing may take a significant amount of time. You may want to select less material to speed up the processing time.

2. Verify whether the outcome is the same as when the report was run on the server. If so, use the other troubleshooting procedures for batch processing to resolve the issue.

13.2.8 Troubleshooting Local Data-Availability Problems

Inspect the local database at \E900\pathcode\data\JDELocal_PD900.mdf to verify that data exists in the tables that the batch application is accessing.

To troubleshoot local data-availability problems:

1. To find the calling SQL statement, open the batch process log.

   JD Edwards EnterpriseOne names this log using these conventions: report name_version_other identifiers.log. This log is located in the local directory, \E900\PrintQueue.

2. Highlight the SQL statement, right-click, and copy the contents to the clipboard.

3. To view data in the local database, open the Universal Table Browser (UTB) and retrieve the table that the batch application is accessing from the local data source.

4. Use the information that you copied from the SQL statement to query the table in UTB.

   If this action causes the expected records to be found, the data that you specified in the data selection matches the SQL statement, which means that data selection is not the cause of the problem.

13.2.9 Troubleshooting .DLL Problems on a Production Workstation

Problems with workstation .DLL files are indicated if you receive an error message such as this:

```
CALLBSFN.DLL Load Lib failed
```

Such a message might indicate that the object does not exist on the workstation. Use a tool such as Explorer to verify whether the file exists. You can find consolidated .DLLs in the \E900\path code\bin32 directory.

If the .DLL does not exist on the workstation or if it exists but you continue to get the error even after restarting JD Edwards EnterpriseOne, you can get the correct parent .DLL by reinstalling JD Edwards EnterpriseOne on the workstation from the deployment server. Another option is to copy the parent .DLL from the deployment server package location or another functioning workstation. This option will be successful if the business functions that are built into the parent .DLL are the same on the workstation that you are copying to as they are on the one that you are copying from. Use caution when copying .DLLs. A workstation installation is the preferred method.

13.2.10 Troubleshooting Data Source Setup Problems

To troubleshoot data source setup problems:

1. From the Control Panel, verify that the ODBC settings are correctly defined and that the data source exists.

The proper settings vary by data source.

2. If other users will sign in to the same workstation, verify that the data sources are set up as system data sources rather than user data sources.

   Data sources that are set up as user data sources must be set up for each user who is accessing JD Edwards EnterpriseOne on the workstation.

13.3 Troubleshooting the Development Workstation

This section provides an overview of development workstation troubleshooting and discusses how to:

- Troubleshoot .DLL problems on a development workstation.
- Troubleshoot event rule problems.
- Troubleshoot business function problems.

13.3.1 Understanding Development Workstation Troubleshooting

The troubleshooting procedures that you use on a workstation depend on whether the workstation is a production or development machine. Production machines contain only JD Edwards EnterpriseOne applications, so the scope of the problems that can occur is limited. In addition to containing prebuilt JD Edwards EnterpriseOne applications, development machines are equipped with JD Edwards EnterpriseOne and third-party tools. These tools enable developers to create, modify, compile, generate, and troubleshoot JD Edwards EnterpriseOne applications.

You can perform troubleshooting procedures to isolate and resolve a problem with a JD Edwards EnterpriseOne development workstation.

Problems with workstation .DLL files are indicated if you receive an error message such as this:

```
CALLBSFN.DLL Load Lib failed
```

Such a message might indicate that the object does not exist on the workstation. Use a tool such as Explorer to verify whether the file exists. You can find consolidated .DLLs in the directory E900\path code\bin32 and E900\system\bin32.

If the .DLL does not exist on the workstation or if it does exist but you continue to get the error even after restarting JD Edwards EnterpriseOne, the workstation has a problem with the build of one or more consolidated .DLLs. You can rebuild libraries or .DLLs using the BusBuild application from Microsoft Windows Explorer. The path to busbuild.exe is E900\system\bin32\busbuild.exe.

13.3.2 Troubleshooting .DLL Problems on a Development Workstation

Use this procedure if you are receiving the error on a specific business function that cannot be found in the parent .DLL.

To troubleshoot .DLL problems on a development workstation:

1. Verify that the correct parent .DLL for the business function that is being run is referenced when you receive the error.
2. If the wrong parent .DLL is referenced, select Synchronize JDEBLC from the Tools menu within BusBuild to correctly synchronize the parent .DLLs.
3. Attempt to rebuild the business function from the BusBuild.exe.

The rebuild should include the business function in the parent .DLL.

4. To verify which business functions are part of a parent .DLL, select Dumpbin from the Tools menu within Busbuild.

   This option lists all of the business functions that are included in the parent .DLL.

13.3.3 Troubleshooting Event Rule Problems

When you encounter problems with event rules on an interactive or batch application, several tools are available to help resolve the problem.

- Review the event rules that are attached to the application or batch process for obvious problems such as disconnected assignments or incorrect parameters that were passed to business functions.
- When the system generates the application, a compile error log is generated, which documents errors in the event rules.

  Review this log for errors within the Event Rules.
- The Debug Application within JD Edwards EnterpriseOne enables you to debug the event rules for an application or batch process.

13.3.4 Troubleshooting Business Function Problems

You might be having business function problems if you are getting unexpected results or getting a .DLL error when you run a business function.

Microsoft Visual C++ enables you to debug a business function. You can use this tool to step through the logic and inspect variables, which often helps you detect the error.

See Also:

- "Troubleshooting Business Function Processing Problems" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

13.4 Working with the Workstation Log Files

This section provides an overview of the workstation log files and discusses how to:

- View log files.
- Set up the workstation jde.log.
- Set up the workstation jdedebug.log.
- Set up the batch process log.
- Troubleshoot with the compile error log.
- Troubleshoot with jdecpy.log.
- Troubleshoot with the sql.log.
- Activate sql.log.
- Troubleshoot ODBC problems using sql.log.
- Troubleshoot with the jdeinst.log.

13.4.1 Understanding the Workstation Log Files

You should be familiar with the various logs that are used to troubleshoot problems on the workstation. By using these logs and the procedures that are outlined in this chapter, you can troubleshoot problems with interactive applications, batch applications, or business functions running locally on the JD Edwards EnterpriseOne workstation. Determine whether you can duplicate the problem consistently or whether it is intermittent.

Do not leave the debugging logs active when the logs are not in use. The logs consume disk and processor resources, and therefore affect performance.

If you do not use data replication in the configuration, ignore error messages that refer to these tables in the jde.log and jdedebug.log:

- F98DRPUB
- F98DRENV
- F98DRSUB
- F98DRLOG
- F98DRPCN

13.4.1.1 Global Tables

Each JD Edwards EnterpriseOne workstation uses global tables (glbltbl.xdb and glbltbl.ddb) to write disk cache files containing internal session-specific and workstation-specific information. For example, information about data dictionary tables and business views is cached. By maintaining a history of this cached session information, individual workstations will improve runtime performance based on their usage.

If you are doing application development, you might need to delete the global tables to see the results of the changes. This is because the system looks first to the disk cache to read certain table information. The information that is contained in the disk cache might not be synchronized with the current development. You cannot edit the contents of the disk cache.

We recommend that normal startup of JD Edwards EnterpriseOne in a production environment *not* include the deletion of these global tables. These files should be deleted only as a troubleshooting technique or a development aid.

If the global table files do not exist when JD Edwards EnterpriseOne is started, they will be created. If they do exist, they will be appended, not overwritten. The files are located in the E900/pathcode/spec directory.

In general, on JD Edwards EnterpriseOne workstations, logs are classified in these categories:

- Logic processing.
- Batch processing.
- Application development (compiling and generating).
- Object Management Workbench transactions.

13.4.1.2 Logic Processing Logs

You use two major log file sources for troubleshooting processing faults on the workstation:

- jde.log

This log displays fatal errors. Jde.log can track any fault that might occur within JD Edwards EnterpriseOne.

- jdedebug.log (JDEDEBUG on IBM i)

  This log tracks API calls and SQL statements as well as other messages. You can use this file to determine the point in time when normal processing stopped. The system does not use jdedebug.log to track errors. Instead, this log is used to track the timing of JD Edwards EnterpriseOne processes.

13.4.1.3 Application Development Logs

For JD Edwards EnterpriseOne workstations in application development environments, you can use these logs to identify faults in processing that are related to compiling and generating applications and business functions:

- compile_error.log

  The compile_error.log contains compile errors for event rules. You can use this log to view event rules that might not properly compile and run. These include Named Event Rules, Table Event Rules, and event rules that are embedded in applications.

- jdecpy.log

  This log is produced each time you run the copy table program (cpytbl.exe). Copy table error messages and IDs are logged. This log also indicates whether any inserts failed that could indicate a possible error.

- sql.log

  You use this log to view exactly what is being sent through the ODBC driver. This is not a JD Edwards EnterpriseOne log; another software vendor provides this log process.

For workstations in production environments, you can use jdeinst.log to identify faults in JD Edwards EnterpriseOne silent installation.

If you use the silent installation process for JD Edwards EnterpriseOne installations on a workstation, you can use this log to view the status of the silent installation.

13.4.1.4 Workstation jdedebug.log

The workstation jdedebug.log file contains messages relating to API calls and SQL statements, as well as other messages. You can use this file to determine the point in time when normal processing stopped. The system does not use jdedebug.log to track errors. Instead, it uses this log to track the timing of JD Edwards EnterpriseOne processes.

You can use jdedebug.log to determine where a process has ended. For example, log data can include what the ODBC was trying to connect to, the SQL statement that was being run for a specific table, and whether memory has been freed.

If the process failed and you have logging turned on, look in the jdedebug.log for these messages:

- Not Found
- Failure

Also, look at the end of the log to see what process ran last. In general, important lines in the log are:

- SELECT

The SELECT lines indicate which table you are selecting. The log tells you in which library (for the IBM i) or environment (for the non- IBM i) the table resides. You should verify that the selected libraries and environments are correct.

- ODBC Version

  The ODBC lines indicate whether you are having problems connecting to the driver.

13.4.1.5 Batch Process log

You can use the batch process log to identify faults in JD Edwards EnterpriseOne processing that are related to batch processes. The batch process log resides in the \E900\PrintQueue directory. The log file name is batch_process.log, where batch_process represents the report name, version name, date, and time.

Based on the setting of the UBESaveLogFile parameter in the [UBE] section of the jde.ini file, this log file is deleted or saved on successful completion of batch processes. This log file displays different types of messages that can help in tracking errors in the batch process. The messages are:

- Section Level Process
- Object Level Process
- ER Level Process
- DB Level Process

The batch process log can contain ER references, batch process flow, and SQL statements, among other messages. You can use the batch process log file to determine when normal processing stopped.

The batch process log file displays the process flow in batch processes. This flow is completed in these steps:

1. When batch processes complete a section, starting with the INIT section, a business view is opened.

   After the INIT Section log, you should see a SQL statement.
2. After INIT Section, the batch engine calls Adv Section to retrieve a record.
3. After the retrieve, batch engine processes the Do Section Processing.
4. From Do Section, each object is processed in Init Object - Do Object - End Object order.
5. After Do Object message, you should see Printed value in the log.

   ER events are logged in a different event level.

13.4.1.6 sql.log

In sql.log, the important lines for you to search are:

- SELECT * FROM (bolded in these example)
- SQLBindCol
- Table not found

Verify that you are reading the correct table. For example, in the sql.log example, a line exists for every column in the selected table, which indicates that the correct table is selected.

If you are having difficulty reading the table, verify that the table has the correct number of columns. If you have added columns to the table and you cannot locate the correct number of columns, you need to configure the table. This information is also provided in jde.log.

13.4.1.7 Use of Log Files to Troubleshoot Strategies

You can create a normal (successful) jde.log by logging on to JD Edwards EnterpriseOne and then immediately logging off. Use this log of successful startup statements to compare against logs that have a problem.

If you know the problem is not related to startup, you can clear and save the log without quitting JD Edwards EnterpriseOne. When you recreate the problem, the contents of the log should contain only errors that occurred since you cleared the log.

You can also rename the log to indicate the kind of problem. For example, you might delete the jde.log and then run a report that causes an error condition. You could rename the jde.log to report.log.

Another alternative is to add comment lines to the jde.log indicating the sequence of events that you are performing. For example, you might be running an application that you know causes an error. Just before you run the application, you could edit the jde.log to add a comment line stating that you are about to start the suspect application.

Most error messages in the jde.log have a unique number assigned to them. You can view an extended description of the error, including possible causes and resolutions, by searching on the error number in the Error Messages application (P92002).

See Also:

- "Troubleshooting Business Function Processing Problems" in the *JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide*.

13.4.2 Viewing Log Files

You can view JD Edwards EnterpriseOne log files from within any application. If you want to view log files outside of JD Edwards EnterpriseOne, you can use a text editor like Notepad or Wordpad.

To view log files:

1. From within any JD Edwards EnterpriseOne application, right-click to open the pop-up menu.
2. On the pop-up menu, select the View System Log option.
3. On Log Viewer, select File, Open to locate and open a JD Edwards EnterpriseOne log file.

   You can also use the View menu selection to select log files. If you have viewed log files previously, the File menu keeps a history of those files.

13.4.3 Setting Up the Workstation jde.log

You can use the workstation jde.log as a general purpose log to track fatal errors that are generated by JD Edwards EnterpriseOne processing. The jde.log tracks any fault that might occur within JD Edwards EnterpriseOne. When you are looking for startup errors, you should read the jde.log from the top down. For other errors, you should read from the bottom up.

The workstation jde.log is created (if it does not exist) or overwritten (if it already exists) at the start of every JD Edwards EnterpriseOne session.

To set up the workstation jde.log:

1. Locate the workstation jde.ini file.

   The JD Edwards EnterpriseOne setup program places this file in the working Microsoft Windows directory; for example, c:\Windows\jde.ini. If you are unsure of the workstation's working Microsoft Windows directory, use the Find command to locate the jde.ini file.

2. Use an ASCII editor (like Notepad or Wordpad) to open the file.

3. In the [DEBUG] section, verify or change this setting for the job file variable:

Setting	Purpose
JobFile=	Specifies the location and name of the jde.log file. The default value is c:\jde.log. **Note:** You can disable the jde.log on the workstation by specifying a blank or invalid value for JobFile. If you delete or disable (comment out) the JobFile parameter, JD Edwards EnterpriseOne automatically creates and writes to a log file called jde.log in the c:\ directory of the workstation.

4. Save the changes and close the jde.ini file.

13.4.4 Setting Up the Workstation jdedebug.log

To set up the workstation jdedebug.log:

1. Locate the workstation jde.ini file.

   The JD Edwards EnterpriseOne setup program places this file in the working Microsoft Windows directory, for example, c:\Windows\jde.ini. If you are unsure of the workstation's working Microsoft Windows directory, use the Find command to locate the jde.ini file.

2. Use an ASCII editor (like Notepad or Wordpad) to open the file.

3. Verify or change the name of the jdedebug.log file.

   The location and file name of the jdedebug.log file is defined by this setting in the jde.ini file:

Setting	Purpose
DebugFile=	Specifies the location and name of the jdedebug.log file. The default value is c:\jdedebug.log.

4. Enable or disable the logging of events to the jdedebug.log file through this setting in the [DEBUG] section:

Setting	Purpose
Output=	Valid values are: ■ NONE: No trace information is written to jdedebug.log. ■ FILE: Database and runtime trace information is written to the file that is specified by the DebugFile= parameter in the [DEBUG] section. ■ EXCFILE: Runtime trace information is written to the file that is specified by the DebugFile= parameter in the [DEBUG] section. ■ BOTH: Trace information is written to both jde.log and jdedebug.log.

Note: The primary method of disabling the jdedebug.log is by using the Output parameter. However, if you set Output=FILE and you leave the DebugFile value blank (or specify an invalid location), JD Edwards EnterpriseOne still performs debug tracing but does not write the results to any jdedebug.log file. If you delete or disable (comment out) the DebugFile parameter, JD Edwards EnterpriseOne automatically creates and writes to a log file called jdedebug.log in the c:\directory of the workstation.

5. Set the level of debugging information that you want written to the jdedebug.log file.

   The debug level is determined by this parameter in the [DEBUG] section:

Setting	Purpose
Output=	Specifies the debug level. You can specify any combination of allowable values using commas as delimiters. The default setting is LEVEL=BSFN,EVENTS. Valid values are: ■ EVENTS ■ BSFN ■ SF_x ■ GRID ■ PARENT_CHILD ■ GENERAL ■ MESSAGING ■ WORKFLOW ■ WORKFLOW_ADMIN ■ MEDIA_OBJ ■ CONTROL

   For example, LEVEL=SF_CONTROL. In addition, you can specify multiple system functions by separating them with commas. For example, LEVEL=SF_GRID,SF_CONTROL. You can also specify numeric values:

   1: Traces critical errors. This is the default level. That is, whether you specify this value or not, the system will always trace critical errors.

2: Traces critical errors. This is the default level. That is, whether you specify this value or not, the system will always trace critical errors.

3: Traces statements as the software enters and exits each event. Specifying this value is the equivalent of specifying the EVENTS value.

4: Traces main messages that the software sends to a controlling parent of a child. These messages concern the processing functions such as the grid.

5: Provides a detailed trace report of every function that the software calls in the interactive runtime module. This setting is applicable only to developers of the runtime module.

6. Save the changes and close the jde.ini file.

13.4.5 Setting Up the Batch Process Log

To set up the batch process log:

1. Locate the workstation jde.ini file.

   The JD Edwards EnterpriseOne setup program places this file in the working Microsoft Windows directory, for example, c:\Windows\jde.ini. If you are unsure of the workstation's working Microsoft Windows directory, use the Find command to locate the jde.ini file.

2. Use an ASCII editor (such as Notepad or Wordpad) to open the file.

3. Set the level of batch report debugging information that you want written to the batch process log file, and set whether you want the file to be saved.

   These settings are controlled by these parameters in the [UBE] section:

Setting	Purpose
UBEDebugLevel=	Specifies the level of UBE debug logging. The default value is 0, and values are: ■ 0: No message in a log file. ■ 1: Error messages, and log entry and section level messages. ■ 2: Object level messages (plus Level 1 messages). ■ 3: ER messages and database mapping messages (plus Level 1 and 2 messages). ■ 4: SQL statements (plus Level 1, 2, and 3 messages). ■ 5: Batch process function calls and printed output values (plus Level 1, 2, 3, and 4 messages). ■ 6: Batch process function calls and printed output values (plus Level 1, 2, 3, 4, and 5 messages).
UBESaveLogFile=	Specifies whether the <batch_report>.log file will be saved. Values are: ■ 0: The <batch_report>.log file is not saved. ■ 1: The <batch_report>.log file is saved in the workstation JD Edwards EnterpriseOne print queue directory (E900\PrintQueue).

4. Save the changes and close the jde.ini file.

13.4.6 Troubleshooting with the Compile Error Log

For JD Edwards EnterpriseOne workstations in development environments, use this log to identify faults in JD Edwards EnterpriseOne processing that are related to compiling and generating applications and business functions. This log for compiled event rules provides an account of event rules (Named Event Rules, Table Event Rules, and applications) that do not properly compile and process. JD Edwards EnterpriseOne generates this log file every time the Code Generator program (cg.exe) is run and errors occur with compiled event rules.

The <compile_error> portion of the log file name refers to a variable value for the name of the event rules being compiled. For example, a name of a log file for compiling NER N3200780 is N3200780.log. The error log from an application containing compiled event rules replaces the first letter of the application name with an E; for example, P0101 generates an error log named E0101.log.

Use this log when errors have occurred within the Code Generator while you were compiling an application, Named Event Rules, or Table Event Rules. When this happens, a message box appears beneath the JD Edwards EnterpriseOne Code Generation form with the source member and the problem description. You can use the log file to keep a record of such problems. The compile error log resides in the log folder under the path code portion of the E900 directory tree, for example, c:\E900\PD900\LOG.

13.4.7 Troubleshooting with jdecpy.log

The system produces output for jdecpy.log each time the copy table program (cpytbl.exe) is run on the workstation. In general, the file contains records of those tables that were successfully copied from the local database to the chosen server. This log also indicates whether any inserts failed. Such failures indicate a possible error. This log is automatically stopped after cpytbl.exe finishes.

The jdecpy.log resides in the root directory of the workstation, usually in c:\. JD Edwards EnterpriseOne automatically generates this log every time you run cpytbl.exe. The log is created or overwritten each time it runs.

After you use jdecpy.log to determine that a copy table error has occurred, you should refer to the jde.log. If a table does not copy properly, the detail of the error text is written to jde.log. The jde.log contains the actual error message and message ID. The message ID relates to the line prefix numbers in the jdecpy.log. This ID will help you locate the applicable error text that was written to the jde.log.

13.4.8 Troubleshooting with the sql.log

You can use sql.log to view exactly what is being sent through the ODBC driver. This log is not a JD Edwards EnterpriseOne log; another software vendor provides this log process. For workstations, sql.log resides in the default root directory of the workstation, usually in c:\. You can direct the output to any file in any location. In general, instead of using the sql.log, you can use the jdedebug.log because it also tracks SQL statements.

In sql.log, the important lines to search for are:

- SELECT * FROM
- SQLBindCol
- Table not found

If you experience a problem with the ODBC settings or can't connect to a JD Edwards EnterpriseOne ODBC database, activate logging for jde.log, jdedebug.log, and sql.log. Duplicate the problem, check jde.log or jdedebug.log to view the ODBC error messages, and check the end of sql.log to determine the last process. The majority of ODBC problems occur when these processes are called:

- Process SQL Statements
- Receive Results

13.4.9 Activating sql.log

To activate sql.log:

1. From the Microsoft Windows Control Panel, select 32bitODBC.
2. On Data Sources, click Options.

   Note: Leave the Stop Tracing Automatically option selected. Because this log grows rapidly, we recommend that you stop the trace in this way to preserve disk space resources and CPU cycles.

   Ensure that Trace ODBC Calls is cleared when you are not debugging. The log files can consume large amounts of disk space as well as CPU cycles.

13.4.10 Troubleshooting ODBC Problems Using sql.log

To troubleshoot ODBC problems using sql.log:

- Ensure that the data source names are set up correctly (as system data sources) and that a driver has been set up in the 32bitODBC in Control Panel.
- Make sure that Client Access has the correct parameters.
- Ensure that the library to which you are pointing is set up correctly.
- Look for these ODBC error messages in jde.log and jdedebug.log:
  - Table not in library

    If the table that is specified couldn't be found in the specified location, go to the appropriate DBMS and attempt to locate the table.

    If the table does not exist, you must generate the table.

    If the table exists but has been moved, you must change the data source to point at the new library.
  - Not Binding Column Data Types

    This error message means that the row is in use and that another program has a lock on that data. As a result, you cannot use this row until the program that is currently using it releases it.

13.4.11 Troubleshooting with the jdeinst.log

Use jdeinst.log to view the status of the JD Edwards EnterpriseOne silent installation. The silent installation mode enables you to submit a workstation installation request through command line arguments. JD Edwards EnterpriseOne creates a log file that records error conditions that were encountered during the silent installation, and it

indicates whether the silent installation was successful. The jdeinst.log file is located in the root directory of the workstation.

14

Troubleshooting the Enterprise Server

This chapter contains the following topics:

14.1 Understanding Enterprise Server Troubleshooting

Using these techniques, you can troubleshoot batch applications and business functions that process on the Oracle JD Edwards EnterpriseOne enterprise server. Platform-specific procedures are presented in other sections of this guide.

You might encounter these types of general problems on a JD Edwards EnterpriseOne enterprise server. The information presented applies to all operating systems:

- Communication failure when submitting a UBE or when trying to run business function logic on the server.
- Error message appearing at the bottom of a form (press F8 or click Bitmap to view an error description).

You should be familiar with the various logs used to troubleshoot problems on the server. Using these logs, you can troubleshoot batch applications and business functions that are executing on the enterprise server.

Types of Enterprise Server Log Files

In general, logs on JD Edwards EnterpriseOne enterprise servers are classified as either logic processing logs or batch processing logs.

Logic Processing Logs

You can use these two major log file sources for troubleshooting processing faults on the enterprise server:

- jde.log

  This log displays fatal errors. It can track any fault that might occur within JD Edwards EnterpriseOne.

- jdedebug.log

  This log tracks API calls and SQL statements as well as other messages. You can use this file to determine the point in time when normal execution stopped. The system does not use jdedebug.log to track errors; instead, this log is used to track the timing of JD Edwards EnterpriseOne processes.

Batch Processing Logs

You can use the batch process log to identify faults in JD Edwards EnterpriseOne processing related to batch processes. This log can contain event rule (ER) references, batch application process flow, and SQL statements, as well as other messages.

14.1.1 The Enterprise Server jde.log File

You can use the enterprise server jde.log to track fatal error messages generated by batch applications and business functions that are executing on the enterprise server. The jde.log tracks any fault that might occur within JD Edwards EnterpriseOne. When you are looking for startup errors, you should read the jde.log from the top down. For other errors, you should read from the bottom up.

If jde.log is enabled, a uniquely identified log file is created each time you start a JD Edwards EnterpriseOne job (including JD Edwards EnterpriseOne startup) on the enterprise server. These logs are associated with an enterprise server process ID (Job Number for IBM i).

The process ID (Job Number for IBM i) is appended to the file name, before the .log extension, with an underscore character (for example, jde_442.log).

jde.log File Creation

The enterprise server jde.log file is created (if it does not exist) or overwritten (if it exists) at the start of every JD Edwards EnterpriseOne session. For a Microsoft Windows enterprise server jde.log file, JD Edwards EnterpriseOne appends new information to the end of the jde.log.

Troubleshooting: Enabling and Disabling jde.log

Normally, the enterprise server should be set to enable the jde.log and disable the jdedebug.log. This example has combinations for the jde.ini parameter setting for enabling or disabling server logs.

Enable jde.log

This is an example of the jde.log file with debug logging enabled:

```
[DEBUG]
Output=NONE
LogErrors=1
JobFile=valid location/name (1)
DebugFile=valid location/name (2)
```

Enable jde.log and jdedebug.log

This is an example of the jde.log file with debug logging enabled and output to a file:

```
[DEBUG]
Output=FILE
LogErrors=1
JobFile=valid location/name (1)
DebugFile=valid location/name (2)
```

Disable jde.log

This is an example of the jde.log file with debug logging disabled:

```
[DEBUG]
Output=NONE
JobFile=blank/invalid location/name (1)
DebugFile= blank/invalid location/name (2)
```

Files and members generated by the jde.log will be located in JobFile. JD Edwards EnterpriseOne uses these naming conventions:

```
jde_process_ID.log
```

Where `jde` is the file or member name prefix, `process_ID` is a uniquely named process ID, and `log` is the file or member suffix or extension.

For non-IBM i enterprise servers, files generated by the jdedebug.log will be located in DebugFile. JD Edwards EnterpriseOne uses these naming conventions:

```
jdedebug_process_ID.log
```

Where jdedebug is the file name prefix, process_ID is a uniquely named process ID, and log is the file suffix or extension.

> **Note:** Verify whether the paths for the JobFile and the DebugFile settings are valid. If the paths for these settings are invalid, JD Edwards EnterpriseOne does not create logs.

For IBM i enterprise servers, the members generated by jdedebug will be located in DebugFile. JD Edwards EnterpriseOne uses these naming conventions:

```
jdedebug_process_ID
```

Where `jdedebug` is the file name prefix and `process_ID` is a uniquely named process ID.

Troubleshooting: Recommendations for the Enterprise Server jde.log

You can create a normal (successful) jde.log by signing on to JD Edwards EnterpriseOne and then immediately signing off. Use this log of successful startup statements to compare against logs that have a problem.

You can also rename the log to indicate the nature of the problem. For example, you might delete the jde.log and then run a report that causes an error condition. Then you could rename the jde.log to report.log.

If you are the only user running an instance of JD Edwards EnterpriseOne, you can add comment lines to the jde.log indicating the sequence of events you are performing. For example, you might be running an application that you know causes an error. Before you run the application, you could edit the jde.log to add a comment line stating you are about to start the suspect application.

Troubleshooting: Recommendations for Setting Up Server Locations

JD Edwards EnterpriseOne recommends that you create a separate directory on the enterprise server for logs. You should set up the jde.ini file to explicitly direct log files to that directory. For jde.log, the location and name of the log file are controlled by this default setting:

```
[DEBUG]
JobFile=jde.log
```

Files generated by the jde.log are located in JobFile. JD Edwards EnterpriseOne uses this syntax for naming files:

```
jde_process_ID.log (jde_jobnumber.log for IBM i)
```

If you do not specify a location, JD Edwards EnterpriseOne places the log files in the directory where you ran the JD Edwards EnterpriseOne startup executable (the default). On a UNIX machine, if you start JD Edwards EnterpriseOne with these commands and if logging is enabled, the system places the log files in the /u13/JDEdwards/E900/system/bin32 directory:

```
cd /u13/JDEdwards/E900/system/bine32
RunOneWorld.sh
```

If you start JD Edwards EnterpriseOne with these commands and if logging is enabled, the system places the log files in the /usr/JDEdwards directory because that is the working directory:

```
cd /usr/JDEdwards

/u13/JDEdwards/E900/system/bin32/RunOneWorld.sh
```

If you set up the UNIX machine to automatically start JD Edwards EnterpriseOne when the machine is started, it is especially important that you specify the full path of the log file in the jde.ini file.

Naming Conventions for jde.log

JD Edwards EnterpriseOne processes create logs as jde_processID.log (jde_JobNumber.log for IBM i), where processID is the process ID of the process that creates the log.

Non-IBM i JD Edwards EnterpriseOne processes move logs for batch jobs to the PrintQueue directory and rename them as report_version_date_time.log, where report is the report name and version is the version name; for example, R014021_XJDE0001_D990312_T161854215.log.

Example: Enterprise Server jde.log

This example of the jde.log from the enterprise server displays errors caused by signon tables that were not properly closed after fetching data. Normally, the only way this can happen is if a business function program did not close the table. Therefore, generated code applications cannot have this problem.

Most entries in the jde.log file are significant, and you should examine them closely. This information is also used by developers to indicate problems with the application that need to be addressed.

Troubleshooting: Recommendations for the Enterprise Server jde.log when a fatal crash occurs

If a fatal crash occurs on a JD Edwards EnterpriseOne Windows Server the Call Stack will be automatically be dumped into the jde.log file. This information in the jde.log file will contain a fully qualified path to the system install location. Therefore, you should take the necessary steps to ensure that the install path information is secured.

14.1.2 The Enterprise Server jdedebug.log File

You can use the enterprise server jdedebug.log to determine the point in time when normal execution stopped. The system does not use jdedebug.log to track errors. Instead, it uses this log to track the timing of JD Edwards EnterpriseOne processes. The log contains API calls and SQL statements as well as other messages.

You can use jdedebug.log to find out where a process ended. For example, log data can include what ODBC was trying to connect to, the SQL statement that was being executed for a specific table, and whether memory has been freed.

If jdedebug is enabled, each jdenet_n job and batch process started on a server creates a uniquely identified jdedebug.log. These logs are associated with an enterprise server process ID. Each time JD Edwards EnterpriseOne is started on the enterprise server and each time a batch process job is executed on the enterprise server, a new jdedebug.log is created.

For enterprise servers, the process ID (Job Number for IBM i) is appended to the file name with an underscore character before the .log extension. For example, the file name might be jdedebug_442.log. The enterprise server jdedebug.log is created (if it doesn't exist) or overwritten (if it exists) at the start of every JD Edwards EnterpriseOne session. For a Microsoft Windows enterprise server jde.log file, JD Edwards EnterpriseOne appends new information to the end of jde.log.

Note: Server administrators are responsible for clearing and deleting jde.log and jdedebug_*.log files from the enterprise server.

Troubleshooting: Reading the jdedebug.log

If the process failed and you have logging turned on, look in the jdedebug.log for these messages:

- Not Found
- Failure

Also, look at the end of the log to see what task was executed last. In general, important lines in the log are:

- SELECT

  The SELECT lines indicate which table you are selecting. The log tells you where the table resides. For the IBM i, this location is a library. For non- IBM i servers, this location is an environment. You should verify that the selected libraries and environments are correct.

- ODBC Version

  The ODBC lines indicate whether you are having problems connecting to the driver.

Troubleshooting: Enabling and Disabling jdedebug.log

Normally, the enterprise server should be set to enable the jde.log and disable the jdedebug.log. This example has valid setting combinations for enabling or disabling server jdedebug.log.

Troubleshooting: Enabling and Disabling jdedebug.log

These are the settings for enabling the jdedebug.log file:

```
[DEBUG]
Output=FILE
```

```
LogErrors=1
JobFile=valid location/name (1)
DebugFile=valid location/name (2)
```

Enable jde.log and jdedebug.log

These are the settings for enabling the jde.log and jdedebug.log files:

```
[DEBUG]
Output=BOTH
LogErrors=1
JobFile=valid location/name (1)
DebugFile=valid location/name (2)
```

Disable jdedebug.log

These are the settings for disabling the jdedebug.log file:

```
[DEBUG]
Output=NONE
LogErrors=0
JobFile=valid location/name (1)
DebugFile=valid location/name (2)
```

The [DEBUG] section of the jde.ini file contains the files and members generated by the jde.log. JD Edwards EnterpriseOne uses these naming conventions:

```
jde_<pid>.log
```

Where `jde` is the file or member name prefix and `<pid>` is a uniquely named process ID.

For enterprise servers, the files generated by the jdedebug.log will be located in the jde.ini file. JD Edwards EnterpriseOne uses these naming conventions:

```
jdedebug_<pid>.log (jdedebug_<JobNumber>.log)
```

Troubleshooting: Recommendations for the Enterprise Server jdedebug.log

You can create a normal (successful) jdedebug.log (JDEDEBUG for IBM i) by logging on to JD Edwards EnterpriseOne and then immediately logging off. Use this log of successful start up statements to compare against logs that have a problem.

You can also rename the log to indicate the nature of the problem. For example, you might delete the jdedebug.log and then run a report that causes an error condition. Then you could rename the jdedebug.log to report.log.

Another alternative is to add comment lines to the jdedebug.log that indicate the sequence of events you are performing. For example, you might be running an application that you know causes an error. Before you run the application, you could edit the jde.log to add a comment line stating that you are about to start the suspected application.

Troubleshooting: Recommendations for Setting Up Server Locations

We recommend that you create a separate directory on the enterprise server for logs. You should set up the jde.ini file to explicitly direct log files to that directory. For jdedebug.log, these setting controls the location:

```
 [DEBUG]

DebugFile=jdedebug.log
```

For enterprise servers, the files generated by the jdedebug.log will be located in DebugFile. JD Edwards EnterpriseOne uses these naming conventions:

```
jdedebug_process_ID.log (jdedebug_JobNumber.log for IBM i)
```

Where `jdedebug` is the file name prefix and `process_ID` is a uniquely named process ID.

By default, JD Edwards EnterpriseOne places the log files in the directory where you ran the startup executable. For example, on a UNIX machine, if you start JD Edwards EnterpriseOne with this command:

```
cd /u13/JDEdwards/E900/system/bin32 RunOneWorld.sh
```

and assuming that logging is enabled, the system places the log files in the /u13/JDEdwards/E900/system/bin32 directory. Similarly, on a UNIX machine, if you start JD Edwards EnterpriseOne with this command:

```
cd /usr/JDEdwards /u13/JDEdwards/E900/system/bin32 RunOneWorld.sh
```

and assuming that logging is enabled, the system places the log files in the /usr/JDEdwards directory. This is the working directory. If you set up the UNIX machine to automatically start JD Edwards EnterpriseOne when the machine is booted, it is especially important that you specify the full path of the log file.

Naming Conventions for jdedebug.log on the Enterprise Server

JD Edwards EnterpriseOne processes create logs as jdedebug_process_ID.log, where process_ID (Job Number for IBM i) is the process ID of the process creating the log. For example, a batch report running on a UNIX server as process 123456 would produce a file named jdedebug_123456.log.

14.1.3 The Batch Process Log File

Whenever you run a batch process requested from a workstation, an individual log file is created in the JD Edwards EnterpriseOne print queue directory (E900\PrintQueue) on that workstation. For any batch process request issued from a workstation, this file is created even if you have specified that the batch process report is to run on the enterprise server. For batch processes requested from a server, the jdedebug.log file is created on the server in the print queue directory.

Based on the setting of the UBESaveLogFile parameter in the [UBE] section of the jde.ini file, this log file is deleted or saved on successful completion of batch processes. This log file displays different types of messages that can help track errors in the batch process. The messages are:

- Section Level Process
- Object Level Process
- ER Level Process
- DB Level Process

The batch process log can contain ER references, batch process flow, and SQL statements, among other messages. You can use the batch process log file to determine when normal execution stopped.

The batch process log file displays the process flow in batch processes. This example describes the event flow within the batch engine and provides sample messages that would be written to the log at each point in the event flow, assuming UBEDebugLevel

is set to 6. Note that each message written to the log file displays the error level of that message in brackets. For example, -UBE--[2]-indicates a section-level message.

When a UBE processes a section, it begins by opening the business view for that section within the INIT section event. As a result, a SELECT statement follows directly after the INIT section for each section.

```
--UBE--[2]-- 355/392 Process Init Section
--UBE--[2]-- 355/392 InitSection for Business Unit Report Driver
--UBE--[2]-- 355/392 InitSection for Business Unit Report LBH
--UBE--[4]-- 355/392 SELECT T0.MCMCU, T0.MCSTYL, T0.MCLDM, T0.MCCO, T0.MCAN8,
T0.MCCNTY, T0.MCADDS, T0.MCFMOD, T0.MCDL01, T0.MCDL02, T0.MCDL03, T0.MCDL04,
T0MCRP01, T0.MCRP02, T0.MCRP03, T0.MCRP04, T0.MCRP05, T0.MCRP06, T0.MCRP07,
T0.MCRP08, T0.MCRP09, T0.MCRP10, T0.MCRP11, T0.MCRP12, T0.MCRP13, T0.MCRP14,
T0.MCRP15, T0.MCRP16, T0.MCRP17, T0.MCRP18, T0.MCRP19, T0.MCRP20, T0.MCRP21,
T0.MCRP22, T0.MCRP23, T0.MCRP24, T0.MCRP25, T0.MCRP26, T0.MCRP27, T0.MCRP28,
T0.MCRP29, T0.MCRP30, T0.MCPECC, T0.MCALS, T0.MCALCL, T0.MCSBLI, T1.CCCO,
 T1.CCNAME,
T1.CCRCD FROM F0006 T0,F0010 T1 WHERE ( T1.CCCO=T0.MCCO ) ORDER BY T0.MCCO ASC,
T0.MCMCU ASC
```

After INIT Section, the engine calls Advance Section to retrieve a record from the SELECT statement:

```
--UBE--[2]-- 355/392 Process Adv Section
--UBE--[2]-- 355/392 Processing Adv Section for Page Header
```

After the retrieve, the engine performs the DO Section processing. This includes any event rules attached to the DO Section event:

```
--UBE--[2]-- 355/392 Process DO Section
--UBE--[2]-- 355/392 Processing DO Section for Page Header
--UBE--[4]-- 355/392 --ER: Line(1): Loading Data Structure for BSFN
--UBE--[4]-- 355/392 --ER: Line(1): Processing BSFN : GetCompanyAndReportDesc
--UBE--[4]-- 355/392 --ER: Line(1): Done Processing BSFN : GetCompanyAndReportDesc
--UBE--[4]-- 355/392 --ER: Line(1): Unloading Data Structure for BSFN
--UBE--[4]-- 355/392 --ER: Line(1): Done Processing ER BSFN
```

Within DO Section, each object is processed and eventually printed in INIT, DO, and END object order:

```
--UBE--[3]-- 355/392 Process Init Object
--UBE--[3]-- 355/392 Processing Init Item SystemTime in Section Page Header
--UBE--[3]-- 355/392 Process DO Object
--UBE--[3]-- 355/392 Processing Do Object SystemTime in Section Page Header
--UBE--[6]-- 355/392 Printing Object Value = 14:35:46
--UBE--[3]-- 355/392 Process End Object
--UBE--[3]-- 355/392 Process Init Object
--UBE--[3]-- 355/392 Processing Init Item SystemDate in Section Page Header
--UBE--[3]-- 355/392 Process Do Object
--UBE--[3]-- 355/392 Processing Do Object SystemDate in Section Page Header
--UBE--[6]-- 355/392 Printing Object Value = 3/6/00
--UBE--[3]-- 355/392 Process End Object
```

After all the objects for a section have been processed, the engine calls Process Last Object and then begins processing for the next section in the report:

```
--UBE--[3]-- 355/392 Processing Do Object
ModelAccountsandConsolid in Section Page Header
--UBE--[6]-- 355/392 Printing Object Value = MD
--UBE--[3]-- 355/392 Process End Object
--UBE--[3]-- 355/392 Process Last Object
```

```
--UBE--[2]-- 355/392 Process End Page Header Section
--UBE--[2]-- 355/392 Process Do Section
--UBE--[2]-- 355/392 Process Do Section for Business Unit Report Driver
```

When all sections have been processed, if the report finishes without errors, these messages are displayed at the end of the log:

```
--UBE--[6]-- Successfully Finishing Engine
...
UBE Job Finished Successfully.
```

The level of detail provided by the batch process log is controlled by the UBEDebugLevel parameter of the jde.ini file. These are values for UBEDebugLevel:

Value	Description
0	No error messages.
3	Object-level messages.
4	Event rule messages and SQL statements (plus levels 1-3).

See Also:

- "Working with Servers" in the *JD Edwards EnterpriseOne Tools System Administration Guide*.
- Understanding Executable Files on the Workstation.
- Understanding Server Administration for IBM i.

14.2 Viewing Enterprise Server Logs from the Workstation

You must log on to the server to view logs for the server. You can also view portions of log files from the workstation that initiated the calls to the server.

To view server logs from the workstation:

1. In the [DEBUG] section of the enterprise server jde.ini file, set the ClientLog parameter to **1**.

   This setting enables the server to send logs to workstations. For example:

   ```
   [DEBUG]

   ClientLog=1
   ```

2. In the [DEBUG] section of the Workstation jde.ini file, set the ServerLog parameter to **1**.

   This setting enables the workstation to receive log information from the enterprise server. For example:

   ```
   [DEBUG]

   ServerLog=1
   ```

14.3 Setting Up the Enterprise Server jde.log

To set up the enterprise server jde.log:

1. Locate the enterprise server jde.log file (JDE member for IBM i) using Server Manager.
2. In Server Manager, in the Management Console, select the Logging hyperlink from the Configuration pane.
3. In the Error and Debug Logging pane, enable or disable the logging of errors to the jde.log file by modifying the Enable JDE.LOG field.

Setting	Purpose
Enable JDE.LOG	A parameter controls whether the logging function is enabled. Valid values are: ■ Disabled (Default) ■ Enabled

4. Click the Apply button to save the changes.

See Also:

- 8.98 Server Manager Guide on My Oracle Support..

14.4 Setting Up the Enterprise Server jdedebug.log

To set up the enterprise server jdedebug.log:

1. Locate the enterprise server jdedebug.log file (JDE member for IBM i) using Server Manager.
2. In Server Manager, in the Management Console, select the Logging hyperlink from the Configuration pane.
3. In the Error and Debug Logging pane, verify or change the name for the debug file using the JDEDEBUG.LOG Filename parameter field.

   The JDEDEBUG.LOG Filename specifies the name of the jdedebug.log file (JDEDEBUG member for IBM i). For non-IBM i enterprise servers, the default value is jdedebug.log. For IBM i enterprise servers, the default value is JDEDEBUG.
4. In the Error and Debug Logging pane, enable or disable the logging of errors to the jde.log file by modifying the Enable JDE.LOG field.

Setting	Purpose
Enable Debug Logging	A parameter controls whether the logging function is enabled. Valid values are: ■ NONE — No Debug Logging – FILE — Enable Debug Logging – ■

5. Click the Apply button to save the changes.

See Also:

- 8.98 Server Manager Guide on My Oracle Support..

14.5 Setting Up the <batch process>.log File

To set up the <batch_process>.log file:

1. Locate the workstation jde.ini file.

   The JD Edwards EnterpriseOne setup program places this file in the working Windows directory (for example, c:\WINNT40\jde.ini). If you are unsure of the workstation's working Microsoft Windows directory, use the Find command to locate the jde.ini file.

2. Use an ASCII editor (such as Microsoft Notepad or Microsoft Wordpad) to open the file.

3. Set the level of batch report debugging information that you want written to the batch process log file and whether you want the file to be saved.

   These settings are controlled by these parameters in the [UBE] section:

Setting	Purpose
UBEDebugLevel=	A parameter that specifies the level of UBE debug logging. Valid values are: ■ 0 (default): No error messages. ■ 1: Warnings and high-level information. ■ 2: Section-level messages (plus Level 1 messages) ■ 3: ER messages and database mapping messages (plus Level 1-2 messages) ■ 4: SQL statements (plus Level 1-3 messages) ■ 5: Database output (plus Level 1-4 messages) ■ 6: Batch process function calls and printed output values (plus Level 1-5 messages)
UBESaveLogFile=	A parameter that specifies whether the batch_report.log file will be saved. Valid values are: ■ 0: The batch_report.log file is not saved. ■ 1: The batch_report.log file is saved in the workstation's JD Edwards EnterpriseOne print queue directory (E900\PrintQueue).

4. Save the changes and close the jde.ini file.

14.6 Troubleshooting the Enterprise Server

This section discusses how to:

- Troubleshoot general problems.
- Troubleshoot communication problems.
- Troubleshoot server map problems.

14.6.1 Troubleshooting General Problems

You can troubleshoot general enterprise server problems using the Server Manager which enables you to monitor server components, processes, and resources.

To troubleshoot general problems:

1. Use Server Manager to verify that you are looking at the correct port and the server is operational on that port.
2. Verify the netTrace setting in the enterprise server jde.ini file:

   ```
   [JDENET]

   netTrace=0/1 (disabled/enabled)
   ```

   When the variable netTrace=0, JD Edwards EnterpriseOne does not generate Net log information. When netTrace=1, JD Edwards EnterpriseOne generates Net log information.

   Note: Using Server Manager, you can turn logging on or off for a particular kernel process.

3. Return to JD Edwards EnterpriseOne and duplicate the problem.

   The trace facilities write debugging information to the jde.log and jdedebug.log files.
4. After running the business function again, look at the jde.log files on the server.

   Search for these message (you must search for lower case): "jdenet_n process."

   If you cannot find this message, bring the server down and back up. If you do find this message, look at the jde.log file with the same process ID as the net process.
5. Verify that the user is running in the correct environment or path code; for example PD900 or DV900.

   If this environment is not set up on the server, you receive errors on the workstation jde.log as well as the enterprise server jde.log.
6. In the jde.logs on the enterprise server, look for a JDENET_SendMSg Failed Error=12 message.

   This message means that the JDENET server is down and you must restart it.
7. In the jde.log file on non-IBM i enterprise servers, look for any "Unable to connect to Oracle" messages. Search on ORA-.

   If you find messages, they indicate problems connecting to Oracle. You get an indication of an Oracle connection problem if, in a business function, you select find/browse, data is not found, and no errors are received from the application. You need help from an Oracle database administrator at this point. To debug this problem, see the section in this document about sql.log.
8. Look in the jdexxx.log file (where xxx is the ID of the process that created the log) on the server for these message: "Could not find symbol in the <BSFN dll name>."

   If present, this message might mean that the business function did not build on the enterprise server.
9. If you have not found a problem indicating why you are unable to run an application on the enterprise server, you will need to debug it on the server.

Note: For Microsoft Windows enterprise servers, if you cannot identify a problem by reading the log, you need to put the business function through debug on the server. This action requires knowledge of C++ and how to debug. See Microsoft documentation for Debugging C++.

14.6.2 Troubleshoot Communication Problems

When you submit an application to an enterprise server through an override of the master business function set in Object Configuration Manager, you might experience communication problems with the enterprise server. The business function then runs locally on the client workstation. JD Edwards EnterpriseOne displays a window to inform you that the business function is running in a new location.

To troubleshoot communication problems:

Note: Use this procedure if JD Edwards EnterpriseOne displays a window to inform you that a business function is running in a new location.

1. Check the jde.ini on the workstation to make sure the JDENET service name (port number) is correct and valid.

   This port number must match the settings in the server jde.ini file, and the JD Edwards EnterpriseOne server must be running to successfully submit reports or to run business logic on a server. Security services and transaction management services also require the JD Edwards EnterpriseOne server to be running:

   ```
   [JDENET]

   serviceNameListen=service name

   serviceNameConnect=service name
   ```

   If serviceNameListen=service name specifies the communications service port on the TCP/IP network, JD Edwards EnterpriseOne uses this port address to listen for requests on the network. Using a file called services, you can associate the port number with a unique name. The default value is jde_server (port number 6003).

   If serviceNameConnect=service name specifies the communications service port on the TCP/IP network, JD Edwards EnterpriseOne uses this port address to connect to the network. Using a file called services, you can associate the port number with a unique name. The default value is jde_server (port number 6003).

2. On the workstation, exit JD Edwards EnterpriseOne and turn logging on in the jde.ini.

3. Run the application on the server again, and then check the jde.log file to see if any of these errors are logged:

   – JDENET_SendMsg Failed Error=8

     This error can mean you are not using the correct TCP/IP service port or that the enterprise server does not have that JDENET listing.

   – JDENET_SendMsgFailed Error=5, 11, or 12

These errors can mean that the message is being sent to the correct port, but the enterprise server JDENET is down.

4. From within Server Manager, change the port address to determine if both the workstation and server are using the same port.
5. Check the services file on the workstation (located in the operating system directory\System32\drivers\etc for Windows).

   Ensure that a blank line exists at the end of the file and that you have the service name mentioned in Step 1 (for example, jde_server) going to the correct port address on the server. Verify the port address with the server administrator.
6. If you receive a Communication Failure message, try resubmitting the application.

   A time-out may have occurred.

   ```
   [JDENET]

   netTrace=0/1 (disabled/enabled)
   ```
7. Look in the log file for this message:

   ```
   Could not find symbol in the <BSFN dll name>
   ```

14.6.3 Troubleshooting Server Map Problems

If you change the Object Configuration Manager or the Data Source Master files in the Server Map data source, you can test the changes using the PORTTEST program. This test is designed to validate the environments.

See the section specific to the platform type for more information about the PORTTEST program.

14.7 Troubleshooting the Enterprise Server Processes

This section provides an overview of resource utilization and performance and discusses how to evaluate JD Edwards EnterpriseOne server performance.

14.7.1 Understanding Resource Utilization and Performance

This section discusses:

- Requirements
- Configuration Setup

The EnterpriseOne Kernel can sometimes encounter certain complex issues which have proved intractable to resolve. Some of these intractable issues are Kernel Crashes, Deadlocked UBEs, or OutMemory conditions that require specialized code to determine the root cause. In addition, the kernel and batch processes utilize resources and place demands on memory and the CPU and execute in the context of other processes that also demand memory and CPU resources. Each process has its own level of impact on the system, and the cumulative effect of all processes currently running will impact performance. Appropriate data can be captured and used with measurement tools to evaluate overall performance as well as the impact of individual processes. When utilization or performance exceeds certain thresholds, it is critical to have tools that can diagnose which process is creating the resource overload in order to correct the problem and restore the performance to normal levels before the system crashes.

A new administration tool called Kernel Resource Management (KRM) has been added to Server Manager to enable you to isolate and determine root causes of CPU and Memory issues, increase system stability and simplify the troubleshooting process. Graphs allow you to quickly identify processes with high resource consumption and recycling allows the ability to reclaim system resources.

There are graphs for the following:

- Summary Graphs for entire Enterprise Server that displays memory and CPU usage.
- At Enterprise Server level, there are graphs that display the Top 10 Processes by Memory and CPU.
- At the Individual Process Level there are graphs for CPU, Memory, caches and DB connections.

New diagnostics allows you to quickly identify root cause of resource consumption issues. For Diagnostics, information about the current resource usage is written to the jdedebug.log:

Cache information:

- Cache name
- Number of records in the cache
- Indices

Database Transactions:

- Commitment type
- User
- Application

KRM enables you to monitor kernel and batch processes and to diagnose CPU and memory usage issues.

14.7.1.1 Requirements

Kernel Resource Management requires JD Edwards EnterpriseOne 8.98.2.0 to be able to capture the diagnostic information. Server Manager 8.98.2.0 or above is required to display the diagnostic information.

If an older Server Manager version is used to monitor a newer Enterprise Server, the new diagnostic data will be captured but will not be viewable from Server Manager. Similarly, if a newer Server Manager version is used to monitor an older Enterprise Server, then the new diagnostic data will not be captured nor be viewable from Server Manager.

14.7.1.2 Configuration Setup

The Kernel Resource Management includes several new graphs of different attributes of the Enterprise Server instance. The interval of the data in these graphs is defined by Server Manager monitors. By default, each monitor collects up to 1440 data points. In the monitor configuration, the user can specify at what interval these data points will be collected. By default, the collection interval is 60 seconds. This default value allows the collection of 24 hours worth of data points. These are two graphs used to present the data for all CPU or memory processes. These graphs can be accessed by selecting an enterprise server in Server Manager.

If a less granular data set is desired, you can increase the collection interval. If more granular data set is desired, you can lower the collection interval. However, it is not recommended to set the collection interval less than 20 seconds. The embedded Server Manager agents (in the Enterprise Server, for example) are designed to collect runtime metrics every 20 seconds. Setting the monitor collection interval to less than 20 seconds will result in duplicate (or stale) data points as well as higher CPU usage on the Enterprise Server.

Note: When the Server Manager Console (SMC) services are stopped and restarted, the collection of data for these monitors is reset. Also, any changes made to the collection interval and to the number of data points are NOT maintained and the default value of 60 seconds is used.

Note: If there are mandatory multiple EnterpriseOne instances, one may want to reduce the number of collection data points and increase the collection frequency. For more information please refer to the solution document on My Oracle Support titled:

E1: SVM: Server Manager Console running out of memory with java.lang.OutOfMemory exception. 1082765.1. (Doc ID 1082765.1).

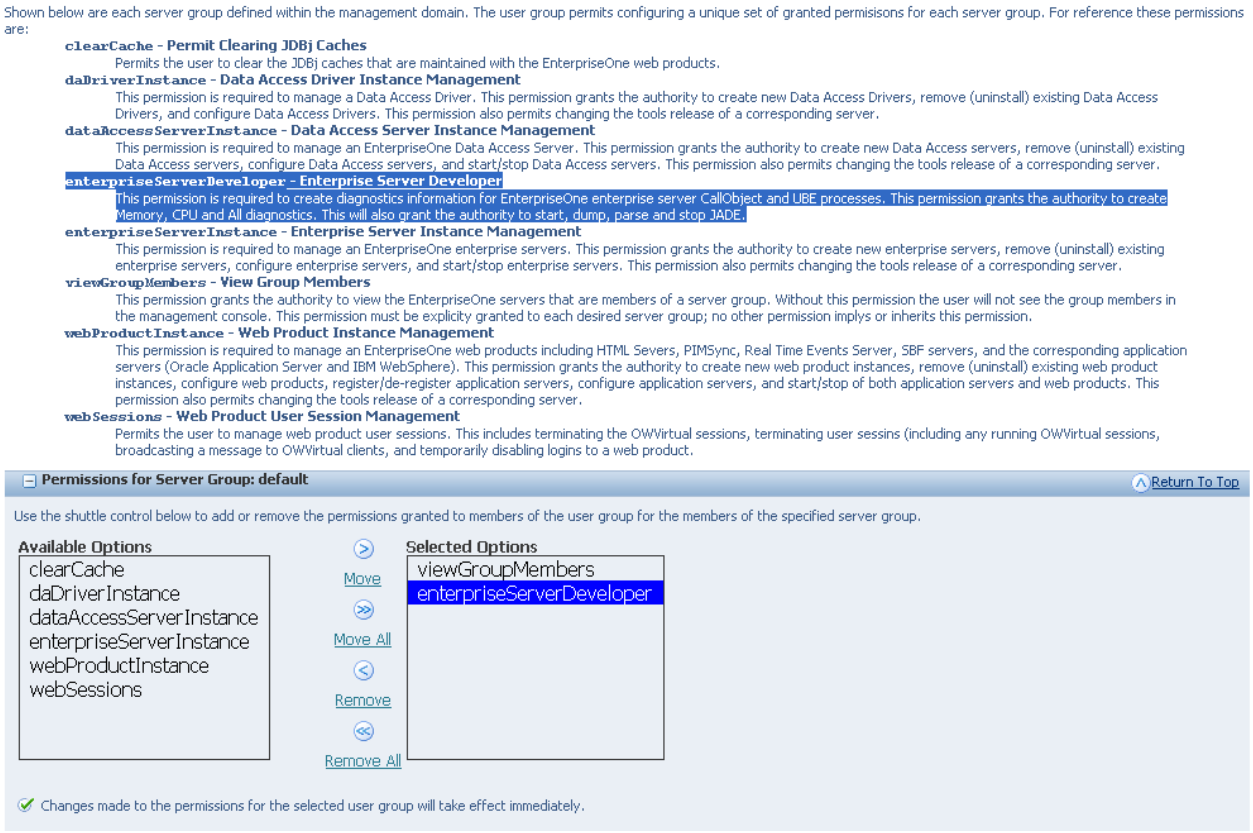
Overhead on Server Manger Console

The data for graphs is saved in XML format in <install_location>/jde_home/data. The XML files are relevant for graphical display purposes until the SMC is bounced. The stale XML files (from a previous run) may be purged or archived. Previously saved XML graph files cannot be viewed.

Server Manager Security Permission Role

A new server manager security permission called enterpriseServerDeveloper is available which will allow you to create memory, CPU and all diagnostics. This security permission will also allow you to start, dump, parse and stop JADE. The buttons used in the JADE section will be disabled if the Server Manager user does not have the enterpriseServerDeveloper Server Manager security permission. This security permission is always assigned to the jde_admin user.

Figure 14–1 Permissions for Server Groups



14.7.2 Evaluating EnterpriseOne Server Performance

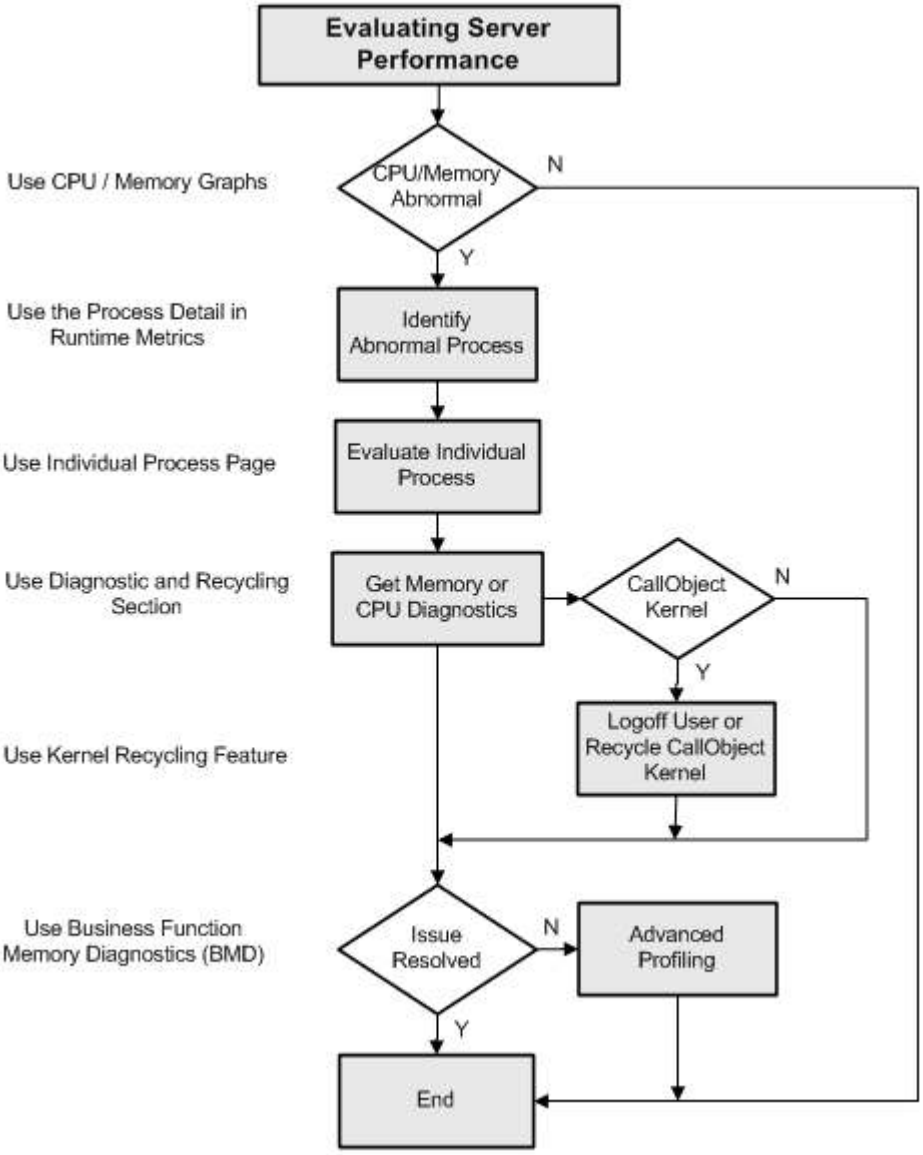
Kernel Resource Management is a set of diagnostic tools implemented in Server Manager 8.98.2.0 that utilizes historical data captured at specific intervals. You can use KRM to drill down through different levels of detail information and isolate the process, user, and attribute to determine the critical decision-making factors when diagnosing and troubleshooting memory problems and CPU issues. KRM functions as a central dashboard regardless of the platform that it is running on.

In order to evaluate JD Edwards EnterpriseOne server performance, you will need to follow a process that will isolate the problem. In this process you will:

- Determine if the CPU or memory is abnormal
- Identify the abnormal process
- Evaluate the individual process
- Get the memory or CPU diagnostics
- Log off or recycle
- Use Advanced Profiling

This is a flow chart of the process using KRM to diagnose the root cause of a problem:

Figure 14–2 Evaluating Server Performance



14.7.2.1 Determine if CPU or Memory is Abnormal

To help you determine what constitutes "abnormal" memory usage, new functionality has been added to the server command-line program porttest. This new functionality will simulate a memory leak until the process can no longer allocate memory. The output of the program will be the number of bytes that the porttest program was able to allocate before failure. This number can be used by the administrator to estimate when an EnterpriseOne process may fail due to excessive memory consumption.

The porttest program can be run in two modes: single-threaded (simulates runbatch and subsystem UBE's) and multi-threaded (simulates CallObject kernels). Since the memory model on the IBM i platform is single-level store, a maximum-limit cannot be determined in this way.

Note: If you are running EnterpriseOne Services on the IBM i platform, please work with your IBM representative to determine the maximum temporary storage that a job can use before failure.

Automatic Method for Memory Limit

To calculate the memory limits:

- Stop the EnterpriseOne server by clicking the Stop button.
- Click the Calculate Memory Limit button.
- Start the EnterpriseOne server by clicking the Start button.

Figure 14–3 General and Instance Properties section - before calculating memory limit

Figure 14–4 Instance Properties section - after calculating memory limit

What constitutes a normal CPU consumption is entirely dependent on the vendor sizing of the EnterpriseOne instance. As a rule of thumb, high sustained CPU usage could indicate under-sized hardware. Determining abnormal CPU usage is challenging because it depends on the load on the server and average usage patterns. For example, it may be normal for the CPU usage to drop below 10% overnight on an application server when no interactive users are on the system. However, if the server is used for nightly batch runs, then dropping below 10% overnight when the batch ube jobs are running might be abnormal. Likewise, it would probably be abnormal for the CPU usage to drop below 10% during the middle of the day on an application server. However, it might be normal if this occurred during lunch hour for most interactive users.

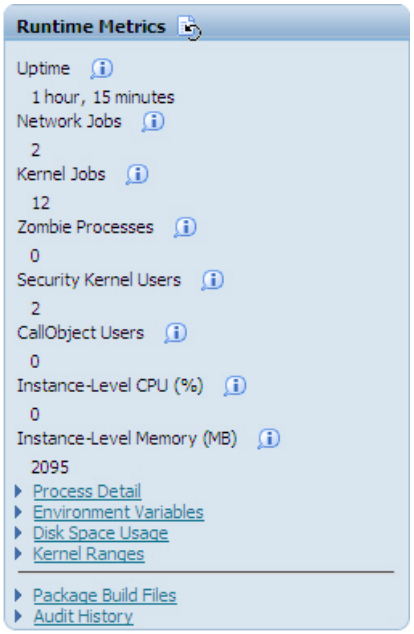
What is "normal" and "abnormal" will be different for each customer and the administrator may need to monitor their EnterpriseOne server usage for a month or so to get a feel for what their normal usage pattern is.

Runtime Metrics Section

Two metrics have been added to the Runtime Metrics section to provide information on Instance-level CPU and memory. The new metrics are:

Metric	Description
Instance-level CPU (%)	The total CPU usage of all processes running in this enterprise server instance.
Instance-level Memory (MB)	The total Memory usage of all processes running in this enterprise server instance.

Figure 14–5 Runtime Metrics Section



Instance-Level Summary Page

The instance-level summary page is used to evaluate instance-level data and has a new section called *Resource Charts - Sum of All EnterpriseOne* Processes. The instance level summary graphs are used to determine if the CPU or Memory have exceeded normal thresholds.

These are the navigation steps to access the instance-level summary page:

- Select the enterprise server instance in server manager.
- You should see the <EnterpriseOne Enterprise Server> instance.
- Make sure that the server is up and running.

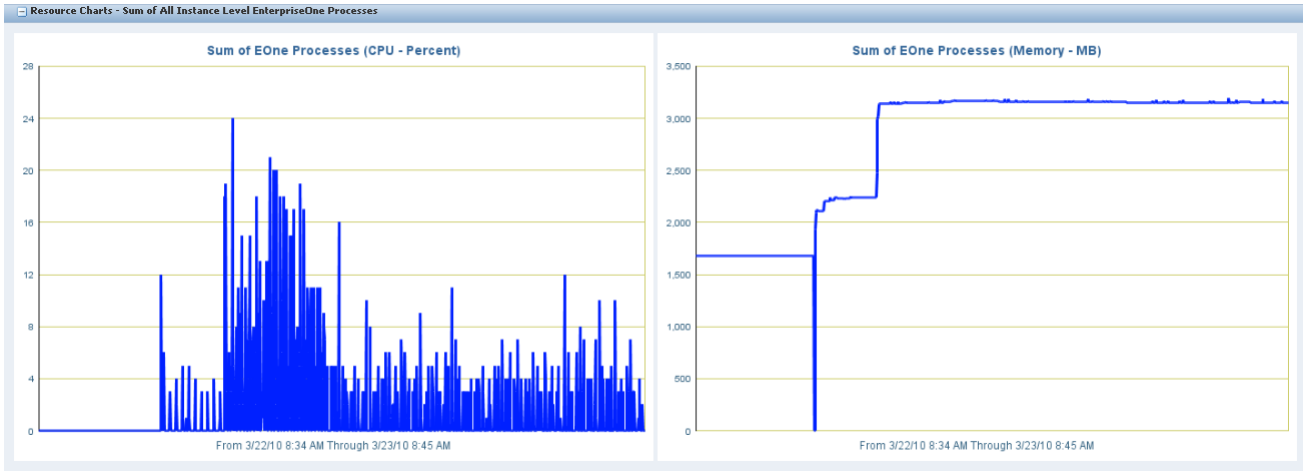
The instance-level summary page has the following sections:

- General and Instance Properties (not modified).
- Resource Charts - Sum of All Instance Level EnterpriseOne Processes (New).
- Available Log Files (not modified).

In the *Resource Charts - Sum of All Processes* section, the instance level summary graph displays the sums of *CPU Usage - Percent and Memory Usage - in MB* for all processes in the EnterpriseOne Server instance. The time-frame of the data collected is indicated

along the x-axis of the graphs. How much data is represented depends on when the Server Manager Console services were started and the collection intervals for the monitors that were specified (refer to Configuration Setup).

Figure 14–6 Resource Charts - Sum of All Instance Level EnterpriseOne Processes



If you want to get more information about processes in the EnterpriseOne Server instance and see where a likely problem might be, you can click on the "Process Detail" link on the navigation pane on the left-ehand side of the Server Manager screen.

14.7.2.2 Identify Abnormal Process

The Enterprise Server Processes page is used to identify abnormal processes through evaluating kernel-level data.

These are the navigation steps to access the enterprise server process page:

- Select the enterprise server instance in server manager.
- You should see the <EnterpriseOne Enterprise Server> instance.
- Make sure that the server is up and running.
- Go to the <Runtime Metrics> section.
- Click on the <Process Detail> link.
- You should see the <Enterprise server Processes> page.

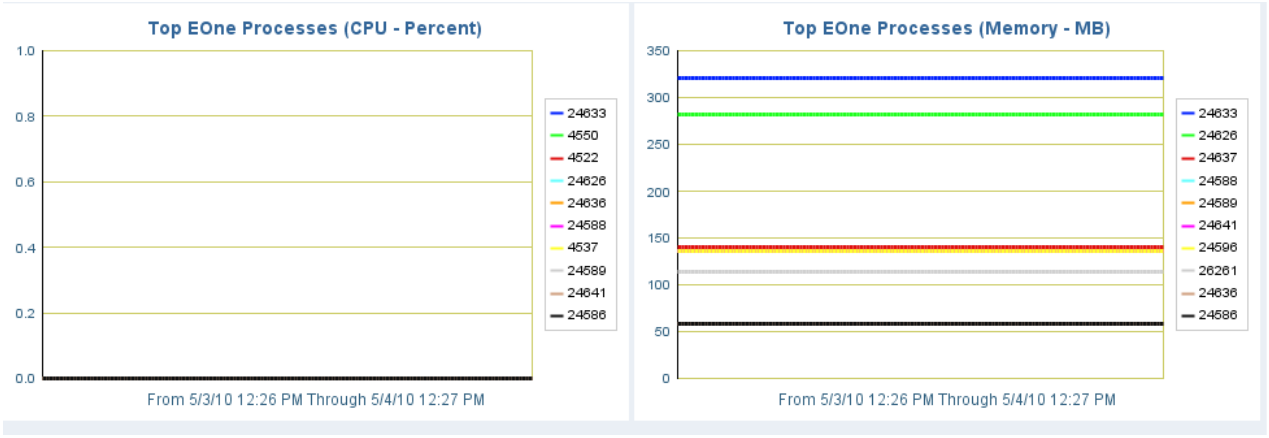
The enterprise server process has the following sections:

- Process and Batch Summary (not modified).
- Resource Charts – Top EnterpriseOne Processes (New).
- Processes (modified).

This page has a new section called *Resource Charts - Top EnterpriseOne Processes* and a modified section called *Process List*. The enterprise server process detail page will have the top ten individual processes for percent CPU usage and memory usage (in MB) designated in graph format. The top ten processes are chosen from all currently running EnterpriseOne Server processes. The time-frame of the data collected will be indicated along the x-axis of the graphs. How much data is represented will depend on when the Server Manager Console services were started and the collection intervals for the monitors that were specified (refer to Configuration Setup).

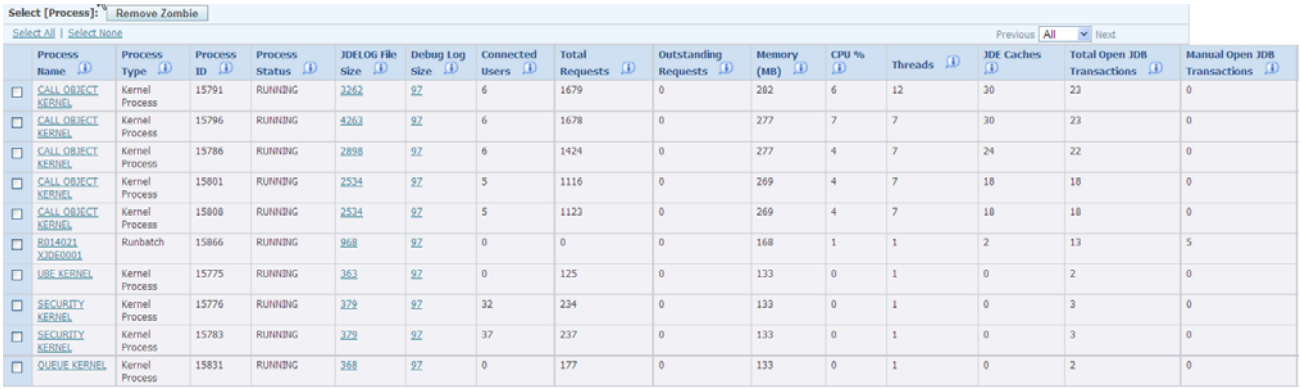
This graph enables the administrator to visually determine which process to troubleshoot, by providing an easy reference for the top memory or CPU consuming processes. The Process ID can be identified and then used to identify the process entry in the table in the Process List section to access more detail.

Figure 14–7 Resource Charts -Top EnterpriseOne Processes



Next on this page is a table of all Enterprise Server processes. Six columns have been added to this table that can be sorted on. This is the *Process List* section:

Figure 14–8 The Process List section.


Select [Process]: Remove Zombie

Select All | Select None

Previous All Next

	Process Name	Process Type	Process ID	Process Status	JDELOG File Size	Debug Log Size	Connected Users	Total Requests	Outstanding Requests	Memory (MB)	CPU %	Threads	JDE Caches	Total Open JDB Transactions	Manual Open JDB Transactions
☐	CALL OBJECT KERNEL	Kernel Process	15791	RUNNING	2262	97	6	1679	0	282	6	12	30	23	0
☐	CALL OBJECT KERNEL	Kernel Process	15796	RUNNING	4263	97	6	1678	0	277	7	7	30	23	0
☐	CALL OBJECT KERNEL	Kernel Process	15786	RUNNING	2898	97	6	1424	0	277	4	7	24	22	0
☐	CALL OBJECT KERNEL	Kernel Process	15801	RUNNING	2534	97	5	1116	0	269	4	7	18	18	0
☐	CALL OBJECT KERNEL	Kernel Process	15808	RUNNING	2534	97	5	1123	0	269	4	7	18	18	0
☐	R014021 XJDE0001	Runbatch	15866	RUNNING	968	97	0	0	0	168	1	1	2	13	5
☐	UBE KERNEL	Kernel Process	15775	RUNNING	363	97	0	125	0	133	0	1	0	2	0
☐	SECURITY KERNEL	Kernel Process	15776	RUNNING	379	97	32	234	0	133	0	1	0	3	0
☐	SECURITY KERNEL	Kernel Process	15783	RUNNING	379	97	37	237	0	133	0	1	0	3	0
☐	QUEUE KERNEL	Kernel Process	15831	RUNNING	368	97	0	177	0	133	0	1	0	2	0

The columns in the *Process List* are:

Field	Description
Process Name	The name of the kernel process. The name indicates which kernel definition the kernel belongs to.
Process Type	The description of the Enterprise Server process type.
Process ID	The operating system assigned process identifier for the kernel process.
Process Status	The current state of the processes. May be RUNNING, STOPPED, or ZOMBIE.
JDE Log File Size	The size of the error log file commonly referred to as jde.log for the enterprise server process. The size is specified in bytes.

Field	Description
Debug Log Size	The size of the debug log file commonly referred to as jdedebug.log for the enterprise server process. The size is specified in bytes.
Connected Users	The number of users that are currently connected to this kernel. This is applicable to security and callobject kernels only; other kernels do not maintain persistent connections with a particular user.
Total Requests	The total number of JDENET messages that have been processed by the process.
Outstanding Requests	The number of JDENET messages 9requests) that are queued up for the kernel process.
Memory (MB)* (applies to all E1 processes)	The virtual memory usage in megabytes for the process. An increase in this value over time could indicate a leak that needs to be analyzed.
CPU %* (applies to all E1 processes)	The amount of %CPU consumed by EnterpriseOne server process. The %CPU is reported directly for SUN and HP platform and is a calculated value in WINDOWS, DB2 for IBM i, AIX and LINUX platform. For example, on a 4-processor machine, a process using all of 1 CPU might be reported as using 25% or 100%.
Threads* (applies to all E1 processes)	The number of OS threads stared in the EnterpriseOne server process including any Java threads if the process loads a JDEJVM.
JDE Caches* (does not apply to jdenet_n processes)	The cache count for the process. This is the total number of JDE Caches in the EnterpriseOne server process which are opened by calling jdeCacheInit API. These caches are generally created by EnterpriseOne Business Functions. They are necessary for Business Functions to do their work. But an increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call jdeCacheTerminate API in business function code to fix the JDECache leak.
Total Open JDB Transactions* (does not apply to jdenet_n processes)	The total number of open JDB transactions which includes both AUTO and MANUAL transaction. This resource is opened by calling JDB_InitUser API which is either called with JDEDB_ COMMIT_AUTO or JDEDEB_COMMIT_MANUAL. An increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call JDB_FreeUser API in business function code to fix the JDB Transactions leak.
Manual Open JDB Transactions* (does not apply to jdenet_n processes)	The number of MANUAL open JDB transactions are opened by calling JDB_InitUser API with JDEDEB_COMMIT_MANUAL. The number of manual open JDB transactions is directly linked with manual-commit database connections. This number should be zero in a normal environment. If this number is growing then it may result in database connection leaks and may cause the enterprise database to reach the maximum number of connections which will bring the entire EnterpriseOne environment (enterprise server and JAS servers) to a halt and consequently no EnterpriseOne user can get their work done. The application developer should call JDB_ FreeUser API in business function code to fix the JDB Transactions leak.

Field	Description
Database Connections**	The total number of open Database Connections includes AUTOcommit, Select For Update (SFU) and MANUAL-commit connections. An increase in this value over time could indicate a leak that needs to be analyzed. The AUTO and SFU connections have a small maximum limit whereas the MANUAL commit connections have no limit and can grow really high and will eventually hit the maximum global connections available in the database server. The manual commit open Database Connections are tied with the number of open manual commit jdb transactions. The next step should be to capture Memory Diagnostics or All Diagnostics from the server manager process detail page. The JDB transactions for different user sessions logged in the process should be reviewed and all of the open manual JDB transactions should be reviewed for a suspect where the apps BSFN code or tools code has called JDB_InitUser with MANUAL COMMIT MODE but has not called the JDB_FreeUser. The JDB transactions has the file name, line number and function name which should help the developer to locate the code faster and review if the specific code does not call JDB_FreeUser at all or does not call when the code returns with an error or exception path.
JDE Cache Records**	This is the total number of JDE Cache Records in the EnterpriseOne server process for the specific user which is created by calling the jdeCacheAdd API, for example. These cache records are generally created by EnterpriseOne Business Functions. They are necessary for Business Functions to do their work. But an increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call the jdeCacheDelete (to delete cache records) or jdeCacheTerminate (to remove the entire cache) API in business function code to fix the JDE Cache Record leak.

Note: The items with an * have been added in the 8.98.2.0 release.

The items with an ** are new in the 8.98.3.0 release.

All of the columns in the table are sortable. To sort rows in the table based on a particular column, simply click the column header. Clicking the column header again will sort the rows in the table in the reverse order. Using the sort capability, the administrator can identify EnterpriseOne Server processes that are using an "abnormal" amount of memory, CPU, JDE caches, or open JDB transactions (manual-commit OR total).

If there is a process that the administrator wants to analyze in more detail, they should click on the link for the process (in the "Process Name" column) to be taken to the Individual Process page.

14.7.2.3 Evaluate Individual Processes

The individual process pages are used to evaluate user-level data to drill down further in order to diagnose and isolate issues.

These are the navigation steps to access the individual enterprise server process page:

- Select the enterprise server instance in server manager.
- You should see the <EnterpriseOne Enterprise Server> instance.
- Make sure that the server is up and running.
- Go to the <Runtime Metrics> section.

- Click on the <Process Detail> link.
- You should see the <Enterprise server Processes> page.
- Select the link for specific process under the <Process Name> column.
- This will present you the <ProcessID> page.

The individual process page has the following sections:

- General Information (modified).
- Resource Charts (New).
- Connected Users (modified).
- Diagnostics and Recycling (New).
- Logging and Configurations (not modified).
- Thread Details (not modified).

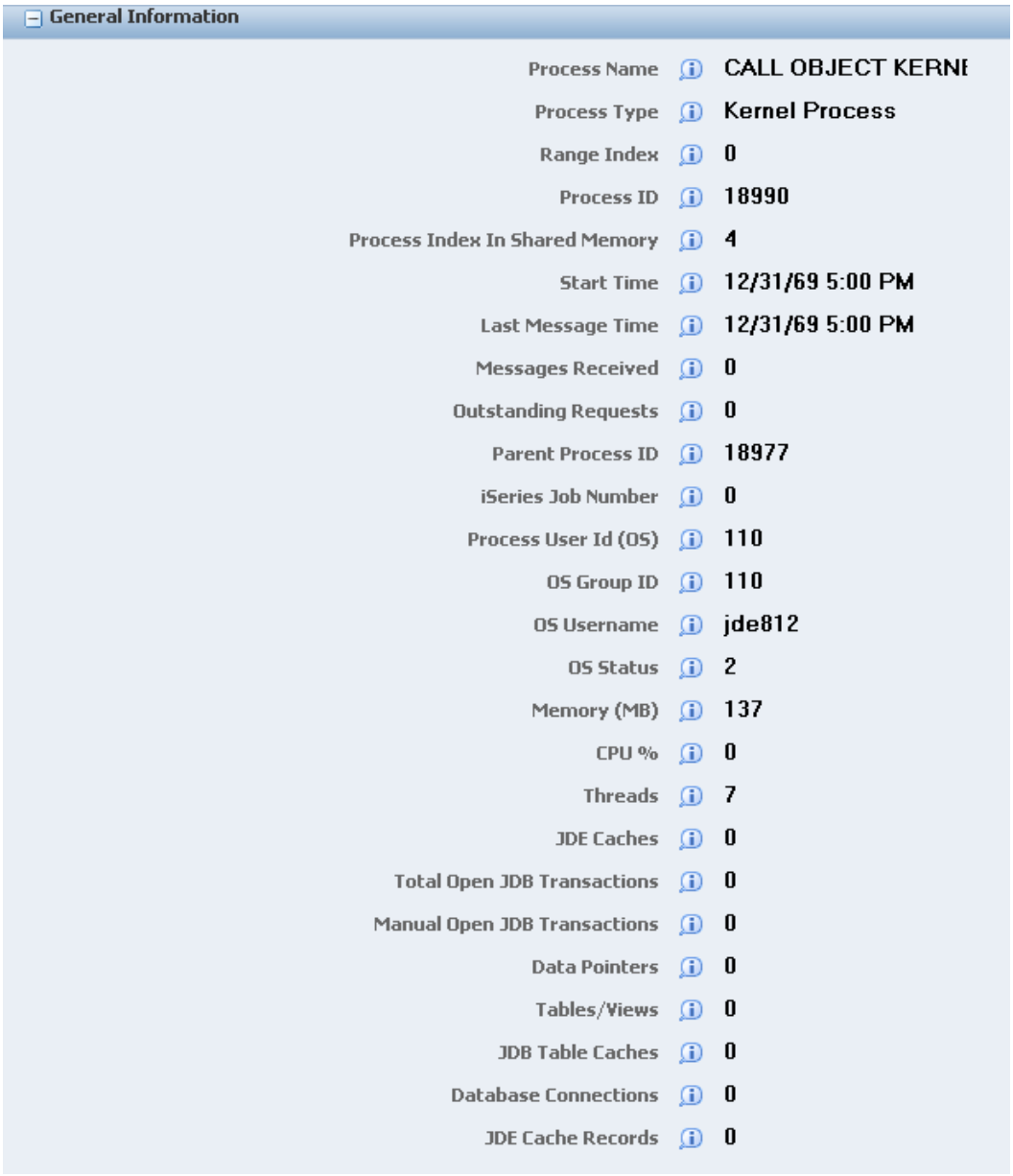
General Information Page

The enterprise server will have an individual process page for all E1 processes. There are several new data items as displayed in the graphic and table below.

The information is collected every 20 seconds. However, new data is presented only when the screen is refreshed.

The individual process page begins with a *General Information* section. The relevant information for the process selected is presented.

Figure 14–9 General Information section



The items in the General Information section are:

Field	Description
Process Name	The name of the kernel process. The name indicates which kernel definition the kernel belongs to.
Process Type	The description of the Enterprise Server process type.
Kernel Range	The kernel definition index. The Enterprise Server is composed of kernels that process messages from other servers and clients. There are more than thirty different types of kernels. The range index indicates which kernel group this kernel belongs to.
Process ID	The operating system assigned process identifier for the kernel process.

Field	Description
Process Index In Shared Memory	An internal identifier used to locate the process's position in the shared memory resources that track kernel and network processes.
Start Time	The time the process was created.
Last Message Time	The last time the kernel performed any activity such as processing incoming JDENET messages.
Messages Received	The total number of messages (requests) that have been processed by the kernel process.
Outstanding Requests	The number or requests that are queued and are waiting to be processed by the kernel process.
Parent Process ID	The operating system assigned process identifier of the parent process.
IBM i Job Number	The job number of the process, valid on the IBM i platform only.
Process User ID (OS)	The operating system user id under which the process is running.
OS Group ID	The group identifier of the os user running the process; valid only on unix based platforms.
OS Username	The operating system user name under which the process is running.
OS Status	The status of the process as reported by the operating system: 0 = Sleeping 1 = Running 2 = Stopped 3 = Zombie 4 = Other
Memory (MB)* (applies to all E1 processes)	The virtual memory usage in megabytes for the process. An increase in this value over time could indicate a leak that needs to be analyzed.
CPU %* (applies to all E1 processes)	The amount of %CPU consumed by EnterpriseOne server process. The %CPU is reported directly for SUN and HP platform and is a calculated value in WINDOWS, DB2 for IBM i, AIX and LINUX platform. For example, on a 4-processor machine, a process using all of 1 CPU might be reported as using 25% or 100%.
Threads* (applies to all E1 processes)	The number of OS threads stared in the EnterpriseOne server process including any Java threads if the process loads a JDEJVM.
JDE Caches* (does not apply to jdenet_n processes)	The cache count for the process. This is the total number of JDE Caches in the EnterpriseOne server process which are opened by calling jdeCacheInit API. These caches are generally created by EnterpriseOne Business Functions. They are necessary for Business Functions to do their work. But an increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call jdeCacheTerminate API in business function code to fix the JDECache leak.

Field	Description
Total Open JDB Transactions* (does not apply to jdenet_n processes)	The total number of open JDB transactions which includes both AUTO and MANUAL transaction. This resource is opened by calling JDB_InitUser API which is either called with JDEDB_COMMIT_AUTO or JDEDEB_COMMIT_MANUAL. An increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call JDB_FreeUser API in business function code to fix the JDB Transactions leak.
Manual Open JDB Transactions* (does not apply to jdenet_n processes)	The number of MANUAL open JDB transactions are opened by calling JDB_InitUser API with JDEDEB_COMMIT_MANUAL. The number of manual open JDB transactions is directly linked with manual-commit database connections. This number should be zero in a normal environment. If this number is growing then it may result in database connection leaks and may cause the enterprise database to reach the maximum number of connections which will bring the entire EnterpriseOne environment (enterprise server and JAS servers) to a halt and consequently no EnterpriseOne user can get their work done. The application developer should call JDB_FreeUser API in business function code to fix the JDB Transactions leak.
Data Pointers* (does not apply to jdenet_n processes)	This reports the number of data pointer slots out of 1000 slots which are used by the current EnterpriseOne server job. The business function code calls jdeStoreDataPtr API to allocate a data pointer slot. The slot index starts at 1001 and goes till a maximum of 2000 for any enterprise server process. The data pointer leak should be fixed by the application developer in the business functions by calling jdeRemoveDataPtr API other wise they will eventually receive the hard error where the process will no not create any new data pointers.
Tables/Views* (does not apply to jdenet_n processes)	This report the number of JDB Table handles or JDB View handles opened in the EnterpriseOne server job. This resource is opened by calling JDB_OpenTable and JDB_OpenView API. The application developer should call JDB_CloseTable API to fix the table and view handle leaks. An increase in this value over time could indicate a leak that needs to be analyzed.
JDB Table Caches* (does not apply to jdenet_n processes)	This reports the number of JDB Table records cached in the EnterpriseOne server manager. The tables' records are cached when the table is registered in F98613 table using P98613 application or the application BSFN code called JDB_AddTableToCache API to cache the table records. If you are seeing lot of records cached then you might have a high memory usage issue in the EnterpriseOne server job. You should review the F98613 table and make sure you do not have any table which is added for caching by mistake. Also you should review if you have any new BSFN code which is calling the JDB_AddTableToCache API.

Field	Description
Database Connections**	The total number of open Database Connections includes AUTOcommit, Select For Update (SFU) and MANUAL-commit connections. An increase in this value over time could indicate a leak that needs to be analyzed. The AUTO and SFU connections have a small maximum limit whereas the MANUAL commit connections have no limit and can grow really high and will eventually hit the maximum global connections available in the database server. The manual commit open Database Connections are tied with the number of open manual commit jdb transactions. The next step should be to capture Memory Diagnostics or All Diagnostics from the server manager process detail page. The JDB transactions for different user sessions logged in the process should be reviewed and all of the open manual JDB transactions should be reviewed for a suspect where the apps BSFN code or tools code has called JDB_InitUser with MANUAL COMMIT MODE but has not called the JDB_ FreeUser. The JDB transactions has the file name, line number and function name which should help the developer to locate the code faster and review if the specific code does not call JDB_ FreeUser at all or does not call when the code returns with an error or exception path.
JDE Cache Records**	This is the total number of JDE Cache Records in the EnterpriseOne server process for the specific user which is created by calling the jdeCacheAdd API, for example. These cache records are generally created by EnterpriseOne Business Functions. They are necessary for Business Functions to do their work. But an increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call the jdeCacheDelete (to delete cache records) or jdeCacheTerminate (to remove the entire cache) API in business function code to fix the JDE Cache Record leak.

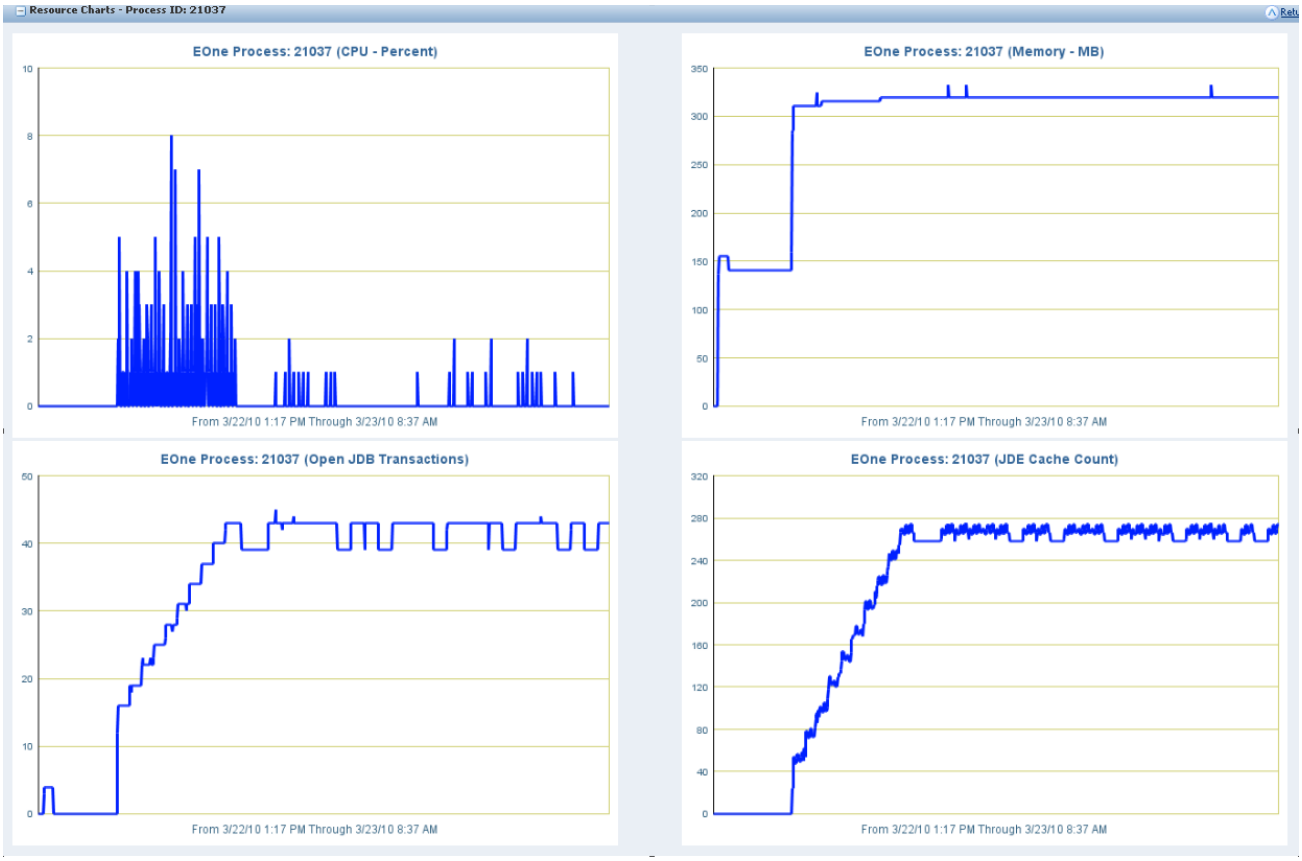
Note: The items with an * have been added in the 8.98.2.0 release.

The items with an ** are new in the 8.98.3.0 release.

Resource Charts

The enterprise server process ID page has a new section called Resource Charts. The charts are available for all actively running EnterpriseOne server processes. The time-frame of the data collected will be indicated along the x-axis of the graphs. How much data is represented will depend on when the Server Manager Console services were started and the collection intervals for the monitors that were specified (refer to Configuration Setup).

Figure 14–10 Resource Charts –Process ID: 21037.



The following two enterprise resource charts are applicable for all EnterpriseOne server process including JDENET_N, JDENET_K, RUNBATCH, PORTTEST etc.

Chart	Description
Eone Process: Process ID (CPU Usage - Percent)	This chart reports the percent CPU used by this EnterpriseOne server process. If you see that a certain EnterpriseOne process is taking a lot of CPU over time like a CallObject kernel, you should then take CPU diagnostics to see if there is any looping pattern in a specific business function. It is normal for a RUNBATCH process to a take an entire CPU, so you should not bother about RUNBATCH unless the RUNBATCH process takes more than the normal execution time. If the percent CPU is low for a RUNBATCH process, it might be a hung RUNBATCH process and you should take CPU diagnostics to review if any specific business function is hung.
Eone Process: Process ID (Memory - MB)	This chart reports the memory used in megabytes by this EnterpriseOne server process. If you see an increasing trend in memory usage, then the process might be leaking memory.

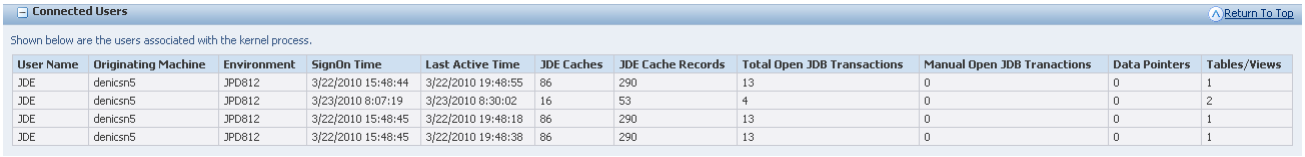
The following two enterprise resource charts are applicable to only non-jdenet_n EnterpriseOne server processes like JDENET_K, RUNBATCH, PORTTEST etc.

Chart	Description
Eone Process: Process ID (Open JDB Transactions)	This chart reports the number of open JDB Transactions in the EnterpriseOne server process. If you see increasing number of open JDB Transactions then the process might be running business function code which is leaking JDB Transactions.
Eone Process: Process ID (JDE Cache Count)	This chart reports the number of open JDE Cache handles in the EnterpriseOne server process. If you see increasing number of open JDE Cache handles then the process might be running business function code which is leaking JDE Cache handles.

Connected Users Section

The Connected Users section provides data for any users associated with the kernel process.

Figure 14–11 Connected Users section


Connected Users

Return To Top

Shown below are the users associated with the kernel process.

User Name	Originating Machine	Environment	SignOn Time	Last Active Time	JDE Caches	JDE Cache Records	Total Open JDB Transactions	Manual Open JDB Tranactions	Data Pointers	Tables/Views
JDE	denicsn5	JPD812	3/22/2010 15:48:44	3/22/2010 19:48:55	86	290	13	0	0	1
JDE	denicsn5	JPD812	3/23/2010 8:07:19	3/23/2010 8:30:02	16	53	4	0	0	2
JDE	denicsn5	JPD812	3/22/2010 15:48:45	3/22/2010 19:48:18	86	290	13	0	0	1
JDE	denicsn5	JPD812	3/22/2010 15:48:45	3/22/2010 19:48:38	86	290	13	0	0	1

The following three attributes are the existing attributes of user session for both CallObject kernel and Security kernel:

Field	Description
User Name	The name of the EnterpriseOne user who is signed on to the kernel process.
Originating Machine	The machine name from where the user signed in. For web client user, this will be the JAS server name.
SignOn Time	The time EnterpriseOne user signed on to this process.

The connected user section has been modified to have eight new attributes only for CallObject kernel process:

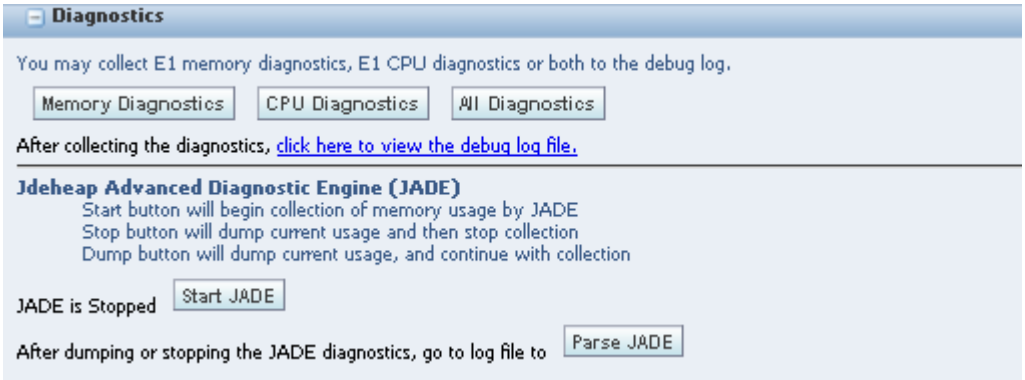
Field	Description
Environment	This reports environment of the logged in User in the CallObject kernel. This information will be useful to debug an issue which is specific to a path code or is specific to an environment.
Last Active Time	Last time the user performed any work on the CallObject kernel process. This will determine the staleness of user session and is a good metric when an EnterpriseOne user is logged for day probably from a third party integration system using EnterpriseOne Java connector, COM connector.
JDE Caches	This reports the number of JDE Caches opened by the specific user. If this number is high for this user compared to other users, then you should collect the memory diagnostics and review the JDE Caches opened by this user. The application developer should fix the business function code to free the JDECache leaks.

Field	Description
JDE Cache Records**	This is the total number of JDE Cache Records in the EnterpriseOne server process for the specific user which is created by calling the jdeCacheAdd API, for example. These cache records are generally created by EnterpriseOne Business Functions. They are necessary for Business Functions to do their work. But an increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call the jdeCacheDelete (to delete cache records) or jdeCacheTerminate (to remove the entire cache) API in business function code to fix the JDE Cache Record leak.
Total Open JDB Transactions	This reports the total number of open JDB Transactions for the specific user. If this number is high for this user compared to other users, then you should collect the memory diagnostics and review the JDB Transactionsopened by this user. The application developer should fix the business function code to free the JDB Transaction leaks.
Manual Open JDB Transactions	This reports the total number of open Manual-Commit JDB Transactions for the specific user. If this number is high for this user compared to other users, then you should collect the memory diagnostics and review the JDB Transactions opened by this user. The application developer should fix the business function code to free the JDB Transaction leaks.
Data Pointers	This reports the number of Data Pointers used by the specific user. If this number is high for this user compared to other users, then you should collect the memory diagnostics and review the Data Pointers opened by this user. The application developer should fix the business function code to free the Data Pointer leaks.
Tables/Views	This reports the number of JDB Tables or JDB Views opened by the specific user. If this number is high for this user compared to other users, then you should collect the memory diagnostics and review the JDB Tables or JDB Views opened by this user. The application developer should fix the business function code to free the JDB Tables or JDB Views leaks.

14.7.2.4 Get Memory / CPU Diagnostics

The enterprise server process ID page is used to evaluate on-demand diagnostics and has a new section called Diagnostics and Recycling.

Figure 14–12 Diagnostics section



The Diagnostics section is only applicable to CallObject Kernels, RUNBATCH processes, and Subsystem UBE's.

The diagnostics section allows the collection of the following four types of diagnostics:

1. Memory diagnostics - Once the User clicks this, the system will write the in-memory EnterpriseOne objects diagnostics data to the EnterpriseOne server process JDEDEBUG log. This should be used to debug EnterpriseOne Object leaks causing EnterpriseOne process memory growth. The diagnostics data has the following structure.
   - Process OS data
     a. Memory (megabytes)
     b. CPU (percent)
     c. Threads (number of threads)
   - Memory data

   a. Process level data shared by all user sessions.

      1. Environment data

      2. JDB Table Cache data

      3. Database Connection data

   b. User Sessions

      1. Open JDB Transactions

      2. Open Tables or Views

      3. Open JDECaches

      4. Open Data Pointers

2. CPU Diagnostics - Once the User clicks this, the system will write the in-memory business function call stack(s) to the EnterpriseOne server process JDEDEBUG log (whether or not debug logging has been enabled). This should be used to debug hanging (low CPU) or looping (high CPU) EnterpriseOne processes. The following data will be displayed in the CPU diagnostics.
   - Process OS data
     a. Memory (megabytes)
     b. CPU (percent)
     c. Threads (number of threads)
   - CPU Diagnostics

     a. BSFN Call Stacks

        1. BSFN call stack for thread 1

        2. BSFN call stack for thread 2 (thread BSFN call stacks beyond the first thread are only applicable to CallObject Kernel processes)

        3. etc.

     b. OS Call Stacks

        1. OS call stack for thread 1

        2. OS call stack for thread 2 (thread OS call stacks beyond the first thread are only applicable to CallObject Kernel processes)

        3. etc.

3. All Diagnostics - Once the user clicks this, the system will generate a combination of Memory AND CPU diagnostics. The following data will be displayed in All diagnostics.
   - Process OS data
     a. Memory (megabytes)
     b. CPU (percent)
     c. Threads (number of threads)
   - Memory data
     a. Process level data shared by all user sessions.
        1. Environment data
        2. JDB Table Cache data
        3. Database connection data
     b. User Sessions
        Open JDB Transactions
        Open Tables of Views
        Open JDECaches
        Open Data Pointers
   - CPU Diagnostics
     a. BSFN Call Stacks
        1. BSFN call stack for thread 1
        2. BSFN call stack for thread 2 (thread BSFN call stacks beyond the first thread are only applicable to CallObject Kernel processes)
        3. etc.
     b. OS Call Stacks
        1. OS call stack for thread 1
        2. OS call stack for thread 2 (thread OS call stacks beyond the first thread are only applicable to CallObject Kernel processes)
        3. etc.
4. JADE -
   - Start button will begin collection of memory usage

     JADE can be set up statically in jde.ini. The functionality is similar to BMD with levels 1, 2, and 3 available (see dvanced Profiling). The default setting is JADE level 2.
   - Stop button will dump current usage and then stop collection
   - Dump button will dump current usage and then stop collection
   - Parse JADE button will bring up the log file after dumping or stopping JADE diagnostics.

After the diagnostics have been collected for Memory Diagnostics, CPU Diagnostics, or All Diagnostics, then click on the *click here to view the debug log file* link to view the diagnostics.

Here is a sample All Diagnostics:

```
Apr 5 16:31:22.423254 jdb_utl1.c13553 - 5504/5 WRK:JDE_000BDB60_P4310 User
initiated process dump
(CPU Diagnostics):
********** Begin OS Data **********
Memory Usage = 257MB
CPU Usage = 1%
Number of Threads = 7
********** End OS Data **********
********** Begin Detailed CPU Data **********
********** Begin BSFN Call Stacks **********
CALLSTACK,5,WRK:JDE_000BDB60_P4310,JDE,*ALL,JPD812,NONE,P4310,denicsn5,ZJDE0001
libCDIST.so/UNKNOWN/F4311EndDoc
********** End BSFN Call Stacks **********
********** Begin OS Call Stacks **********
5504: jdenet_k 6014
----------------- lwp# 2 / thread# 2 --------------------
fc340408 lwp_park (0, 0, 0)
fc33a49c cond_wait_queue (24e8d0, bd298, 0, 0, 0, 0) + 28
fc33aa1c cond_wait (24e8d0, bd298, 0, 0, 20, 0) + 10
fc33aa58 pthread_cond_wait (24e8d0, bd298, 0, 0, 20, 1) + 8
fe0f3db8 psthread_cond_wait (24e6c8, bd088, 1, 1, 1, fe10d810) + 274
fe352da4 ps_blocking_queue_dequeue (24e4a0, f637ff24, bd088, 0, fe10d7d8, 20) +
224
fe354214 psthread_pool_worker_function (24e4a0, 0, d5cd4, d5cc8, fe10d7b8,
fe367de8) + 4a8
fe0f48fc threadFunctionWrapper (c3548, 204, 0, c4, c354b, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
----------------- lwp# 5 / thread# 3 --------------------
fc341850 waitid (0, 15c9, f58e6ab0, 3)
fc334758 waitpid (15c9, f58e6c04, 0, 0, f58e6c5c, faa82740) + 60
fc327eec system (1b3360, fc36ff74, 20000, 1, fc368288, f58e6c5c) + 2ec
fd9e6350 jdeSystem (f58e6db8, 6a, 0, 1b3360, 4b, f58e6db8) + 50
ff30f6bc logCallStackOnUnix (1580, 0, f58e7694, f58e74b1, 24fc, 2500) + 130
ff30fb0c allocCallStackUnix (0, f58e81e4, ff316d56, 0, 2518, 2400) + 88
ff30fd7c jdeAllocCallStack (1580, f58e81e4, f58e83ec, 8c00, 50e598, f58e83ec) + 20
fed27dd8 logProcessDumpData (2, 0, 0, ffffffff, 810060, 8c00) + e04
f6571310 doQueryDiagnostics (238, 810060, 0, f74bb544, 234, 0) + 3d8
f656eaf0 performRequestInternal (82f3e0, 1, 0, ae6f88, ff0ed1c0, 0) + 51c
f656e46c dballPerformRequest (82f3e0, 8303d0, 0, 8303d0, 8303d3, 3) + 3e4
feb937a4 JDB_DBPerformRequest (aeaad8, 82f3e0, ae6f88, 18a268, 18a340, 18a268) +
e4
fe13a894 TM_DBPerformRequest (65be18, 0, ae6f88, 8ab250, 7d70c0, 5) + 41c
fec083fc InsertTable (8ab250, 7d70c0, 0, ae6f88, f58eb1f0, 0) + 392c
fec0bf74 JDB_InsertTable (8ab250, f32d3854, 0, f58ed714, 0, ff338dfc) + 208
f214177c IXT4311Z1_EndDocWriteNewOrderDetail_2 (f32d3818, 7d0868, 675720, a53750,
f58f4c28,
f58f6bfc) + dd4
f213efbc IXT4311Z1_EndDocProcessCurrentDetailLine (3e5d48, 7d0868, 675720,
f58f770c, a53750,
f58f62e0) + 1338
f2139694 F4311EndDoc (7d0868, 675720, a53750, 31, 0, 1) + 16d4
fefb88f8 jdeCallObjectV2 (1b8df8, 7d0868, 506be0, 1, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (1b8df8, 0, 7d0868, 0, fe963658, f58feea4) + 38
fe79f4dc JDEK_ProcessCallRequest (f58fef58, 6e1018, bdb60, 3e5d48, 7d0868, 20) +
3d30
JD Edwards EnterpriseOne Tools 8.98 Update 3
Kernel Resource Management
Copyright © 2010, Oracle. All rights reserved 22
fe79a5e8 JDEK_StartCallRequest (f58ffc40, 6e1018, 0, 3e5d48, bdb60, 0) + f74
```

```
fe77db24 runBusinessFunction (94ef0, 0, 0, 7174, ff338dff, fe93c6ec) + 328
fe77dea8 runCallObjectJob (94ef0, 0, 94f00, 94f00, 94f00, 20) + 5c
fe354394 psthread_pool_worker_function (0, 0, 74, d5cc8, fe10d7b8, fe367de8) + 628
fe0f48fc threadFunctionWrapper (c34c8, 204, 0, c4, c34cb, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
********** End OS Call Stacks **********
********** End Detailed CPU Data **********
Apr 5 16:32:06.733801 jdb_utl1.c13485 - 5504/3 WRK:Free Remote Env User initiated
process dump
(Memory Diagnostics):
********** Begin OS Data **********
Memory Usage = 264MB
CPU Usage = 1%
Number of Threads = 7
********** End OS Data **********
********** Begin Detailed Memory Data **********
********** Begin Process Data **********
ENVIRONMENT,Ptr=0026fbd8,Env=JPD812,PathCode=PD812
JDBTABLECACHE,Ptr=005a5250,Name=JDB_BV_1270506634JPD812F4009,#Records=1
JDBTABLECACHE,Ptr=005ae618,Name=JDB_BV_1270506634JPD812F40203,#Records=2
JDBTABLECACHE,Ptr=005b8fb0,Name=JDB_BV_1270506634JPD812F40205,#Records=1
JDBTABLECACHE,Ptr=005eb458,Name=JDB_BV_1270506634JPD812F41001,#Records=2
JDBTABLECACHE,Ptr=00184f10,Name=JDB_BV_1270506637JPD812F41002,#Records=8
JDBTABLECACHE,Ptr=00647ae8,Name=JDB_BV_1270506637JPD812F7306,#Records=1
JDBTABLECACHE,Ptr=0065c138,Name=JDB_BV_1270506637JPD812F99410,#Records=2
JDBTABLECACHE,Ptr=0067c3a0,Name=JDB_BV_1270506637JPD812F0004,#Records=25
JDBTABLECACHE,Ptr=00685f60,Name=JDB_BV_1270506637JPD812F0005,#Records=27
JDBTABLECACHE,Ptr=0068b8d0,Name=JDB_BV_1270506637JPD812F0006,#Records=1
JDBTABLECACHE,Ptr=006cf0f0,Name=JDB_BV_1270506637JPD812F0008,#Records=1
JDBTABLECACHE,Ptr=006d7bc8,Name=JDB_BV_1270506637JPD812F0010,#Records=2
JDBTABLECACHE,Ptr=003c1a38,Name=JDB_BV_1270506638JPD812F1609,#Records=1
OCIDBCONN,Ptr=00426850,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=AutoInUse
,Comm
itMode=A,RefCount=39
OCIDBCONN,Ptr=006e2780,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Co
mmitMode=M,RefCount=1
OCIDBCONN,Ptr=0052fc50,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualAva
ilable,C
ommitMode=M,RefCount=0
OCIDBCONN,Ptr=00871ec0,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Co
mmitMode=M,RefCount=1
OCIDBCONN,Ptr=00abf430,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Com
mitMode=M,RefCount=1
********** End Process Data **********
********** Begin Session Data **********
SESSION,Ptr=000bdb60,User=JDE,Env=JPD812,Role=*ALL,Machine=denicsn5,SignOnTime= 4/
5/2010
16:30:38,LastActiveTime= 4/ 5/2010 16:32:06
OPENJDBTRANSACTION,Ptr=001819c0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile
=jdb_ctl.c,Function=JDB_LoadEnv,Line=5829
TABLE,Ptr=0048d0a0,Name=F0013,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0013Ro
wUsingCurrencyCode,Line=73
TABLE,Ptr=00aa41f0,Name=F0010,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0010Ro
wUsingCompanyNumber,Line=246
```

```
OPENJDBTRANSACTION,Ptr=007e88a0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile
=jdb_ctl.c,Function=CallStartupBusinessFunction,Line=8551
JDECACHE,Ptr=0071d1e0,Name=1XT4311Z1A,#Cursors=1,#Records=0,#Indices=1,#References
=1,File=x
t4311z1.c,Function=F4311InitializeCaching,Line=23018
JDECACHE,Ptr=00743aa8,Name=1XT4311Z1B,#Cursors=1,#Records=0,#Indices=2,#References
=1,File=x
t4311z1.c,Function=F4311InitializeCaching,Line=23027
JDECACHE,Ptr=008d28c0,Name=1XT4311Z1C,#Cursors=1,#Records=0,#Indices=1,#References
=4,File=x
t4311z1.c,Function=IXT4311Z1_InitiateOrderCache,Line=5758
JDECACHE,Ptr=009426b0,Name=1B4302570Cache,#Cursors=1,#Records=0,#Indices=1,#Refere
nces=2,F
ile=b4302570.c,Function=ApprovalsFieldConstants,Line=158
JDECACHE,Ptr=006801f0,Name=1F45UI73,#Cursors=1,#Records=0,#Indices=2,#References=4
,File=b450
0200.c,Function=F4573GetNextFreeGood,Line=100
JD Edwards EnterpriseOne Tools 8.98 Update 3
Kernel Resource Management
Copyright © 2010, Oracle. All rights reserved 23
JDECACHE,Ptr=009399d8,Name=1B3201470213088,#Cursors=1,#Records=0,#Indices=7,#Refer
ences=3,
File=b3201470.c,Function=I3201470_CreateInitCache,Line=732
JDECACHE,Ptr=008c87d0,Name=1B3201470213090,#Cursors=1,#Records=0,#Indices=7,#Refer
ences=3,
File=b3201470.c,Function=I3201470_CreateInitCache,Line=732
********** End Session Data **********
********** End Detailed Memory Data **********
```

Contextual Diagnostics

Please note that each of the memory objects listed in the Memory and All Diagnostics file will show the File, Function, and Line number for JDECACHE, TABLE/VIEW, OPENJDBTRANSACTION, and DATAPOINTER objects. For example:

```
JDECACHE,Ptr=0071d1e0,Name=1XT4311Z1A,#Cursors=1,#Records=0,#Indices=1,#References
=1,File=x
t4311z1.c,Function=F4311InitializeCaching,Line=23018

TABLE,Ptr=01eb8a48,Name=F0013,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0013Ro
wUsingCurrencyCode,Line=73

OPENJDBTRANSACTION,Ptr=02d05328,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile
=jdb_ctl.c,Function=JDB_LoadEnv,Line=5845
```

In order to get all of the added contextual information in 8.98.3.0 and higher releases on Windows Platform, please ensure that /Oy- is there under OptimizationFlags and /MAP is there under LinkFlags in [BSFN BUILD] section.

```
[BSFN BUILD]
OptimizationFlags=/FD /Gz /O2 /Zi /MD /W4 /EHs /Gy /Oy-
LinkFlags=/DLL /DEBUG /SUBSYSTEM:windows /FORCE:MULTIPLE
/FORCE:UNRESOLVED /INCREMENTAL:YES /VERBOSE /MAP /WARN:3
```

Note: The columns with an * are new columns that have been added in the 8.98.3.0 release.

14.7.2.5 Corrective Actions

Cache or Recycling actions are available in the Corrective Actions section.

Clear Cache

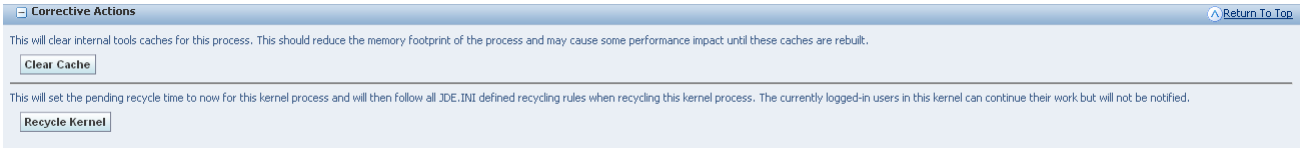
The Clear Cache button is now available to clear internal tools caches for the process that has been selected. This should recuce the memory footprint of the process and may cause some performance impact while the caches are being rebuilt.

Recycle Kernel

You can now recycle an individual kernel which prevents a single process from impacting or bringing down the entire system. It also prevents new users from being associated with the kernel and possibly being impacted if the kernel zombies. This allows the system to gracefully shut down the kernel and reclaim resources.

There is a new button called Recycle Kernel that is used to begin recycling CallObject kernel processes on demand. The purpose of the recycling button is to allow administrators to gracefully shut down a process that appears to have problems.

Figure 14–13 Clear Cache and Recycle Kernel corrective actions



Previously, the only options for an administrator were to allow the process to continue running or to kill the process from the OS. If the process were allowed to continue, it could become attached to new user sessions, which may be detrimental to those sessions, and which may make the perceived problems within the kernel process even worse than what they might have been. On the other hand, if the process were killed manually, the applications that were currently running would be ungracefully stopped, which would prevent the applications from completing, and would force any open transactions to be rolled back.

The recycling option tries to avoid both of those issues. When a kernel process begins recycling, there will be no additional user sessions attached to it. But, the kernel is not stopped immediately, which allows the current users to complete their processing. When all users have completed their processing, then that kernel will be shut down. Before KRM, kernel recycling was only available based on JDE.INI settings. The CallObject kernels could be recycled at a scheduled time. With the new KRM "Recycle Kernel" button, a selected kernel process can begin recycling on demand. Because runbatch and subsystem processes cannot be recycled, the Recycle Kernel button is not available for these processes.

There are additional JDE.INI settings that can affect how the kernels are recycled. After the time to begin recycling has passed, the state of the kernel goes into a "pending recycling" state. While recycling is pending, the kernel accepts no new user sessions. When all existing user sessions have logged off the kernel will shut down. There is a JDE.INI setting that specifies the length of a period of inactivity, before each user session is considered inactive. When that is set, and there are only inactive user sessions, the kernel will also shut down. The default time for the inactivity period is six hours. Another JDE.INI setting is the length of time before a forced termination occurs. When that period expires, the kernel will shut down, even if there are active users at that time. The assumption is that those user sessions have some reason why they will never initiate their own session logoff. For instance, the user could be stuck

in a deadlock, or the user could be stuck in a loop that it cannot exit. There could also be users that are intentionally never logged off (there are some use cases for this with Interop user sessions). The default time for the forced termination is twelve hours.

Within the context of the KRM Recycle Kernel button, those JDE.INI recycling parameters will apply, even if scheduled kernel recycling is not set. The beginning of the recycling period will be when the Recycle Kernel button is pressed.

14.7.2.6 Inline Corrective / Diagnostic Actions

Call Object or Runbatch Crash due to Memory Corruption

CallObject may go to a Zombie or Crashed state when it encounters a bad memory in the business function code. The same behavior is to be expected for a Runbatch process. This crash will produce an extra log file with .dmp.log as its extension. Viewing the .dmp file for a crashed CallObject will show all the threads which were running at the time of the crash. On System-i enterprise server, there will not be any .dmp file. Instead the callstacks for all threads will be logged in jde.log.

The thread which encountered the bad memory and crashed will be highlighted in "red" color and often times will serve as a good starting point for investigation of the bug.

Figure 14–14 Log .dmp files highlighted in red.

When analyzing the .dmp file, begin by looking for the thread which contains the jdeLogCallStack function, as this is the thread which initiated the crash.

This thread may not always be the root cause of the problem, as it could be a victim of another thread which caused memory corruption but did not crash itself. Crashes of this kind will most likely have a message from the given Operating System in their corresponding jde.logs such as "ACCESS VIOLATION".

This is a dmp.log for a crash due to Memory Corruption:

```
****************************************
Tue Mar 30 17:27:28 MDT 2010
****************************************
Generating call stacks for PID 28774
****************************************
28774: jdenet_k 6014
```

```
----------------- lwp# 1 / thread# 1 --------------------
fc342d74 msgsys (2, 34000023, 13f2d58, 200c, 0, 0)
fc333b84 msgrcv (34000023, 13f2d58, 200c, 0, 0, 0) + 68
ff23e7b4 receiveMessage (5, 34000023, 200c, 0, 0, 0) + 1dc
ff21d73c ipcGetQueueEntry (92d60, ffbfc0c4, ffbfc0c8, ffbfe0d4, 5, 13f2d58) + 334
ff139a24 getExternalQueueEntry (0, 5, 1, ffbff28c, fb3a00ab, ff1d1b58) + 35c
ff195c60 getKernelQueueEntry (0, 8fd68, ffbff28c, 92d60, 8fd54, 8fd58) + 49c
ff19606c processKernelQueue (ffbff28c, fb3a0000, 0, 0, 0, ff1cd2c8) + 300
ff16e250 JDENET_RunKernel (20, 8fd68, 1, ff1cd2c8, 0, 5) + 428
00011cc0 main (0, 0, 4c, 0, 2218c, 0) + 6bc
000111d4 _start (0, 0, 0, 0, 0, 0) + 108
----------------- lwp# 2 / thread# 2 --------------------
fc340408 lwp_park (0, 0, 0)
f6667034 kpuexec (f8e9d800, f1aa34, f14ce0, 0, 0, 18814c) + 2b8
f65c5e50 OCIStmtExecute (188118, f1aa34, 0, 0, 0, 0) + 2c
f657e348 BFOCIStmtExecute (188118, f1aa34, f14ce0, 0, 0, 0) + 28
f656eacc performRequestInternal (116d930, 0, 116d938, 116ca98, ff0ed1c0, 0) + 4f8
f656e46c dballPerformRequest (116d930, 116e920, 0, 116e920, 116e923, 1000) + 3e4
feb937a4 JDB_DBPerformRequest (93bc18, 116d930, 116ca98, 181240, 181318, 181240) +
e4
fe13a894 TM_DBPerformRequest (17d050, 1, 116ca98, a6f318, 1b3ab8, 5) + 41c
feba75b4 SelectKeyed (a6f318, 17d050, f635eecc, 1, 1, 1) + 1ec8
febb3c00 FetchKeyed (0, 1, a7cb68, 116ca98, feddd4dc, 0) + 286c
febb114c JDB_FetchKeyed (a6f318, 1, a7cb68, ff338dfc, f636128c, 0) + 214
ee688ab4 EditSystemExistenceF99410 (a7cb68, 689800, f6367910, 3, ff976770,
110c6e0) + 30c
fefb88f8 jdeCallObjectV2 (f1f3d5ec, d38030, a73c40, 3, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1f3d5ec, 0, d38030, 0, 0, 0) + 38
f0b6b804 I4500250_GetGrowerSystemConstant (d38030, a9be40, f6367a80, f6367912,
ff990790, 0) + e8
f0b65e50 CalculatePurchasePrice (d38030, a9be40, f6370b3c, f6367a84, ff98bfa8,
f6370bda) + 220
fefb88f8 jdeCallObjectV2 (f1ff38fc, d38030, e1af50, 2, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1ff38fc, 0, d38030, 0, 0, 0) + 38
f0f500cc IXTF4311Z1_CalcPurchasePrice (a9be40, f63765ce, 1243948, 1243688,
f6376ce8, f63756f0) +
15d8
f0f39fd0 IXT4311Z1_F4311EditLineInternalFunctions (d38030, a9be40, 1243688,
f6376f38, 0, f63756f0) +
23b8
fefb88f8 jdeCallObjectV2 (6f7a20, d38030, d28008, 1, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (6f7a20, 0, d38030, 0, fe963658, f637eea4) + 38
fe79f4dc JDEK_ProcessCallRequest (f637ef58, f076b0, 7d9d08, a80f90, d38030, 20) +
3d30
fe79a5e8 JDEK_StartCallRequest (f637fc40, f076b0, 0, a80f90, 7d9d08, 0) + f74
fe77db24 runBusinessFunction (c44eb8, 0, 0, 7174, ff338dff, fe93c6ec) + 328
fe77dea8 runCallObjectJob (c44eb8, 0, c44ec8, c44ec8, c44ec8, 20) + 5c
fe354394 psthread_pool_worker_function (0, 0, 74, d5cc8, fe10d7b8, fe367de8) + 628
fe0f48fc threadFunctionWrapper (c3548, 204, 0, c4, c354b, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
----------------- lwp# 3 / thread# 3 --------------------
fc340408 lwp_park (0, 0, 0)
fc33a49c cond_wait_queue (2c2350, 2fc040, 0, 0, 0, 0) + 28
fc33aa1c cond_wait (2c2350, 2fc040, 0, 0, 20, 0) + 10
fc33aa58 pthread_cond_wait (2c2350, 2fc040, 0, 0, 20, 1) + 8
fe0f3db8 psthread_cond_wait (2c2148, 2fbe30, 1, 1, 1, fe10d810) + 274
fe352da4 ps_blocking_queue_dequeue (2c1f20, f5ffff24, 2fbe30, 0, fe10d7d8, 20) +
224
fe354214 psthread_pool_worker_function (2c1f20, 0, d5cd4, d5cc8, fe10d7b8,
fe367de8) + 4a8
```

```
fe0f48fc threadFunctionWrapper (c3588, 204, 0, c4, c358b, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
----------------- lwp# 4 / thread# 4 --------------------
fc340408 lwp_park (0, 0, 0)
f6667034 kpuexec (f8e9d800, f84e4c, f4e190, 0, 0, 18814c) + 2b8
f65c5e50 OCIStmtExecute (188118, f84e4c, 0, 0, 0, 0) + 2c
f657e348 BFOCIStmtExecute (188118, f84e4c, f4e190, 0, 0, 0) + 28
f656eacc performRequestInternal (11c2750, 0, 11c2758, 11c0b10, ff0ed1c0, 0) + 4f8
f656e46c dballPerformRequest (11c2750, 11c3740, 0, 11c3740, 11c3743, 1000) + 3e4
feb937a4 JDB_DBPerformRequest (a6fa80, 11c2750, 11c0b10, 181240, 181318, 181240) +
e4
fe13a894 TM_DBPerformRequest (3511d8, 1, 11c0b10, c93e90, 1b3ab8, 5) + 41c
feba75b4 SelectKeyed (c93e90, 3511d8, f58deecc, 1, 1, 1) + 1ec8
febb3c00 FetchKeyed (0, 1, a7ce68, 11c0b10, feddd4dc, 0) + 286c
febb114c JDB_FetchKeyed (c93e90, 1, a7ce68, ff338dfc, f58e128c, 0) + 214
ee688ab4 EditSystemExistenceF99410 (a7ce68, 689800, f58e7910, 3, ff976770, 500e90)
+ 30c
fefb88f8 jdeCallObjectV2 (f1f3d5ec, 112a060, a73e48, 3, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1f3d5ec, 0, 112a060, 0, 0, 0) + 38
f0b6b804 I4500250_GetGrowerSystemConstant (112a060, 7aa228, f58e7a80, f58e7912,
ff990790, 0) + e8
f0b65e50 CalculatePurchasePrice (112a060, 7aa228, f58f0b3c, f58e7a84, ff98bfa8,
f58f0bda) + 220
fefb88f8 jdeCallObjectV2 (f1ff38fc, 112a060, f07708, 2, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1ff38fc, 0, 112a060, 0, 0, 0) + 38
f0f500cc IXTF4311Z1_CalcPurchasePrice (7aa228, f58f65ce, a9a898, a9a5d8, f58f6ce8,
f58f56f0) + 15d8
f0f39fd0 IXT4311Z1_F4311EditLineInternalFunctions (112a060, 7aa228, a9a5d8,
f58f6f38, 0, f58f56f0) + 23b8
fefb88f8 jdeCallObjectV2 (b81968, 112a060, 928860, 1, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (b81968, 0, 112a060, 0, fe963658, f58feea4) + 38
fe79f4dc JDEK_ProcessCallRequest (f58fef58, 13e3c58, 8a7968, 7471c8, 112a060, 20)
+ 3d30
fe79a5e8 JDEK_StartCallRequest (f58ffc40, 13e3c58, 0, 7471c8, 8a7968, 0) + f74
fe77db24 runBusinessFunction (1116ff8, 0, 0, 7174, ff338dff, fe93c6ec) + 328
fe77dea8 runCallObjectJob (1116ff8, 0, 1117008, 1117008, 1117008, 20) + 5c
fe354394 psthread_pool_worker_function (0, 0, 74, d5cc8, fe10d7b8, fe367de8) + 628
fe0f48fc threadFunctionWrapper (c35c8, 204, 0, c4, c35cb, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
----------------- lwp# 5 / thread# 5 --------------------
fc341850 waitid (0, 7170, f555c420, 3)
fc334758 waitpid (7170, f555c564, 0, 0, 0, f7fe0c00) + 60
ff30f39c logCallStackOnUnixSigSafe (0, ff316748, f555c581, 24dc, 7170, ff3351f4) +
b8
ff310038 jdeLogCallStack (7066, ff338e64, 2400, ff338dfc, 0, ff3351f4) + 1e8
Start Here
ff1470ec krnlSignalHandler (b, 1000, 7066, ff1d217c, 3000, 1154) + 4e0
fc340494 __sighndlr (b, 0, f555ca38, ff146c0c, 0, 1) + c
JD Edwards EnterpriseOne Tools 8.98 Update 3
Kernel Resource Management
Copyright © 2010, Oracle. All rights reserved 27
fc33558c call_user_handler (b, 0, 0, 0, f7fe0c00, f555ca38) + 3b8
f66304ac kpuhhalp (f74bb544, 1774, 16c00, f73eb320, 8048, 16800) + 8ec
f708b174 ttcrbur (182c28, d6784, f662fbc0, 24, f753ac5c, d6898) + 1104
f708b6f8 ttcbur (182c28, d6784, d8b60, 24, 185, 1026) + fc
f6664740 kpuexCallback (182c28, d86e8, f753d1b8, d6784, 1, d8be8) + 1030
f69b36e8 ttcdrv (d86e8, 1894e8, d6784, 182c28, 0, ecb88) + 1da4
f6770b6c nioqwa (0, 189490, f69b1944, d86e8, d67ac, 0) + 40
f65e6f18 upirtrc (d6784, 5e, 0, 182c28, 1, 0) + 544
f66b6380 kpurcsc (188118, 0, 0, d86e8, d93bc, 0) + 6c
```

```
f6665c80 kpuexecv8 (188118, 5ad2f8, 5ad344, 17790, 0, f74bb544) + 1390
f66682c0 kpuexec (0, 5ad2f8, f85200, 0, 0, 1000) + 1544
f65c5e50 OCIStmtExecute (188118, 5ad2f8, 0, 0, 0, 0) + 2c
f657e348 BFOCIStmtExecute (188118, 5ad2f8, f85200, 0, 0, 0) + 28
f656eacc performRequestInternal (1257408, 0, 1257410, 11ce838, ff0ed1c0, 0) + 4f8
f656e46c dballPerformRequest (1257408, 12583f8, 0, 12583f8, 12583fb, 1000) + 3e4
feb937a4 JDB_DBPerformRequest (116ec18, 1257408, 11ce838, 181240, 181318, 181240)
+ e4
fe13a894 TM_DBPerformRequest (9e2d28, 1, 11ce838, f40f18, 4a7e48, 5) + 41c
feba75b4 SelectKeyed (f40f18, 9e2d28, f5564be4, 0, 1, 1) + 1ec8
feba3ed4 JDB_SelectKeyed (f40f18, 7, 0, a000, a118, feddd4dc) + 20c
fe1c969c RTK_CER_FIOSelect (f40f18, 7, 7, f5567620, af7a9c, 8) + ac
f0d5cab4 DetermineIfBlanketPOExists (928a28, c47588, f5571780, 0, f5571814, 24000)
+ 1934
fefb88f8 jdeCallObjectV2 (f1ff2c98, 928a28, 13fee98, 2, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1ff2c98, 0, 928a28, 0, 0, 0) + 38
f0f420c4 IXTF4311Z1_EditLineMultipleBlanketRelease (928a28, c47588, e5fc28,
f5576f38, e5feb6,
f55756f0) + 7c0
f0f39be4 IXT4311Z1_F4311EditLineInternalFunctions (928a28, c47588, e5fc28, 53, 0,
f55756f0) + 1fcc
fefb88f8 jdeCallObjectV2 (b81788, 928a28, 1106270, 1, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (b81788, 0, 928a28, 0, fe963658, f557eea4) + 38
fe79f4dc JDEK_ProcessCallRequest (f557ef58, dda5b8, 5a7e68, 9e2ae8, 928a28, 20) +
3d30
fe79a5e8 JDEK_StartCallRequest (f557fc40, dda5b8, 0, 9e2ae8, 5a7e68, 0) + f74
fe77db24 runBusinessFunction (c59d88, 0, 0, 7174, ff338dff, fe93c6ec) + 328
fe77dea8 runCallObjectJob (c59d88, 0, c59d98, c59d98, c59d98, 20) + 5c
fe354394 psthread_pool_worker_function (0, 0, 74, d5cc8, fe10d7b8, fe367de8) + 628
fe0f48fc threadFunctionWrapper (c3478, 204, 0, c4, c347b, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
----------------- lwp# 6 / thread# 6 --------------------
fc340408 lwp_park (0, 0, 0)
f6667034 kpuexec (f8e9d800, f1492c, 5ad6ac, 0, 0, 18814c) + 2b8
f65c5e50 OCIStmtExecute (188118, f1492c, 0, 0, 0, 0) + 2c
f657e348 BFOCIStmtExecute (188118, f1492c, 5ad6ac, 0, 0, 0) + 28
f656eacc performRequestInternal (d05080, 0, d05088, d03440, ff0ed1c0, 0) + 4f8
f656e46c dballPerformRequest (d05080, d06070, 0, d06070, d06073, 1000) + 3e4
feb937a4 JDB_DBPerformRequest (93bc50, d05080, d03440, 181240, 181318, 181240) +
e4
fe13a894 TM_DBPerformRequest (9e2a20, 1, d03440, 116e990, 1b3ab8, 5) + 41c
feba75b4 SelectKeyed (116e990, 9e2a20, f385eecc, 1, 1, 1) + 1ec8
febb3c00 FetchKeyed (0, 1, 1562d8, d03440, feddd4dc, 0) + 286c
febb114c JDB_FetchKeyed (116e990, 1, 1562d8, ff338dfc, f386128c, 0) + 214
ee688ab4 EditSystemExistenceF99410 (1562d8, 689800, f3867910, 3, ff976770, 618700)
+ 30c
fefb88f8 jdeCallObjectV2 (f1f3d5ec, 13741c8, a6fb20, 3, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1f3d5ec, 0, 13741c8, 0, 0, 0) + 38
f0b6b804 I4500250_GetGrowerSystemConstant (13741c8, b86dd8, f3867a80, f3867912,
ff990790, 0) + e8
f0b65e50 CalculatePurchasePrice (13741c8, b86dd8, f3870b3c, f3867a84, ff98bfa8,
f3870bda) + 220
fefb88f8 jdeCallObjectV2 (f1ff38fc, 13741c8, 117a878, 2, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (f1ff38fc, 0, 13741c8, 0, 0, 0) + 38
f0f500cc IXTF4311Z1_CalcPurchasePrice (b86dd8, f38765ce, 1265f00, 1265c40,
f3876ce8, f38756f0) + 15d8
f0f39fd0 IXT4311Z1_F4311EditLineInternalFunctions (13741c8, b86dd8, 1265c40,
f3876f38, 0, f38756f0) + 23b8
fefb88f8 jdeCallObjectV2 (6f7908, 13741c8, eb79e8, 1, ff0c36fc, 20) + a3d0
fefae518 jdeCallObject (6f7908, 0, 13741c8, 0, fe963658, f387eea4) + 38
```

```
fe79f4dc JDEK_ProcessCallRequest (f387ef58, 13e3c78, bdb60, 9e2768, 13741c8, 20) +
3d30
fe79a5e8 JDEK_StartCallRequest (f387fc40, 13e3c78, 0, 9e2768, bdb60, 0) + f74
fe77db24 runBusinessFunction (14c288, 0, 0, 7174, ff338dff, fe93c6ec) + 328
fe77dea8 runCallObjectJob (14c288, 0, 14c298, 14c298, 14c298, 20) + 5c
fe354394 psthread_pool_worker_function (0, 0, 74, d5cc8, fe10d7b8, fe367de8) + 628
fe0f48fc threadFunctionWrapper (36cdb8, 204, 0, c4, 36cdbb, fe353d6c) + f4
fc340368 _lwp_start (0, 0, 0, 0, 0, 0)
```

Analysis of Call Object Crash - CallStacks (for above .dmp file)

1. We start by looking at the thread which contains jdeLogCallStack() function as this is the thread which encountered the error first.
2. From that point in the callstack please traverse down to the businessfunction which is being invoked by looking top -> bottom until one hits jdeCallobjectV2() in this example its DetermineIfBlanketPOExists.
3. Check for any MEM SAR fix for this Bsfn and note the ESU.
4. As mentioned earlier - this may not be the root cause of the crash - it could be a victim of some other thread/ bsfn corrupting the memory before it.
5. Search for other active business functions in other threads for this callobject. Use the same methodology:
   a. Evaluate from top to bottom.
   b. Keep going until you hit the jdeCallObjectV2() Function.
   c. The BusinessFunction right above jdeCallObjectV2() is the one closest the point in time when the crash occurred.
6. In our example these are:
   a. No running BSFN in Thread1. Main thread is waiting for work.
   b. EditSystemExistenceF99410 in Thread2.
   c. No running BSFN in Thread3. Thread is waiting for work.
   d. EditSystemExistenceF99410 in Thread4.
   e. DetermineifBlanketPOExists in Thread 5 (This is the thread which initiated the crash).
   f. EditSystemExistenceF99410 in Thread6.

So from this example, we see that the thread running DetermineifBlanketPOExists initiated a process crash, but there were three instances of EditSystemExistenceF99410 running in three other threads. Therefore, one should look for SARs containing fixes for these two business functions and apply those ESUs.

It's also possible that the corruption in this crashed CallObject was caused by some other business function which was no longer running at the point of crash. Therefore, it is helpful to analyze a few CallObject crash .dmp files so that an overall pattern of suspect business functions appear and we can then use these to select a narrow range of ESUs to apply. This method of analysis is quite focused and presents one with a given set of fixes to apply rather than choose wide swathes of possible ESUs.

Call Object or RunBatch Crash due to Out of Memory

CallObjects or Runbatch processes may also crash when they are unable to allocate memory. This can happen for two reasons:

1. If this process hits the 'per-process' memory limit for the given OS. This may happen due to the process leaking memory or consuming excessive memory.
2. General lack of memory in the machine environment where it is running. This is usually due to an undersized box or a side effect of leaking processes on other sibling processes.

Similar to memory corruption, out of memory also produces a .dmp.log file. But in this case, it produces a Memory Dump which contains the list of all EnterpriseOne objects in the process memory at the time of crash. If the crash is due to raw allocation of memory or a third party leak, then this memory diagnostic dump will not show any large quantities of EnterpriseOne objects.

This is a dmp.log for a crash due to Out of Memory:

```
Apr 5 17:31:27.459991 DEBUG INIT0 - 6329 **** jdeDebugInit -- output disabled in
INI file.
Apr 5 17:31:27.461443 jdemem.c131 - 6329 BMD OFF - Not running BSFN MEMORY
DIAGNOSTICS v8.98.2.0 level 0
Apr 5 17:59:03.229258 jdb_utl1.c13485 - 6329/2 MEMORY ALLOCATION FAILURE
Apr 5 17:59:03.229365 jdb_utl1.c13485 - 6329/2 File: jdb_rq1.c Line: 192
********** Begin OS Data **********
Memory Usage = 4281MB
CPU Usage = 0%
Number of Threads = 7
********** End OS Data **********
********** Begin Detailed Memory Data **********
********** Begin Process Data **********
ENVIRONMENT,Ptr=00652d78,Env=JPD812,PathCode=PD812
JDBTABLECACHE,Ptr=007ccec8,Name=JDB_BV_1270510706JPD812F4009,#Records=1
JDBTABLECACHE,Ptr=003e6f38,Name=JDB_BV_1270510706JPD812F40203,#Records=2
JDBTABLECACHE,Ptr=003ed418,Name=JDB_BV_1270510706JPD812F40205,#Records=1
JDBTABLECACHE,Ptr=004fb430,Name=JDB_BV_1270510706JPD812F41001,#Records=2
JDBTABLECACHE,Ptr=00501078,Name=JDB_BV_1270510706JPD812F41002,#Records=3294
JDBTABLECACHE,Ptr=0051ead0,Name=JDB_BV_1270510706JPD812F7306,#Records=1
JDBTABLECACHE,Ptr=00319358,Name=JDB_BV_1270510706JPD812F99410,#Records=2
JDBTABLECACHE,Ptr=00338690,Name=JDB_BV_1270510706JPD812F0004,#Records=25
JDBTABLECACHE,Ptr=00342d40,Name=JDB_BV_1270510706JPD812F0005,#Records=27
JDBTABLECACHE,Ptr=00346cf8,Name=JDB_BV_1270510706JPD812F0006,#Records=1
JDBTABLECACHE,Ptr=00378de1,Name=JDB_BV_1270510706JPD812F0901,#Records=948722
JDBTABLECACHE,Ptr=00349ca0,Name=JDB_BV_1270510706JPD812F0008,#Records=1
JDBTABLECACHE,Ptr=0034e6a8,Name=JDB_BV_1270510706JPD812F0010,#Records=2
JDBTABLECACHE,Ptr=006454d8,Name=JDB_BV_1270510707JPD812F1609,#Records=1
OCIDBCONN,Ptr=0012e530,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=AutoInUse
,Comm
itMode=A,RefCount=196
OCIDBCONN,Ptr=0082ee18,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Co
mmitMode=M,RefCount=1
OCIDBCONN,Ptr=006aca68,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Co
mmitMode=M,RefCount=1
OCIDBCONN,Ptr=00a46d08,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Co
mmitMode=M,RefCount=1
OCIDBCONN,Ptr=00880b98,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualInU
se,Co
mmitMode=M,RefCount=1
OCIDBCONN,Ptr=015f64c0,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualAva
ilable,C
ommitMode=M,RefCount=0
```

```
OCIDBCONN,Ptr=01b150c8,DBServer=densun28,DBUser=JDE,TNSDB=orcl,ConnState=ManualAva
ilable,
CommitMode=M,RefCount=0
********** End Process Data **********
********** Begin Session Data **********
SESSION,Ptr=000bdb60,User=JDE,Env=JPD812,Role=*ALL,Machine=denicsn5,SignOnTime= 4/
5/2010 17:38:27,LastActiveTime= 4/ 5/2010 17:58:55
OPENJDBTRANSACTION,Ptr=0014eb28,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdb_ctl.c,Function=JDB_LoadEnv,Line=5829
TABLE,Ptr=00a2bac0,Name=F0013,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0013RowUsingCurrencyCode,Line=73
TABLE,Ptr=013f0830,Name=F0010,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0010RowUsingCompanyNumber,Line=246
OPENJDBTRANSACTION,Ptr=007f4538,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdb_ctl.c,Function=CallStartupBusinessFunction,Line=8551
OPENJDBTRANSACTION,Ptr=004299c0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdekinit.c,Function=JDEK_ProcessInitUserRequest,Line=356
OPENJDBTRANSACTION,Ptr=00783a08,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdekinit.c,Function=JDEK_ProcessInitUserRequest,Line=356
JDECACHE,Ptr=000fabe8,Name=2XT4311Z1A,#Cursors=1,#Records=0,#Indices=1,#References
=1,File=xt4311z1.c,Function=F4311InitializeCaching,Line=23018
JDECACHE,Ptr=005ce150,Name=2XT4311Z1B,#Cursors=1,#Records=0,#Indices=2,#References
=1,File=xt4311z1.c,Function=F4311InitializeCaching,Line=23027
JDECACHE,Ptr=00ef3390,Name=2XT4311Z1C,#Cursors=1,#Records=1,#Indices=1,#References
=5,File=xt4311z1.c,Function=IXT4311Z1_InitiateOrderCache,Line=5758
JDECACHE,Ptr=014397a8,Name=2B4302570Cache,#Cursors=1,#Records=0,#Indices=1,#Refere
nces=2,File=b4302570.c,Function=ApprovalsFieldConstants,Line=158
JDECACHE,Ptr=014f8000,Name=2F45UI73,#Cursors=1,#Records=0,#Indices=2,#References=4
,File=b4500200.c,Function=F4573GetNextFreeGood,Line=100
JDECACHE,Ptr=00beae30,Name=2B3201470213107,#Cursors=1,#Records=0,#Indices=7,#Refer
ences=3,File=b3201470.c,Function=I3201470_CreateInitCache,Line=732
JDECACHE,Ptr=00b65350,Name=2B3201470213144,#Cursors=1,#Records=0,#Indices=7,#Refer
ences=3,File=b3201470.c,Function=I3201470_CreateInitCache,Line=732
JDECACHE,Ptr=004d1438,Name=2B4302180F632910,#Cursors=1,#Records=1,#Indices=1,#Refe
rences=19,File=b4302180.c,Function=CacheProcessPOHeaderCache,Line=1173
JDECACHE,Ptr=01876570,Name=2213190PricingDecimals,#Cursors=2,#Records=2,#Indices=1
,#References=1,File=b4504500.c,Function=CreateDataMapCDIST,Line=229
JDECACHE,Ptr=006d05d0,Name=2213190DataMapCDISTCache,#Cursors=1,#Records=3,#Indices
=1,#References=1,File=b4504500.c,Function=InitDataMapCacheCDIST,Line=2108
JDECACHE,Ptr=01765910,Name=2213190PricingHistory,#Cursors=2,#Records=1,#Indices=1,
#References=1,File=b4504500.c,Function=CreateDataMapCDIST,Line=229
JDECACHE,Ptr=017ed260,Name=2213190PricingCatCodes,#Cursors=2,#Records=1,#Indices=1
,#References=1,File=b4504500.c,Function=CreateDataMapCDIST,Line=229
JDECACHE,Ptr=0104aaf0,Name=2B4302180G632910,#Cursors=1,#Records=6,#Indices=2,#Refe
rences=12,File=b4302180.c,Function=CacheProcessPODetailCache,Line=1510
JDECACHE,Ptr=015b0138,Name=2B4302180H632910,#Cursors=1,#Records=0,#Indices=3,#Refe
rences=6,File=b4302180.c,Function=CacheProcessBlanketCache,Line=1848
JDECACHE,Ptr=013c0960,Name=2F40UI74_
213190,#Cursors=1,#Records=6,#Indices=3,#References=1,File=b4504610.c,Function=F40
UI74_Init,Line=719
JDECACHE,Ptr=013b0f80,Name=2F40UI70-213190,#Cursors=1,#Records=0,#Indices=5,#Refer
ences=18,File=b4500720.c,Function=ProcessPriceAdjustmentListCache,Line=335
SESSION,Ptr=00811f78,User=JDE,Env=JPD812,Role=*ALL,Machine=denicsn5,SignOnTime= 4/
5/2010 17:38:44,LastActiveTime= 4/ 5/2010 17:58:56
OPENJDBTRANSACTION,Ptr=0084a660,CommitMode=Auto,Owner=InitUser,AppName=UNKNOWN),Fi
le=jdb_ctl.c,Function=JDB_LoadEnv,Line=5829
TABLE,Ptr=00d4a3b8,Name=F0013,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0013RowUsingCurrencyCode,Line=73
TABLE,Ptr=0142e000,Name=F0010,CommitStatus=Active,File=jdb_curr.c,Function=JDB_
RetrieveF0010RowUsingCompanyNumber,Line=246
```

```
OPENJDBTRANSACTION,Ptr=0084a968,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdb_ctl.c,Function=CallStartupBusinessFunction,Line=8551
OPENJDBTRANSACTION,Ptr=00822b00,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdekinit.c,Function=JDEK_ProcessInitUserRequest,Line=356
OPENJDBTRANSACTION,Ptr=00878840,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),F
ile=jdekinit.c,Function=JDEK_ProcessInitUserRequest,Line=356
JDECACHE,Ptr=0085a448,Name=5XT4311Z1A,#Cursors=1,#Records=0,#Indices=1,#References
=1,File=xt4311z1.c,Function=F4311InitializeCaching,Line=23018
JDECACHE,Ptr=00a3a360,Name=5XT4311Z1B,#Cursors=1,#Records=0,#Indices=2,#References
=1,File=xt4311z1.c,Function=F4311InitializeCaching,Line=23027
JDECACHE,Ptr=0104ea50,Name=5XT4311Z1C,#Cursors=1,#Records=1,#Indices=1,#References
=5,File=xt4311z1.c,Function=IXT4311Z1_InitiateOrderCache,Line=5758
JDECACHE,Ptr=00e868a8,Name=5B4302570Cache,#Cursors=1,#Records=0,#Indices=1,#Refere
nces=2,File=b4302570.c,Function=ApprovalsFieldConstants,Line=158
********** End Session Data **********
********** End Detailed Memory Data **********
```

Analysis of Memory Dumps (for preceding .dmp file)

1. Look for the memory usage at the point of the crash. If this number is much lower than the "per-process" memory limit as previously determined in SM Console, then it means that the system as a whole is running low on memory and this process is a victim of low memory available to it.
2. If the process crashed at memory utilization close to the "per-process" memory limit, then we need to evaluate the likely suspects which might be excessively consuming the memory in the process.
3. Look at the global Process level objects such as JDBTABLECACHE or Database connections for any extreme numbers.
4. After that, analyze the perSession objects such as JDBTRANSACTIONS(manual) or JDECACHEs or DATAPOINTERs or JDBRequests(TABLE/VIEW) in each session.
5. In our example above, the culprit is the JDB Caching of F0901 Table which has lead to caching of close to a million (948,722) data rows in memory, leading to a "memory exhaustion" condition . This is easily solved by either Dynamically Clearing Cache from Server Manager or removing this table from JDB Cache from P98613 Application.

Out of Threads

A Callobject can crash if it consumes an excessive number of threads. Usually this limit is twice the number of threadpoolsize as set in jde.ini. This will also produce a .dmp.log file which will contain a list of all threads and their callstacks - this will help in determining where the source of the threads. On System-i enterprise server, there will not be any dmp file. Instead the callstacks for all threads will be logged in jde.log.

Query Exceeding a Threshold

To determine which DB queries are taking an excessive amount of time to execute, the following setting is available in Server Manager:

Figure 14–15 Query Execution Time Threshold

Setting this value greater than 0 will start a timer for each DB query. If the amount of time taken to execute the DB query equals or exceeds this value, two messages will be written to the jde log for the E1 process. The first line will contain the text indicating that the DB query threshold has been exceeded. This line will also contain the E1 user and DB proxy user that executed the query. The second line will contain the actual DB query that exceeded the threshold.

This is a sample of these messages:

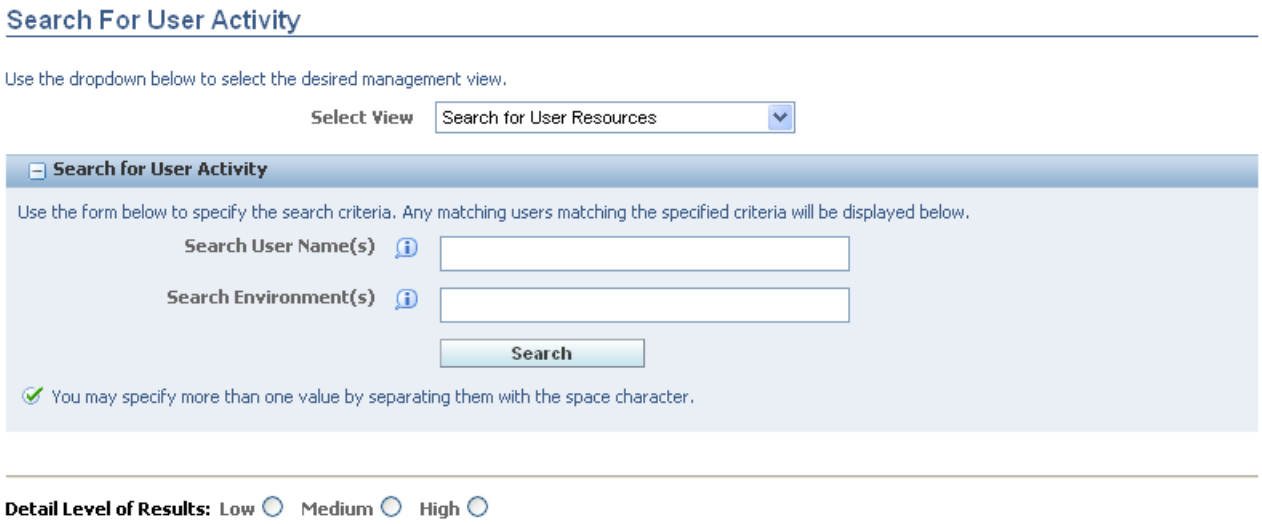
```
859026/452 MAIN_THREAD Tue Apr 20 16:41:26.044816
dbdrv_log.c196
OS400AG016 - doQueryDiagnostics: The following SQL query took 5 seconds which is
equal to or greater than QueryExecutionTimeThreshold (1 seconds) for
E1User(JOESMITH) with DBProxyUser(JDE).

859026/452 MAIN_THREAD Tue Apr 20 16:41:26.044960
dbdrvag.c1394
SELECT * FROM SVM812/F986110 WHERE ( JCJOBSTS = 'P' AND JCFUNO = 'UBE' AND JCPRTQ
= '6003' AND JCEXEHOST = 'JDESERVER1' ) ORDER BY JCEXEHOST ASC,JCJOBNBR ASC
```

Terminating HTML User Sessions

An administrator can terminate user sessions on EnterpriseOne HTML servers. This can be used in conjunction with the Recycle Kernel button, to stop a user session that appears to be out of control, while allowing other user sessions on the same CallObject kernel to complete normally.

User sessions can be found using Server Manager, by selecting the view Search for User Resources from the main Management Dashboard page.

Figure 14–16 Search for User Activity



The search list can be narrowed by specifying user names and environments. Clicking on the link HTML Server Session will bring up a page where those users on that same HTML server instance can be selected for termination. An alternative way to get to the same page is to select the link to User Sessions in the Runtime Metrics block of the HTML server instance main page. On that page, user sessions can be selected by check box. The CNC admin can send a message to the user that their session will be terminated. Then the CNC admin can terminate the selected user sessions.

For a CallObject kernel that has already begun recycling, that kernel will shut down when all of its user sessions have either logged off or been terminated by an administrator.

14.7.2.7 Logging and Diagnostics

Logging Improvements

At the time of user signout, all EnterpriseOne related memory structures should be freed. If they are not, then the three new settings listed in the Error and Debug Logging section can help isolate the value and origin of these outstanding memory objects. These memory objects are gracefully removed when a user signs out. However, these represent a potential buildup of objects in processes that do not represent a problem in long running processes such as subsystem jobs, long running batch jobs, or if an interactive user does not sign out for a long time (such as interop jobs).

Error and Debug Logging section

Three new fields have been added to the Error and Debug Logging section:

Field	Description
Log memory diagnostics at signoff	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - DEBUG INI Entry - logMemDiagsAtSignoff Default Value - FALSE Allowed Values: ■ TRUE ■ FALSE This setting specifies whether or not memory diagnostics will be logged when a CallObject Kernel user session is ended or a UBE process ends. The memory diagnostics will be written to the jdedebug log.
Log JDE Cache leaks at signoff	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - DEBUG INI Entry - logCacheLeaksAtSignoff Default Value - FALSE Allowed Values: ■ TRUE ■ FALSE This setting specifies whether or not JDE Cache leaks will be logged when a user session is freed in an E1 Server process. Details about the leaked JDE Caches will be written to the jde log.
Log Data Pointer leaks at signoff	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - DEBUG INI Entry - logDPLeaksAtSignoff Default Value - FALSE Allowed Values: ■ TRUE ■ FALSE This setting specifies whether or not Data Pointer leaks will be logged when a user session is freed in an E1 Server process. Details about the leaked Data Pointers will be written to the jde log.

14.7.2.8 Advanced Profiling

The advanced Profiling section discusses:

- JDEHEAP Advanced Diagnostics Engine (JADE)
- Business Function Memory Diagnostics (BMD)

JDEHEAP Advanced Diagnostics Engine (JADE)

This section discusses:

- Description of JDEHEAP Advanced Diagnostics Engine (JADE)
- JADE Standard Mode
- JADE Advanced Mode

- JDEHEAP ADVANCED DIAGNOSTICS ENGINE (JADE) Configuration

Description of JDEHEAP Advanced Diagnostics Engine (JADE)

JADE (JDEdwards Heap Advanced Diagnostic Engine) and BMD (Business Function Memory Diagnostic) are two advanced profiling tools which work cross-platform with the production level builds to diagnose memory leak issues.

These Profilers are to be used when the normal memory diagnostics do not yield adequate information.

BMD and JADE have a common problem domain of finding the source of memory consumption or leak within JDEdwards Enterpriseone Kernel Space. However, they specialize in different use cases:

- BMD Level 1 gives and overall idea of memory consumption.
- BMD Level 2 and 3 provide growth of memory as a function of business functions.

Thus, BMD helps isolate which business functions are likely candidates for a memory consumption or leak.

On the other hand, JADE is a finer diagnostic tool and it works by collecting all memory allocations from the given Enterprise Server Kernel.

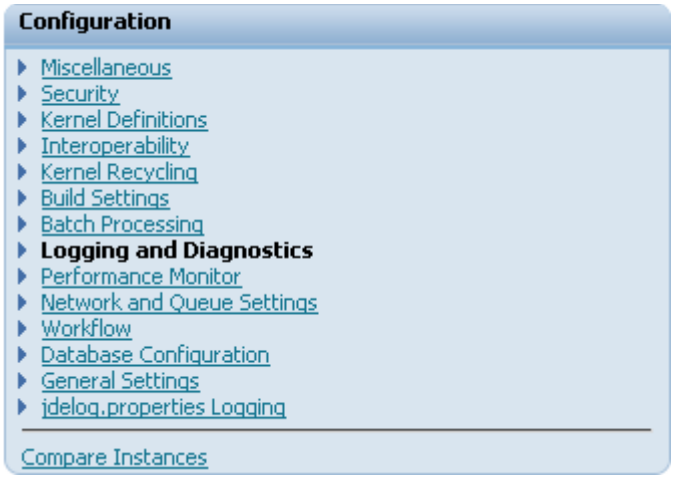
All memory that is left unallocated at the end of the run can be considered as a memory leak or excessive consumption - if start and stop points are chosen judiciously.

JADE Standard Mode

JDEHEAP ADVANCED DIAGNOSTICS ENGINE (JADE) provides a method to identify the location of JdeHeap memory leaks. Note: do not use these settings unless investigating a "Tools-level" problem, as system performance may be affected.

Configuration settings for JADE are accessed by selecting Logging and Diagnostics in the Configuration section.

Figure 14–17 Configuration section



For memory investigations within the scope of BSFNs, use the buttons on the Server Manager process screen instead of these INI settings. JADE tracks memory usage of all jdeHeap functions (jdeAlloc, jdeCalloc, jdeRealloc, and jdeFree). The JADE JDE.INI settings take affect whenever a process starts. They will be ignored and deactivated whenever using a JADE button on the Server Manager process screen.

All JADE logging is placed in the jdedebug.log files, even if jdedebug logging is turned off in the JDE.INI.

Use Case 1

If an Application is known to have a memory leak:

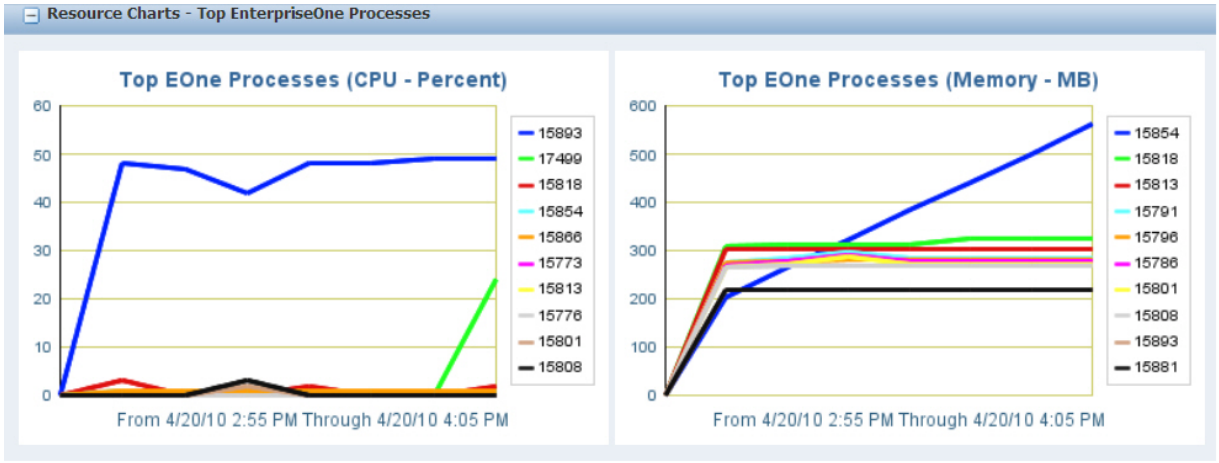
1. Launch the Application to the App Entry point.
2. Associate the Application to a given CallObject by correlating under "Remote Env" section of HTML WebSessions.
3. Drill into the Callobject Kernel and click on "Start JADE" Button.
4. Execute the pre-defined transactions on the application.
5. When done, click on "Dump JADE" Button.
6. Exit the Application and click on on "Stop JADE" Button.
7. Click "Parse JADE" to see the possible leak locations.

In the previous use case we can determine if there is any unallocated memory at the end of Application's run. If yes, then it can be determined from where it is being allocated.

Use Case2

If a running process, such as RunBatch, has a runaway memory leak:

Figure 14–18 Top memory usage identified by graph and by sorting on Memory column.



Select [Process]: Remove Zombie

Select All | Select None Previous 1 - 10 Next

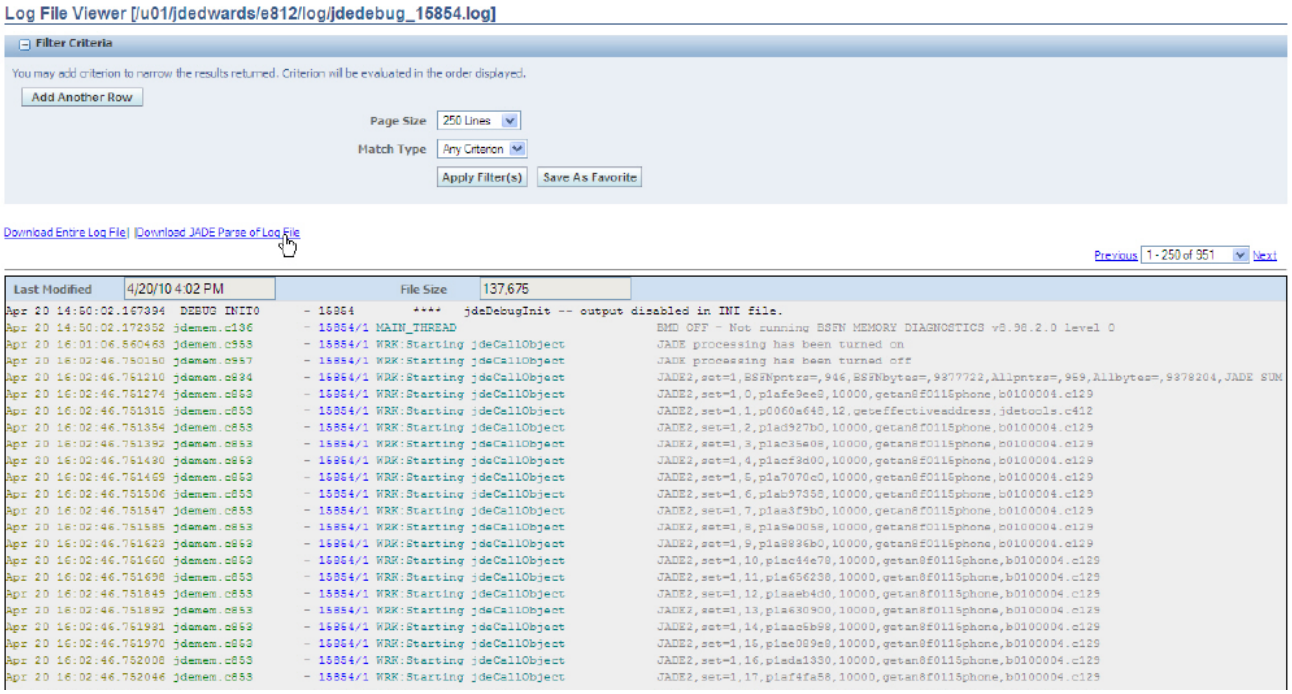
	Process Name	Process Type	Process ID	Process Status	JDELOG File Size	Debug Log Size	Connected Users	Total Requests	Outstanding Requests	Memory (MB)
☐	R014021 XJDE0001	Runbatch	15854	RUNNING	968	97	0	0	0	561
☐	MANAGEMENT KERNEL	Kernel Process	15818	RUNNING	391	97	0	812	0	326
☐	METADATA KERNEL	Kernel Process	15813	RUNNING	866	97	0	2377	0	303

1. Drill into the known process.

Figure 14–19 Press Start JADE button.

2. Enable JADE by pressing on "Start JADE".
3. Let the process run for representative period of leak.
4. Press "Stop JADE" Button.
5. Click "Parse JADE" to see the possible leak locations.

Figure 14–20 Click Download JADE Parse of Log File.



6. Download JADE Parse file.

Figure 14–21 Parsed JADE Log File

	A	B	C	D	E	F	G	H
1	seqnum	month/date	setNumber	.	BSFNpntrs	.	BSFNbytes	.
2	1	02:46.8	set=1	BSFNpntrs	946	BSFNbytes=	9377722	Allpntrs=
3	seqnum	month/date	setNumber	indexInSet	pointer	memSize	bsfnName	fileNameLineNumber
4	2	02:46.8	set=1	0	p1afe9ee8	10000	getan8f0115phone	b0100004.c129
5	3	02:46.8	set=1	1	p0060a648	12	geteffectiveaddress	jdetools.c412
6	4	02:46.8	set=1	2	p1ad927b0	10000	getan8f0115phone	b0100004.c129
7	5	02:46.8	set=1	3	p1ac35e08	10000	getan8f0115phone	b0100004.c129
8	6	02:46.8	set=1	4	p1acf3d00	10000	getan8f0115phone	b0100004.c129
9	7	02:46.8	set=1	5	p1a7070c0	10000	getan8f0115phone	b0100004.c129
10	8	02:46.8	set=1	6	p1ab97358	10000	getan8f0115phone	b0100004.c129
11	9	02:46.8	set=1	7	p1aa3f9b0	10000	getan8f0115phone	b0100004.c129
12	10	02:46.8	set=1	8	p1a9e0058	10000	getan8f0115phone	b0100004.c129
13	11	02:46.8	set=1	9	p1a8836b0	10000	getan8f0115phone	b0100004.c129
14	12	02:46.8	set=1	10	p1ac44e78	10000	getan8f0115phone	b0100004.c129
15	13	02:46.8	set=1	11	p1a656238	10000	getan8f0115phone	b0100004.c129
16	14	02:46.8	set=1	12	p1aaeb4d0	10000	getan8f0115phone	b0100004.c129
17	15	02:46.8	set=1	13	p1a630900	10000	getan8f0115phone	b0100004.c129
18	16	02:46.8	set=1	14	p1aac5b98	10000	getan8f0115phone	b0100004.c129
19	17	02:46.8	set=1	15	p1ae089e8	10000	getan8f0115phone	b0100004.c129
20	18	02:46.8	set=1	16	p1ada1330	10000	getan8f0115phone	b0100004.c129
21	19	02:46.8	set=1	17	p1af4fa58	10000	getan8f0115phone	b0100004.c129
22	20	02:46.8	set=1	18	p1a7c3450	10000	getan8f0115phone	b0100004.c129
23	21	02:46.8	set=1	19	p1ac54ee8	10000	getan8f0115phone	b0100004.c129
24	22	02:46.8	set=1	20	p1a666aa8	10000	getan8f0115phone	b0100004.c129
25	23	02:46.8	set=1	21	p1aafbd40	10000	getan8f0115phone	b0100004.c129
26	24	02:46.8	set=1	22	p1a67a008	10000	getan8f0115phone	b0100004.c129
27	25	02:46.8	set=1	23	p1ae8ece8	10000	getan8f0115phone	b0100004.c129
28	26	02:46.8	set=1	24	p1ad05080	10000	getan8f0115phone	b0100004.c129
29	27	02:46.8	set=1	25	p1ab0f2a0	10000	getan8f0115phone	b0100004.c129
30	28	02:46.8	set=1	26	p1ae9f248	10000	getan8f0115phone	b0100004.c129
31	29	02:46.8	set=1	27	p1aba86d8	10000	getan8f0115phone	b0100004.c129
32	30	02:46.8	set=1	28	p1a892230	10000	getan8f0115phone	b0100004.c129

7. Open the downloaded Parse file in favorite CSV reader and analyze memory allocation patterns which may suggest the location of the leaks.

JADE Advanced Mode

JADE Advanced Mode provides a deeper level of information for analysis of situations when Standard Mode is not able to resolve the issue. For example, if one is working to resolve an issue of high memory usage, but doesn't see any corresponding rise in memory of any known EnterpriseOne objects either in analyzing the graphs or using the Diagnostic buttons, then Advanced Mode is needed.

Three levels of JADE logging are available:

- JADE level 1 logs a summary of memory usage.
- JADE level 2 logs the summary, plus detail lines for each BSFN-scoped memory pointer.
- JADE level 3 logs the summary, plus detail lines for all jdeHeap memory pointers.

There is an option to start JADE memory tracking based on specified BSFN triggers, and also an option to turn on Debug logging when the BSFN trigger is met. JADE data is dumped to the log file based on a minimum time interval. The INI-initiated JADE tracking is terminated whenever a JADE button on the Server Manager process screen is used.

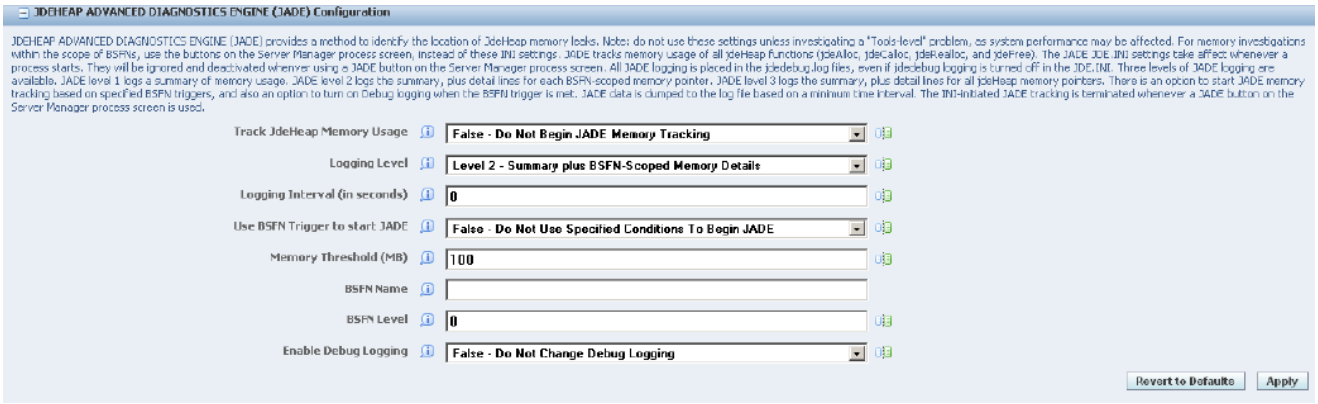
JDEHEAP Advanced Diagnostics Engine (JADE) Configuration

JADE can be configured by accessing the JDEHEAP Advanced Diagnostics Engine (JADE) Configuration section.

To access this section:

1. In Server Manager, select the EnterpriseOne server.
2. Select Logging and Diagnostics in the Configuration Section.

Figure 14–22 JDEHEAP ADVANCED DIAGNOSTICS ENGINE (JADE) Configuration section



The fields used to configure JADE are:

Field	Description
Track JdeHeap Memory Usage	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - trackMemUsage Default Value - 0 Allowed Values: • False - Do Not Begin JADE Memory Tracking (0) • True - Begin JADE Memory Tracking At Process Start (1) Track jdeHeap memory usage by JADE. The tracking will start with the first jdeHeap usage of the process. It will terminate when the process shuts down and when any of the JADE buttons on the Server Manager process screen are used.
Logging Level	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - logLevel Default Value - 2 Allowed Values: • Level 1 - Summary of JdeHeap Memory Usage (1) • Level 2 - Summary plus BSFN-Scoped Memory Details (2) • Level 3 - Summary plus All JdeHeap Memory Details (3) Level of logging the JADE diagnostics. Level 1 logs a summary of BSFN-scoped pointers and of all pointers. Level 2 logs the summary, plus detail lines for each BSFN-scoped jdeHeap usage. Level 3 logs the summary, plus detail lines for all jdeHeap usage.

Field	Description
Logging Level (in seconds)	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - logInterval Default Value - 0 Logging interval (in seconds) between dumping of JADE data to the debug log file. This is a minimum interval, not an exact interval. No dumping will be done if there has been no jdeHeap activity since the last dumped data. This interval-based dumping will be terminated if any of the JADE buttons on the Server Manager process screen are used.
Use BSFN Trigger to start JADE	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - bsfnTriggerUseTrigger Default Value - 0 Allowed Values; • False - Do Not Use Specified Conditions To Begin JADE (0) • True - Use Specified Conditions To Begin JADE (1) Use BSFN Trigger to begin JADE memory tracking. The BSFN-related trigger conditions are given in this section of the INI file. When the trigger conditions are met, JADE memory tracking begins. The trigger does not cause any JADE data to be dumped to the log files.
Memory Threshold (MB)	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - bsfnTriggerMemThresholdMB Default Value - 100 Memory threshold (in megabytes) to begin JADE tracking of jdeHeap memory usage. This can be combined with a specified BSFN name and level to trigger when teacking starts.
BSFN Name	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - bsfnTriggerBsfnName Business Function name. When combined with the BSFN level and memory threshold, triggers the JADE tracking.
BSFN Level	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - bsfnTriggerBsfnLevel Default Value - 0 Business Function level. When combined with the BSFN name and memory threshold, triggers JADE tracking. Not used when set to zero.

Field	Description
Enable Debug Logging	INI Filename - /u01/jdedwards/e812/ini/JDE.INI INI Section Name - JADE INI Entry - bsfnTriggerEnableDebug Default Value - 0 Allowed Values: • False - Do Not Change Debug Logging (0) • True - Turn On Debug Logging When JADE Is Triggered (1) Option to turn on debug logging. If selected, debug logging will be dynamically turned on when trigger conditions for JADE are met. This does not change the debug logging settings of JDE.INI.

Business Function Memory Diagnostics (BMD)

This section discusses:

- Business Function Memory Diagnostics (BMD)
- Logged Values in BMD
- Configuration of BMD
- BMD Parsing

Description of Business Function Memory Diagnostics (BMD)

The KRM system enables you to engage BMD Profiling and includes BMD (BSFN Memory Diagnostics) in an easy-to-use format. BMD is a memory profiler which can track and isolate memory leaks to a particular business function and provides output of memory in .csv format. The BMD is considered a last resort to identify the cause of memory problems.

There are three levels of BMD:

- The first level provides memory usage through the lifetime of all processes, to help narrow down where the problem is occurring.
- The second level provides information about Business Functions (BSFNs) (for BSFN levels 1 and 2) after the kernel memory exceeds a given threshold.
- The third level provides information about BSFNs (all BSFN levels) after the kernel memory exceeds a given threshold within a specified BSFN running at a specified level.

The first level is sampled memory logging that can be used for all types of kernel processes. The second and third levels are BSFN-specific memory logging. The types of kernel processes that run BSFNs are CallObject kernels, UBEs, and Subsystems.

The BMD data is written to the debug log file, even if debug logging is not turned on. The trigger for the third BMD level can also be used to activate full debug logging, to provide a detailed context around the memory problems.

Waiting to turn on debug logging until after the trigger conditions are met can eliminate huge debug-log files, which might take a lot of analysis to identify the portion of the log file that is related to the actual problem.

Logged Values in BMD

BMD Level 1 - collected at the allocation frequency specified in the BMD configuration - Allocation Frequency:

- Current CPU percent
- Current total process memory being used

BMD Level 2 - collected at the exit point of each Business Function after the specified Memory Threshold is hit for Business Functions at Level 1 and 2 only:

- Current CPU percent
- Current total process memory being used
- Change in the memory during the BSFN (delta-memory)
- BSFN name
- BSFN level

BMD Level 3 - collected at the exit point of each Business Function after the specified Threshold of Memory AND Business Function Name AND Business Function Level (any level) is hit:

- Current CPU percent
- Current total process memory being used
- Change in the memory during the BSFN (delta-memory)
- BSFN name
- BSFN level
- Debug Logging (Optional)

Configuration of BMD

These are the navigation steps to access the configuration of BMD:

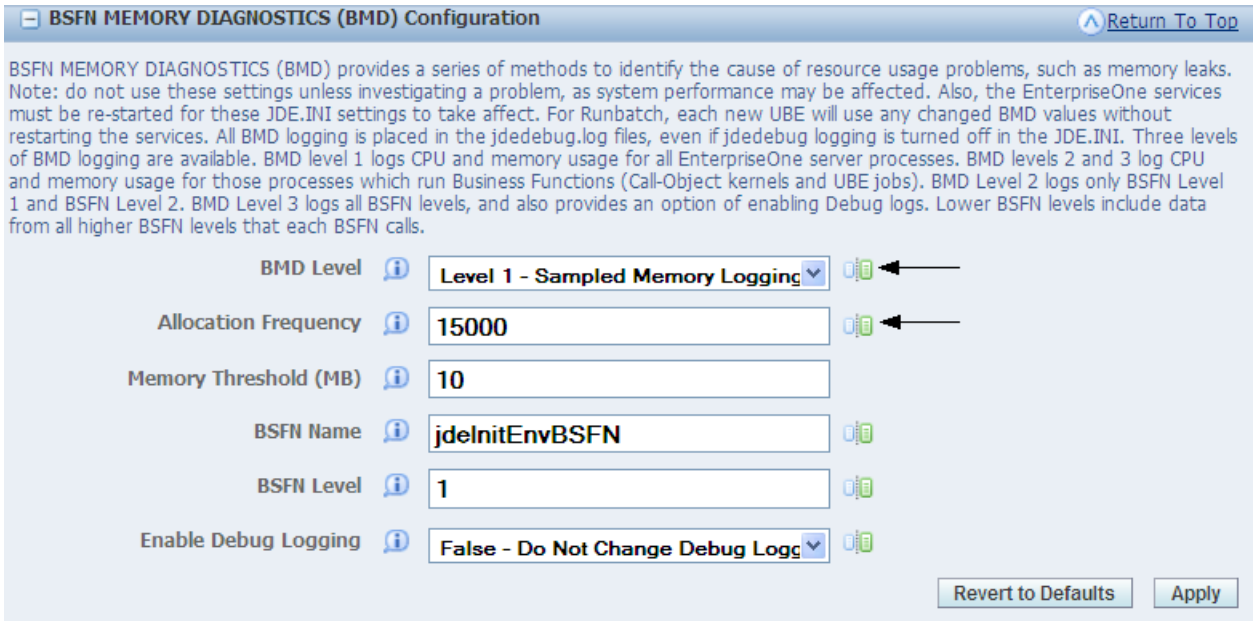
- Select the enterprise server instance in server manager.
- You should see the <EnterpriseOne Enterprise Server> instance.
- Make sure that the server is up and running.
- Go to the <Configuration> section.
- Click on the <Logging and Diagnostics> link.
- Find the BSFN Memory Diagnostics (BMD) Configuration section.

The BMD JDE.INI settings specify the BMD level, and then the trigger conditions within each level. The settings are found in the JDE.INI in a block called "[BSFN MEMORY DIAGNOSTICS]". They are found in Server Manager in the configuration link called "Logging and Diagnostics".

The primary BMD setting is the BMD level, called "bmdLevel". That can be 0 (BMD off), or 1, 2, or 3 (BMD level).

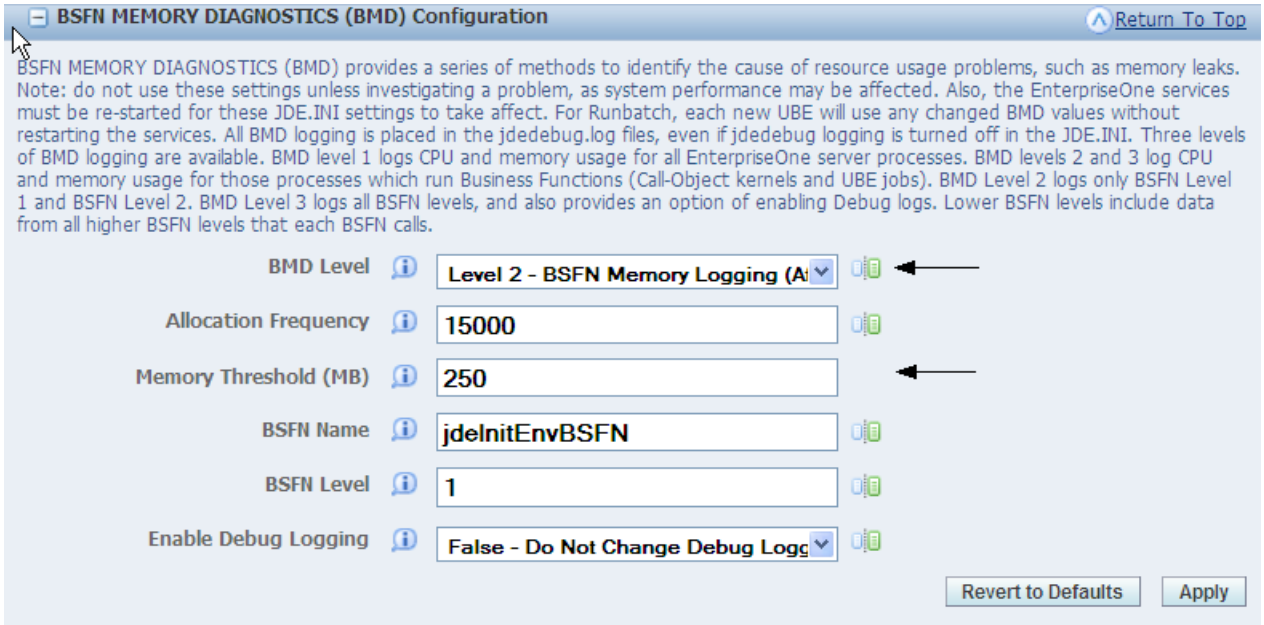
For BMD level 1, the trigger condition is based on the allocation frequency (called "allocFrequency"). That is a count of all calls to the system-level allocation functions (malloc, calloc, and realloc). The default setting is 15000, with a minimum of 7000. At 15000, for active CallObject kernels, this can result in memory-level logging several times per minute.

Figure 14–23 BSFN Memory Diagnostics (BMD) Configuration section – Level 1



For BMD level 2, the trigger condition is passing a specified memory threshold (called "memThresholdMB"). After that memory threshold is first passed, there will be BMD logging each time a BSFN level 1 or 2 exits.

Figure 14–24 BSFN Memory Diagnostics (BMD) Configuration section – Level 2



For BMD level 3, the trigger is based on a combination three conditions: a specified memory threshold (called "memThresholdMB", same as BMD level 2), a specified BSFN name (called "bsfnName"), and a specified BSFN level (called "bsfnLevel"). After those three conditions are met, there will be BMD logging each time a BSFN (all BSFN levels) exits. BMD level 3 has the option of turning on full debug logging starting when the trigger conditions are met. The optional setting is called "enableDebug", with

possible values of 0 (debug logging unchanged) or 1 (debug logging turned on if it had been off).

Figure 14–25 BMD Configuration section – Level 3, Enable Debug Logging = False (debug off)

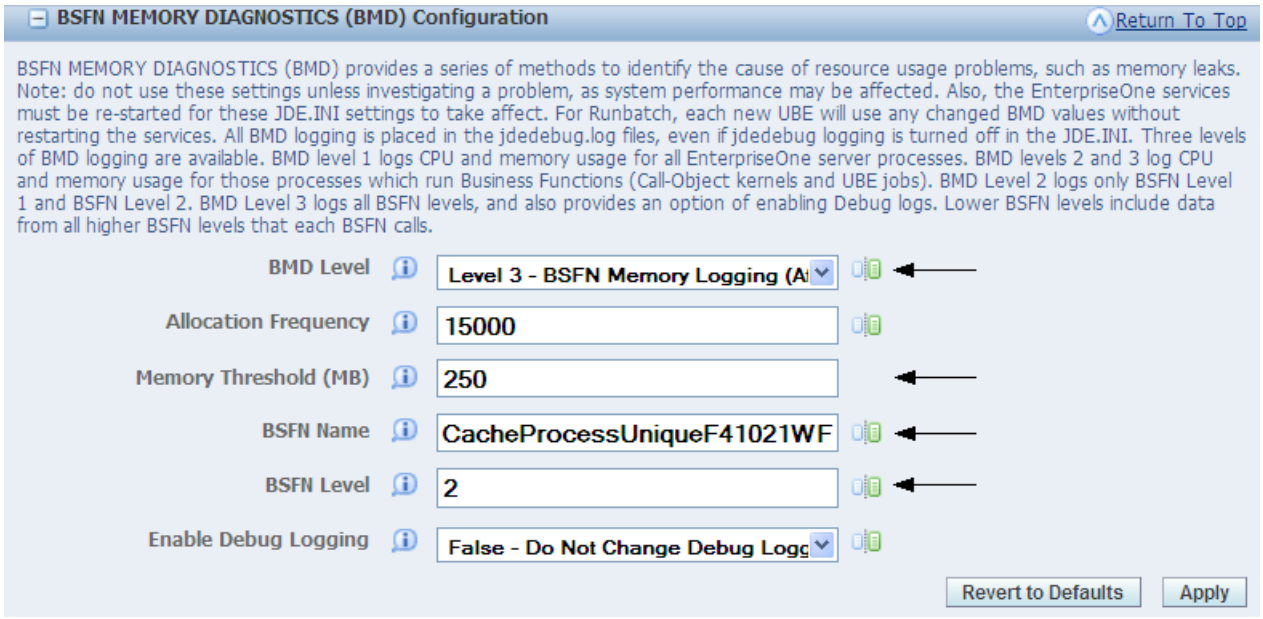
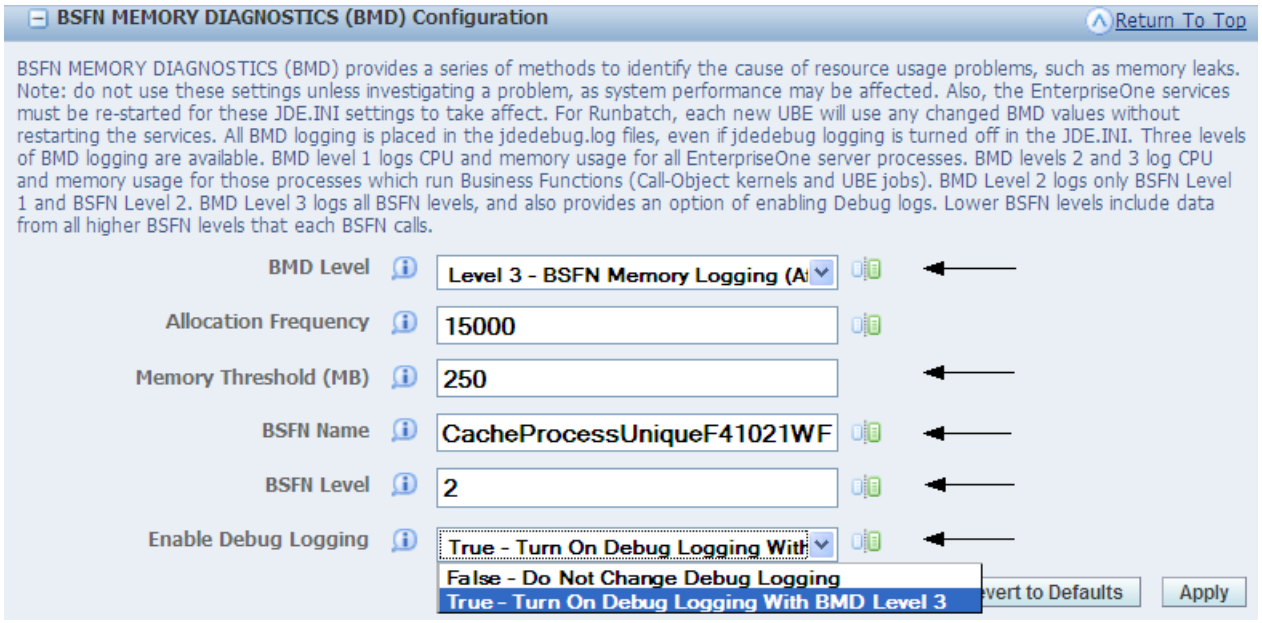


Figure 14–26 BMD Configuration section – Level 3, Enable Debug Logging = True (debug on)



The EnterpriseOne services must be re-started for any changed BMD settings to take affect. For Runbatch, each new UBE will use any BMD values that are changed before that UBE starts.

14.7.2.9 BMD Parsing

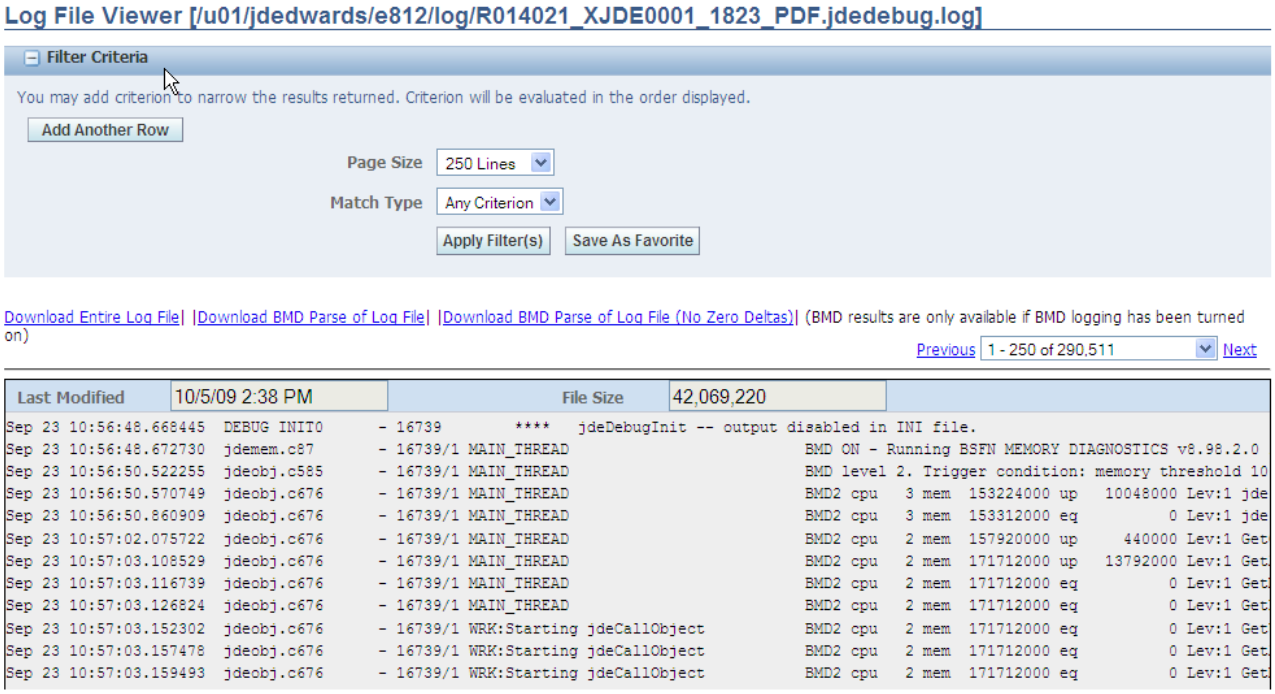
These are the navigation steps to access BMD Parsing:

- Select the enterprise server instance in server manager.

- You should see the <EnterpriseOne Enterprise Server> instance.
- Make sure that the server is up and running.
- Go to the <Available Log Files> section.
- Select a BMD parsing option link.

The BMD feature also includes an easy parsing system that is set up through links on the Server Manager log file screen.

Figure 14–27 Log File Viewer



For level 1 BMD data, there is one parsing option that puts all of the data in the CSV format. For levels 2 and 3, there are two parsing options. One brings all BMD data for the BSFNs into the CSV format. The other excludes those BMD data rows which have a zero delta. The second option presumably keeps the most interesting BMD data, while reducing the CSV file size substantially.

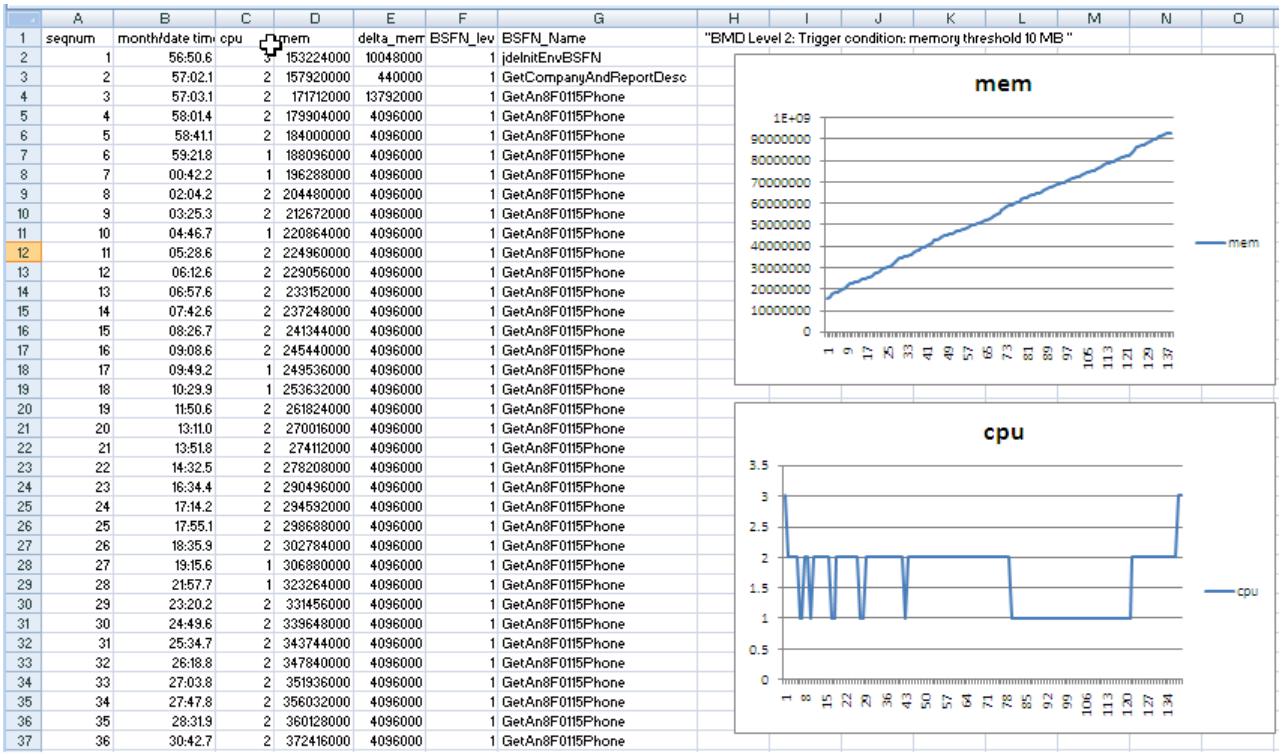
Figure 14–28 BMD Parsing Option Links

For CallObject kernels, the corresponding jdedebug_pid.log file contains the BMD information.

If a UBE is still running, the corresponding jdedebug_pid.log file contains the BMD information. If the UBE has completed, the log file is moved to the /printqueue folder. Copy the UBE-BMD log manually from the /printqueue folder to the /log folder so that the log file can be processed by BMD parsing within Server Manager.

The BMD data is parsed and retrieved in a CSV format. That parsed data can be saved to a file on the local machine, or it can be entered directly into a spreadsheet tool, such as Microsoft® Excel or OpenOffice Calc. Within Microsoft® Excel, the BMD data is easily searched and graphed.

Figure 14–29 BMD data parsed and retrieved in a CSV format



	A	B	C	D	E	F	G	H
1	seqnum	month/date tim	cpu	mem	delta_mem	BSFN_lev	BSFN_Name	"BMD Level 2: Trigger condition: memory threshold 10 MB "
2	1	56:50.6	3	153224000	10048000	1	jdeInitEnvBSFN	
3	2	57:02.1	2	157920000	440000	1	GetCompanyAndReportDesc	
4	3	57:03.1	2	171712000	13792000	1	GetAn8F0115Phone	
5	4	58:01.4	2	179904000	4096000	1	GetAn8F0115Phone	
6	5	58:41.1	2	184000000	4096000	1	GetAn8F0115Phone	
7	6	59:21.8	1	188096000	4096000	1	GetAn8F0115Phone	
8	7	00:42.2	1	196288000	4096000	1	GetAn8F0115Phone	
9	8	02:04.2	2	204480000	4096000	1	GetAn8F0115Phone	
10	9	03:25.3	2	212672000	4096000	1	GetAn8F0115Phone	
11	10	04:46.7	1	220864000	4096000	1	GetAn8F0115Phone	
12	11	05:28.6	2	224960000	4096000	1	GetAn8F0115Phone	
13	12	06:12.6	2	229056000	4096000	1	GetAn8F0115Phone	
14	13	06:57.6	2	233152000	4096000	1	GetAn8F0115Phone	
15	14	07:42.6	2	237248000	4096000	1	GetAn8F0115Phone	
16	15	08:26.7	2	241344000	4096000	1	GetAn8F0115Phone	
17	16	09:08.6	2	245440000	4096000	1	GetAn8F0115Phone	
18	17	09:49.2	1	249536000	4096000	1	GetAn8F0115Phone	
19	18	10:29.9	1	253632000	4096000	1	GetAn8F0115Phone	
20	19	11:50.6	2	261824000	4096000	1	GetAn8F0115Phone	
21	20	13:11.0	2	270016000	4096000	1	GetAn8F0115Phone	
22	21	13:51.8	2	274112000	4096000	1	GetAn8F0115Phone	
23	22	14:32.5	2	278208000	4096000	1	GetAn8F0115Phone	
24	23	16:34.4	2	290496000	4096000	1	GetAn8F0115Phone	
25	24	17:14.2	2	294592000	4096000	1	GetAn8F0115Phone	
26	25	17:55.1	2	298688000	4096000	1	GetAn8F0115Phone	
27	26	18:35.9	2	302784000	4096000	1	GetAn8F0115Phone	
28	27	19:15.6	1	306880000	4096000	1	GetAn8F0115Phone	
29	28	21:57.7	1	323264000	4096000	1	GetAn8F0115Phone	
30	29	23:20.2	2	331456000	4096000	1	GetAn8F0115Phone	
31	30	24:49.6	2	339648000	4096000	1	GetAn8F0115Phone	
32	31	25:34.7	2	343744000	4096000	1	GetAn8F0115Phone	
33	32	26:18.8	2	347840000	4096000	1	GetAn8F0115Phone	
34	33	27:03.8	2	351936000	4096000	1	GetAn8F0115Phone	
35	34	27:47.8	2	356032000	4096000	1	GetAn8F0115Phone	
36	35	28:31.9	2	360128000	4096000	1	GetAn8F0115Phone	
37	36	30:42.7	2	372416000	4096000	1	GetAn8F0115Phone	

14.8 Troubleshooting the IBM i Enterprise Server

This section provides an overview of IBM i enterprises server troubleshooting and discusses how to:

- Troubleshoot IBM i enterprise server installation.
- Troubleshoot multiple release setup.
- Troubleshoot JDBNET.
- Troubleshoot interprocess communications.
- Troubleshoot jde.ini file.

14.8.1 Understanding IBM i Enterprise Server Troubleshooting

This subsection explains how to troubleshoot problems that can occur on an IBM i enterprise server. When troubleshooting, follow these guidelines:

- Try to narrow the definition of any problem that you have, particularly when communicating the issue to someone, such as JD Edwards Worldwide Customer Support Services.

  For example, rather than reporting that the batch application failed, explain how the batch application failed. The more specific the information, the faster the

problem can be solved. Rather than reporting that "The report had the wrong data," say that "The batch status is E."

- When communicating an error message to someone, be sure to include all parts of the error message exactly as they appear in the log file or on the screen.

  Parts of the message that might not seem important might actually hold the key as to why an error occurs. Also, distinguish between characters that might be misinterpreted, for example, the capital letter "O" and the numeral zero.

- As soon as you notice an error, examine the log files.

  Messages near the end of the log files sometimes reveal the most important information about the cause of the error.

- Before you restart JD Edwards EnterpriseOne on the server, either delete or move all the files from the log directories. Refer to the JDE.INI file for the locations of the log files.

- When you first try to get JD Edwards EnterpriseOne running, verify that you have logging turned on. Examine the jde.log and jdedebug.log files carefully.

- Carefully examine the JOBLOGs and jde.log files of the JD Edwards EnterpriseOne jobs to ensure that authorities and OCM are set correctly. Look for messages such as these in the jde.log files:

```
JDB3100011 - Failed to get location of table F983051 for environment PD900
Look for messages similar to these in the JOBLOGs:
File F98306 not found in library PRODDTA.
```

You might want to temporarily modify the job description of the JD Edwards EnterpriseOne user profile to always write the joblog until you are satisfied that all setup is correct.

Note: To complete the resolutions provided for this issues, you must sign on to the enterprise server using an account that has administrative privileges.

14.8.2 Troubleshooting IBM i Enterprise Server Installation

This section explains topics that might create issues during the installation of an IBM i enterprise server.

14.8.2.1 Troubleshooting: Library Installation Verification

Issue	Resolution
You want to verify that the correct libraries and data dictionary items are installed on the IBM i.	See the list of libraries and data dictionary items and descriptions of their contents.

14.8.2.2 Troubleshooting: Database Table Configuration

Issue	Resolution
Strange database results or errors imply that Object Configuration Manger (OCM) is not set up correctly. For example, you see these message in the jde.log file: Databases: IBM i:table configuration problems IBM i JDB3300011 - Failed to get location of table F983051 for environment PD900	■ Verify that environments set up in the OCM are correct. ■ Review the description of how OCM is used by JD Edwards EnterpriseOne in JD Edwards EnterpriseOne Initialization. ■ Run the VerifyOCM program to ensure that the OCM tables are set up correctly. You must have one valid environment available to run VerifyOCM.

14.8.2.3 Troubleshooting: Setting up the IBM i .INI File

Issue	Resolution
You cannot find the .INI file	Find it in IFS. The file should be located in the /<release>/ ini directory. For example, /E900sys/ini/JDE.INI.
You need more information on using the IBM i .INI file	Review the notes and descriptions of .INI settings.

14.8.2.4 Troubleshooting: You Cannot Find the Log Files

The logging is performed to the IBM i Integrated File System (IFS). The naming convention is similar to that of the UNIX enterprise servers. That is, the default names of the files are JDE_AS400JobNumber.log, JDEDEBUG_AS400JobNumber.log, and JDETS.LOG, where AS400JobNumber is the IBM i Job Number of the job that generated the file. The system creates these files automatically, but the path to the files must exist before logging begins. The created log file directory should match the JOBLOG and JDELOG settings in the JDE.INI file.

The path to the log files stored in the IFS can be created by performing successive calls to the IBM i command MKDIR. For example, to create the path /PSFT900/LogFiles, enter this command:

MKDIR DIR('/PSFT900') DTAAUT(*RWX) OBJAUT(*ALL)

Followed by:

MKDIR DIR('/PSFT900/LogFiles') DTAAUT(*RWX) OBJAUT(*ALL)

Logging might be turned off in the .INI. Activate logging in the .INI using these settings in the [DEBUG] section:

[DEBUG]

LogErrors=1

Output=FILE

Where variable names and descriptions are as follows:

LogErrors values are:

0 = Do not generate logs.

1 = Create logs.

Output values (always in upper case) are:

NONE = Do not write debug messages to any output device.

FILE = Write messages to log files.

14.8.2.5 Troubleshooting: Not Enough Relevant Information Is Written to the Log Files

Additional logging information may need to be turned on in the .INI. Set these keys in the .INI for additional information to be output to the log files:

[JDENET]

netTrace=1

[JDEIPC]

ipcTrace=1

[DEBUG]

TAMTraceLevel=1

[UBE]

UBEDebugLevel=6

[TCEngine]

TraceLevel=10

Where variable names and descriptions are as follows:

netTrace values are:

0 = Do not generate JDENet error messages (that is, communication between platforms).

1 = Generate JDENet error messages.

ipcTrace values are:

0 = Do not generate Interprocess Communication (IPC) error messages (that is, communication between processes on a single platform).

1 = Generate IPC error messages.

TAMTraceLevel values are:

0 = Do not generate Table Access Management (TAM) error messages (that is, regarding specification files).

1 = Generate TAM error messages.

UBEDebugLevel values are:

0 = Do not generate batch application error messages.

1-6 = Generate increasingly detailed error messages (1 indicates the least specific message; 6 indicates the most detailed message).

TraceLevel values are:

0 = Do not generate Table Conversion (TC) error messages.

1-10 = Generate increasingly detailed error messages (1 indicates the least specific message; 10 indicates the most detailed message).

Note: Because NetTrace and ipcTrace messages are written to the debug log associated with that job, the [DEBUG] section of the jde.ini file requires the Output=FILE setting.

14.8.2.6 Troubleshooting: Testing with PORTTEST

In general, activate logging when running the PORTTEST program. Review the jde.log and jdedebug members generated by running the PORTTEST program. Also review the IBM i job log generated by running the PORTTEST program. These logs provide valuable information about the JD Edwards EnterpriseOne IBM i configuration and setup.

Issue	Resolution
An error with OCM occurred.	Verify that the OCM is correct for the environment. Disable the security server in the JDE.ini file and make sure that porttest runs successfully. For this work, you must log on with a User ID that has administrative privileges.
An error with the security server occurred.	The JD Edwards EnterpriseOne network might not be running. Clear the Interprocess Communication (IPC) structures using the JD Edwards EnterpriseOne IBM i CLRIPC command and restart JD Edwards EnterpriseOne. If you have different versions of JD Edwards EnterpriseOne running, make sure that they are on different ports and have different values for startIPCKeyValue. In the [JDEIPC] section of the JDE.INI file. Also, note that the different versions of JD Edwards EnterpriseOne should have different JD Edwards EnterpriseOne libraries, database files, and IFS directories Successful running of CLRIPC should result in the appearance of no messages on the screen. If messages appear as a result of CLRIPC, one or more jobs (including an interactive job that ran the PORTTEST program) might have locked some of the IPC shared memory. Determine which job locked shared memory and end it. Try logging off of a session in which you ran the PORTTEST program and running CLRIPC. If all attempts fail, you may change the .INI setting [JDEIPC] startIPCKeyValue to at least 1000 more or less than the current setting. Log off and back on again to ensure the new value is read. Attempt CLRIPC again, and restart JD Edwards EnterpriseOne if CLRIPC is successful.
An error with the security server occurred.	The JD Edwards EnterpriseOne network might be running as a service under one library list and you are trying to run the PORTTEST program under another library list. Display all the libraries in the current library list and correct the list if the displayed library list is wrong. Then run the PORTTEST program. If the library list is correct, the problem could be because the activation group under which the job is running on the IBM i may retain some of the information from previous attempts. Log off, log on, and run the PORTTEST program again.
An error with the security server occurred.	The supplied user name or password might not match any names or passwords in the JD Edwards EnterpriseOne security table. Try one of these: ■ Run the PORTTEST program with a valid user name and password. ■ Add the given user name and password to the JD Edwards EnterpriseOne security table.

Issue	Resolution
You get these message on the screen: Invalid parms PORTTEST <USER> <PWD> <ENV>	You might not have included the correct number of arguments in the PORTTEST program. Use these arguments: User - A valid JD Edwards EnterpriseOne user ID. Password - Password for the JD Edwards EnterpriseOne user ID. Environment - A valid JD Edwards EnterpriseOne environment.
Fewer than 99 F0902 records are written to the screen by PORTTEST.	A possible PORTTEST program failure. Examine the log files. ■ Fewer than 99 records might exist in the F0902 table. This is not an error, but you should review the log files for any errors. ■ The F0902 database table might not be accessible. Verify that you can query the F0902 table using SQL. Use the STRSQL command on the IBM i.
An error initializing the environment occurs in the log file.	The environment might not be set up correctly. Any errors in the affected .INI keys or database tables could cause the JD Edwards EnterpriseOne initialization to fail. The environment that the PORTTEST program uses is passed as a command line argument.

14.8.2.7 Troubleshooting: Running JDENET

Issue	Resolution
NETWORK dies immediately.	IPCs might not have been cleared before starting JD Edwards EnterpriseOne (that is, starting JDENET using the JD Edwards EnterpriseOne IBM i command STRNET). End JD Edwards EnterpriseOne (ENDNET). Clear IPCs (using the CLRIPC command) and restart JD Edwards EnterpriseOne. The startIPCKeyValue in the .INI file could be used by another version of JD Edwards EnterpriseOne. Try one of these: ■ Change the startIPCKeyValue and restart the software. This problem is not easily evident by examining the log files or reviewing error messages. Symptoms of the problem include: ■ You attempt to run more than one version of JD Edwards EnterpriseOne on the IBM i. ■ One environment can be successfully started by itself. A second environment cannot be successfully started (that is, the JDENET_N job ends almost immediately after starting) for the second version. ■ Look in the jde_xxx and jdedebug_xxx log files for specific error messages. ■ Determine if the PORTTEST program runs correctly. If not, correct those problems, and then try restarting JD Edwards EnterpriseOne using STRNET. ■ The configuration for the local host name, local domain name, and IP address might be incorrect. In the command line, enter CFGTCP to access the Configure TCP/IP form. Select option 12 (Change local domain and host names) and verify the settings for the local domain name and the local host name (for example, YOURCOMPANY.COM and SRVR1 respectively). Then select option 10 (Work with TCP/IP host table entries) and verify that two names exist in connection with the IP address for the IBM i. One name is a combination of the local host name and the local domain name (for example, SRVR1.YOURCOMPANY.COM). The other name is just the local host name (for example, SRVR1). ■ Verify that the Relational Database Directory name is set up correctly. Use the WRKRDBDIRE command to verify that the name of the *LOCAL database is the same as the server. If they are different, refer to the IBM i Configuration guide to determine how to set this up correctly.
An error initializing the environment occurs in the log file.	■ Examine the issues in this section about the PORTTEST program. ■ Determine if the PORTTEST program runs correctly. If not, correct those problems, and then try restarting JD Edwards EnterpriseOne using STRNET.

14.8.2.8 Troubleshooting: Testing JD Edwards EnterpriseOne by Submitting a Report

Issue	Resolution
You get these message: Communication Failure with <server name>	You might see a message referencing an error of 11, indicating a time out occurred because the server was started after the client was run. Try resubmitting the report. A time out might have occurred because of heavy network traffic or server load. Increase the time out value for the JDENETTimeout setting in the [NETWORK QUEUE SETTINGS] section of the jde.ini file on the workstation. The wrong communications port might have been used. Verify that the serviceNameListen value in the [JDENET] section of the jde.ini file on the workstation matches the serviceNameConnect value in the [JDENET] section of the jde.ini on the server. In addition, the serviceNameConnect value in the client jde.ini must match the serviceNameListen in the jde.ini on the server. Other communications problems might exist. Run SERVERADMINISTRATIONWORKBENCH.exe (found in the system\bin32 directory on the workstation). This program displays only the machines on the specified port (also known as service) that are running JD Edwards EnterpriseOne (either client or server). Use this information to track down the problem: ■ If the remote machine is visible, a time-out probably occurred. Rerun the report. ■ If the remote machine is not visible, try to ping the remote machine using the name of the machine. ■ If the ping fails, try to ping the remote machine using its IP address. ■ With this information, determine if the client and server agree on the IP address for the server. ■ If none of these steps identify the problem, a general network error probably occurred (for example, the network is down or a machine is disconnected). Track it down.
The report does not display any data.	No data might exist in the database for the report that you are running, or you do not have access to the data. Try these: ■ Select a different report to verify that some reports do produce data. ■ Verify the database contains data that should be included in the report. Add data if necessary. ■ Change the processing options for the report. ■ Change the OCM and data sources to reference the correct library. ■ If the report is launched on the server, make sure that the vertical tables in the server OCM match those of the OCM for the workstation. If no data is found, it could be because: ■ No data exists. ■ The processing options are incorrect. ■ The OCM for either the client or server is pointing to the wrong data source. ■ The data sources for either the client or server are pointing to the wrong database. ■ The SQL statement is incorrect (possibly due to a program bug). ■ The database drivers are out of date.

Issue	Resolution
The report does not display any data.	An error might have occurred with the report. Review the jdedebug.log and jde.log files for errors.
An error initializing the environment occurs in the log file.	The environment might not be set up correctly: ■ Look for errors in .INI keys or database tables that can cause an initialization failure. ■ Stop JD Edwards EnterpriseOne and determine if the PORTTEST program runs correctly. If not, correct the problems, and then rerun JD Edwards EnterpriseOne manually.
You get this message: Communication Failure with <server name> This error occurs sometimes on the workstation Restarting JDENET_N sometimes gets rid of the error You can ping the server from the workstation	The server might have two network cards, which can confuse JDENET when the net communications are initialized between the client and server. One machine tries to connect using one network card, and the other machine connects using the other network card. The hosts file on the server should list two different IP addresses for the server: one for each network card. The solution for the error involves setting the NetHostName field in the [JDENET] section of the JDE.INI to one of the names for the server given in the hosts file. JDENET then uses the IP address associated with the given network card.

14.8.2.9 Troubleshooting: Shutting Down JDENET

Running the IBM i command CLRIPC immediately after shutdown (that is, after running the IBM i command ENDNET) each time you shut down will help you avoid most restart problems.

14.8.2.10 Troubleshooting: Email and PPAT

Issue	Resolution
The batch application, server package installation, or table conversion log file (in the PrintQueue directory) displays the message: PPAT:troubleshooting IBM i XE Email:troubleshooting: IBM i DoSendMessage Error: User 5600427 does not exist in the address book file (F0101).	The particular user may not be found in the F0101 table. Add the user to the F0101 table.

See Also:

- Understanding the IBM i Library Structure for JD Edwards EnterpriseOne.
- Troubleshooting the JDE.INI File.

14.8.3 Troubleshooting Multiple Release Setup

This table explains how to troubleshoot problems that can occur with multiple releases on the IBM i:

Issue	Resolution
When you try to run multiple releases of JD Edwards EnterpriseOne at the same time, conflicts seem to occur between each release.	Each installed release of JD Edwards EnterpriseOne may not have its own unique set of keys in the .INI. Change these keys in one or both .INI files: [JDEIPC] startIPCKeyValue [JDENET] serviceNameListen serviceNameConnect Variable names and descriptions: startIPCKeyValue An integer value that indicates an arbitrary starting memory offset for interprocess communications. For multiple instances of JD Edwards EnterpriseOne server, be sure that the differences between these values are 1000 or more. The default value is 5000. **Note:** IBM Opti-Connect and Opti-Mover products use the IPC shared memory address 9999. Avoid setting the jde.ini file setting IPCStartKey to a starting value using the range of 9000 to 9999. serviceNameListen Port through which JDENet listens for communications attempts. The default is jde_server (translated using the services file). Each instance of the JD Edwards EnterpriseOne server needs to communicate with JD Edwards EnterpriseOne clients through different ports. serviceNameConnect Port through which JDENet tries to initialize connections with other platforms. The default is jde_server (which is translated using the services file). Each instance of JD Edwards EnterpriseOne server needs to communicate with JD Edwards EnterpriseOne clients through different ports. Also, verify that each version of JD Edwards EnterpriseOne has a unique set of libraries and database files. This is done using the ApplicationPathAddendum setting in the JDE.INI file.

14.8.4 Troubleshooting JDBNET

This table explains how to troubleshoot problems that can occur with JDBNET:

Issue	Resolution
You do not know how JDBNET is used.	JDBNET processes database requests using a client and server. It can also be configured to process server-to-server requests. This is, one server functions as a JDBNET client and the other as a JDBNET server. JDBNET eliminates the need for database-specific network software. All database requests are transported to the JDBNET server, processed in a local database, and the results are transported back to the JDBNET client.
You get an error that the data source on the JDBNET server is not found.	The correct data source on the JDBNET server may not exist. Create a data source on the server that will be used by JDBNET. This is a normal configuration for a server data source that can be accessed by JDENet running on that server. Note the data source name (OMDATP) that will be used for the JDBNET client configuration.

Issue	Resolution
You get an error that the data source on the JDBNET client is not found.	The correct data source on the JDBNET client may not exist. Use the P98611 application to create a JDBNET data source in the F98611 table using this information: ■ Data source name (OMDATP field) is used to access tables as specified in the F986101 table. ■ Server name (OMSRVR field) identifies the JDBNET server. ■ Database name (OMDATB field) matches exactly the data source name (that is, the OMDATP field) to be used by the JDBNET server. ■ All other columns must match the values in the corresponding columns of the server data source. Set this data source as an active override in the F986101 table for all tables that will be accessed through JDBNET.
JDBNET does not transfer any data	The network may not be running. End JD Edwards EnterpriseOne, clear IPC (using the IBM i CLRIPC command), and restart JD Edwards EnterpriseOne.
JDBNET does not transfer any data	The JDBNET server and client may not be using the same server port number. Modify the serviceNameListen and serviceNameConnect fields in the [JDENET] section of both the JDBNET jde.ini files on the server and on the workstation. These values must match on both the JDBNET server and JDBNET client.

14.8.5 Troubleshooting Interprocess Communications

This subsection explains how to troubleshoot problems that can occur with Interprocess Communication (IPC).

Issue	Resolution
JD Edwards EnterpriseOne jobs cannot communicate with one another with these symptoms: ■ The PORTTEST program fails. ■ The security server on the IBM i fails. ■ UBE submissions fail. If you activated ipcTrace in the [JDEIPC] section of the server jde.ini file, an error similar to this should appear in the JDEDEBUG.log: IPC2100017 createIPC Msgq (name Port6005) failed, errno=3484: A damaged object was encountered	This could be because the IBM i release is pre-V4R2. In these releases, damaged IPC message queues might result when you end JD Edwards EnterpriseOne jobs using the command ENDJOB* IMMED. ■ Use the *CNTRLD option to end an IBM i job. **Note:** You might still have damaged IPC message queues if the IBM i-controlled ending times out. ■ Run these program to verify whether a damaged message queue exists. You must have V4R1 PTF# SF45946. CALL QPOZIPCS PARM('-aqE') This program generates a spool file called IPCS that contains information about message queues on the system. Look for these output: KEY MODE 0x00000000 -------- 0x00000000 --RW------- 0x00000000 --RW------- 0x00000000 --RW-------- 0x00001234 D-RW----RW- In this example, the message queue 0x00001234 is damaged. To fix, stop, and restart JDENET using these commands: ENDNET CLRIPC STRNET Also, if the ipcTrace setting in the [JDEIPC] section of the jde.ini file on the server is not set, activate the setting and run the PORTTEST program to determine whether any message queues are damaged. Look for the word damage in the JDEDEBUG.log file. **Note:** Some of the message queues might be damaged even if the JDEDEBUG.log file does not indicate that any damage exists.

14.8.6 Troubleshooting the JDE.INI File

This section explains how to troubleshoot problems that can occur with the JDE.INI file. These notes apply to the .INI file in the E900SYS library:

- It is composed of several sections.

  The section names are enclosed in square brackets, for example [JDENET].

- Within each section are one or more keys or settings.

  The key name is on the left side of the equals sign, and the value of the key is on the right side.

- Do not include spaces in the names or values of the keys unless you know that a space is required.

  Do not include spaces immediately before or after the equals sign.

- Keys may be commented out by adding a semicolon (;) at the start of the key name.

- We recommend that you place any incidental comments on a separate line adjacent to the key to which the comment applies.

  Be sure to include a preceding semicolon. Comments can be included at the end of the key's values, but these comments can be wrongly interpreted if they are not separated from the keys' values by enough white space. Because the amount of white space needed between the keys' values and the comments is not strictly defined, we recommend that you do not place comments after the values of the keys.
- The section and key names are not case sensitive.
- Many key values are case sensitive.
- Although all of the following values may be used to turn a feature on, they may not be interchangeable as values in the .INI. Use a value that is comparable to the default value provided in the original .INI. Also, many values are case sensitive. If you have any questions about values, contact JD Edwards Global Support Center.
  - YES
  - ON
  - TRUE
  - 1

Likewise, these values may be used to turn a feature off. They are not necessarily interchangeable as values in the .INI.

- NO
- OFF
- FALSE
- NONE
- 0

If you are told by JD Edwards Worldwide Customer Support Services to modify a key that does not exist, you can add the key. Just be sure that it is in the correct section.

14.9 Troubleshooting the UNIX/Linux Enterprise Server

This section provides an overview for UNIX/Linux enterprise server troubleshooting and discusses how to:

- Troubleshoot the jde.ini file.
- Troubleshoot JD Edwards EnterpriseOne file copying to a server.
- Troubleshoot database table configuration.
- Troubleshoot printer setup.
- Troubleshoot email.
- Troubleshoot multiple release setup.
- Troubleshooting report file output location.
- Troubleshoot JDBNET server not found.
- Troubleshoot JD Edwards EnterpriseOne testing.

14.9.1 Understanding UNIX/Linux Enterprise Server Troubleshooting

This section discusses some typical problems that you might encounter and their solutions. When troubleshooting, follow these guidelines:

- Check the logs.

  Many times, the logs point to the problem. As soon as you notice an error, examine the log files. Messages near the end of the log files will probably reveal the most important information about the cause of the error.

- Try to narrow down the definition of any problem that you may have, particularly when communicating the issue to someone, such as JD Edwards Worldwide Customer Support Services.

  For example, rather than reporting that the batch application failed, explain how the batch application failed. The more specific the information, the faster the problem can be solved. For example, rather than reporting that "The report had the wrong data," say that "The batch status is E."

- When communicating an error message to someone, include all parts of the error message exactly as they appear in the log file or on the screen.

  Parts of the message that may not seem important may actually hold the key as to why an error occurred. Also, distinguish between characters that might be misinterpreted, such as the capital letter O and the numeral zero (0).

- Before you restart JD Edwards EnterpriseOne on the server, either delete or move the jde_xxx.log and jdedebug_xxx.log files (where xxx is a number).

  Do not rename the log files because it is easier to work with logs that use the standard naming convention (jde_xxx.log and jdedebug_xxx.log). If you need to save the log files until the problem is solved, then create a temporary directory and move the files.

- Clear the log directory regularly to avoid filling the file system.

  If the file system fills up, then the specification files can become corrupted.

- Always keep a backup of the specification files handy in case they become corrupted.

  Specification files should be backed up regularly for easy recovery of specification installs. If spec files have to be replaced, all specification installations will be lost if backups are not kept.

- To find problems that occur due to server failure, go to the system/bin32 directory:

  grep -n failed *log* > problems.txt

  The file problems.txt will contain a list of errors with the file and line number.

- Remember that UNIX is case-sensitive: jde.ini is not the same file as JDE.INI.

Important: To complete the resolutions provided for this issues, you must sign on to the UNIX enterprise server using an account that has administrative privileges.

14.9.2 Troubleshooting the JDE.INI File

To locate the JDE.INI file, search in the system/bin32 subdirectory. For example, /u01/JDEdwards/E900/ini/JDE.INI. These notes apply to the JDE.INI:

- It is composed of several sections.

  The section names are enclosed in square brackets; for example, [JDENET].

- The environment variable $JDE_BASE should contain the location of the JDE.INI file.

- If you copy the JDE.INI file to other directories (for example, the $SYSTEM/bin32 directory), the JD Edwards EnterpriseOne programs could read the wrong JDE.INI file.

  This error occurs because some programs might look for the JDE.INI file in the current directory before looking at the JDE_BASE environment variable.

- Each section contains one or more keys.

  The key name is on the left side of the equal sign, and the value of the key is on the right side.

- Do not include spaces in the key names or key values unless you know that a space is required.

  Do not include spaces immediately before or after the equal sign.

- Keys can be commented out by adding a semicolon (;) at the start of the key name.

- We recommend that you place incidental comments on a separate line adjacent to the key to which the comment applies.

  Be sure to include a preceding semicolon. Comments can be included at the end of the key value, but these comments can be incorrectly interpreted if they are not separated from the values of the keys by sufficient white space. Because the amount of white space between the values of the keys and the comments is not strictly defined, we recommend that you do not place comments after the values of the keys.

- Section and key names are not case sensitive.

- Many key values are case sensitive.

- Although all of the following values activate a feature, they may not be interchangeable as values in the JDE.INI.

  Use a value that is comparable to the default value provided in the original JDE.INI. Also, many of these values are case sensitive. If you have any questions about values, contact JD Edwards Worldwide Customer Support Services.

  - YES
  - ON
  - TRUE
  - 1

These values turn a feature off. They are not necessarily interchangeable as values in the JDE.INI.

- NO
- OFF
- FALSE
- NONE
- 0

If you are told by JD Edwards Worldwide Customer Support Services to modify a key that does not exist, you can add the key. Ensure that the key is in the correct section and entered with the correct spelling and case.

14.9.3 Troubleshooting JD Edwards EnterpriseOne File Copying to a Server

If you cannot copy files from the deployment server to the temporary directory on the enterprise server, this could be because ftp cannot connect. See the system administrator.

14.9.4 Troubleshooting Database Table Configurations

If results or errors occur that imply that OCM is not set up correctly, review the description in this guide of how OCM is used by JD Edwards EnterpriseOne.

14.9.5 Troubleshooting Printer Setup

If reports do not print from a server, verify the name of the default printer. Send a simple text file to the default printer using the lp command. If you get an error similar to these, then the printer is not configured on the server or is not online:

```
"lp: destination aPrinter non-existent"
```

Contact the system administrator for assistance.

For Linux, do not set up a print queue that translates files to postscript. The Linux print queues that are used by JD Edwards EnterpriseOne should generally be "raw" print queues that simply redirect the output of the file to the printer.

14.9.6 Troubleshooting Email

If the report, server package installation, or table conversion log file (in the PrintQueue directory) displays the message DoSendMessage Error: User 5600427 does not exist in the address book file (F0101), the particular user might not be found in the F0101 table. Add the user to the F0101 table.

14.9.7 Troubleshooting Multiple Release Setup

Each installed release of JD Edwards EnterpriseOne has its own JDE.INI in its ini directory. Point the user entries in the JDE.INI files to the directories of the log and other files. If the log files do not go to separate directories, change the appropriate keys in one or both JDE.INI files to point to unique directories for each installed instance of JD Edwards EnterpriseOne.

14.9.8 Troubleshooting Report File Output Location

If you cannot find the report output files, consider this information:

- The location is specified as the OutputDirectory key of the [NETWORK QUEUE SETTINGS] section in the JDE.INI on the server.

  If this key is not found, the location is the PrintQueue subdirectory of the JD Edwards EnterpriseOne base directory (for example, /u01/JDEdwards/E900SYS/PrintQueue).

- The JDE.INI file on the workstation may have the SaveOutput key of the [NETWORK QUEUE SETTINGS] section set to FALSE.

This is because a problem after the report has been printed. After the report is printed, then the record will be deleted, as will the .PDF file. Change the value of the SaveOutput key of the [NETWORK QUEUE SETTINGS] section in the JDE.INI on the workstation to TRUE.

14.9.9 Troubleshooting JDBNET Server Not Found

If you get an error that the data source on the JDBNET server is not found, the correct data source on the JDBNET server might not exist. Create a data source on the server that will be used by JDBNET. This is a normal configuration for a server data source that can be accessed by JDENet running on that server. Note the data source name (OMDATP) that will be used for the JDBNET client configuration.

If you get an error that the data source on the JDBNET client is not found, the correct data source on the JDBNET client might not exist. Create a JDBNET data source in the F98611 table using this information:

- Data source name (OMDATP field).

  Used to access tables as specified in the F986101 table.

- Server name (OMSRVR field).

  Identifies the JDBNET server.

- Database name (OMDATB field).

  Matches exactly the data source name (that is, the OMDATP field) to be used by the JDBNET server.

- Shared library name (OMDLLNAME field).

  Identifies the JDBNET client .DLL. (libjdbnet.sl on HP-UX, libjdbnet.so on AIX).

- All other columns must match the values in the corresponding columns of the server data source.

Set this data source as an active override in the F986101 table for all tables that will be accessed through JDBNET.

14.9.10 Troubleshooting JD Edwards EnterpriseOne Testing

If the PORTTEST program does not run successfully after startup:

- If you have Oracle or UDB running on the enterprise server and the database and JD Edwards EnterpriseOne services are set to start automatically at system startup, JD Edwards EnterpriseOne services may start before the database is running completely.

  You must ensure that the database software is running before starting any JD Edwards EnterpriseOne processes.

- If JD Edwards EnterpriseOne loses the connection to the database because either the network or database went down, you should see some sort of network or database error in the log files.

- Stop the JD Edwards EnterpriseOne services, clear the logs, and then restart the JD Edwards EnterpriseOne services to see if the problem is resolved.

14.10 Troubleshooting the Microsoft Windows Enterprise Server

This section provides an overview of Microsoft Windows enterprise server troubleshooting and discusses how to:

- Troubleshoot JD Edwards EnterpriseOne account setup.
- Troubleshoot JD Edwards EnterpriseOne file copying to a server.
- Troubleshoot database table configuration.
- Troubleshoot printer setup.
- Troubleshoot jde.ini file setup.
- Troubleshoot finding the log files.
- Troubleshoot testing with the PORTTEST program.
- Troubleshoot running JD Edwards EnterpriseOne manually.
- Troubleshoot finding report files.
- Troubleshoot testing JD Edwards EnterpriseOne by submitting a report.
- Take ownership of a printer.
- Stop all JD Edwards EnterpriseOne processes.
- Stop all JD Edwards EnterpriseOne processes without rights.
- Troubleshoot email.

14.10.1 Understanding Microsoft Windows Enterprise Server Troubleshooting

This section discusses some typical problems that you might encounter and their solutions. When troubleshooting, follow these guidelines:

- Narrow the definition of any problem that you might have, particularly when communicating the issue to someone, such as JD Edwards Worldwide Customer Support Services.

  For example, rather than reporting that the batch application failed, explain how the batch application failed. The more specific the information, the faster the problem can be solved. For example, rather than reporting that "The report had the wrong data," say that "The batch status is E."

- When communicating an error message to someone, be sure to include all parts of the error message exactly as they appear in the log file or on the screen.

  Parts of the message that may not seem important may actually hold the key to why an error occurs. Also, distinguish between characters that might be misinterpreted (for example, the capital letter O and the number 0).

- As soon as you notice an error, examine the log files.

  Messages near the end of the log files sometimes reveal the most important information about the cause of the error.

- Before you restart JD Edwards EnterpriseOne on the server, either delete or move the jde_xxx.log and jdedebug_xxx.log files (where xxx is a number).

  Do not rename the log files; it is easier to work with logs that use the standard naming convention (jde_xxx.log and jdedebug_xxx.log). If you need to save the log files until the problem is solved, create a temporary directory and move the files there.

- Clear the log directory regularly to avoid filling the file system. If the file system fills up, the specification files become corrupt.
- Always keep a backup of the specification files in case they become corrupt.

  Specification files should be backed up regularly for easy recovery of spec installs. If specification files have to be replaced, all specification installations are lost unless backups are kept.

Note: To complete the resolutions provided for this issues, you must sign on to the Microsoft Windows enterprise server using an account that has administrative privileges.

14.10.2 Troubleshooting JD Edwards EnterpriseOne Account Setup

If you cannot set up any accounts in the User Manager program, the account you are logged into in Microsoft Windows may not have the privileges to modify or add accounts. Log out of Microsoft Windows and log back on under the Administrator account or an account in the Administrators group.

14.10.3 Troubleshooting JD Edwards EnterpriseOne File Copying to a Server

If you cannot copy files from the CD to the JD Edwards EnterpriseOne directory on the enterprise server, verify that the CD is in the CD-ROM drive. Another cause is that one or more of the files to be copied is currently open on the CD:

- Close any files on the CD that are open.
- Close any applications that may have files open on the CD.

If one or more of the files that will be overwritten in the target directory is open:

- Close any files in the target directory that are open.
- Close any applications that may have files open in the target directory.

If the target disk is full:

- Delete or move files from the target disk.
- Copy JD Edwards EnterpriseOne to a different disk.

14.10.4 Troubleshooting Database Table Configuration

If the OCM is not set up correctly and errors occur, run the VerifyOCM program to ensure that the OCM tables are set up correctly.

14.10.5 Troubleshooting Printer Setup

If you cannot set up a printer:

- The printer may not be attached (local printer) or the print server may not be available (network printer).

  Attach to the local printer or determine why the print server is not available.
- The printer drivers may not be installed.

  Install the correct printer drivers.

14.10.6 Troubleshooting jde.ini File Setup

If you cannot find the jde.ini file:

- Search in the system\bin32 subdirectory in the JD Edwards EnterpriseOne tree. For example, z:\JDEdwards\E900\ddp\system\bin32\ jde.ini.
- Make sure you have access rights to the system\bin32 directory by logging on to Microsoft Windows as a user who has administrative rights.

14.10.7 Troubleshooting Finding the Log Files

If you cannot find the log files:

- Log files are listed in the DebugFile and JobFile keys in the [DEBUG] section of the jde.ini.

  If there are no paths, the logs are in the system\bin32 directory. The log files are named according to these scheme:

  An underscore (_) and the process ID of the process that creates the log file are inserted before the period for example, jde_123.log or jdedebug_123.log for a process with an ID of 123.

  The log file associated with the DebugFile key contains the sequence of JD Edwards EnterpriseOne events. The default value for this key is jdedebug.log. The log file associated with the JobFile key contains error messages that occur in JD Edwards EnterpriseOne. The default value for this key is jde.log.

- When a batch application is run and the jde.ini on the workstation has [NETWORK QUEUE SETTINGS] SaveOutput=TRUE, the jde_xxx.log and jdedebug_xxx.log files for the runbatch that processed the batch application is copied to a file in the PrintQueue directory.

  The root name of the files are the same as the name of the PDF file. The extension is .jde.log and .jdedebug.log. The duplication of these log files does not occur if the batch application runbatch.exe dies before duplication.

- Verify that logging in the jde.ini is turned on using these settings in the [DEBUG] section:

```
[DEBUG]

LogErrors=1

Output=FILE

Variables and their descriptions:

LogErrors

0 = Do not generate logs.

1 = Create logs.

Output

NONE = Do not write messages to any output device.

AUX = Write messages to a console window.

FILE = Write messages to log files.
```

```
    BOTH = Write messages to log files and console window.
```

If not enough relevant information is written to the log files, this could be because additional logging information needs to be turned on in the jde.ini. Set these keys in the jde.ini for additional output to the log files:

```
[JDENET]

netTrace=1

[JDEIPC]

ipcTrace=1

[DEBUG]

TAMTraceLevel=1

[UBE]

UBEDebugLevel=6

[TCEngine]

TraceLevel=10
```

These are the variables that you use to set logging options::

- netTrace
- 0 = Do not generate JDENet error messages (that is, communication between platforms).
- 1 = Generate JDENet error message.
- ipcTrace
- 0 = Do not generate Interprocess Communication (IPC) error messages (that is, communication between processes on a single platform).
- 1 = Generate IPC error messages.
- TAMTraceLevel
- 0 = Do not generate Table Access Management (TAM) error messages (that is, regarding specification files).
- 1 = Generate TAM error messages.
- UBEDebugLevel
- 0 = Do not generate batch application error messages.
- 1 = Generate increasingly detailed error messages (1 gives the least specific messages, whereas 6 gives the most detailed messages).
- TraceLevel
- 0 = Do not generate Table Conversion (TC) error messages.
- 1-10 = Generate increasingly detailed error messages (1 gives the least detail, whereas 10 gives the most detail).

14.10.8 Troubleshooting Testing with the PORTTEST Program

If an error with the security server occurred:

- Verify the JD Edwards EnterpriseOne network is running either as a service or started from a command prompt.
- If the security server is inactive, or if it is active on a server and port that is different from the ones the PORTTEST program uses, perform one of these tasks:

  Start JD Edwards EnterpriseOne net on the server and port where the PORTTEST program is being run. The security server key in the [SECURITY] section of the jde.ini specifies the security server, and the serviceNameListen and serviceNameConnect settings in the [JDENET] section specify the ports.

  Change the name of the security server or the names of the ports, or both, in the jde.ini file to point to the correct security server.
- Make sure that the JD Edwards EnterpriseOne network and the PORTTEST program are running under the same account:

  To determine under which account PORTTEST is running, press the Control, Alt, and Delete keys at the same time. If the JD Edwards EnterpriseOne network is running as a service, determine under which account it is running. To do this, select the service in Microsoft Windows Control Panel, then go to Services and click Startup.

  For initial testing, you can stop the JD Edwards EnterpriseOne network service, open a Windows command prompt, cd to the system\bin32 directory, run jdenet_n without any parameters, and rerun the PORTTEST program. When finished, stop jdenet_n from the Microsoft Windows Task Manager.

  To run the PORTTEST program under the same account as the JD Edwards EnterpriseOne network service, log out of Windows, log into the same account under which the service is running, open a Microsoft Windows command prompt, cd to the system\bin32 directory, and rerun the PORTTEST program.
- To make sure the supplied user name and password, or both, match names and passwords, or both, in the JD Edwards EnterpriseOne security table:

  Run the PORTTEST program with a valid user name and password. Add the given user name and password to the JD Edwards EnterpriseOne security table.

If you get the message Invalid parms PORTTEST: <USER> <PWD> <ENV>, the correct number of arguments to PORTTEST may not have been included. Use these arguments:

- User - A valid JD Edwards EnterpriseOne account name.
- Password - Password for the JD Edwards EnterpriseOne account.
- Environment - A valid JD Edwards EnterpriseOne environment.
- Fewer than 99 records are written to the screen by PORTTEST.

If PORTTEST failed, examine the log files.

If fewer than 99 records exist in the F0902 table, this is not an error. You should review the log files for errors.

If the F0902 table is not accessible, verify that you can query the F0902 table using SQL.

If an error initializing the environment occurs in the log file, the environment may not have been set up correctly. See the chapter Understanding the JD Edwards EnterpriseOne Initialization for Windows in this guide for more information about

how JD Edwards EnterpriseOne programs use OCM. Any errors in the affected jde.ini keys or database tables could cause the JD Edwards EnterpriseOne initialization to fail. The environment that PORTTEST uses is passed as a command line argument.

14.10.9 Troubleshooting Running JD Edwards EnterpriseOne Manually

- If the JD Edwards EnterpriseOne network is not running, start the JD Edwards EnterpriseOne network service.
- Verify the JD Edwards EnterpriseOne network is running by doing these:
  - The JD Edwards EnterpriseOne network should either be running as a service or from a Windows command prompt.
  - If it is running as a service, determine under which account it is running.

    To do this, select the JD Edwards EnterpriseOne network service in Microsoft Windows Control Panel, select Services, and then select Startup. Note the account name. If you are using Microsoft Windows 2000, select the JD Edwards EnterpriseOne network service in the Windows Control Panel, select Services, and then select Properties.
  - If it is run from a command prompt, the network is running under the Microsoft Windows account you signed on to.

    When you log off Microsoft Windows, network processes started from a command prompt and all child processes will terminate.
- If the setup of some part of JD Edwards EnterpriseOne, such as the jde.ini file or OCM, is incorrect, determine if PORTTEST runs correctly.

  If not, correct those problems and then try running JD Edwards EnterpriseOne manually.

If an error initializing the environment occurs in the log file, the setup for some part of JD Edwards EnterpriseOne, such as the jde.ini file or OCM, may be incorrect. Examine the applicable problems in the "Testing with the PORTTEST Program" section in this chapter. Determine if the PORTTEST programs runs correctly. If not, correct those problems, and then try running JD Edwards EnterpriseOne manually.

14.10.10 Troubleshooting Finding the Report Files

If you cannot find the report output files:

- Check the OutputDirectory key of the [NETWORK QUEUE SETTINGS] section in the jde.ini file on the server.

  If there is no location, listed, the files are in the PrintQueue directory of the JD Edwards EnterpriseOne base directory. For example, z:\JDEdwards\E900\ddp\PrintQueue.
- Verify that SaveOutput in the [NETWORK QUEUE SETTINGS] section in the jde.ini file on the workstation is TRUE.

14.10.11 Troubleshooting Testing JD Edwards EnterpriseOne by Submitting a Report

- If a time-out occurred because the JD Edwards EnterpriseOne server was started after the client, resubmit the report.

- If a time-out occurred due to heavy network traffic or server load, increase the time-out value in the jde.ini file on the workstation and resubmit the report.

  Use the JDENETTime-out setting in the [NETWORK QUEUE SETTINGS] section.

- If the wrong communications port is being used, perform one of these tasks:

  – Verify that the serviceNameListen value in the [JDENET] section of the jde.ini file on the workstation matches the serviceNameConnect value in the [JDENET] section of the jde.ini file on the server.

    In addition, the serviceNameConnect value in the jde.ini file on the workstation must match serviceNameListen in the jde.ini file on the server. If the values of these keys are strings, the numeric value is retrieved from the services file in the c:\winnt\system32\drivers\etc directory (Microsoft Windows: client or server).

  – The services file contains a list of strings and their corresponding port numbers.

    If the port that you are interested in is on the last line of the services file, be sure to include a return at the end of the line or else the string will not be translated to the corresponding port number.

- If the client is using Dynamic Host Configuration Protocol (DHCP) and the server does not have an entry for itself in its hosts file in the c:\winnt\system32\drivers\etc directory, add an entry for the server in the hosts file on the server.

- These situations can occur:

  – Communications failure error message on the workstation.

  – Restarting Network Service or jdenet_n sometimes gets rid of the error.

  – You can ping the server from the workstation.

  These issues can occur because the server has two network cards, which confuses JDENET when the net communications are initialized between the client and server. One machine tries to connect using one network card, and the other machine connects using the other network card.

  The hosts file on the server should list two different IP addresses for the server--one for each network card. Resolve the error by setting the NetHostName field in the [JDENET] section of the jde.ini to one of the names for the server given in the hosts file. JDENET then uses the IP address associated with the given network card.

- For the error cannot connect to printer in the jde_xxx.log or the log file in the PrintQueue subdirectory:

  – If a general printing error occurred, try to print a text document from Notepad.

    Resolve any issues.

  – If no default printer is set up on the enterprise server, set up a printer using the task Add a new printer or Modify an existing printer in the JD Edwards EnterpriseOne Tools 8.94 Implementation Guide: System Administration.

  – If you do not have privileges to the printer, define the owner as a local or network account.

    The type of account depends on the type of printer. If the printer is a local printer, the owner could be either a local or network account but either type

must have privileges to access the printer. If the printer is a network printer, the owner must be a network account with access privileges.

- All jobs sent to this printer using the current server will conform to the selected orientation.

  Note that the report template or other programs may override this default orientation. If you cannot change the printer orientation, you may not have the right to change the orientation. Log on to Microsoft Windows in an account that has administrative rights for the printer. For a local printer, use an account that has administrative privileges. For a network printer, use an account given administrative privileges by a network administrator.

  If the report does not list any data, the data may not exist in the database for the report that you are running, or you do not have access to the data. Perform one or more of these tasks to resolve the data issue:

  – Select a different report.

  – Add data to the database.

  – Change the processing options for the report.

  – Change the OCM and data sources to point to the correct database.

- If the report is launched on the server, verify the vertical tables in the server OCM match those in the workstation OCM.

  If you believe data should have been found, edit the report jdedebug.log found in the PrintQueue subdirectory.

  Search for the SQL select statement used to retrieve data from the database. You must have some idea what data is being read to do this.

  – Copy the SQL statement.

  – Open the specific database SQL command interface - for example, SQL Plus or ISQL_w.

  – Paste the SQL statement into the SQL command interface.

  – Submit the SQL statement.

- If no data is found, one of these conditions may be true:

  – No data exists.

  – The processing options are incorrect.

  – The OCM for either the client or server is pointing to the wrong data source.

  – The data sources for either the client or server are pointing to the wrong database.

  – The SQL statement is incorrect (possibly due to a program bug).

  – The database drivers are out of date.

  – If an error occurred with the report, look in the jde_xxx.log for error messages.

- If an error initializing the environment occurs in the log file, the environment may not be set up correctly.

  Stop JD Edwards EnterpriseOne and determine if the PORTTEST program runs correctly. If not, correct those problems and then run JD Edwards EnterpriseOne manually.

14.10.12 Taking Ownership of a Printer

To take ownership of a printer:

1. From the Microsoft Windows Start menu, select Settings, Printers.
2. Right-click the desired printer.
3. Select Properties,, and then select the Privileges tab.
4. Click Ownership and Take Ownership.

   If the printer drivers are not installed, see the section Database Driver Files in this guide for information about which drivers you need.

   If the report printouts are in portrait mode but should be in landscape mode (or vice versa), verify that the orientation specified in RDA for the report is correct.

   If the default printer is set to the wrong orientation, set the orientation using these task:
5. From the Microsoft Windows Start menu, select Settings, Printers.
6. Right-click the desired printer.
7. Select Document Defaults.
8. Select the desired default orientation.
9. Click OK.

14.10.13 Stopping All JD Edwards EnterpriseOne Processes

If you need to stop the JD Edwards EnterpriseOne processes that you started from the command prompt, for example, jdenet_n, stop any of these processes that are running:

- Jdenet_n.exe
- Jdenet_k.exe
- Runbatch.exe
- ipcsrv.exe

These additional processes, such as jdenet_k and runbatch, are started by jdenet_n and queue kernel.

To stop all JD Edwards EnterpriseOne processes:

1. Run the Microsoft Windows Task Manager.
2. Select the Processes tab.
3. Select one of the running processes.
4. Click End Process.
5. Repeat for each process to be stopped.

14.10.14 Stopping JD Edwards EnterpriseOne Processes Without Rights

Use this task if you do not have the rights to stop the processes.

To stop JD Edwards EnterpriseOne processes without rights:

1. Log on to Microsoft Windows in an account that has rights to stop processes.
2. Stop processes using Visual C++.

3. Run the Microsoft Windows Task Manager.
4. Select the Processes tab.
5. Select one of the running processes.
6. Click Debug Process.

   Visual C++ starts.
7. Click the X in the upper right-hand corner to close Visual C++.

   Do not save the project workspace. This ends the runaway process.
8. Repeat these steps for each runaway process.

   If they still do not end, reboot the machine.

14.10.15 Troubleshooting Email

If the report, server package installation, or table conversion log file in the PrintQueue directory displays the message DoSendMessage Error:

```
User 5600427 does not exist in the address book file (F0101).
```

This could be because the particular user is not found in the F0101 table. Add the user to the F0101 table.

14.11 Troubleshooting Web Servers

This section provides an overview of web server troubleshooting and discusses how to:

- Troubleshoot IIS and IBM HTTP web servers.
- Troubleshoot JAS.
- Troubleshoot serialized database and generation issues.
- Troubleshoot SQL server issues.
- Troubleshoot problems using log files.

14.11.1 Understanding Web Server Troubleshooting

This section discusses some typical issues you might encounter when using WebSphere and Java Application Server (JAS). It also explains other issues you might encounter with web servers and how to track down problems by using the log files in Server Manager.

14.11.2 Troubleshooting IIS and IBM HTTP Web Servers

If you need to configure with IIS and an IBM HTTP Server, refer to the installation documentation.

If you receive the message Recursive error - page not found, you need to make sure IIS is running for a particular instance of JAS. IIS instances can be stopped easily, and the user may forget to restart them. To make sure IIS is running for the particular instance of JAS, verify IIS instance properties by selecting the appropriate instance, and then right-clicking and choosing Properties. Confirm that the correct paths are listed for the desired JAS code.

14.11.3 Troubleshooting JAS

If no logs appear, verify that the [LOGS] setting in the jas.ini has logging turned on and points the log files to reside in the desired location (for example, ;log=d:\E900\internet\jas.log or ;debuglog=d:\E900\internet\jasdebuglog). If the log file paths are not correctly stipulated, the logs may be writing to a file located elsewhere.

If JAS seems slow, check to see whether jdbcTrace is set to TRUE or FALSE. If tracing is turned on or set to TRUE, the additional logging will dramatically slow JAS performance.

14.11.4 Troubleshooting Serialized Database and Generation Issues

If you receive the message "Form is out of date...most likely needs to be regenerated," this error usually occurs because the specifications used to construct the serialized database do not match the JAS code. Ensure that the date the JAS code was written matches the date of the jdecom.dll that resides in the E900\system\bin32 directory of the generating machine.

Also be sure to register the jdecom.dll. After you run the regsvr32 jdecom.dll command, the eGenerator recognizes the jdecom.dll and uses it to fetch JD Edwards EnterpriseOne specs and convert them into Java serialized objects.

If the menu does not appear when the user signs on to JD Edwards EnterpriseOne, check for these conditions:

- [JDBC URL] section in jde.ini is set correctly or [JDBC DRIVERS] is set correctly.

  The [JDBC URL] points to the serialized database (the one you just set up).

- Bounce the WebSphere application server.

  Menus are cached, and by bouncing the server you clear the cached information.

- Ensure that the host database for serialized objects is running.

14.11.5 Troubleshooting SQL Server Issues

If SQL Server process or Oracle process consumes excess CPU in a web server environment, the serialized objects for the web server are stored in either SQL server or the Oracle database. The web server must access these tables frequently when running an application. Indexes may be missing, which can cause severe performance problems.

Ensure that all existing JD Edwards EnterpriseOne indexes are created for tables F989998 and F989999. You should have one index for F989998 for columns WBJOBID and WBOID. You should also have one index for F989999 for columns WBUID, WBOID, WBLNGPREF. If these indexes do not exist in the database, generate them using Object Librarian.

Add a new index to the F989999. This index should include columns WBOID, WBUID, and WBJVER. Generate this index over the F9899999 table.

Update statistics on both tables as follows:

- For Oracle, issue these commands in SQL *Plus:

  ```
  ANALYZE TABLE owner.F989999 COMPUTE STATISTICS
  ```

- For SQL Server, issue these commands:

  ```
  UPDATE STATISTICS owner.F989999
  ```

```
UPDATE STATISTICS owner.F989998
```

Improvements vary depending on the number of users accessing the serialized database.

14.11.6 Troubleshooting Problems Using Log Files

If you need to view logging information for the Java client, open the Java Console by choosing Java Console from the View menu in Internet Explorer. The Java Console displays all problems that the Java Virtual Machine on the client is having. Errors appear as uncaught exceptions in the console.

Note: You must have the appropriate internet options turned on to view the Java Console.

To enable the Java Console in Internet Explorer, select Tools, and then select Internet Options. In Internet Options, click the Advanced tab, scroll down to the section titled Java VM, and select these options:

- Java Console enabled.
- Java logging enabled.
- JIT compiler for virtual machine enabled.

If you need to troubleshoot errors in web applications:

- Verify that the problem is only a problem on the web.

  Test the fat client version of the same application against the same enterprise server that the web is using. Make sure that you use the same JD Edwards EnterpriseOne accounts and environments.

- Determine whether the problem happens in HTML, Java, or both.

  Since both Java and HTML use the Java runtime engine, they should behave the same. Some variation exists based on the inherent differences between the Portal, HTML page processing and Java interactive processing, but underlying functionality and processing should be the same.

- Re-create the problem on the web server.

  The logs will work in the Portal, HTML, and Java.

- Open a separate Internet Explorer browser and use it to access the Web Server Monitor for the web server being used.

- Check the Standard Error Log (stderr.log) for errors.

  A common error you might see here is BSFN Failed. If you see this error, verify that the enterprise server is up and that the BSFN is not a T1 BSFN.

  T1 refers to Type 1 business functions, which are client-only business functions. They cannot run on a server.

- Check the Standard Output Log (stdout.log) for more information.

  For example, you can view the time and date stamps from the errors found in both the Jas.log and the standard error log to find more detailed information about what was occurring at about the same time that the errors occurred.

If you need more information, enable Debug.log and set Net Trace, which you can do in the [LOGS] section of jas.ini file. Re-create the problem, view the Debug.log, and look for more information.

You can also use the Server Manager to monitor web servers.

Try to find SQL statement information. SQL statements can give you an idea of what values are being passed. Some common failures include:

- Form Interconnects are passing incorrect information.

  Verify that the fat client is working correctly. Watch especially for null, blank, and zero problems, as well as special characters.

- String is too big.

  Note carefully what you did to get this error.

- Null values are being passed.

  The SQL statement information search will result in nothing being found. Check the SQL statements and make sure that correct values were passed. Determine where the failure occurred and make a note of it.

- The application stops responding.

  Check logs for BSFN failures.

A

Using the Microsoft Visual C++ 2005 or Higher Level Compiler

This appendix contains the following topics:

Note: All references to Microsoft Visual C++ 2005, VS2005, (v.8) contained within this Appendix refer to the defined JD Edwards EnterpriseOne minimum technical requirement Microsoft Windows platform compiler, e.g. Microsoft Visual C++ 2005 SPn, Microsoft Visual C++ 2008 SPn (where n represents a service pack). Please refer to the JD Edwards EnterpriseOne minimum technical requirements to identify supported versions of the Microsoft Windows platform compiler.

A.1 Understanding Microsoft Visual C++ 2005 or Higher Level Runtime Libraries

This section discusses:

- Microsoft Visual C++ runtime libraries background.
- Redistribution of Microsoft Visual C++ 2005 or higher runtime libraries.

A.1.1 Microsoft Visual C++ Runtime Libraries Background

Historically, the Microsoft Visual C++ compiler runtime libraries have been redistributed as part of our JDEdwards OneWorld Xe and EnterpriseOne products. For past Visual C++ compiler releases (before Visual C++ 2005), the redistribution of the compiler specific runtime libraries has been quite simple. Our installer process placed the runtime libraries in a location found within the server's path, (for example, %SYSTEMROOT%\system32), to make them accessible.

Microsoft Visual C++ compiler release 2005 (v.8) or higher runtime libraries are not backward and forward compatible. When JD Edwards EnterpriseOne objects are compiled using Microsoft Visual C++ 2005 and linked into a dynamic link library (DLL), a manifest file is created for each DLL. This DLL-specific manifest identifies the runtime library version used to compile and link the objects that were built. Unlike past compilers, the runtime libraries must not only be release-specific but also version-specific. For instance, when the Microsoft Visual C++ 2005 Compiler and SPn, (where n represents a service pack) are installed on a machine, the Windows\WinSxS (side-by-side) folder is updated to include the associated compiler runtime libraries with the release level of Visual Studio 2005 SPn.

With the Visual C++ 2005 or higher compiler, manifests associated with our DLLs must now be created. Each DLL-specific manifest identifies a specific runtime library release and version.

This association requires that all Microsoft Windows machines that are part of a JD Edwards EnterpriseOne solution and that perform business function builds share a Microsoft Visual C++ 2005 compiler with identical service pack releases.

When you build new packages using the Microsoft Visual C++ 2005 SPn (or higher) compiler, you must ensure that all machines receiving these packages have the corresponding runtime libraries installed. This is important to note. Assuming Microsoft makes a new service pack available or requires an update for its Visual C++ 2005 compiler and it is installed on your JD Edwards EnterpriseOne Microsoft Windows build machines, you must do the following:

- Ensure that all JD Edwards EnterpriseOne Microsoft Windows build machines, both servers and workstations, have the identical compiler service pack release levels installed.
- Distribute the new Visual C++ 2005 SPn runtime libraries to all Microsoft Windows machines that are receiving packages built by Visual C++ 2005 SPn and do not have a compiler installed.

> **Note:** Microsoft does make the Visual C++ 2005 SPn redistributable package available from the Microsoft Download Center. From the Microsoft Download Center, search for Microsoft Visual C++ 2005 SP1 Redistributable Package (x86), Microsoft Visual C++ 2008 SP1 Redistributable Package (x86), etc.

A.1.2 Redistribution of Microsoft Visual C++ 2005 or Higher Runtime Libraries

This process applies to customers adopting Microsoft Visual C++ 2005 or a higher level compiler. All JD Edwards EnterpriseOne Microsoft Windows machines receiving application foundation packages built with Microsoft Visual C++ 2005 or higher require the runtime libraries to be installed. For example, the Microsoft Visual C++ 2005 Redistributable Package (vcredist_x86.exe) installs runtime components of Visual C++ required to run applications developed with Visual C++ on a computer that does not have Visual C++ 2005 compiler installed.

The absence of the Microsoft Visual C++ 2005 runtime libraries from a machine using a JD Edwards EnterpriseOne application foundation package built by the same compiler will result in "Business function Library load failed..." error messages. Once the Visual C++ 2005 runtime libraries are installed on a Microsoft Windows machine, only new service pack updates or Microsoft Updates to the Microsoft Visual C++ 2005 compiler require redistribution of new runtime libraries.

Note: All references to Microsoft Visual C++ 2005 within this Appendix refer to the JD Edwards EnterpriseOne defined Microsoft Windows platform compiler minimum technical requirements, for example, Microsoft Visual C++ 2005 SPn, Microsoft Visual C++ 2008 SPn. Please refer to the JD Edwards EnterpriseOne minimum technical requirements to identify supported versions of the Microsoft Windows platform compiler.

A Microsoft or third-party system management tool such as SMS can be used to distribute the Microsoft Visual C++ 2005 runtime libraries. This is generally the recommended approach for the distribution of Microsoft packaged product. The JD Edwards EnterpriseOne package build feature can also be used to push Microsoft's redistributable runtime library package to all Microsoft Windows client machines. Delivery of Microsoft Visual C++ 2005 runtime libraries for JD Edwards EnterpriseOne enterprise, logic, application, or batch servers is also explained in this appendix.

See .

A.2 Creating a VS2005 Runtime Library Package Feature

The JD Edwards EnterpriseOne package build feature makes it possible to distribute third party applications with the deployment of a client package. Create a package feature for the Microsoft Visual C++ 2005 compiler runtime libraries to leverage this facility.

The deployment server should have a copy of the Microsoft Visual C++ 2005 SPn vcredist.exe. By default the Runtime Libraries path for the 2005 compiler release is C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\BootStrapper\Packages\vcredist_x86, while the 2008 compiler release is C:\Program Files\Microsoft SDKs\Windows\v6.0A\Bootstrapper\Packages\vcredist_x86.

To create a VS2005 runtime library package feature:

1. On the deployment server, open Windows Explorer and navigate to your JD Edwards EnterpriseOne solution's shared folder. For example, E900.
2. Expand the shared node and open OneWorld Client Installs\ThirdParty.
3. Under the folder ThirdParty, create a new folder with the name **VS2005RTL.**
4. Locate and copy the vcredist_x86.exe file from your installed compiler path. For example, C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\BootStrapper\Packages\vcredist_x86.
5. Copy the vcredist_x86.exe file into the VS2005RTL folder.
6. Log into the deployment server's DEP900 environment.
7. Fast path to menu GH9083, Package and Deployment Tools, and select the Package Assembly application.
8. On the Work with Packages form, select Form and then Features.
9. On the Work with Features form, click Add.
10. Click Next to begin the Feature Based Deployment Director.
11. On the Feature Information form, complete these fields and click Next:

Field	Value
Feature	**VS2005_RTL**
Feature Type	**1**
Description	**Visual C++ 2005 Runtime Libraries**
Feature Installation Options	**Required**
Components	**Additional Install Processes**

12. On the Additional Install Processes form, select the Execute After Install option.
13. Complete these fields and click Next:

Field	Value
Third Party	**VS2005RTL**
Description	**Visual C++ 2005 Runtime Libraries**
Sequence	**1**
Executable Name	**vcredist_x86.exe** This is the name of the executable file found in the ...\ThirdParty\VS2005RTL folder.
Source Path	\\<deploymentservername>\E900\OneWorld Client Install\ThirdParty\VS2005RTL **Note:** Do not use the Select Directory browse function to obtain the aforementioned path. Instead type the UNC path or cut and paste the Universal Naming Convention (UNC) path from Windows Explorer into the Source Path field.
Parameters	**/Q** /Q denotes Quiet Mode and does not require any user intervention.

14. Click Save to preserve the feature settings and then click Next to continue.
15. On the Feature Summary form, click End to complete the feature definition.

> **Note:** If you expand the nodes describing each package feature you may inspect the feature definition. You may notice that the UNC share path has been truncated for your newly created entry. This is NOT an issue as this line entry serves only as a description. The complete UNC share path has been properly preserved in System table F96604.

A.3 Creating an Update Package with the VS2005 Runtime Library Feature

Once the VS2005 runtime library package feature has been created, it can be associated with either an update or full package. Creating an update package containing this feature will cause the full parent package assembly information to include this same feature.

To create an update package with the VS2005 runtime library feature:

1. Go to menu GH9083, Package and Deployment Tools, and select the Package Assembly application.
2. On the Work with Packages form, click Add.
3. On the Package Assembly Director, click Next to begin the package assembly process.
4. On the Package Information form, select the Express option, complete these fields, and click Next:

Field	Example Value
Package Name	**DV2005RTL**
Description	**Visual C++ 2005 RTL for DV**
Path Code	**DV900**

5. On the Package Component Revisions form, select the Update option and type or select the parent package. For example, **DV900FA.**
6. Click the Features button.
7. On the Features Components form, click the Browse button.
8. On the Feature Component Selection form, click Find.
9. Highlight the entry associated with VS2005_RTL and click Select to mark the entry with a check mark.
10. Click Close and Close again to return to the Package Component Revisions form.
11. Verify that the form shows "Individual Features Selected" and click End to complete the package assembly process.
12. On the Work with Packages form, select Row and Active/Inactive to activate the package.

A.4 Building and Deploying an Update Package with the VS2005 Runtime Library Feature

After creating the update package, you will need to build and deploy the package.

To build and deploy an update package with the VS2005 runtime library feature:

1. Highlight your package and select Row and Define Build.
2. On the Package Build Definition Director, click Next to begin the build definition process.
3. On the Package Build Revisions form, ensure that the Build Location Client check box is checked and click Next.
4. On the Build Features tab, select the Build Feature INFs option.
5. On the Package Build Revisions form, click End to complete the build definition process.
6. On the Work with Package Build Definition form, select Row and Active/Inactive to activate the package build definition.
7. Select Row and Submit Build to build the package.
8. On the Report Output Destination form, select On Screen and click OK.

9. Once the package build has completed, review the R9621 PDF report file to verify that the build completed successfully.

   The successful package build with the included package feature results in the creation of a feature-specific INF file.

10. After building the package, the appropriate person must approve it for client deployment.

    Afterwards, both the update and associated parent package will automatically include the Microsoft Visual C++ 2005 runtime libraries as part of the client installation process.

    Since this feature was configured to install in Quiet Mode (/Q), it does not require any user intervention. If the Microsoft Visual C++ 2005 runtime libraries are already installed on the machine, the feature-specific installer will exit.

A.5 Installing the VS2005 Runtime Library on an Enterprise Server

All JD Edwards EnterpriseOne Microsoft Windows machines receiving application foundation packages built by Microsoft Visual C++ 2005 require the runtime libraries to be installed.

Customers using a JDEdwards EnterpriseOne enterprise, logic, application, or batch server that does not have a Microsoft Visual C++ 2005 compiler installed, must install the associated Microsoft redistributable runtime library package (vcredist_x86.exe). Failure to install the runtime libraries while using a Microsoft Visual C++ 2005 built application foundation package on a machine without the supported compiler will result in "Business function Library load failed ..." error messages.

A Microsoft or third-party system management tool such as SMS could be used to distribute the Microsoft Visual C++ 2005 runtime libraries to these machines. This is generally the recommended approach for the distribution of Microsoft packaged products. In the absence of a system management tool, simply install the Microsoft redistributable runtime library package (vcredist_x86.exe).

To install the Microsoft redistributable runtime library package:

1. Locate the vcredist_x86.exe file on a machine with the Microsoft Visual C++ 2005 compiler installed. The default installed compiler path is C:\Program Files\Microsoft SDKs\Windows\v6.0A\Bootstrapper\Packages\vcredist_x86.
2. Copy the vcredist_x86.exe file to a folder on your local machine.
3. Double-click the vcredist_x86.exe file executable to install the Microsoft Visual C++ 2005 runtime libraries.

Glossary

Accessor Methods/Assessors

Java methods to “get” and “set” the elements of a value object or other source file.

activity rule

The criteria by which an object progresses from one given point to the next in a flow.

add mode

A condition of a form that enables users to input data.

Advanced Planning Agent (APAg)

A JD Edwards EnterpriseOne tool that can be used to extract, transform, and load enterprise data. APAg supports access to data sources in the form of rational databases, flat file format, and other data or message encoding, such as XML.

application server

Software that provides the business logic for an application program in a distributed environment. The servers can be Oracle Application Server (OAS) or WebSphere Application Server (WAS).

Auto Commit Transaction

A database connection through which all database operations are immediately written to the database.

batch processing

A process of transferring records from a third-party system to JD Edwards EnterpriseOne.

In JD Edwards EnterpriseOne Financial Management, batch processing enables you to transfer invoices and vouchers that are entered in a system other than JD Edwards EnterpriseOne to JD Edwards EnterpriseOne Accounts Receivable and JD Edwards EnterpriseOne Accounts Payable, respectively. In addition, you can transfer address book information, including customer and supplier records, to JD Edwards EnterpriseOne.

batch server

A server that is designated for running batch processing requests. A batch server typically does not contain a database nor does it run interactive applications.

batch-of-one

A transaction method that enables a client application to perform work on a client workstation, then submit the work all at once to a server application for further processing. As a batch process is running on the server, the client application can continue performing other tasks.

best practices

Non-mandatory guidelines that help the developer make better design decisions.

BPEL

Abbreviation for Business Process Execution Language, a standard web services orchestration language, which enables you to assemble discrete services into an end-to-end process flow.

BPEL PM

Abbreviation for Business Process Execution Language Process Manager, a comprehensive infrastructure for creating, deploying, and managing BPEL business processes.

Build Configuration File

Configurable settings in a text file that are used by a build program to generate ANT scripts. ANT is a software tool used for automating build processes. These scripts build published business services.

build engineer

An actor that is responsible for building, mastering, and packaging artifacts. Some build engineers are responsible for building application artifacts, and some are responsible for building foundation artifacts.

Build Program

A WIN32 executable that reads build configuration files and generates an ANT script for building published business services.

business analyst

An actor that determines if and why an EnterpriseOne business service needs to be developed.

business function

A named set of user-created, reusable business rules and logs that can be called through event rules. Business functions can run a transaction or a subset of a transaction (check inventory, issue work orders, and so on). Business functions also contain the application programming interfaces (APIs) that enable them to be called from a form, a database trigger, or a non-JD Edwards EnterpriseOne application. Business functions can be combined with other business functions, forms, event rules, and other components to make up an application. Business functions can be created through event rules or third-generation languages, such as C. Examples of business functions include Credit Check and Item Availability.

business function event rule

See named event rule (NER).

business service

EnterpriseOne business logic written in Java. A business service is a collection of one or more artifacts. Unless specified otherwise, a business service implies both a published business service and business service.

business service artifacts

Source files, descriptors, and so on that are managed for business service development and are needed for the business service build process.

business service class method

A method that accesses resources provided by the business service framework.

business service configuration files

Configuration files include, but are not limited to, interop.ini, JDBj.ini, and jdelog.properties.

business service cross reference

A key and value data pair used during orchestration. Collectively refers to both the code and the key cross reference in the WSG/XPI based system.

business service cross-reference utilities

Utility services installed in a BPEL/ESB environment that are used to access JD Edwards EnterpriseOne orchestration cross-reference data.

business service development environment

A framework needed by an integration developer to develop and manage business services.

business services development tool

Otherwise known as JDeveloper.

business service EnterpriseOne object

A collection of artifacts managed by EnterpriseOne LCM tools. Named and represented within EnterpriseOne LCM similarly to other EnterpriseOne objects like tables, views, forms, and so on.

business service framework

Parts of the business service foundation that are specifically for supporting business service development.

business service payload

An object that is passed between an enterprise server and a business services server. The business service payload contains the input to the business service when passed to the business services server. The business service payload contains the results from the business service when passed to the Enterprise Server. In the case of notifications, the return business service payload contains the acknowledgement.

business service property

Key value data pairs used to control the behavior or functionality of business services.

Business Service Property Admin Tool

An EnterpriseOne application for developers and administrators to manage business service property records.

business service property business service group

A classification for business service property at the business service level. This is generally a business service name. A business service level contains one or more business service property groups. Each business service property group may contain zero or more business service property records.

business service property key

A unique name that identifies the business service property globally in the system.

business service property utilities

A utility API used in business service development to access EnterpriseOne business service property data.

business service property value

A value for a business service property.

business service repository

A source management system, for example ClearCase, where business service artifacts and build files are stored. Or, a physical directory in network.

business services server

The physical machine where the business services are located. Business services are run on an application server instance.

business services source file or business service class

One type of business service artifact. A text file with the .java file type written to be compiled by a Java compiler.

business service value object template

The structural representation of a business service value object used in a C-business function.

Business Service Value Object Template Utility

A utility used to create a business service value object template from a business service value object.

business services server artifact

The object to be deployed to the business services server.

business view

A means for selecting specific columns from one or more JD Edwards EnterpriseOne application tables whose data is used in an application or report. A business view does not select specific rows, nor does it contain any actual data. It is strictly a view through which you can manipulate data.

central objects merge

A process that blends a customer's modifications to the objects in a current release with objects in a new release.

central server

A server that has been designated to contain the originally installed version of the software (central objects) for deployment to client computers. In a typical JD Edwards EnterpriseOne installation, the software is loaded on to one machine—the central

server. Then, copies of the software are pushed out or downloaded to various workstations attached to it. That way, if the software is altered or corrupted through its use on workstations, an original set of objects (central objects) is always available on the central server.

charts

Tables of information in JD Edwards EnterpriseOne that appear on forms in the software.

check-in repository

A repository for developers to check in and check out business service artifacts. There are multiple check-in repositories. Each can be used for a different purpose (for example, development, production, testing, and so on).

checksum

A fixed-size datum computed from an arbitrary block of digital data for the purpose of detecting accidental errors that may have been introduced during its transmission or storage. JD Edwards EnterpriseOne uses the checksum to verify the integrity of packages that have been downloaded by recomputing the checksum of the downloaded package and comparing it with the checksum of the original package. The procedure that yields the checksum from the data is called a checksum function or checksum algorithm. JD Edwards EnterpriseOne uses the MD5 and STA-1 checksum algorithms.

connector

Component-based interoperability model that enables third-party applications and JD Edwards EnterpriseOne to share logic and data. The JD Edwards EnterpriseOne connector architecture includes Java and COM connectors.

Control Table Workbench

An application that, during the Installation Workbench processing, runs the batch applications for the planned merges that update the data dictionary, user-defined codes, menus, and user override tables.

control tables merge

A process that blends a customer's modifications to the control tables with the data that accompanies a new release.

correlation data

The data used to tie HTTP responses with requests that consist of business service name and method.

credentials

A valid set of JD Edwards EnterpriseOne username/password/environment/role, EnterpriseOne session, or EnterpriseOne token.

cross-reference utility services

Utility services installed in a BPEL/ESB environment that access EnterpriseOne cross-reference data.

database credentials

A valid database username/password.

database server

A server in a local area network that maintains a database and performs searches for client computers.

Data Source Workbench

An application that, during the Installation Workbench process, copies all data sources that are defined in the installation plan from the Data Source Master and Table and Data Source Sizing tables in the Planner data source to the system-release number data source. It also updates the Data Source Plan detail record to reflect completion.

deployment artifacts

Artifacts that are needed for the deployment process, such as servers, ports, and such.

deployment server

A server that is used to install, maintain, and distribute software to one or more enterprise servers and client workstations.

direct connect

A transaction method in which a client application communicates interactively and directly with a server application.

See also batch-of-one and store-and-forward.

Do Not Translate (DNT)

A type of data source that must exist on the iSeries because of BLOB restrictions.

embedded application server instance

An OC4J instance started by and running wholly within JDeveloper.

edit code

A code that indicates how a specific value for a report or a form should appear or be formatted. The default edit codes that pertain to reporting require particular attention because they account for a substantial amount of information.

edit mode

A condition of a form that enables users to change data.

edit rule

A method used for formatting and validating user entries against a predefined rule or set of rules.

Electronic Data Interchange (EDI)

An interoperability model that enables paperless computer-to-computer exchange of business transactions between JD Edwards EnterpriseOne and third-party systems. Companies that use EDI must have translator software to convert data from the EDI standard format to the formats of their computer systems.

embedded event rule

An event rule that is specific to a particular table or application. Examples include form-to-form calls, hiding a field based on a processing option value, and calling a business function. Contrast with the business function event rule.

Employee Work Center

A central location for sending and receiving all JD Edwards EnterpriseOne messages (system and user generated), regardless of the originating application or user. Each user has a mailbox that contains workflow and other messages, including Active Messages.

enterprise server

A server that contains the database and the logic for JD Edwards EnterpriseOne.

Enterprise Service Bus (ESB)

Middleware infrastructure products or technologies based on web services standards that enable a service-oriented architecture using an event-driven and XML-based messaging framework (the bus).

EnterpriseOne administrator

An actor responsible for the EnterpriseOne administration system.

EnterpriseOne credentials

A user ID, password, environment, and role used to validate a user of EnterpriseOne.

EnterpriseOne development client

Historically called "fat client," a collection of installed EnterpriseOne components required to develop EnterpriseOne artifacts, including the Microsoft Windows client and design tools.

EnterpriseOne extension

A JDeveloper component (plug-in) specific to EnterpriseOne. A JDeveloper wizard

is a specific example of an extension.

EnterpriseOne object

A reusable piece of code that is used to build applications. Object types include tables, forms, business functions, data dictionary items, batch processes, business views, event rules, versions, data structures, and media objects.

EnterpriseOne process

A software process that enables JD Edwards EnterpriseOne clients and servers to handle processing requests and run transactions. A client runs one process, and servers can have multiple instances of a process. JD Edwards EnterpriseOne processes can also be dedicated to specific tasks (for example, workflow messages and data replication) to ensure that critical processes don't have to wait if the server is particularly busy.

EnterpriseOne resource

Any EnterpriseOne table, metadata, business function, dictionary information, or other information restricted to authorized users.

Environment Workbench

An application that, during the Installation Workbench process, copies the environment information and Object Configuration Manager tables for each environment from the Planner data source to the system-release number data source. It also updates the Environment Plan detail record to reflect completion.

escalation monitor

A batch process that monitors pending requests or activities and restarts or forwards them to the next step or user after they have been inactive for a specified amount of time.

event rule

A logic statement that instructs the system to perform one or more operations based on an activity that can occur in a specific application, such as entering a form or exiting a field.

explicit transaction

Transaction used by a business service developer to explicitly control the type (auto or manual) and the scope of transaction boundaries within a business service.

exposed method or value object

Published business service source files or parts of published business service source files that are part of the published interface. These are part of the contract with the customer.

fast path

A command prompt that enables the user to move quickly among menus and applications by using specific commands.

file server

A server that stores files to be accessed by other computers on the network. Unlike a disk server, which appears to the user as a remote disk drive, a file server is a sophisticated device that not only stores files, but also manages them and maintains order as network users request files and make changes to these files.

final mode

The report processing mode of a processing mode of a program that updates or creates data records.

foundation

A framework that must be accessible for execution of business services at runtime. This includes, but is not limited to, the Java Connector and JDBj.

FTP server

A server that responds to requests for files via file transfer protocol.

HTTP Adapter

A generic set of services that are used to do the basic HTTP operations, such as GET, POST, PUT, DELETE, TRACE, HEAD, and OPTIONS with the provided URL.

instantiate

A Java term meaning “to create.” When a class is instantiated, a new instance

is created.

integration developer

The user of the system who develops, runs, and debugs the EnterpriseOne business services. The integration developer uses the EnterpriseOne business services to develop these components.

integration point (IP)

The business logic in previous implementations of EnterpriseOne that exposes a document level interface. This type of logic used to be called XBPs. In EnterpriseOne 8.11, IPs are implemented in Web Services Gateway powered by webMethods.

integration server

A server that facilitates interaction between diverse operating systems and applications across internal and external networked computer systems.

integrity test

A process used to supplement a company's internal balancing procedures by locating and reporting balancing problems and data inconsistencies.

interface table

See Z table.

internal method or value object

Business service source files or parts of business service source files that are not part of the published interface. These could be private or protected methods. These could be value objects not used in published methods.

interoperability model

A method for third-party systems to connect to or access JD Edwards EnterpriseOne.

in-your-face error

In JD Edwards EnterpriseOne, a form-level property which, when enabled, causes the text of application errors to appear on the form.

jargon

An alternative data dictionary item description that JD Edwards EnterpriseOne appears based on the product code of the current object.

Java application server

A component-based server that resides in the middle-tier of a server-centric architecture. This server provides middleware services for security and state maintenance, along with data access and persistence.

JDBNET

A database driver that enables heterogeneous servers to access each other's data.

JDEBASE Database Middleware

A JD Edwards EnterpriseOne proprietary database middleware package that provides platform-independent APIs, along with client-to-server access.

JDECallObject

An API used by business functions to invoke other business functions.

jde.ini

A JD Edwards EnterpriseOne file (or member for iSeries) that provides the runtime settings required for JD Edwards EnterpriseOne initialization. Specific versions of the file or member must reside on every machine running JD Edwards EnterpriseOne. This includes workstations and servers.

JDEIPC

Communications programming tools used by server code to regulate access to the same data in multiprocess environments, communicate and coordinate between processes, and create new processes.

jde.log

The main diagnostic log file of JD Edwards EnterpriseOne. This file is always located in the root directory on the primary drive and contains status and error messages from the startup and operation of JD Edwards EnterpriseOne.

JDENET

A JD Edwards EnterpriseOne proprietary communications middleware package. This package is a peer-to-peer, message-based, socket-based, multiprocess communications middleware solution. It handles client-to-server and server-to-server communications for all JD Edwards EnterpriseOne supported platforms.

JDeveloper Project

An artifact that JDeveloper uses to categorize and compile source files.

JDeveloper Workspace

An artifact that JDeveloper uses to organize project files. It contains one or more project files.

JMS Queue

A Java Messaging service queue used for point-to-point messaging.

listener service

A listener that listens for XML messages over HTTP.

local repository

A developer's local development environment that is used to store business service artifacts.

Location Workbench

An application that, during the Installation Workbench process, copies all locations that are defined in the installation plan from the Location Master table in the Planner data source to the system data source.

logic server

A server in a distributed network that provides the business logic for an application program. In a typical configuration, pristine objects are replicated on to the logic server from the central server. The logic server, in conjunction with workstations, actually performs the processing required when JD Edwards EnterpriseOne software runs.

MailMerge Workbench

An application that merges Microsoft Word 6.0 (or higher) word-processing documents with JD Edwards EnterpriseOne records to automatically print business documents. You can use MailMerge Workbench to print documents, such as form letters about verification of employment.

Manual Commit transaction

A database connection where all database operations delay writing to the database until a call to commit is made.

master business function (MBF)

An interactive master file that serves as a central location for adding, changing, and updating information in a database. Master business functions pass information between data entry forms and the appropriate tables. These master functions provide a common set of functions that contain all of the necessary default and editing rules for related programs. MBFs contain logic that ensures the integrity of adding, updating, and deleting information from databases.

master table

See published table.

media storage object

Files that use one of the following naming conventions that are not organized into table format: Gxxx, xxxGT, or GTxxx.

message center

A central location for sending and receiving all JD Edwards EnterpriseOne messages (system and user generated), regardless of the originating application or user.

messaging adapter

An interoperability model that enables third-party systems to connect to JD Edwards EnterpriseOne to exchange information through the use of messaging queues.

messaging server

A server that handles messages that are sent for use by other programs using a messaging API. Messaging servers typically employ a middleware program to perform their functions.

Monitoring Application

An EnterpriseOne tool provided for an administrator to get statistical information for various EnterpriseOne servers, reset statistics, and set notifications.

named event rule (NER)

Encapsulated, reusable business logic created using event rules, rather that C programming. NERs are also called business function event rules. NERs can be reused in multiple places by multiple programs. This modularity lends itself to streamlining, reusability of code, and less work.

Object Configuration Manager (OCM)

In JD Edwards EnterpriseOne, the object request broker and control center for the runtime environment. OCM keeps track of the runtime locations for business functions, data, and batch applications. When one of these objects is called, OCM directs access to it using defaults and overrides for a given environment and user.

Object Librarian

A repository of all versions, applications, and business functions reusable in building applications. Object Librarian provides check-out and check-incapabilities for developers, and it controls the creation, modification, and use of JD Edwards EnterpriseOne objects. Object Librarian supports multiple environments (such as

production and development) and enables objects to be easily moved from one environment to another.

Object Librarian merge

A process that blends any modifications to the Object Librarian in a previous release into the Object Librarian in a new release.

Open Data Access (ODA)

An interoperability model that enables you to use SQL statements to extract JD Edwards EnterpriseOne data for summarization and report generation.

Output Stream Access (OSA)

An interoperability model that enables you to set up an interface for JD Edwards EnterpriseOne to pass data to another software package, such as Microsoft Excel, for processing.

package

JD Edwards EnterpriseOne objects are installed to workstations in packages from the deployment server. A package can be compared to a bill of material or kit that indicates the necessary objects for that workstation and where on the deployment server the installation program can find them. It is point-in-time snapshot of the central objects on the deployment server.

package build

A software application that facilitates the deployment of software changes and new applications to existing users. Additionally, in JD Edwards EnterpriseOne, a package build can be a compiled version of the software. When you upgrade your version of the ERP software, for example, you are said to take a package build.

Consider the following context: "Also, do not transfer business functions into the production path code until you are ready to deploy, because a global build of business functions done during a package build will automatically include the new functions." The process of creating a package build is often referred to, as it is in this example, simply as "a package build."

package location

The directory structure location for the package and its set of replicated objects. This is usually \\deployment server\release\path_code\package\package name. The subdirectories under this path are where the replicated objects for the package are placed. This is also referred to as where the package is built or stored.

Package Workbench

An application that, during the Installation Workbench process, transfers the package information tables from the Planner data source to the system-release number data source. It also updates the Package Plan detail record to reflect completion.

Pathcode Directory

The specific portion of the file system on the EnterpriseOne development client where EnterpriseOne development artifacts are stored.

patterns

General repeatable solutions to a commonly occurring problem in software design. For business service development, the focus is on the object relationships and interactions.

For orchestrations, the focus is on the integration patterns (for example, synchronous and asynchronous request/response, publish, notify, and receive/reply).

print server

The interface between a printer and a network that enables network clients to connect to the printer and send their print jobs to it. A print server can be a computer, separate hardware device, or even hardware that resides inside of the printer itself.

pristine environment

A JD Edwards EnterpriseOne environment used to test unaltered objects with JD Edwards EnterpriseOne demonstration data or for training classes. You must have this environment so that you can compare pristine objects that you modify.

processing option

A data structure that enables users to supply parameters that regulate the running of a batch program or report. For example, you can use processing options to specify default values for certain fields, to determine how information appears or is printed, to specify date ranges, to supply runtime values that regulate program execution, and so on.

production environment

A JD Edwards EnterpriseOne environment in which users operate EnterpriseOne software.

Production Published Business Services Web Service

Published business services web service deployed to a production application server.

program temporary fix (PTF)

A representation of changes to JD Edwards EnterpriseOne software that your organization receives on magnetic tapes or disks.

project

In JD Edwards EnterpriseOne, a virtual container for objects being developed in Object Management Workbench.

promotion path

The designated path for advancing objects or projects in a workflow. The following is the normal promotion cycle (path):

11>21>26>28>38>01

In this path, 11 equals new project pending review, 21 equals programming, 26 equals QA test/review, 28 equals QA test/review complete, 38 equals in production, 01 equals complete. During the normal project promotion cycle, developers check objects out of and into the development path code and then promote them to the prototype path code. The objects are then moved to the productions path code before declaring them complete.

proxy server

A server that acts as a barrier between a workstation and the internet so that the enterprise can ensure security, administrative control, and caching service.

published business service

EnterpriseOne service level logic and interface. A classification of a published business service indicating the intention to be exposed to external (non-EnterpriseOne) systems.

published business service identification information

Information about a published business service used to determine relevant authorization records. Published business services + method name, published business services, or *ALL.

published business service web service

Published business services components packaged as J2EE Web Service (namely, a J2EE EAR file that contains business service classes, business service foundation, configuration files, and web service artifacts).

published table

Also called a master table, this is the central copy to be replicated to other machines. Residing on the publisher machine, the F98DRPUB table identifies all of the published tables and their associated publishers in the enterprise.

publisher

The server that is responsible for the published table. The F98DRPUB table identifies all of the published tables and their associated publishers in the enterprise.

QBE

An abbreviation for query by example. In JD Edwards EnterpriseOne, the QBE line is the top line on a detail area that is used for filtering data.

real-time event

A message triggered from EnterpriseOne application logic that is intended for external systems to consume.

refresh

A function used to modify JD Edwards EnterpriseOne software, or subset of it, such as a table or business data, so that it functions at a new release or cumulative update level.

replication server

A server that is responsible for replicating central objects to client machines.

rules

Mandatory guidelines that are not enforced by tooling, but must be followed in order to accomplish the desired results and to meet specified standards.

secure by default

A security model that assumes that a user does not have permission to execute an object unless there is a specific record indicating such permissions.

Secure Socket Layer (SSL)

A security protocol that provides communication privacy. SSL enables client and server applications to communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery.

selection

Found on JD Edwards EnterpriseOne menus, a selection represents functions that you can access from a menu. To make a selection, type the associated number in the Selection field and press Enter.

serialize

The process of converting an object or data into a format for storage or transmission across a network connection link with the ability to reconstruct the original data or objects when needed.

Server Workbench

An application that, during the Installation Workbench process, copies the server configuration files from the Planner data source to the system-release number data source. The application also updates the Server Plan detail record to reflect completion.

SOA

Abbreviation for Service Oriented Architecture.

softcoding

A coding technique that enables an administrator to manipulate site-specific variables that affect the execution of a given process.

source repository

A repository for HTTP adapter and listener service development environment artifacts.

Specification merge

A merge that comprises three merges: Object Librarian merge, Versions List merge, and Central Objects merge. The merges blend customer modifications with data that accompanies a new release.

specification

A complete description of a JD Edwards EnterpriseOne object. Each object has its own specification, or name, which is used to build applications.

Specification Table Merge Workbench

An application that, during the Installation Workbench process, runs the batch applications that update the specification tables.

SSL Certificate

A special message signed by a certificate authority that contains the name of a user and that user's public key in such a way that anyone can "verify" that the message was signed by no one other than the certification authority and thereby develop trust in the user's public key.

store-and-forward

The mode of processing that enables users who are disconnected from a server to enter transactions and then later connect to the server to upload those transactions.

subscriber table

Table F98DRSUB, which is stored on the publisher server with the F98DRPUB table and identifies all of the subscriber machines for each published table.

super class

An inheritance concept of the Java language where a class is an instance of something, but is also more specific. “Tree” might be the super class of “Oak” and “Elm,” for example.

table access management (TAM)

The JD Edwards EnterpriseOne component that handles the storage and retrieval of use-defined data. TAM stores information, such as data dictionary definitions; application and report specifications; event rules; table definitions; business function input parameters and library information; and data structure definitions for running applications, reports, and business functions.

Table Conversion Workbench

An interoperability model that enables the exchange of information between JD Edwards EnterpriseOne and third-party systems using non-JD Edwards EnterpriseOne tables.

table conversion

An interoperability model that enables the exchange of information between JD Edwards EnterpriseOne and third-party systems using non-JD Edwards EnterpriseOne tables.

table event rules

Logic that is attached to database triggers that runs whenever the action specified by the trigger occurs against the table. Although JD Edwards EnterpriseOne enables event rules to be attached to application events, this functionality is application specific. Table event rules provide embedded logic at the table level.

terminal server

A server that enables terminals, microcomputers, and other devices to connect to a network or host computer or to devices attached to that particular computer.

transaction processing (TP) monitor

A monitor that controls data transfer between local and remote terminals and the applications that originated them. TP monitors also protect data integrity in the distributed environment and may include programs that validate data and format terminal screens.

transaction processing method

A method related to the management of a manual commit transaction boundary (for example, start, commit, rollback, and cancel).

transaction set

An electronic business transaction (electronic data interchange standard document) made up of segments.

trigger

One of several events specific to data dictionary items. You can attach logic to a data dictionary item that the system processes automatically when the event occurs.

triggering event

A specific workflow event that requires special action or has defined consequences or resulting actions.

user identification information

User ID, role, or *public.

User Overrides merge

Adds new user override records into a customer's user override table.

value object

A specific type of source file that holds input or output data, much like a data structure passes data. Value objects can be exposed (used in a published business service) or internal, and input or output. They are comprised of simple and complex elements and accessories to those elements.

versioning a published business service

Adding additional functionality/interfaces to the published business services without modifying the existing functionality/interfaces.

Versions List merge

The Versions List merge preserves any non-XJDE and non-ZJDE version specifications for objects that are valid in the new release, as well as their processing options data.

visual assist

Forms that can be invoked from a control via a trigger to assist the user in determining what data belongs in the control.

vocabulary override

An alternate description for a data dictionary item that appears on a specific JD Edwards EnterpriseOne form or report.

web application server

A web server that enables web applications to exchange data with the back-end systems and databases used in eBusiness transactions.

web server

A server that sends information as requested by a browser, using the TCP/IP set of protocols. A web server can do more than just coordination of requests from browsers; it can do anything a normal server can do, such as house applications or data. Any computer can be turned into a web server by installing server software and connecting the machine to the internet.

Web Service Description Language (WSDL)

An XML format for describing network services.

Web Service Inspection Language (WSIL)

An XML format for assisting in the inspection of a site for available services and a set of rules for how inspection-related information should be made.

web service softcoding record

An XML document that contains values that are used to configure a web service proxy. This document identifies the endpoint and conditionally includes security information.

web service softcoding template

An XML document that provides the structure for a soft coded record.

Where clause

The portion of a database operation that specifies which records the database operation will affect.

Windows terminal server

A multiuser server that enables terminals and minimally configured computers to display Windows applications even if they are not capable of running Windows software themselves. All client processing is performed centrally at the Windows terminal server and only display, keystroke, and mouse commands are transmitted over the network to the client terminal device.

wizard

A type of JDeveloper extension used to walk the user through a series of steps.

workbench

A program that enables users to access a group of related programs from a single entry point. Typically, the programs that you access from a workbench are used to complete a large business process. For example, you use the JD Edwards EnterpriseOne Payroll Cycle Workbench (P07210) to access all of the programs that the system uses to process payroll, print payments, create payroll reports, create journal entries, and update payroll history. Examples of JD Edwards EnterpriseOne workbenches include Service Management Workbench (P90CD020), Line Scheduling Workbench (P3153), Planning Workbench (P13700), Auditor's Workbench (P09E115), and Payroll Cycle Workbench.

workflow

The automation of a business process, in whole or in part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules.

workgroup server

A server that usually contains subsets of data replicated from a master network server. A workgroup server does not perform application or batch processing.

XAPI events

A service that uses system calls to capture JD Edwards EnterpriseOne transactions as they occur and then calls third-party software, end users, and other JD Edwards EnterpriseOne systems that have requested notification when the specified transactions occur to return a response.

XML CallObject

An interoperability capability that enables you to call business functions.

XML Dispatch

An interoperability capability that provides a single point of entry for all XML documents coming into JD Edwards EnterpriseOne for responses.

XML List

An interoperability capability that enables you to request and receive JD Edwards EnterpriseOne database information in chunks.

XML Service

An interoperability capability that enables you to request events from one JD Edwards EnterpriseOne system and receive a response from another JD Edwards EnterpriseOne system.

XML Transaction

An interoperability capability that enables you to use a predefined transaction type to send information to or request information from JD Edwards EnterpriseOne. XML transaction uses interface table functionality.

XML Transaction Service (XTS)

Transforms an XML document that is not in the JD Edwards EnterpriseOne format into an XML document that can be processed by JD Edwards EnterpriseOne. XTS then transforms the response back to the request originator XML format.

Z event

A service that uses interface table functionality to capture JD Edwards EnterpriseOne transactions and provide notification to third-party software, end users, and other JD Edwards EnterpriseOne systems that have requested to be notified when certain transactions occur.

Z table

A working table where non-JD Edwards EnterpriseOne information can be stored and then processed into JD Edwards EnterpriseOne. Z tables also can be used to retrieve JD Edwards EnterpriseOne data. Z tables are also known as interface tables.

Z transaction

Third-party data that is properly formatted in interface tables for updating to the JD Edwards EnterpriseOne database.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W