

JD Edwards EnterpriseOne Tools

Administration Guide

9.2

Copyright © 2011, 2022, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface	i
<hr/>	
1 Introduction to JD Edwards EnterpriseOne Server Administration	1
EnterpriseOne Server Administration Overview	1
EnterpriseOne Server Administration Implementation	1
Understanding Java Runtime Engine Installation for JD Edwards EnterpriseOne Tools 9.2	1
2 Administering the Server	3
Understanding Server Administration for IBM i	3
Starting the Enterprise Server for IBM i	6
Shutting Down the Enterprise Server for IBM i	10
Using IBM i Integrated File System Logging Support	10
Cleaning Up the Enterprise Server for IBM i	11
Setting Up a Printer for IBM i	12
Administering Batch Processes for IBM i	13
Running Multiple Instances of JD Edwards EnterpriseOne on the IBM i	16
Administering JD Edwards EnterpriseOne Database Security for IBM i	27
3 Administering the UNIX and Linux Servers	45
Understanding Server Administration for UNIX and Linux	45
Starting the Enterprise Server for UNIX or Linux	50
Shutting Down the Enterprise Server for UNIX or Linux	53
Setting Up a Printer for UNIX or Linux	54
Administering Batch Processes for UNIX or Linux	54
Maintaining File Security for UNIX and Linux	60
Working with HP-UX and Solaris Kernel Parameter Settings	63
Working with Linux Kernel Parameter Settings	69
Working with AIX Kernel Parameter Settings for JD Edwards EnterpriseOne	70
Running Multiple Instances of the JD Edwards EnterpriseOne Enterprise Server	73
4 Administering the Windows Server	77
Understanding Server Administration for Windows	77

Setting Up a Printer for Windows	85
Working with Network Services	89
Administering Batch Processes for Windows	93
Maintaining File Security for Windows	97
Running Multiple Instances of JD Edwards EnterpriseOne on Windows	98
5 Backing Up JD Edwards EnterpriseOne Tables	105
Understanding Backup Requirements for Servers	105
Backing Up JD Edwards EnterpriseOne Tables on Servers	112
6 Troubleshooting the Enterprise Server	119
Understanding Enterprise Server Troubleshooting	119
Viewing Enterprise Server Logs from the Workstation	127
Setting Up the Enterprise Server jde.log	127
Setting Up the Enterprise Server jdedebug.log	127
Setting Up the <batch process>.log File	128
Troubleshooting the Enterprise Server	129
Troubleshooting the Enterprise Server Processes	131
Troubleshooting the Enterprise Server	179
Troubleshooting the UNIX/Linux Enterprise Server	190
Troubleshooting the Microsoft Windows Enterprise Server	195
Troubleshooting Web Servers	205
7 Working with Servers	209
Understanding the Work With Servers Program (P986116)	209
Understanding the Work With Locations and Machines Web Program (Release 9.2.5)	209
Understanding the Work With Virtual Hosts Program (Release 9.2.5)	212
Managing Server Jobs	213
Managing Job Queues	222
Managing JD Edwards EnterpriseOne Subsystems	228
8 Glossary	235
EnterpriseOne extension	235
JDBNET	235
JDEIPC	235
program temporary fix (PTF)	235

replication server	235
serialize	235
terminal server	236
Index	237

Preface

Welcome to the JD Edwards EnterpriseOne documentation.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Related Information

For additional information about JD Edwards EnterpriseOne applications, features, content, and training, visit the JD Edwards EnterpriseOne pages on the JD Edwards Resource Library located at:

<http://learnjde.com>

Conventions

The following text conventions are used in this document:

Convention	Meaning
Bold	Boldface type indicates graphical user interface elements associated with an action or terms defined in text or the glossary.
<i>Italics</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
Monospace	Monospace type indicates commands within a paragraph, URLs, code examples, text that appears on a screen, or text that you enter.
> Oracle by Example	Indicates a link to an Oracle by Example (OBE). OBEs provide hands-on, step-by-step instructions, including screen captures that guide you through a process using your own environment. Access to OBEs requires a valid Oracle account.

1 Introduction to JD Edwards EnterpriseOne Server Administration

EnterpriseOne Server Administration Overview

EnterpriseOne Server 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. EnterpriseOne Server Administration uses the flexibility of Oracle's JD Edwards EnterpriseOne administration to optimize Oracle's JD Edwards EnterpriseOne installation for the enterprise.

Access

Voiding Joint Venture Distribution Documents Using a Batch Process (R09J414)

Abbreviations Used in Formulas

Abbreviations Used in Formulas

EnterpriseOne Server 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.

Understanding Java Runtime Engine Installation for JD Edwards EnterpriseOne Tools 9.2

With JD Edwards EnterpriseOne Tools 9.2, the Java Runtime Engine is no longer bundled into the JD Edwards EnterpriseOne Tools code. Therefore, before installing EnterpriseOne Tools 9.2, access the vendor website to download and install the JRE for the appropriate platform, which includes:

- Oracle LINUX
- Oracle Solaris
- Microsoft Windows
- HP-UX Itanium
- IBM AIX
- IBM AS400 (already pre-installed)

Once the JRE has been downloaded and installed, you must then configure the `InProcessJVMHome` setting in the `[JDE JVM]` section of the `jde.ini` file:

[JDE JVM]

InProcessJVMHome=<Path to JVM library>

Below is the <Path to JVM library> by platform:

- Oracle Windows

<JAVA_INSTALL_HOME>\JRE\bin\client\jvm.dll

- Oracle Linux

<JAVA_INSTALL_HOME>/jre/lib/i386/server/libjvm.so

- Oracle Solaris

<JAVA_INSTALL_HOME>/jre/lib/sparc/server/libjvm.so

- HP-UX Itanium

<JAVA_INSTALL_HOME>/jre/lib/IA64N/server/libjvm.so

- IBM AIX

<JAVA_INSTALL_HOME>/jre/lib/ppc/j9vm/libjvm.so

- IBM AS400

Because Java comes as a part of AS400 OS, you do not need to install or configure the JRE as this is already hardcoded in the following path:

/QOpenSys/QIBM/ProdData/JavaVM/jdk60/32bit

2 Administering the Server

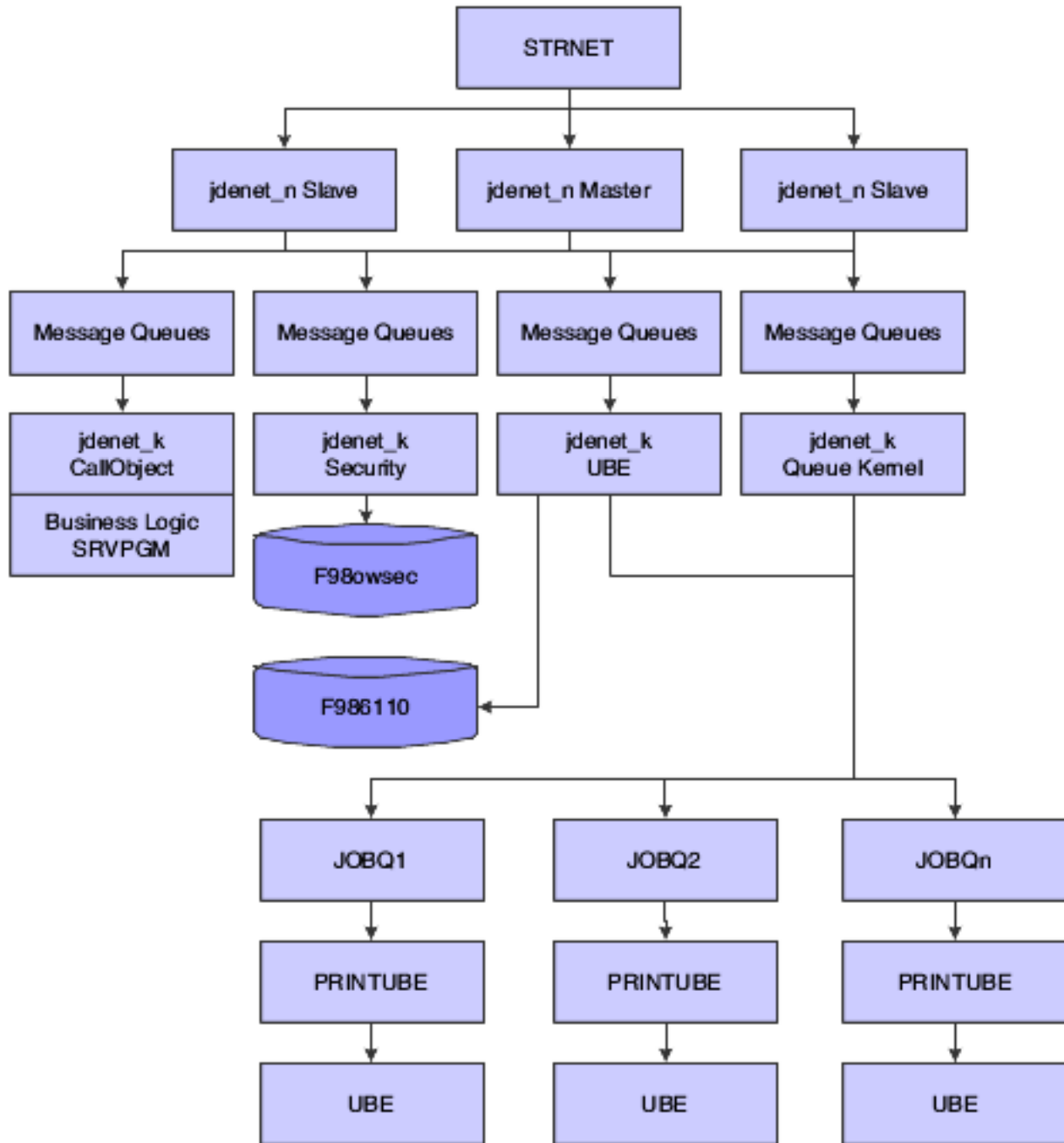
Understanding Server Administration for IBM i

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 EnterpriseOne initialization for *IBM i*.

JD Edwards EnterpriseOne Architecture and Process Flow for IBM i

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



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.

- 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.

- 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.
- 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.
- 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.

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

JD Edwards EnterpriseOne Initialization for IBM i

This initialization occurs when you start JD Edwards EnterpriseOne 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. As of Release 9.2.3, JDBNET is no longer supported.

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.

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 E920SYS.

The releaseSYS library contains these objects:

Object	Description
INI	The jde.ini file used to initialize JD Edwards EnterpriseOne on the <i>IBM i</i> enterprise server.
*PGM and *SRVPGM	The various programs and service programs required to run the JD Edwards EnterpriseOne <i>IBM i</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.

Object	Description
DPSSTMF (*CMD)	<p>The display stream file, which displays <i>IBM i</i> Integrated File System (IFS) text stream files.</p> <p>The JD Edwards EnterpriseOne log files, <code>jde.log</code> and <code>jdedebug.log</code>, typically reside in a directory called <code>JDEErelease</code>, where <code>release</code> represents the JD Edwards EnterpriseOne release, such as <code>/JDEE920</code>.</p>
LINKBSFN (*CMD)	<p>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.</p>
PID2JOB (*CMD)	<p>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 <code>JDE.LOG</code> files. This command returns job information only while the job is still active.</p>
PORTTEST (*CMD)	<p>The command that runs the JD Edwards EnterpriseOne test program <code>PORTTEST</code>.</p>
RUNUBE (*CMD)	<p>The command that interactively runs a batch program. If you need to run a batch program, use the <code>SBMJOB</code> command to submit the <code>RUNUBE</code> command to batch.</p>
PRINTQUEUE (*FILE)	<p>The file that contains the output from a batch program. This output is stored as ASCII PDF members.</p>
*PGM and *SRVPGM	<p>The programs and server programs required to run the JD Edwards EnterpriseOne network.</p>
JDENET (*JOBQ)	<p>The job queue used by the JD Edwards EnterpriseOne <i>IBM i</i> network jobs.</p>
NETJOBQ (*JOBQ)	<p>The job description used by JD Edwards EnterpriseOne <i>IBM i</i> network jobs.</p>
JDENET (*CLS)	<p>The class used to create the routing entry for the <code>JDENET</code> subsystem.</p>
ENDNET (*CMD)	<p>The command that ends the JD Edwards EnterpriseOne <i>IBM i</i> network jobs and cleans up the network runtime structures.</p>
IPCS (*CMD)	<p>The utility command that indicates the status of objects used by the JD Edwards EnterpriseOne <i>IBM i</i> network jobs and as a backup method for cleaning up the <code>IPCS</code> objects.</p>
STRNET (*CMD)	<p>The command that starts the JD Edwards EnterpriseOne <i>IBM i</i> network jobs.</p>
CLRIPC (*CMD)	<p>The command used to clear <code>IPC</code> structures.</p>
DSPIPC (*CMD)	<p>The command used to display <code>IPC</code> structures.</p>
JDEErelease (*SBSD)	<p>The subsystem description under which the JD Edwards EnterpriseOne network jobs run. The variable <code>release</code> is the JD Edwards EnterpriseOne release level, such as <code>JDEE920</code>.</p>

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 (ADDLIB) 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*.

Prerequisites

Before you complete the tasks in this section:

- Install JD Edwards EnterpriseOne as described in the appropriate install guide at <http://docs.oracle.com/cd/E61420-01/index.html>. In the appropriate guide, you should have performed all of the 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.

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! Portttest 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(JDEErelease)
```

The variable `release` is the JD Edwards EnterpriseOne release level that the site is using, such as `JDEE920`.

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).

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(E920SYS 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.

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.

Prerequisite

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

Note:

- *Understanding the Library Structure for JD Edwards EnterpriseOne.*

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.

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=/JDE920_a/jdedebug
JobFile=/JDE920_a/jde.log
JD Edwards EnterpriseOne generates log files in the IFS directory called /JDE920_
a.
```

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

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*.

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. Inter-process 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.

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.

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
```

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 `/JDEErelease/jde_*
```

Where release is the JD Edwards EnterpriseOne release, such as JDEE920.

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

```
DEL `/JDEErelease/jdedebug_*
```

Where release is the JD Edwards EnterpriseOne release, such as JDEE920.

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.

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.

Note:

- ["Defining Print Properties for Reports" in the JD Edwards EnterpriseOne Tools Report Printing Administration Technologies Guide](#) .

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.

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 '/JDEErelease, where release is the JD Edwards EnterpriseOne release, as in JDEE920.

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.

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.

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, JDEE920, 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 E920SYS/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, /E920SYS/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 depends 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 and when there is no OUTQ associated with the *IBM i* proxy user profile.

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 E920SYS/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)
```

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(PD920)
REPORT(R0006P) VERSION(XJDE0001)
```

This command begins processing version XJDE0001 of the report in the PD920 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.

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(PD920)
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.

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)
```

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 `/E920SYS/PrintQueue`), and view the PDF files.

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
```

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.

Note:

- *JD Edwards EnterpriseOne Tools Server Manager Guide*

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

Section in server JDE.INI file	Parameter	Purpose
		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]	JDTSTFile	Specifies the location of the lock manager trace file on the <i>IBM i</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.

Note:

- [Administering JD Edwards EnterpriseOne Database Security for](#).
- ["Creating a New Environment Using the Director Mode" in the JD Edwards EnterpriseOne Tools Configurable Network Computing Implementation Guide](#) .
- [Setting Up Database Security for Multiple JD Edwards EnterpriseOne Instances](#).
- [Managing JD Edwards EnterpriseOne Subsystems](#).

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.

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 datapath 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.

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.

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.

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

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.

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.

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.

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.

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.

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.

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.

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.

Copying Libraries and Directories (Release 9.2.6.0)

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 E920SYS E920CST
```

Where E920CST 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/E920SYS)
```
6. Save the system directory into the save file by typing this command:

```
SAV DEV ('/QSYS.LIB/TEMPLIB/E920SYS.FILE') OBJ((' /E920SYS)) 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/E920SYS.FILE') OBJ((' /E920sys/*' *INCLUDE/E920cst'))
```

Where E920cst 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 package name library in the QSYS file system by typing this command:

```
CPYLIB <PD package name> <CST new package name>
```

In the above command, **CST new package name** is the name of the package library specified in the spec.ini file for the new instance of JD Edwards EnterpriseOne. The name of the library for the new instance cannot exceed 10 characters.

Note: The package name library for any environment that you intend to use for a new instance of JD Edwards EnterpriseOne must be copied to the new library. You cannot share package name directories between two or more instances of JD Edwards EnterpriseOne because such sharing 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/PRD920)
```

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/PRD920.FILE') OBJ('/prd920/*') 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/PRD920.FILE') OBJ('/prd920/* INCLUDE '/cst920'))
```

Where **cst920** 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>
```

14. Open the `spec.ini` file under the `pathcode` directory and change the package name to the new package name.

Applying Security to Multiple Instances of JD Edwards EnterpriseOne on the

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 `SYS920`, `920MAP`, 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('/920CSTLOG')
```

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.

Creating a JD Edwards EnterpriseOne Sub-System 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:
 - o SUBSYSTEM
 - o 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`

Administering 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.

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.

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 not for subsequent releases of JD Edwards EnterpriseOne.

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.

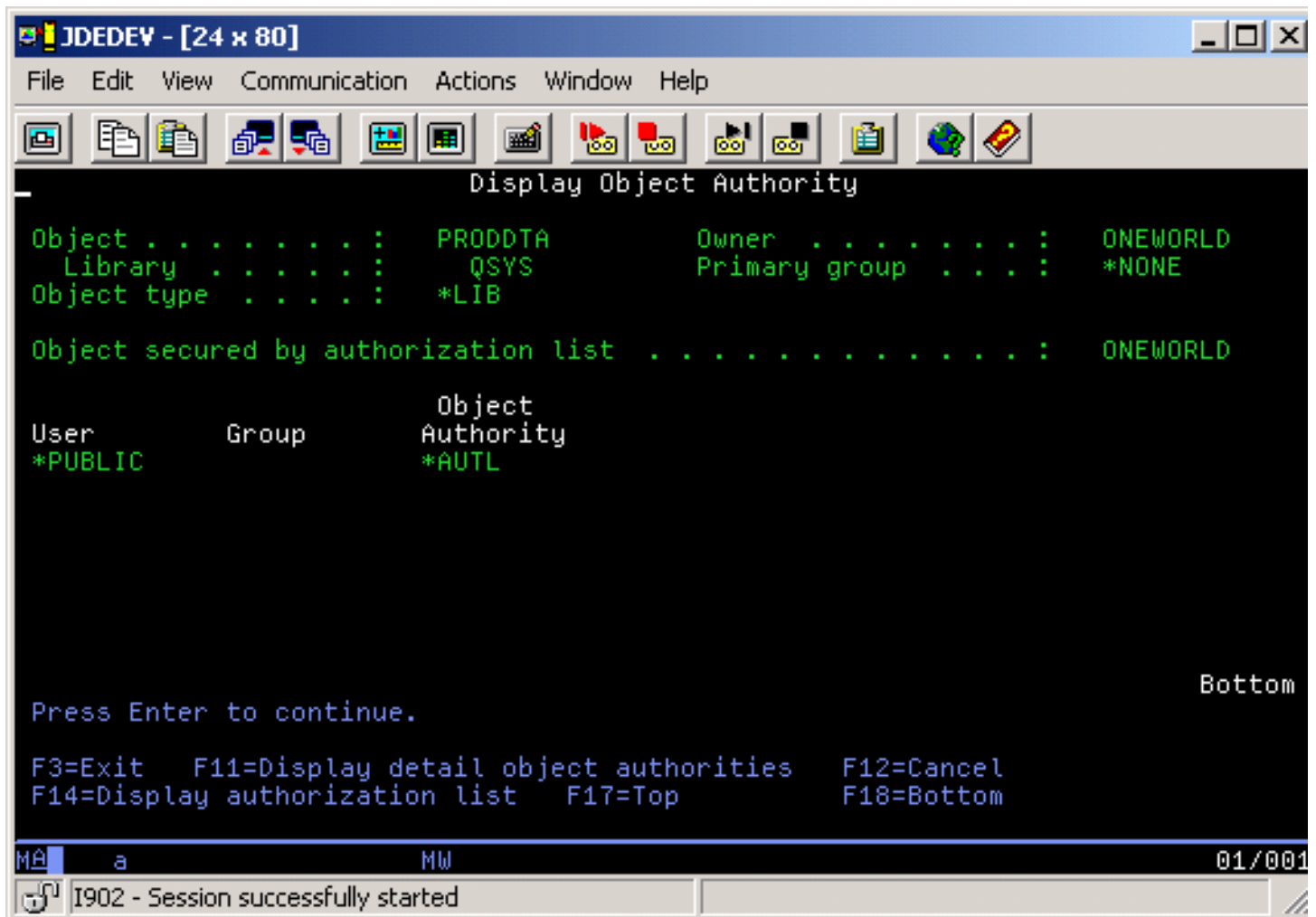
Sample Results for SETOWAUT

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.

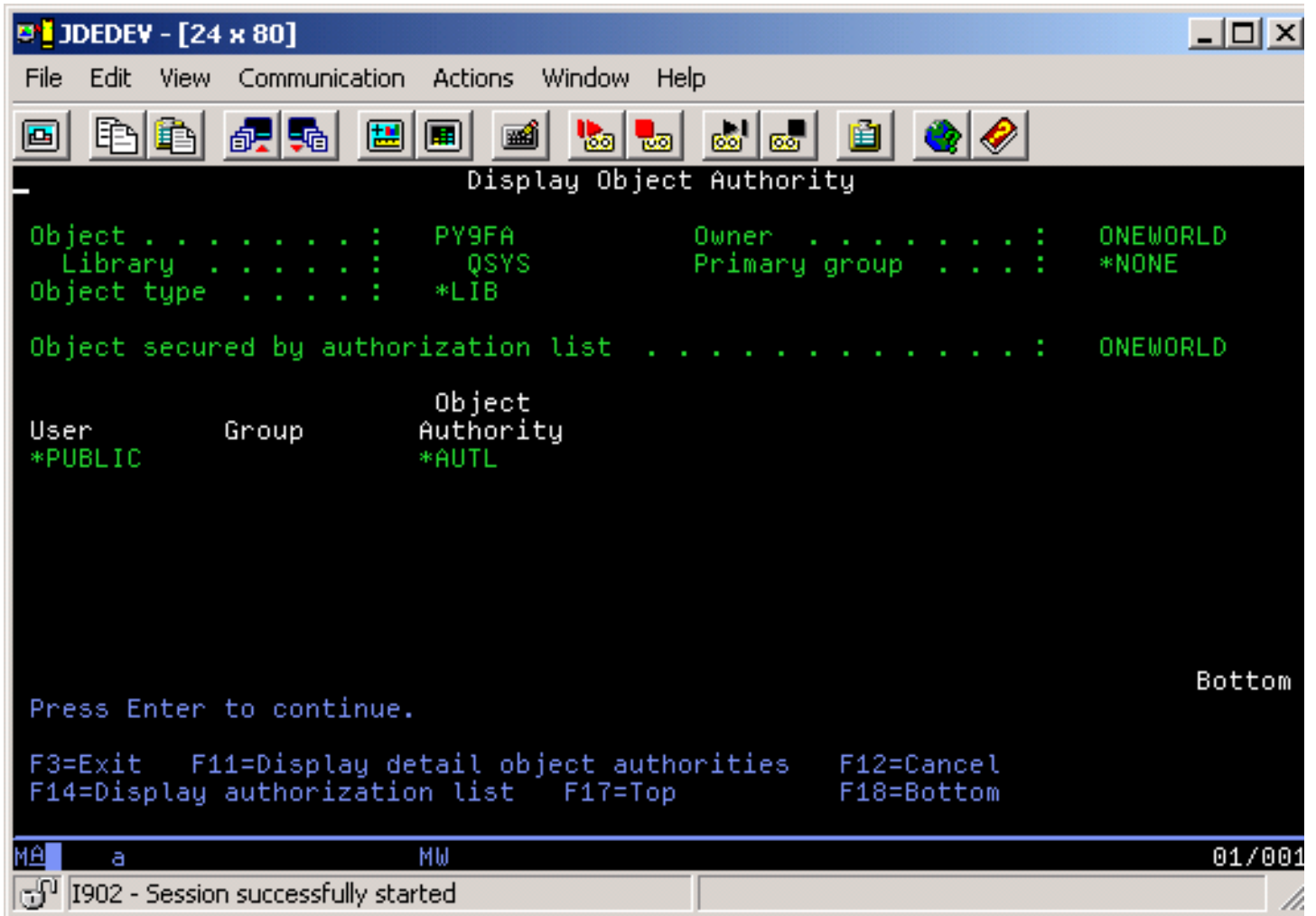
If you set up multiple instances of JD Edwards EnterpriseOne, the owner of each instance is the user profile that you entered in the JD Edwards EnterpriseOne User Profile parameter during the authority setup. If you set up a single instance of JD Edwards EnterpriseOne, the owner is ONEWORLD.

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.

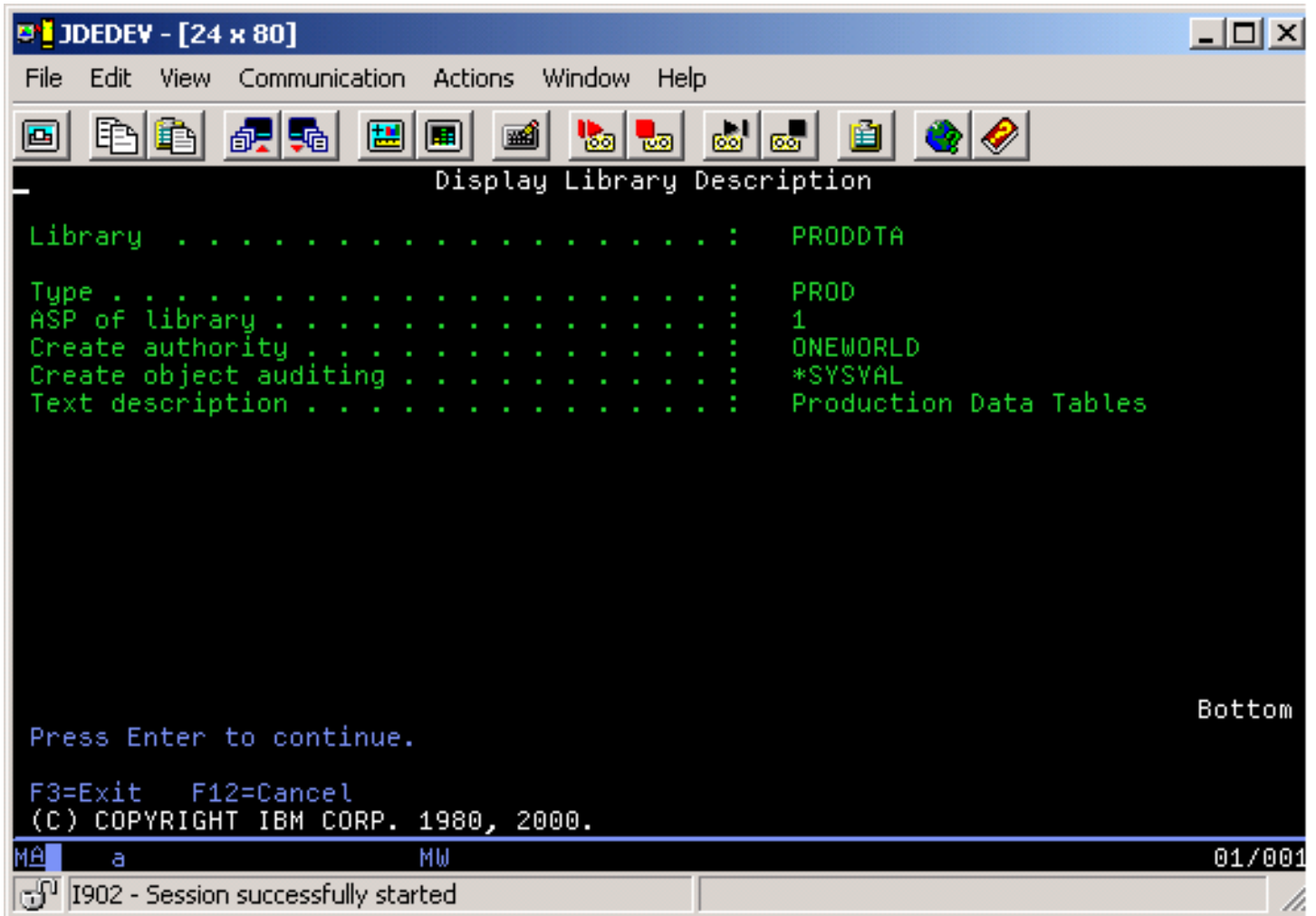
This is an example of the data source DSPOBJAUT:



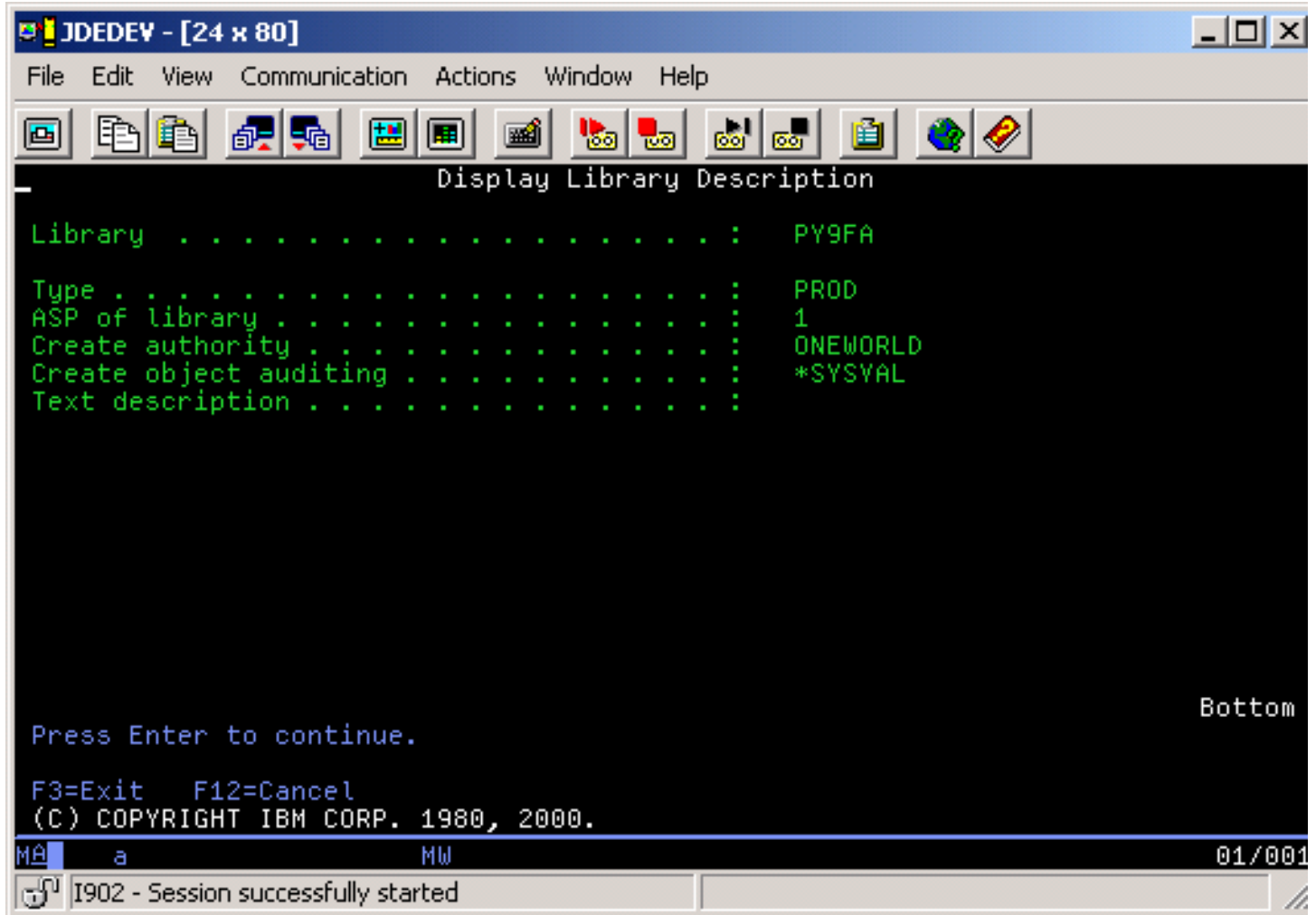
This is an example of the data source DSPOBJAUT:



This is an example of the data source DSPLIBD:



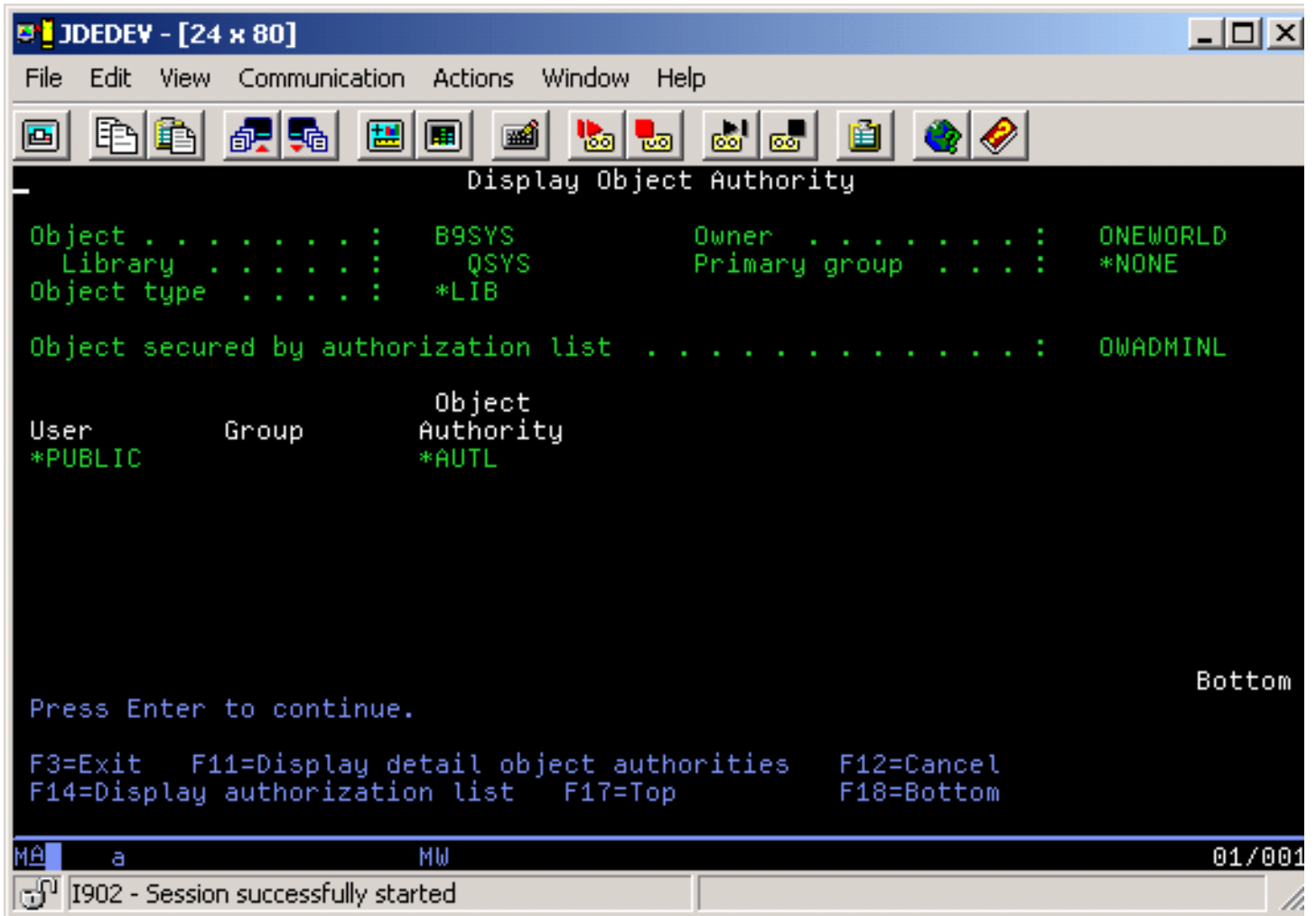
This is an example of the pathcode DISLIBD:



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:



This is an example of the system library DSPLIBD:

```

JDEDEV - [24 x 80]
File Edit View Communication Actions Window Help
Library Description
Library . . . . . : B9SYS
Type . . . . . : PROD
ASP of library . . . . . : 1
Create authority . . . . . : ONEWORLD
Create object auditing . . . . . : *SYSVAL
Text description . . . . . :
Press Enter to continue.
F3=Exit F12=Cancel
(C) COPYRIGHT IBM CORP. 1980, 2000.
a MW 01/001
I902 - Session successfully started
    
```

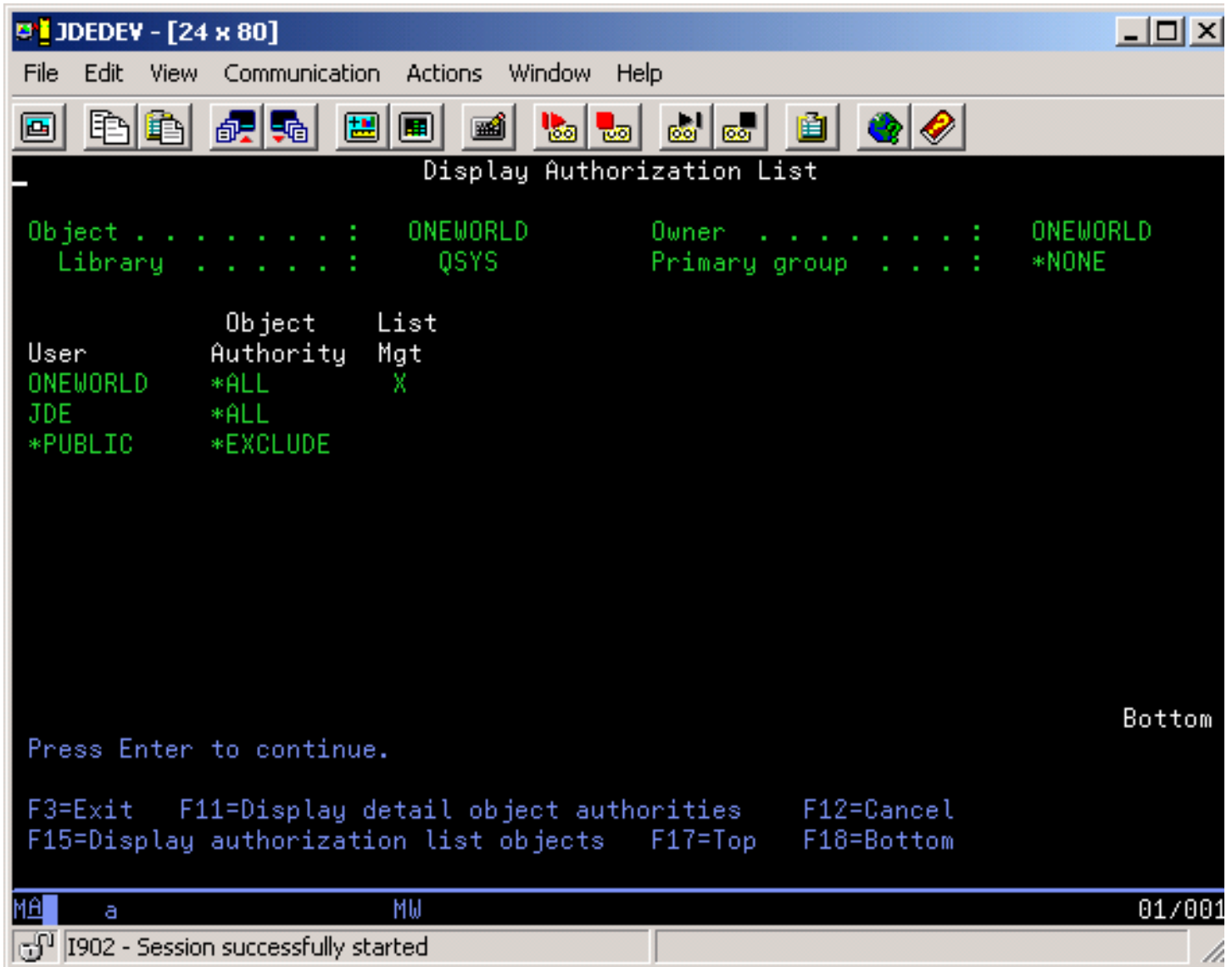
The authority changes for objects within system libraries that either contain the attributes CLLE or CLP or that share the same name. You can use commands to review the authority on these objects. The required parameters are object name, object type (*PGM or *CMD), and the name of the library in which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

Sample Results for Authorization Lists

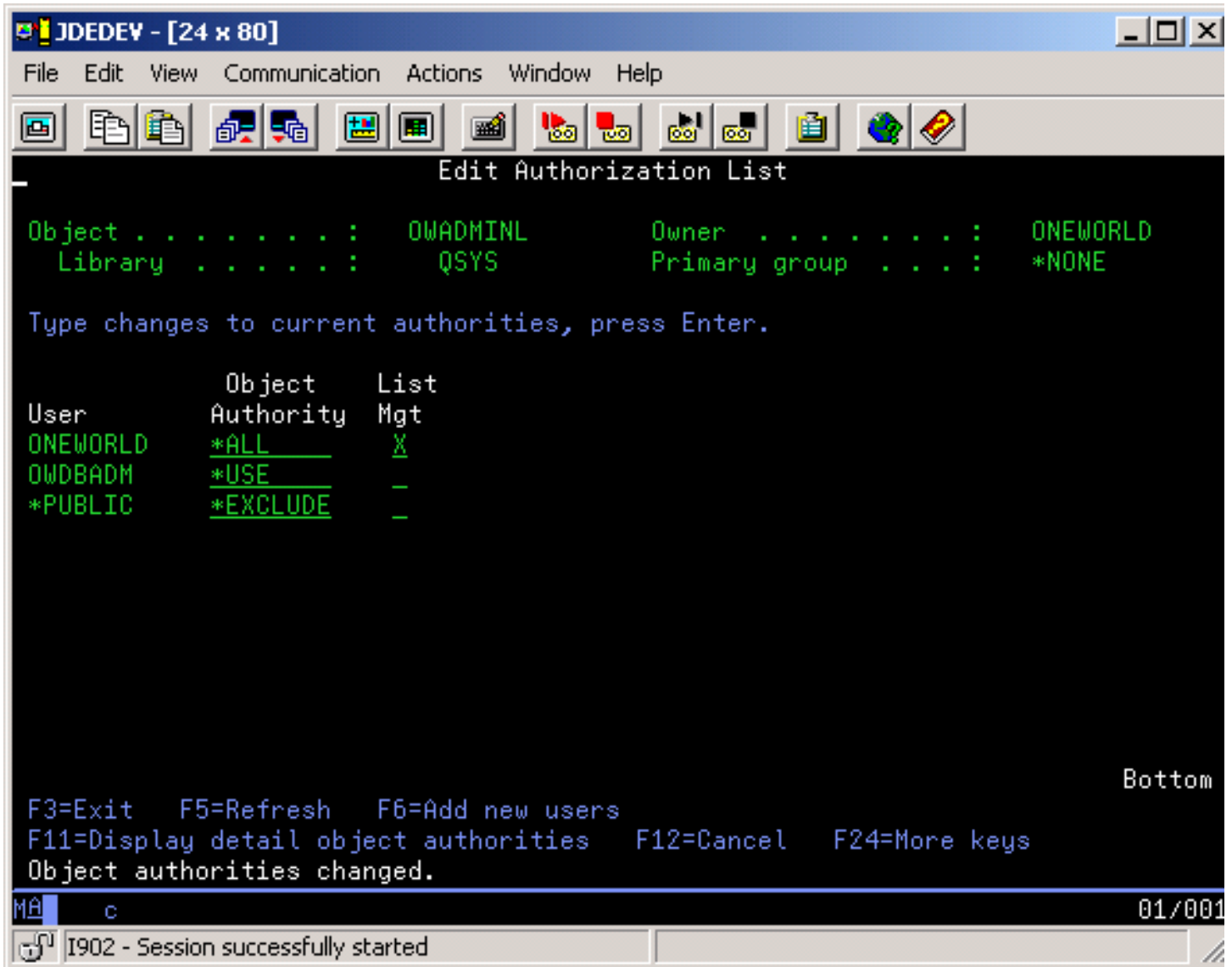
Use these commands to view the authorization list authorities. The name of the list is the only necessary parameter:

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

This is an example of DSPAUTL:



This is an example of DSPAUTL:



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.

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

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:
 - o USRAUTL
Enter a name that identifies the administrative authorization list.
 - o TYPE
 - o INILIB

5. Complete any additional fields, if necessary.
6. Press Enter.

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, CLRPC, 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.

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.

Field	Description
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.

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 . . . . . : QDFTJOB
  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 . . . . . : QDFTJOB
  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

Understanding Server Administration for UNIX and Linux

Oracle supports JD Edwards EnterpriseOne enterprise servers for UNIX operating systems on Oracle SPARC servers running Solaris, IBM Power Systems running IBM AIX, and HP Integrity servers running HP-UX. Oracle also supports JD Edwards EnterpriseOne enterprise servers running on x86-64 based hardware running Oracle Linux or RedHat Enterprise Linux. 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.

JD Edwards EnterpriseOne Directory Structure for UNIX and Linux

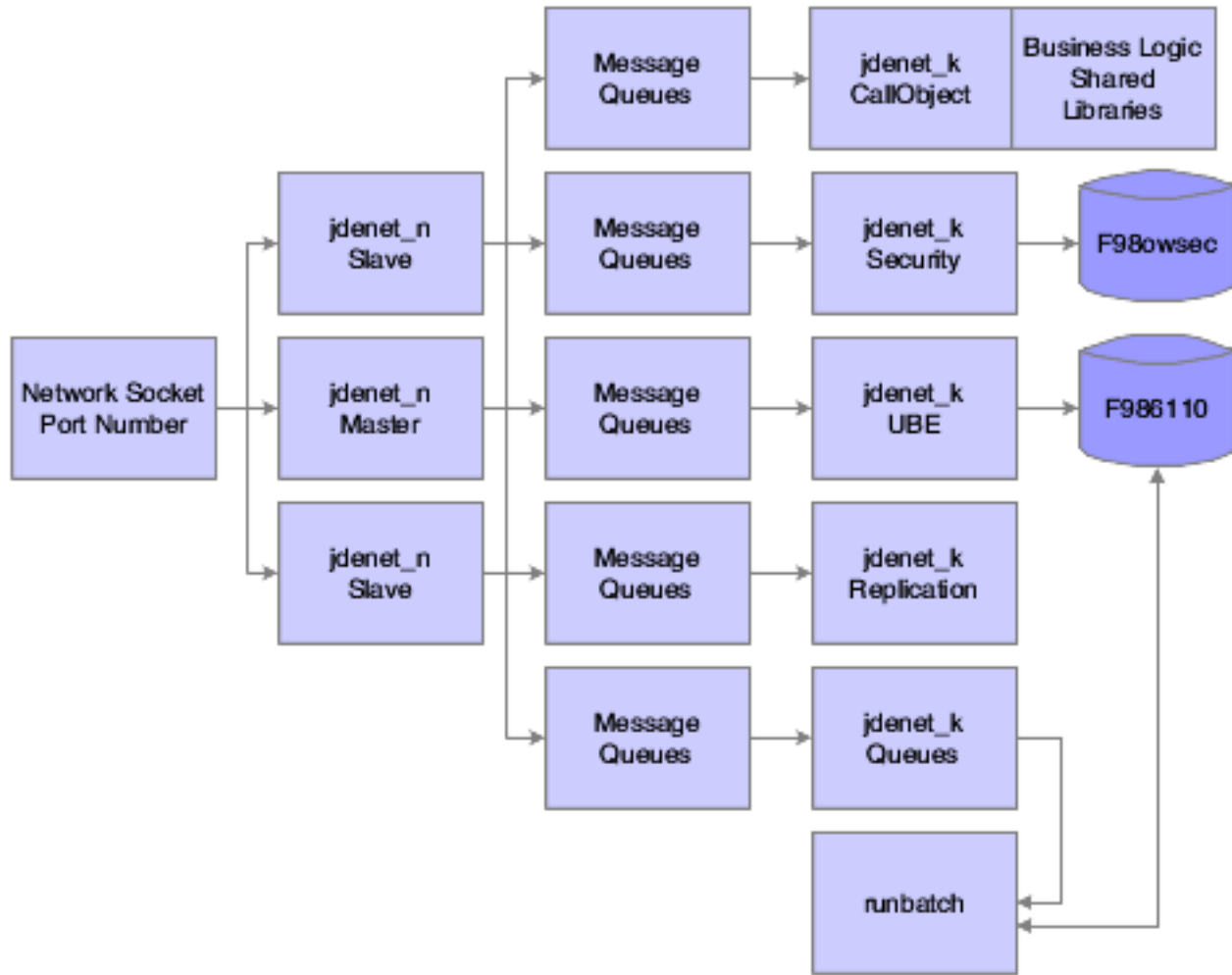
This is a list of directories that are shipped on the UNIX and Linux JD Edwards EnterpriseOne Server Installation CD. They should be installed under the JD Edwards EnterpriseOne base directory; for example, install them in /u01/JDEdwards/E910. Indented names indicate subdirectories of the directories, which are not indented.

Directory	Description
pathcode	<p>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 PD910 and DV910. These subdirectories are included:</p> <ul style="list-style-type: none"> • bin32/bin64 - Business function shared libraries. • Spec files - Binary data files in a JD Edwards proprietary format.
system	<p>The main directory for the system-level executables, shared libraries, C header files, libraries, and localization files. These subdirectories are included:</p> <ul style="list-style-type: none"> • bin32 - System-level executables, scripts, and log files such as jdenet_n.log and startstop.log. • classes - java.jar and jdelog.properties files. • include - System-level C header files. • includev - System-level C header files provided by third-party vendors, such as Vertex. • jde.ini - configure the jde.ini to point to the <Path to JVM library>

Directory	Description
	<p>[JDE JVM]</p> <p>InProcessJVMHome=<Path to JVM library></p> <p>Below is the <Path to JVM library> by platform:</p> <p>Oracle LINUX</p> <p><JAVA_INSTALL_HOME>/jre/lib/i386/server/libjvm.so</p> <p>Oracle Solaris</p> <p><JAVA_INSTALL_HOME>/jre/lib/sparc/server/libjvm.so</p> <p>HP-UX Itanium</p> <p><JAVA_INSTALL_HOME>/jre/lib/IA64N/server/libjvm.so</p> <p>IBM AIX</p> <p><JAVA_INSTALL_HOME>/jre/lib/ppc/j9vm/libjvm.so</p> <ul style="list-style-type: none"> • lib - System-level shared libraries and export files. • libv32 - System-level shared libraries provided by third-party vendors. • locale - icu locale configuration. • resource - PDF resource files. • Xts Repository - XML style sheets.
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	<p>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:</p> <ul style="list-style-type: none"> • 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. Each subdirectory contains a link .log file for the link step of the BSFN build.) • 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.)
SharedScripts	Location for Shell Scripts. Includes enterpriseone.sh script for configuring environment variables.

JD Edwards EnterpriseOne Architecture and Process Flow for UNIX and Linux

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



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. The `maxNumberOfSemaphores` setting in the `JDE.INI` file allows the default IPL key range to be increased over the default of 1000. The KEY displayed by the command `ipcs` will be displayed in hex, while the value for `startIPCKeyValue` is in decimal. For example, an IPC value of 5000 in the `JDE.INI` file will show up as a KEY of `0x1388` when running `ipcs`.

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.

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.

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). Starting with Oracle Database 12.1.0.1, a tnsnames.ora entry must be manually added for the pluggable (PDB) database that must match the OMDATB field from the F98611 table. This will have its own SID that is different than the SID for the container database (CDB). For more details, see the "Pluggable Databases in Oracle 12cR1 and Higher" section of the *JD Edwards EnterpriseOne Applications Installation Guide for UNIX with Oracle*.

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 for 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.

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 or *.processes 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 or spfile. These paths are an example of where you might typically find these files:

```
/u01/app/oracle/product/12.1.0/dbhome/dbs/init.ora  
/u01/app/oracle/product/12.1.0/dbhome/dbs/spfileORASID.ora
```

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 default user ID, the user ID is jde910.

2. Enter these commands:

- o `cd $EVRHOME/log`

This command moves the user's current directory to the default log directory.

CAUTION: The administrator can override this default location by changing the `$SYSTEM/classes/jdelog.properties` file and the `[DEBUG]` section in the `JDE.INI` file.

- o `rm *log*`

This command deletes the log files in the directory.

Note: Use extreme care when you enter this command. Running this command in the wrong directory can cause severe problems on the system.

- o `RunOneWorld.sh`

This script starts the JD Edwards EnterpriseOne system.

3. Sign off the system.

Starting the Enterprise Server for HP-UX Automatically

To start the enterprise server for HP-UX automatically:

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

The script should contain only these:

```
#!/sbin/sh
/bin/su - jde910 -c ` $SYSTEM/bin32/RunOneWorld.sh `
```

The value `jde910` is the name of the user who owns the shell script `$SYSTEM/bin32/RunOneWorld.sh`. Make sure that no interactive commands appear in the `jde910` user's `~/.profile` (or `~/.bash_profile`), and that `RunOneWorld.sh` has all necessary permissions for execution.

2. Using this command, create a soft link named `S995jde` to the `jdedwards` script in the directory named `/sbin/rc2.d`.

```
ln -s /sbin/init.d/jdedwards /sbin/rc2.d/S995jde
```

3. Verify that this 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
```

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.jde` in `/etc` with all necessary permissions for execution.

The script should contain only these:

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

```
/bin/su - jde910 -c `$$SYSTEM/bin32/RunOneWorld.sh`
```

The value `jde910` is the name of the user who owns the shell script `$$SYSTEM/bin32/RunOneWorld.sh`. Make sure there are no interactive commands in the `jde910 ~/.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`:

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

3. Verify that this 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**.

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
```

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 - jde910 -c `$$SYSTEM/bin32/RunOneWorld.sh`
```

The value `jde910` is the name of the user who owns the shell script `$$SYSTEM/bin32/RunOneWorld.sh`. Make sure there are no interactive commands in the `jde910 ~/.profile` (or `~/.bash_profile`), and that `RunOneWorld.sh` has all the necessary permissions for execution.

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
portttest userID password environment
```

The portttest 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.

Understanding Java Runtime Engine Installation Issue on Unix

With JD Edwards 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/elt_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`.

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.

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
```

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 smh to help in setting up a printer; AIX uses a tool called SMIT; Solaris uses a tool called Admintool; and Linux uses CUPS through utilities like lpadmin or system-config-printer. 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, Linux, or Solaris documentation.

Note:

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

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 from the command line for UNIX or Linux.
- Schedule reports from the command line for UNIX or Linux.

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:

```
JDE920 (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).

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.

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
```

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|-fe|-FE|-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 or -p	Prompts for user/password information.
-F or -f	Reads user/password information from the plain text file that is specified in passfile (FilePath).
-Fe or -fe	Reads user/password (Encrypted) information from the plain text file that is specified in FilePath.
-D or -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).
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.

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
```

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).
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.

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
```

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
```

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.

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.

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-- jde910 jde910 jdeblc.xdb  
-rw-r--r-- jde910 jde910 jdeblc.ddb
```

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-- jde910 jde910 b4200100.c  
-rw-r--r-- jde910 jde910 b4200100.h
```

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----- jde910 jde910 RunOneWorld.sh  
-r-xr----- jde910 jde910 EndOneWorld.sh
```

Setting jde.ini File Security

CAUTION: 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----- jde910 jde910 JDE.INI
```

Denying write access to the user jde910 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 jde910 (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.

Working with HP-UX and Solaris Kernel Parameter Settings

Beginning in Solaris 10, Solaris had a major change in the way kernel and IPC parameters are handled. Many of these parameters should no longer be changed through the `/etc/system` file. Some kernel settings have been removed, and some are now obsolete and should now be changed through the Solaris resource controls facility. A complete 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 Enterprise 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 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, 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. To see the resource control limits for a given user, sign on to that user and run the following commands:

```
prctl $$  
  
id -p  
projects -l
```

Other resource control groups besides "default" can be configured, but for the purposes of these examples, the default group is shown. See the "Oracle® Solaris Tunable Parameters Reference Manual".

More examples of common tuning settings that may be set include:

```
projmod -a -K "process.max-msg-messages=(privileged,16384,deny)" default  
projmod -a -K "process.max-msg-qbytes=(privileged,1048576,deny)" default  
projmod -a -K "project.max-msg-ids=(privileged,2048,deny)" default  
projmod -a -K "project.max-sem-ids=(privileged,15320,deny)" default  
projmod -a -K "project.max-shm-ids=(privileged,1024,deny)" default  
projmod -a -K "project.max-shm-memory=(privileged,4294967296,deny)" default
```

These settings are stored in the `/etc/project` file, and correspond to kernel parameter settings with different names previously set in the `/etc/system` file (see sections 3.7.1, 3.7.2, and 3.7.3 below).

CAUTION: Some of these corresponding `/etc/system` settings that are obsolete, can still be set in the `/etc/system` file despite being obsolete. Other settings in the `/etc/system` file have been removed starting with Solaris 10, and are ignored by the operating system. Setting a corresponding obsolete setting to different values in both the `/etc/system` and `/etc/project` files may lead to unexpected results.

The kernels for HP-UX and Solaris 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 Management Homepage (`smh`) 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. Changes made using `projmod` require the user to stop all processes running under the user's id, log off, and log back onto the server to take effect. For the E1 Enterprise Server, this includes shutting down the Server Manager Agent. 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 `smh` 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 `smh` 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 `smh`. 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
<code>jdenet_n</code>	The maximum number of <code>jdenet_n</code> (net) processes that can be created for an instance JD Edwards EnterpriseOne server software running on the system. This is controlled by the <code>maxNetProcesses</code> parameter in the <code>JDENET</code> section of the <code>JDE.INI</code> file for each instance of JD Edwards EnterpriseOne.
<code>jdenet_k</code>	The maximum number of <code>jdenet_k</code> (kernel) processes that can be created for an instance of JD Edwards EnterpriseOne server software running on the system. This is controlled by the <code>maxKernelProcesses</code> parameter in the <code>JDENET</code> section of the <code>JDE.INI</code> file for each instance of JD Edwards EnterpriseOne. Note that the <code>maxNumberOfProcesses</code> parameters in the <code>JDENET_Kernel_Def</code> sections do not matter here.

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/project.max-msg-ids	The value of msgmni (project.max-msg-ids) 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: $1 + jdenet_n + 2 \times jdenet_k + (\text{max number of concurrent runbatch, runube, and runprint processes})$
msgtql/process.max-msg-messages	The value of msgtql (process.max-msg-messages) 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 $10 \times \text{msgmni}$ 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 $\text{msgtql} + 2$ should be used. If the value of msgmap is less than the value of $\text{msgtql} + 2$, attempts to create a message queue or to send a message might fail. Note: This parameter was removed in Solaris 10.
msgmnb/process.max-msg-qbytes	The value of msgmnb (process.max-msg-qbytes) 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 $\text{msgseg} \times \text{msgssz}$. For JD Edwards EnterpriseOne, a value of 32768 is reasonable. You can set a larger value as long as the product of $\text{msgseg} \times \text{msgssz}$ 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 $\text{msgmax} = \text{msgmnb}$. The minimum value is 1024. Additional requirements of this parameter might increase the value of msgmax. Note: This parameter was removed in Solaris 10.

Note: Not all parameters are configurable in every operating system release.

Inside the HP-UX, messages in message queues reside in message segments. These parameters 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.

Parameter	Description
	Note: This parameter was removed in Solaris 10.
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. Note: This parameter was removed in Solaris 10.

Semaphores

Note: Not all parameters are configurable in every operating release.

These definitions apply to semaphores:

Parameter	Description
sema	This value must be 1. System-V style message queues are valid.
semnmi/project.max-sem-ids	The value of semnmi (project.max-sem-ids) 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 semnmi + 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.
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. Note: This parameter was removed in Solaris 10.
semnmu	The value of semnmu 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:

Parameter	Description
	<p>$1 + \text{jdenet_n} + \text{jdenet_k} + \text{maximum number of runbatch processes} + \text{maximum number of runprint processes} + \text{maximum number of runube processes}$</p> <p>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.</p> <p>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.</p> <p>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.</p> <p>Note: This parameter was removed in Solaris 10.</p>
semume	<p>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.</p> <p>Note: This parameter was removed in Solaris 10.</p>
semmsl/process.max-sem-nsem	<p>The value for semmsl (process.max-sem-nsem), 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.</p>

Shared Memory

These definitions apply to shared memory:

Parameter	Description
shmem	The shmem value must be 1 to enable shared memory.
shmmax/project.max-shm-memory	The value of shmmax (project.max-shm-memory) 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/project.max-shm-id	The value of shmmni (project.max-shm-id) 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.

Parameter	Description
	Note: This parameter was removed in Solaris 10.

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.

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.

Working with Linux Kernel Parameter Settings

This section provides an overview of Linux kernel parameter settings.

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 Oracle Linux and RedHat 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.

IPC Resources

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

Parameter	Description
kernel.sem	<p>This setting controls these four different semaphore limits:</p> <ul style="list-style-type: none"> • 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). <p>For JD Edwards EnterpriseOne, you might need to increase the first value, semaphores per array, particularly if you increase the value of <code>maxNumberOfSemaphores</code> in the <code>jde.ini</code> file. Some database products also require that the fourth value, number of semaphore arrays, be increased from the default value.</p>
kernel.shmmax	<p>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.</p>
kernel.msgmax	<p>This parameter defines the maximum size of a message. The recommendation for JD Edwards EnterpriseOne is 65535.</p>
kernel.msgmnb	<p>This parameter defines the maximum number of bytes on a message queue. The recommendation for JD Edwards EnterpriseOne is 65535.</p>
kernel.msgmni	<p>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:</p> $1 + jdenet_n + 2 \times jdenet_k + (\text{max number of concurrent runbatch, runube, and runprint processes})$

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 a minimum of 32768, and this number might need to be increased to 6815744 for larger installations.

Example: `/etc/sysctl.conf`

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

```
fs.file-max = 6815744
kernel.shmmax = 4398046511104
kernel.shmall = 1073741824
kernel.shmmni = 4096
kernel.sem = 1024 32000 100 1024
kernel.msgmax = 65535
kernel.msgmnb = 65535
kernel.msgmni = 2878
```

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.

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.

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.

Parameter	Description
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.

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.

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.

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

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.

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.

Note:

- *JD Edwards EnterpriseOne Tools Server Manager Guide*
- *"Adding an Environment" in the JD Edwards EnterpriseOne Tools System Administration Guide* .

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.

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`, `smh`, `admintool`, or `useradd`.

Note: Although you can run multiple instances of the JD Edwards EnterpriseOne server using the same UNIX or Linux user 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` 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 to reference the new directory path in which you will create the new JD Edwards EnterpriseOne instance.

For example:

Original `.psft` file:

```
if [ -f /u01/JDEdwards/E910/SharedScripts/enterpriseone.sh ] ; then . /u01/  
JDEdwards/E910/SharedScripts/enterpriseone.sh
```

New `.profile` file:

```
if [ -f /u02/JDEdwards/E910/SharedScripts/enterpriseone.sh ] ; then . /u02/  
JDEdwards/E910/SharedScripts/enterpriseone.sh
```

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

For example, type these:

```
mkdir -p /u02/JDEdwards/E910
```

6. 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/E910/system /u02/JDEdwards/E910
cp -R /u01/JDEdwards/E910/ini /u02/JDEdwards/E910
cp -R /u01/JDEdwards/E910/DV910 /u02/JDEdwards/E910
```

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 will corrupt specification files.

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

```
mkdir -p /u02/JDEdwards/E910/log
```

8. 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/E910/log/ jdedebug.log
JobFile=/u02/JDEdwards/E910/log/jde.log
```

```
[INSTALL]
B9=/u02/JDEdwards/E910
```

```
[BSFN BUILD]
BuildArea=/u02/JDEdwards/E910/packages
```

9. 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
```

10. From the client workstation JDE.INI file, change the serviceName parameters to match those of the server JDE.INI file.
11. Ensure you are signed on using the new user ID.
12. Perform a fresh installation of the Server Manager Agent on the Enterprise Server machine. This is required since the earlier Server Manager Agent, which is managing the original Enterprise Server, is running with a different UserId and cannot be used to manage the newly created Enterprise Server.
13. Once the Installation of the Server Manager Agent is completed, the Server Manager Agent will be running and will start showing up in the Server Manager Console.
14. The newly created Enterprise Server can be registered in the new Server Manager Agent and can be used for managing the new Enterprise Server (Configuration, Logs, Runtime Metrics, etc).
15. This completes the setup required from the Server Manager side for managing the new Enterprise Server.

4 Administering the Windows Server

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.

JD Edwards EnterpriseOne Directory Structure for Windows

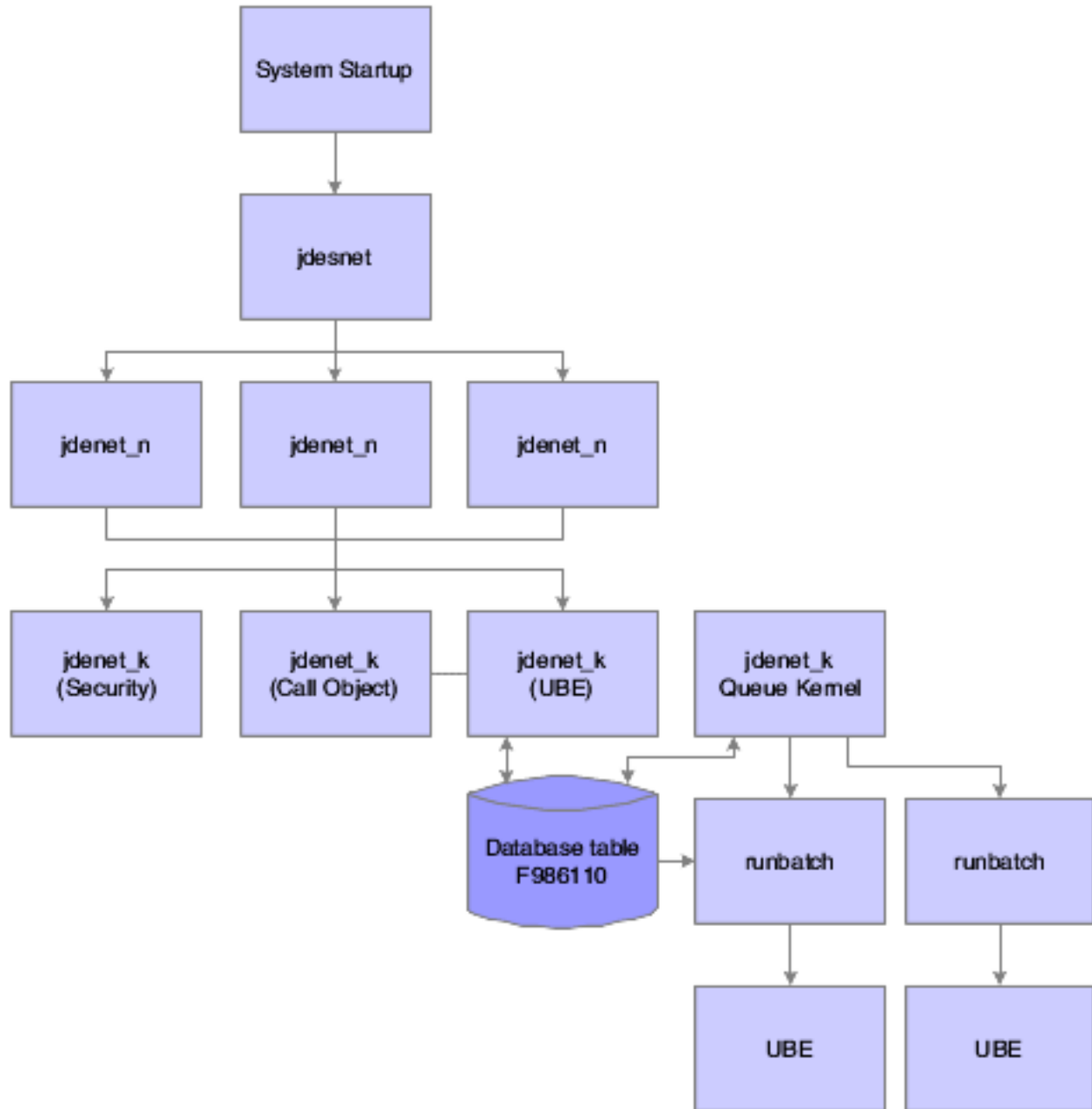
This table lists the directories that are copied to the Windows enterprise server when Oracle's JD Edwards EnterpriseOne is installed. They should be installed under the JD Edwards EnterpriseOne base directory (such as z:\JDEdwards\E920\ddp). Indented names indicate subdirectories of the directories.

Directory	Description
pathcode	<p>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 PD920 and DV920. These subdirectories are included:</p> <ul style="list-style-type: none"> • bin32, which includes business function shared libraries. • spec, which includes specification files. These binary data files are in a JD Edwards proprietary format.
system	<p>The main directory for the system-level executables, shared libraries, C header files, libraries, and localization files. These subdirectories are included:</p> <ul style="list-style-type: none"> • 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	<p>The directory to which all .PDF file output for reports is written.</p>
log	<p>The directory to which jde_xxx.log and jdedbug_ xxx.log files are written.</p>

Directory	Description
packages	<p>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:</p> <ul style="list-style-type: none"> • 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.

JD Edwards EnterpriseOne Architecture and Process Flow for Windows

These host server processes perform the indicated actions:



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\jdesqueue.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.

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 *DB2 for Linux, UNIX, and Windows database* 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 *DB2 for Linux, UNIX, and Windows database* 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.

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.
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.

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.

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.

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      JDEEDWARDS_ENTERPRISEONE_920_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.

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) (Prior to Tools Release 9.2.2).

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_920_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_920_SP1
SecurityServerSCP	Security server	JDEDWARDS_ ENTERPRISEONE_920_SP1
LockManagerSCP	Lock manager	JDEDWARDS_ ENTERPRISEONE_920_SP1
UnifiedLogonServerSCP	Unified logon server	JDEDWARDS_ ENTERPRISEONE_920_SP1

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.

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.

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.

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.

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.

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.

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.

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:
 - o User name
 - o Full Name
 - o Description
 - o Password
 - o Confirm Password
5. Complete these options, as appropriate for the installation:
 - o User must change password at next logon.

- User cannot change password.
 - Password never expires.
 - Account disabled.
6. Click Create.
 7. Click Cancel.

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.

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.

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.

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 E920 is JDE 920 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.

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 E920 is JDE 920 Network.
2. From the Action menu, click Start.
3. Use the Windows Task Manager to ensure that these processes are running:
 - jdesnet.exe.

- o jdenet_k.exe processes. (None, one, or more might exist.)

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 9.20 is JDE 920 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:

- o jdesnet.exe
- o jdenet_n.exe
- o jdenet_k.exe
- o runbatch.exe
- o ipcsrv.exe

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.

Uninstalling the Network Service

To uninstall the Network services:

Run this program from the \system\bin32 directory:

```
jdesnet -u
```

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.

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\E920\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.

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.

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.

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.

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 output 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
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 Reader.

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).
-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.

Parameter	Description
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.

Example: Running Reports from the Command Line for Windows

This example lists commands for executing a batch process report:

```
cd \JDEdwards\E920\ddp\system\bin32
runube KL5595218 KL5595218 PROD R0006P XJDE0001 QBATCH Interactive
Print Delete printer_1
```

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.

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
```

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.

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.

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.

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.

JDE.INI File (Enterprise Server) Security

CAUTION: 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.

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.

Note:

- "Working with the Server Manager Management Console" in the *JD Edwards EnterpriseOne Tools Server Manager Guide* .

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\E920 directory tree.

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\E920_2\log\jdedebug.log
[DEBUG]	JobFile=	Name of the log file that contains log data.	z:\JDEdwards\E920_2\log\jde.log

Section of server JDE.INI file	Parameter	Purpose	Example Value
[INSTALL]	StartServicePrefix=	Prefix that is used for names of the JD Edwards EnterpriseOne network and queue services.	Instance 2
[INSTALL]	B9=	Base directory of the JD Edwards EnterpriseOne installation.	z:\JDEdwards\E920_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=

Note: *Setting Up the Network Service.*

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 uuidgen-? 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.

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 920. The default value produces the service names JDE 920 Network and JDE 920 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.

Parameter	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.

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:
 - o serviceNameListen=
 - o serviceNameConnect=

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:\E920\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.

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:\E920\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 Backing Up JD Edwards EnterpriseOne Tables

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.

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:\windows.
- Services file on c:\winnt\system32\drivers\etc.
- Registry export file.
- JD Edwards EnterpriseOne files in the root directory (c:\):
 - jdeapp.xdp
 - jdeauth.xda
 - jdemod.ddm
 - jdemod.xdm
 - jdsec.dds
 - jdsec.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 PD920, DV920, and PY920 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.

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
 - SY920
 - E920SYS
 - SVM920
 - JD Edwards EnterpriseOne data dictionary library: DD920.
 - JD Edwards EnterpriseOne Object Management Workbench library: OL920.
- (Release 9.2.6.0) All JD Edwards EnterpriseOne package libraries that are active:
 - Look in spec.ini of each of the pathcode(s) to retrieve the active package library that you must backup.
- All JD Edwards EnterpriseOne production libraries (This example is for pristine and production):
 - PD920
 - PS920
 - PRODDTA
 - PRSTDTA
- All JD Edwards EnterpriseOne business data libraries:
 - PRODDTA
 - CRPDTA
 - PRSTDTA
 - TESTDTA
- All JD Edwards EnterpriseOne control libraries:
 - PRODCTL

- CRPCTL
- TESTCTL
- PRSTCTL
- IFS (Integrated File System) libraries:
 - PD920
 - PY920
 - PS920
 - TS920
 - DV920
- IBM libraries that require backups:
 - QGPL
 - 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/E920 directory. For example, /u03/JDEdwards/E920/*.

- 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\E920 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, PSFT920) to a backup database.

Note: We recommend that you use the backup tool provided by the RDBMS vendor.

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 SY920.

- 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 obj920.

- F00165
- F9860
- F9861
- F9862
- F9863
- F9865

Data Dictionary Tables

The Object Owner for the Data Dictionary tables is dd920.

- F00165
- F9200
- F9202
- F9203
- F9207
- F9210
- F9211

Server Map Tables

The Object Owner for Server Map tables is svm920.

- 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 - PS920: 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 - PD920: PD920
- Versions - PY920: PY920
- Versions - DV920: DV920
- Versions - PS920: PS920

The Versions tables are listed:

- F983051
- F98306

Central Objects

The Object Owners of the Central Objects tables are:

- Central Objects - PD920: pd920
- Central Objects - PY920: py920
- Central Objects - DV920: dv920
- Central Objects - PS920: PS920

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 - PS920: prstdta

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.

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.

Creating a Backup for

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
SY920	System library
SVM920	Server Map
OL920	Object Librarian
DD920	Data Dictionary
COPY920	Central Objects - Prototype
COPS920	Central Objects - PS920
COPD920	Central Objects - PROD
CODV920	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

Library name	Description
PRSTCTL	Pristine Control Tables
E920SYS	Server system library
JDEOW	JD Edwards Installation
PY920	Server modules - Prototype
PY920FA	Package Library - Prototype
PS920	Server modules - PS920
PS920FA	Package Library - PS920
PD920	Server modules - PROD
PD920FA	Package Library - PROD
DV920	Server modules - DEV
DV920FA	Package Library - DEV

2. Back up these IFS structure with the subdirectories:

Library name	Description
JDE920	Logging directory
E920SYS	Kernel spec and XML
PY920	Spec files for Prototype
PS920	Spec files for PSFT
PD920	Spec files for PROD

Library name	Description
DV920	Spec files for DEV
JD Edwards	Contains the spec files for each path code. \JDEdwards\PACKAGES\PY920FA\SPEC*.* \JDEdwards\PACKAGES\PS920FA\SPEC*.* \JDEdwards\PACKAGES\PD920FA\SPEC*.* \JDEdwards\PACKAGES\DV920FA\SPEC*.*

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.

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:
 - o When the process completes the transfer, click the View Logs button to review the transfer process.
 - o Run a SELECT statement to verify that the backup tables transferred to the new database with data.

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.

Restoring a Backup File for

To restore a backup file for *IBM i*:

Restore the libraries and IFS directories that you backed up from tape.

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:
 - o When the process completes the transfer, click the View Logs button to review the transfer process.
 - o Run a SELECT statement to verify that the backup tables transferred to the new database with data.

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.

6 Troubleshooting the Enterprise Server

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.

- `jddebug.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 `jddebug.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.

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 `jddebug` will be located in `DebugFile`. JD Edwards EnterpriseOne uses these naming conventions:

```
jddebug_process_ID
```

Where `jddebug` 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/E920/system/bin32` directory:

```
cd /u13/JDEdwards/E920/system/bin32
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/E920/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.

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) `jddebug.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 `jddebug.log` and then run a report that causes an error condition. Then you could rename the `jddebug.log` to `report.log`.

Another alternative is to add comment lines to the `jddebug.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 `jddebug.log`, these setting controls the location:

```
[DEBUG]
```

```
DebugFile=jddebug.log
```

For enterprise servers, the files generated by the `jddebug.log` will be located in `DebugFile`. JD Edwards EnterpriseOne uses these naming conventions:

```
jddebug_process_ID.log (jddebug_JobNumber.log for  
                        IBM i  
                        )
```

Where `jddebug` 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/E920/system/bin32 RunOneWorld.sh
```

and assuming that logging is enabled, the system places the log files in the `/u13/JDEdwards/E920/system/bin32` directory. Similarly, on a UNIX machine, if you start JD Edwards EnterpriseOne with this command:

```
cd /usr/JDEdwards /u13/JDEdwards/E920/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 jddebug.log on the Enterprise Server

JD Edwards EnterpriseOne processes create logs as `jddebug_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 `jddebug_123456.log`.

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 (`E920\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 `jddebug.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,
T0.MCRP01, 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.MCAL, 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).

Note:

- *Working with Servers*
- *Understanding Server Administration for IBMi.*
- *"Understanding Executable Files on the Workstation" in the JD Edwards EnterpriseOne Development Client Installation Guide for Oracle WebLogic Server (WLS) and WebSphere Application Server (WAS) Express .*

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
```

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: <ul style="list-style-type: none"> <input type="radio"/> Disabled (Default) <input type="radio"/> Enabled

4. Click the Apply button to save the changes.

Setting Up the Enterprise Server jddebug.log

To set up the enterprise server jddebug.log:

1. Locate the enterprise server jddebug.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	<p>A parameter controls whether the logging function is enabled. Valid values are:</p> <ul style="list-style-type: none"> ○ NONE — No Debug Logging - FILE — Enable Debug Logging - ○

5. Click the Apply button to save the changes.

Note:

- *JD Edwards EnterpriseOne Tools Server Manager Guide*

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=	<p>A parameter that specifies the level of UBE debug logging. Valid values are:</p> <ul style="list-style-type: none"> ○ 0 (default): No error messages. ○ 1: Warnings and high-level information. ○ 2: Section-level messages (plus Level 1 messages)

Setting	Purpose
	<ul style="list-style-type: none"> ○ 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=	<p>A parameter that specifies whether the batch_report.log file will be saved. Valid values are:</p> <ul style="list-style-type: none"> ○ 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 (E920\PrintQueue).

4. Save the changes and close the jde.ini file.

Troubleshooting the Enterprise Server

This section discusses how to:

- Troubleshoot general problems.
- Troubleshoot communication problems.
- Troubleshoot server map problems.

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 PD920 or DV920.
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++.

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:
 - o 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.
 - o 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>
```

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.

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.

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 Out-of-Memory 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 `jededebug.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.

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.

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.

Shown below are each server group defined within the management domain. The user group permits configuring a unique set of granted permissions for each server group. For reference these permissions are:

clearCache - Permit Clearing JDBJ Caches

Permits the user to clear the JDBJ caches that are maintained with the EnterpriseOne web products.

daDriverInstance - Data Access Driver Instance Management

This permission is required to manage a Data Access Driver. This permission grants the authority to create new Data Access Drivers, remove (uninstall) existing Data Access Drivers, and configure Data Access Drivers. This permission also permits changing the tools release of a corresponding server.

dataAccessServerInstance - Data Access Server Instance Management

This permission is required to manage an EnterpriseOne Data Access Server. This permission grants the authority to create new Data Access servers, remove (uninstall) existing Data Access servers, configure Data Access servers, and start/stop Data Access servers. This permission also permits changing the tools release of a corresponding server.

enterpriseServerDeveloper - Enterprise Server Developer

This permission is required to create diagnostics information for EnterpriseOne enterprise server CallObject and UBE processes. This permission grants the authority to create Memory, CPU and All diagnostics. This will also grant the authority to start, dump, parse and stop JADE.

enterpriseServerInstance - Enterprise Server Instance Management

This permission is required to manage an EnterpriseOne enterprise servers. This permission grants the authority to create new enterprise servers, remove (uninstall) existing enterprise servers, configure enterprise servers, and start/stop enterprise servers. This permission also permits changing the tools release of a corresponding server.

viewGroupMembers - View Group Members

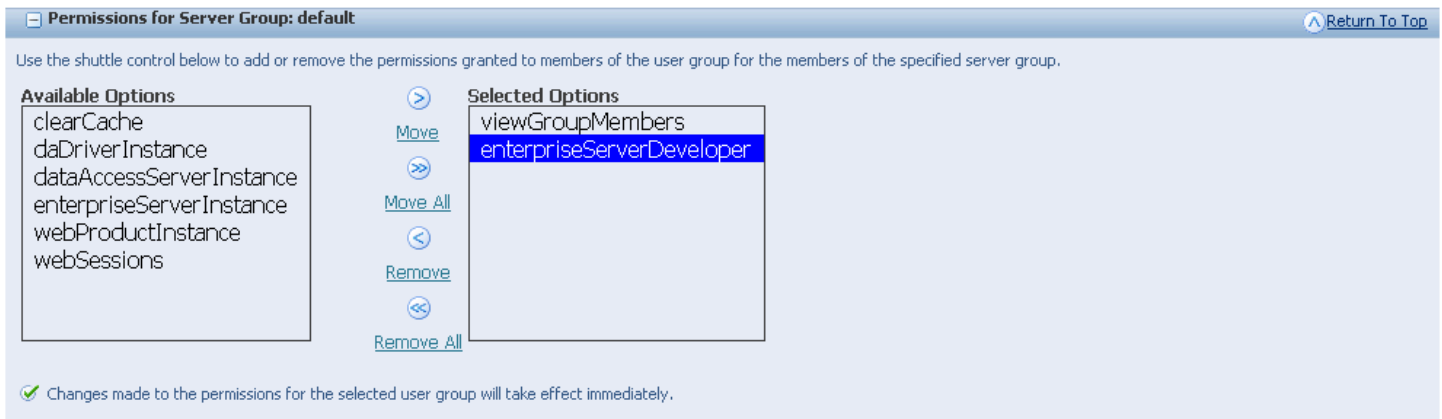
This permission grants the authority to view the EnterpriseOne servers that are members of a server group. Without this permission the user will not see the group members in the management console. This permission must be explicitly granted to each desired server group; no other permission implies or inherits this permission.

webProductInstance - Web Product Instance Management

This permission is required to manage an EnterpriseOne web products including HTML Servers, PIMSync, Real Time Events Server, SBF servers, and the corresponding application servers (Oracle Application Server and IBM WebSphere). This permission grants the authority to create new web product instances, remove (uninstall) existing web product instances, configure web products, register/de-register application servers, configure application servers, and start/stop of both application servers and web products. This permission also permits changing the tools release of a corresponding server.

webSessions - Web Product User Session Management

Permits the user to manage web product user sessions. This includes terminating the OWVirtual sessions, terminating user sessins (including any running OWVirtual sessions, broadcasting a message to OWVirtual clients, and temporarily disabling logins to a web product.



Permissions for Server Group: default [Return To Top](#)

Use the shuttle control below to add or remove the permissions granted to members of the user group for the members of the specified server group.

Available Options	Selected Options
clearCache	viewGroupMembers
daDriverInstance	enterpriseServerDeveloper
dataAccessServerInstance	
enterpriseServerInstance	
webProductInstance	
webSessions	

Changes made to the permissions for the selected user group will take effect immediately.

Evaluating EnterpriseOne Server Performance

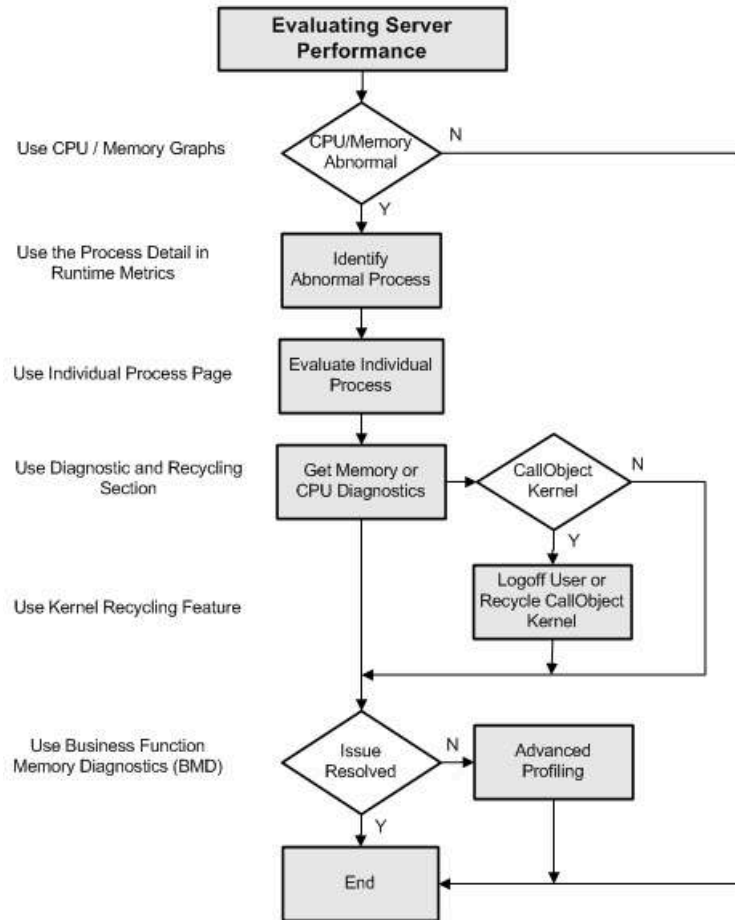
Kernel Resource Management is a set of diagnostic tools implemented in Server Manager 8.98.2.0 that utilizes historical data captured at specific intervals. You can use KRM to drill down through different levels of detail information and isolate the process, user, and attribute to determine the critical 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:



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.

EnterpriseOne Enterprise Server: E812EnterpriseServer

Available Log Files

General

Version
8.98.3.0

Status
Running

Software Component Version
EnterpriseOne Enterprise Server 100316_Release

Instance Properties

Install Location
/u01/jdedwards/e812

Instance Name
[E812EnterpriseServer](#)

CallObject Kernel Memory Limit
Unknown

Runbatch Memory Limit
Unknown

Instance Properties

Install Location
/u01/jdedwards/e812

Instance Name
[E812EnterpriseServer](#)

CallObject Kernel Memory Limit
4098 MB

Runbatch Memory Limit
4169 MB

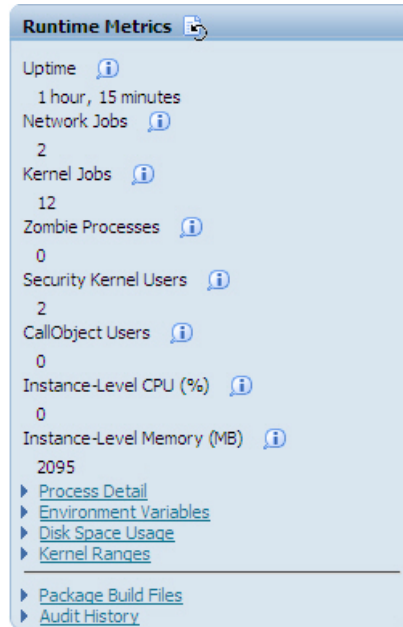
What constitutes a normal CPU consumption is entirely dependent on the vendor sizing of the EnterpriseOne instance. As a rule of thumb, high sustained CPU usage could indicate under-sized hardware. Determining abnormal CPU usage is challenging because it depends on the load on the server and average 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 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.



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.

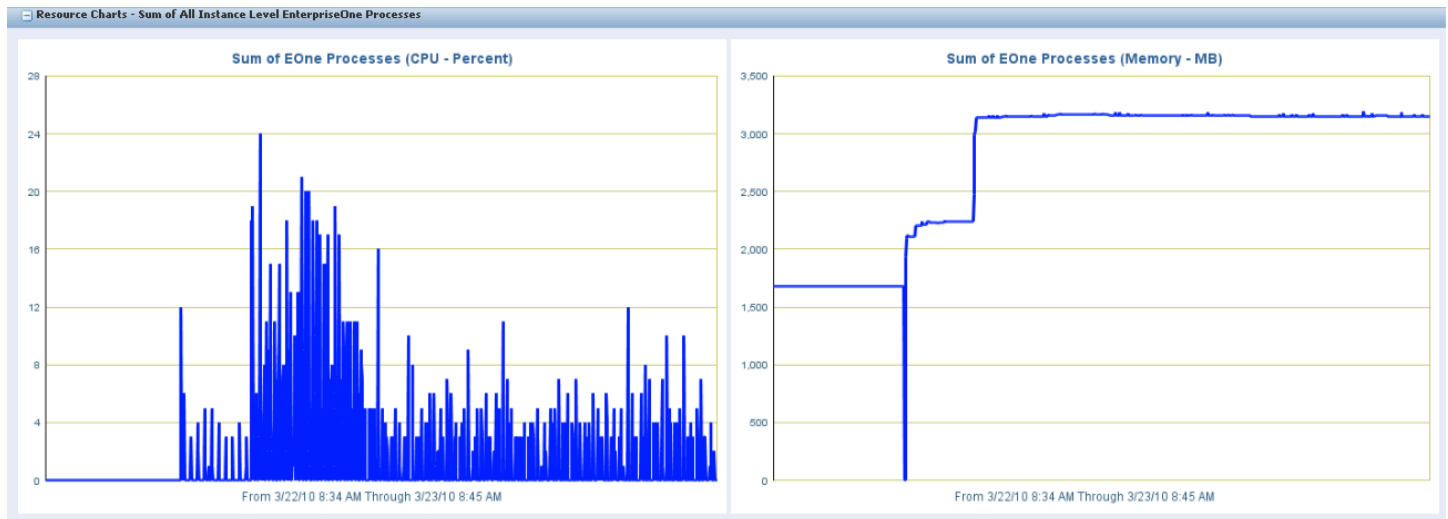
These are the navigation steps to access the instance-level summary page:

- 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).



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-hand side of the Server Manager screen.

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:

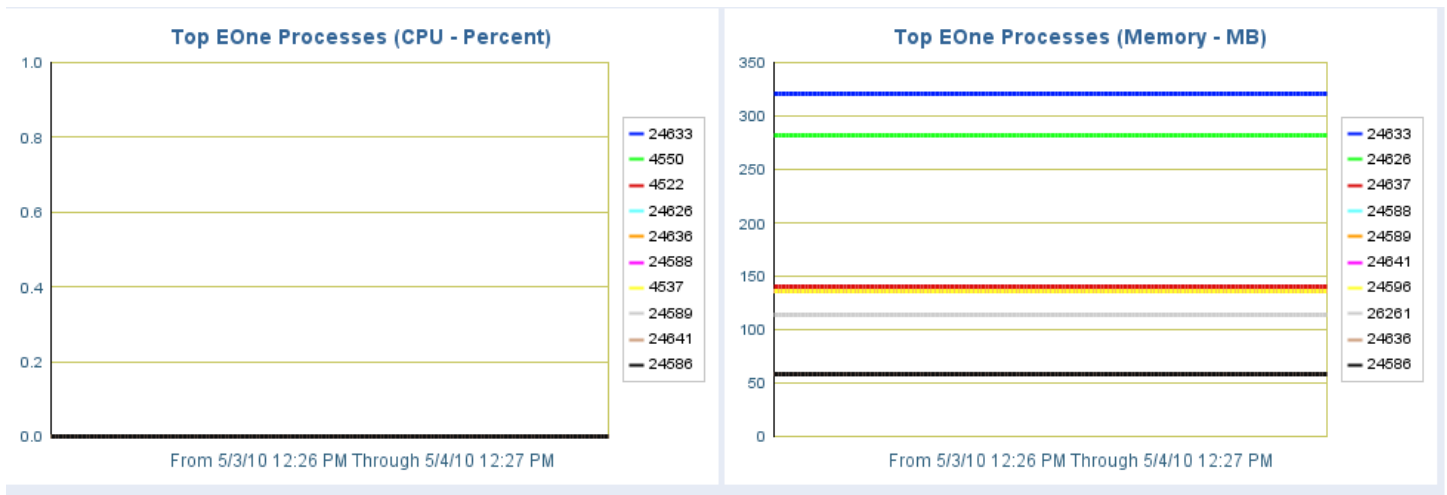
- Select the enterprise server instance in server manager.
- You should see the <EnterpriseOne Enterprise Server> instance.
- Make sure that the server is up and running.
- Go to the <Runtime Metrics> section.
- Click on the <Process Detail> link.
- You should see the <Enterprise server Processes> page.

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.



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:

Select [Process]: ^\ Remove Zombie

Select All | Select None Previous All Next

<input type="checkbox"/>	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
<input type="checkbox"/>	CALL_OBJECT KERNEL	Kernel Process	15791	RUNNING	3262	92	6	1679	0	282	6	12	30	23	0
<input type="checkbox"/>	CALL_OBJECT KERNEL	Kernel Process	15796	RUNNING	4263	92	6	1678	0	277	7	7	30	23	0
<input type="checkbox"/>	CALL_OBJECT KERNEL	Kernel Process	15786	RUNNING	2898	92	6	1424	0	277	4	7	24	22	0
<input type="checkbox"/>	CALL_OBJECT KERNEL	Kernel Process	15801	RUNNING	2534	92	5	1116	0	269	4	7	18	18	0
<input type="checkbox"/>	CALL_OBJECT KERNEL	Kernel Process	15808	RUNNING	2534	92	5	1123	0	269	4	7	18	18	0
<input type="checkbox"/>	R014021 XJDE0001	Runbatch	15866	RUNNING	968	92	0	0	0	168	1	1	2	13	5
<input type="checkbox"/>	UBE KERNEL	Kernel Process	15775	RUNNING	363	92	0	125	0	133	0	1	0	2	0
<input type="checkbox"/>	SECURITY KERNEL	Kernel Process	15776	RUNNING	379	92	32	234	0	133	0	1	0	3	0
<input type="checkbox"/>	SECURITY KERNEL	Kernel Process	15783	RUNNING	379	92	37	237	0	133	0	1	0	3	0
<input type="checkbox"/>	QUEUE KERNEL	Kernel Process	15831	RUNNING	368	92	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.
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 (requests) 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, 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 started 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

Field	Description
	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.
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 BSN 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.

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.

General Information	
Process Name	CALL OBJECT KERNE
Process Type	Kernel Process
Range Index	0
Process ID	18990
Process Index In Shared Memory	4
Start Time	12/31/69 5:00 PM
Last Message Time	12/31/69 5:00 PM
Messages Received	0
Outstanding Requests	0
Parent Process ID	18977
iSeries Job Number	0
Process User Id (OS)	110
OS Group ID	110
OS Username	jde812
OS Status	2
Memory (MB)	137
CPU %	0
Threads	7
JDE Caches	0
Total Open JDB Transactions	0
Manual Open JDB Transactions	0
Data Pointers	0
Tables/Views	0
JDB Table Caches	0
Database Connections	0
JDE Cache Records	0

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 of 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.
<i>IBM i</i> Job Number	The job number of the process, valid on the <i>IBM i</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, <i>IBM i</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 started 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 jdeCachelnit API. These caches are generally created by EnterpriseOne Business Functions. They are necessary for Business Functions to do their work. But

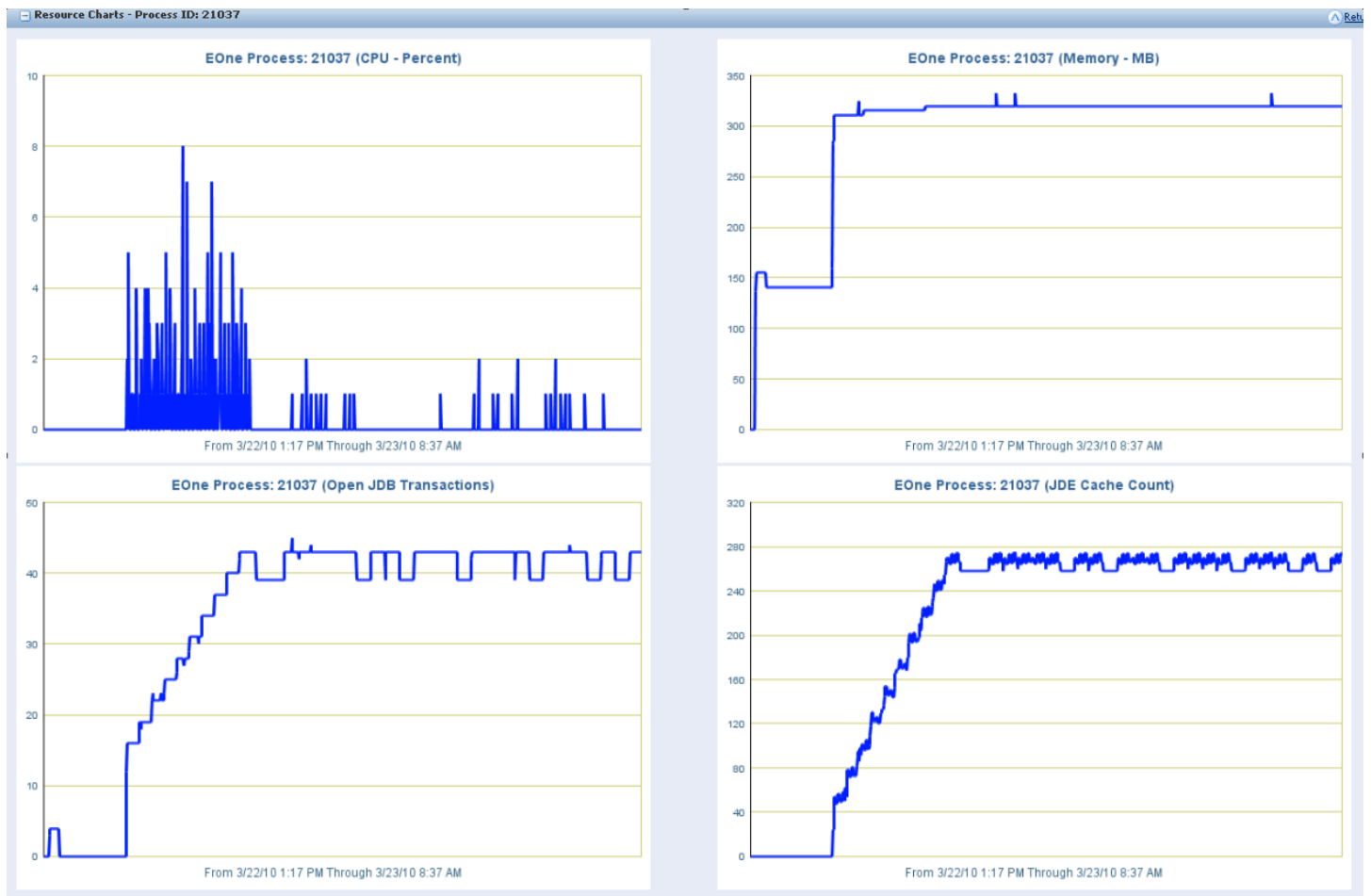
Field	Description
	an increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call <code>jdeCacheTerminate</code> API in business function code to fix the <code>JDECache</code> leak.
Total Open JDB Transactions* (does not apply to <code>jdenet_n</code> processes)	The total number of open JDB transactions which includes both <code>AUTO</code> and <code>MANUAL</code> transaction. This resource is opened by calling <code>JDB_InitUser</code> API which is either called with <code>JDEDB_COMMIT_AUTO</code> or <code>JDEDEB_COMMIT_MANUAL</code> . An increase in this value over time could indicate a leak that needs to be analyzed. The application developer should call <code>JDB_FreeUser</code> API in business function code to fix the JDB Transactions leak.
Manual Open JDB Transactions* (does not apply to <code>jdenet_n</code> processes)	The number of <code>MANUAL</code> open JDB transactions are opened by calling <code>JDB_InitUser</code> API with <code>JDEDEB_COMMIT_MANUAL</code> . 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 <code>JDB_FreeUser</code> API in business function code to fix the JDB Transactions leak.
Data Pointers* (does not apply to <code>jdenet_n</code> 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 <code>jdeStoreDataPtr</code> 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 <code>jdeRemoveDataPtr</code> 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 <code>jdenet_n</code> 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 <code>JDB_OpenTable</code> and <code>JDB_OpenView</code> API. The application developer should call <code>JDB_CloseTable</code> 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 <code>jdenet_n</code> 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 <code>JDB_AddTableToCache</code> 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 <code>JDB_AddTableToCache</code> API.
Database Connections**	The total number of open Database Connections includes <code>AUTOcommit</code> , <code>Select For Update (SFU)</code> and <code>MANUAL-commit</code> connections. An increase in this value over time could indicate a leak that needs to be analyzed. The <code>AUTO</code> and <code>SFU</code> connections have a small maximum limit whereas the <code>MANUAL</code> 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 <code>JDB_InitUser</code> with <code>MANUAL COMMIT MODE</code> but has not called the <code>JDB_FreeUser</code> . 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 <code>JDB_FreeUser</code> 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 <code>jdeCacheAdd</code> 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

Field	Description
	application developer should call the <code>jdeCacheDelete</code> (to delete cache records) or <code>jdeCacheTerminate</code> (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).



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 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.

Chart	Description
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.

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 Transactions	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.

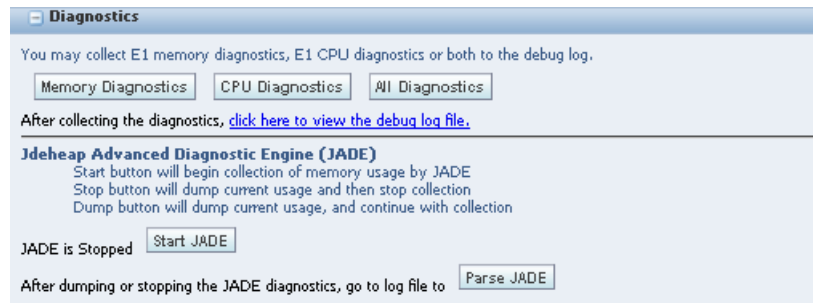
Field	Description
Last Active Time	Last time the user performed any work on the CallObject kernel process. This will determine the staleness of user session and is a good metric when an EnterpriseOne user is logged for day probably from a third party integration system using EnterpriseOne Java connector, 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.
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 Transactions opened 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.

In Tools Release 9.2.6.0 and above, the CallObject kernel process will have an additional column showing which Applications package the EnterpriseOne user is using:

Connected Users Return To Top										
Shown below are the users associated with the kernel process.										
User Name	Originating Machine	Environment	Environment Package	SignOn Time	Last Active Time	JDE Caches	Total Open JDB Transactions	Manual Open JDB Transactions	Data Pointers	Tables/Views
USR01	den60202jems	JDV92WN2	D926PKGB	9/ 7/2021 11:57:56	9/ 7/2021 12:13:39	0	4	0	0	2
USR02	den60202jems	JDV92WN2	D926PKGA	9/ 7/2021 12:16:15	9/ 7/2021 12:16:18	0	3	0	0	0

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.



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.
 - o Process OS data
 - i. Memory (megabytes)
 - ii. CPU (percent)
 - iii. Threads (number of threads)
 - o 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.

- o Process OS data
 - i. Memory (megabytes)
 - ii. CPU (percent)
 - iii. Threads (number of threads)
- o CPU Diagnostics
 - i. 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.
 - ii. 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.
 - o Process OS data
 - i. Memory (megabytes)
 - ii. CPU (percent)
 - iii. Threads (number of threads)
 - o Memory data
 - i. Process level data shared by all user sessions.
 1. Environment data
 2. JDB Table Cache data
 3. Database connection data
 - ii. User Sessions
 - Open JDB Transactions
 - Open Tables of Views
 - Open JDECaches
 - Open Data Pointers
 - o CPU Diagnostics
 - i. 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.
 - ii. 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 -
 - o 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 advanced Profiling). The default setting is JADE level 2.
 - o Stop button will dump current usage and then stop collection
 - o Dump button will dump current usage and then stop collection
 - o 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.

(Release 9.2.6.0) Here is a sample of All Diagnostics:

```
Sep 7 13:00:46.002000 - 6116/3768 WRK:eNetKernelLevel3Dump User initiated process dump
(All Diagnostics):
***** Begin OS Data *****
Memory Usage = 193MB
CPU Usage = 0%
Number of Threads = 10
***** End OS Data *****
***** Begin Detailed Memory Data *****
***** Begin Process Data *****
ENVIRONMENT,Ptr=000000008C85980,Env=JDV92WN2,PathCode=DV920,Package=D926PKGB,Latest=FALSE
ENVIRONMENT,Ptr=00000000C521FF0,Env=JDV92WN2,PathCode=DV920,Package=D926PKGA,Latest=TRUE
ENVIRONMENT,Ptr=00000000D084730,Env=DV92WN2,PathCode=DV920,Package=D926PKGA,Latest=TRUE
JDBTABLECACHE,Ptr=000000009F67680,Name=JDB_BV_1631033875JDV92WN2F0004,#Records=2
JDBTABLECACHE,Ptr=000000009F67940,Name=JDB_BV_1631033875JDV92WN2F0005,#Records=5
JDBTABLECACHE,Ptr=000000009DFF0E0,Name=JDB_BV_1631033876JDV92WN2F40039,#Records=1
JDBTABLECACHE,Ptr=000000009DFF3A0,Name=JDB_BV_1631033876JDV92WN2F4009T1,#Records=1
JDBTABLECACHE,Ptr=00000000C159C90,Name=JDB_BV_1631033876JDV92WN2F4009,#Records=1
JDBTABLECACHE,Ptr=00000000C80F3C0,Name=JDB_BV_1631033876JDV92WN2F40205,#Records=1
JDBTABLECACHE,Ptr=00000000C80F7E0,Name=JDB_BV_1631033876JDV92WN2F7306,#Records=1
JDBTABLECACHE,Ptr=00000000C8109C0,Name=JDB_BV_1631033876JDV92WN2F99410,#Records=8
JDBTABLECACHE,Ptr=00000000A034720,Name=JDB_BV_1631034974JDV92WN2F0004,#Records=1
JDBTABLECACHE,Ptr=00000000A0349E0,Name=JDB_BV_1631034974JDV92WN2F0005,#Records=2
JDBTABLECACHE,Ptr=00000000A01E3C0,Name=JDB_BV_1631034975JDV92WN2F7306,#Records=1
JDBTABLECACHE,Ptr=00000000E4EF8C0,Name=JDB_BV_1631035877DV92WN2F7306,#Records=1
OCIDBCONN,Ptr=00000000A2FD060,DBServer=den60202jems,DBUser=JDE,TNSDB=ems2649,ConnState=AutoInUs
OCIDBCONN,Ptr=00000000A2FE1D0,DBServer=den60202jems,DBUser=TESTDTA,TNSDB=ems2649,ConnState=Auto
OCIDBCONN,Ptr=00000000A300A80,DBServer=den60202jems,DBUser=DV920,TNSDB=ems2649,ConnState=AutoIn
***** End Process Data *****
***** Begin Session Data *****
SESSION,Ptr=0000000094F82B0,User=USR01,Env=JDV92WN2,Role=*ALL,EnvPtr=000000008C85980,Machine=d
9/ 7/2021 11:57:56,LastActiveTime= 9/ 7/2021 13:00:09,ThreadedBSFN=0,InlinedBSFN=0
OPENJDBTRANSACTION,Ptr=00000000C417F80,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
TABLE,Ptr=00000000A305C70,Name=F9200,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
TABLE,Ptr=00000000A308670,Name=F9210,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
TABLE,Ptr=00000000D2FAA70,Name=F9203,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
TABLE,Ptr=00000000D2F5370,Name=F9207,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
OPENJDBTRANSACTION,Ptr=00000000C41DE70,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000C41EC00,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000C418D10,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000D8C9560,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
JDECACHE,Ptr=00000000E4EE9A0,Name=231DBC09834E4EAE21F1CC9A207A205B918SalesOrderLine,#Cursors=1,
JDECACHE,Ptr=00000000E4ED240,Name=231DBC09834E4EAE21F1CC9A207A205B918SalesOrderHeader,#Cursors=
DATAPOINTER,Ptr=00000000C5E6550,Index=1001,File=b42x0080.c,Function=GetSalesAdvisorDocumentType
SESSION,Ptr=00000000DD6E990,User=JDE,Env=JDV92WN2,Role=*ALL,EnvPtr=00000000C521FF0,Machine=den
9/ 7/2021 12:32:30,LastActiveTime= 9/ 7/2021 12:32:31,ThreadedBSFN=0,InlinedBSFN=0
OPENJDBTRANSACTION,Ptr=00000000D8C4400,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000D8CB080,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000D8C7A40,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
SESSION,Ptr=00000000DA0CFB0,User=JDE,Env=JDV92WN2,Role=*ALL,EnvPtr=00000000C521FF0,Machine=den
9/
7/2021 12:30:57,LastActiveTime= 9/ 7/2021 12:30:58,ThreadedBSFN=0,InlinedBSFN=0
OPENJDBTRANSACTION,Ptr=00000000C41D0E0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000C41C350,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=00000000D8C6CB0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
```

```

***** End Session Data *****
***** End Detailed Memory Data *****
***** Begin Detailed CPU Data *****
***** Begin BSFN Call Stacks *****
No BSFN Call Stack ***** End BSFN Call Stacks *****
***** Begin OS Call Stacks *****
=====Call stack of thread 3768=====
GetNTProcessCallStack! C:\JDEdwards\E920\system\bin64\jdel.dll
allocCallStackWindows! C:\JDEdwards\E920\system\bin64\jdel.dll
jdeAllocCallStack! C:\JDEdwards\E920\system\bin64\jdel.dll
logProcessDumpData! C:\JDEdwards\E920\system\bin64\jdekrnl.dll
ProcessLevel3DataDump! C:\JDEdwards\E920\system\bin64\jdekrnl.dll
JDEK_DispatchCallObjectMessage! C:\JDEdwards\E920\system\bin64\jdekrnl.dll
XMLCallObjectDispatch! C:\JDEdwards\E920\system\bin64\XMLCallObj.dll
=====Call stack of thread 7356=====
NtDelayExecution! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fd998c1.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa78da7028.<nosymbols>! C:\app\client\product\19.0.0\client_1\bin\OraClient19.Dll
0x7ffa7994e0a0.<nosymbols>! C:\app\client\product\19.0.0\client_1\bin\oracore19.dll
0x7ffa90497974.<nosymbols>! C:\Windows\System32\KERNEL32.DLL
=====Call stack of thread 6548=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 7528=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 1464=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 7780=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 5724=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 4352=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 2868=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
psthread_cond_wait! C:\JDEdwards\E920\system\bin64\psthread.dll
=====Call stack of thread 7584=====
NtWaitForMultipleObjects! C:\Windows\SYSTEM32\ntdll.dll
0x7ffa8fdbd5ee.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
0x7ffa8fdbd4de.<nosymbols>! C:\Windows\System32\KERNELBASE.dll
receiveMessage! C:\JDEdwards\E920\system\bin64\jdeipc.dll

```

```
***** End OS Call Stacks *****
***** End Detailed CPU Data *****
```

Contextual Diagnostics

(Release 9.2.6.0) 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=00000000E4EE9A0, Name=231DBC09834E4EAE21F1CC9A207A205B918SalesOrderLine, #Cursors=1,
TABLE, Ptr=00000000A305C70, Name=F9200, CommitStatus=Active, File=jdeddapi.c, Function=CheckAndOpenD
OPENJDBTRANSACTION, Ptr=00000000C417F80, CommitMode=Auto, Owner=InitUser, AppName=(UNKNOWN), File=jd
```

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.

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 reduce 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.

The screenshot shows a dialog box titled "Corrective Actions" with a "Return To Top" link in the top right corner. The main text reads: "This will clear internal tools caches for this process. This should reduce the memory footprint of the process and may cause some performance impact until these caches are rebuilt." Below this text is a "Clear Cache" button. A horizontal line separates this from the second section, which reads: "This will set the pending recycle time to now for this kernel process and will then follow all JDE.INI defined recycling rules when recycling this kernel process. The currently logged-in users in this kernel can continue their work but will not be notified." Below this text is a "Recycle Kernel" button.

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.

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.

Available Log Files Return To Top			
Display Logs Modified Within <input type="radio"/> 1 Hour <input checked="" type="radio"/> 24 Hours <input type="radio"/> 48 Hours <input type="radio"/> 1 Week <input type="radio"/> No Limit			
Select [Log File]: <input type="button" value="Delete"/>			
Select All Select None			Previous 1 - 10 Next
Filename	File Size	Last Modified	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_28774_1269991648_1_dmp.log	14,993	Apr 20, 2010 3:07:54 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_5504.log	9,212	Apr 20, 2010 3:07:54 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_6329.dmp.log	17,800	Apr 20, 2010 3:07:54 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_15791.log	18,277	Apr 20, 2010 3:07:00 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_15786.log	20,916	Apr 20, 2010 3:06:49 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jdedebug_15791.log	21,074	Apr 20, 2010 3:05:05 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_15866.log	5,360	Apr 20, 2010 3:04:57 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_16309.log	954	Apr 20, 2010 3:03:57 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jde_16064.log	363	Apr 20, 2010 3:03:54 PM	
<input type="checkbox"/> /u01/jdedwards/e812/log/jdedebug_16309.log	97	Apr 20, 2010 3:03:54 PM	

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, flaa34, f14ce0, 0, 0, 18814c) + 2b8
f65c5e50 OCISmtExecute (188118, flaa34, 0, 0, 0, 0) + 2c
f657e348 BFOCISmtExecute (188118, flaa34, 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, f635eccc, 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 OCISstmtExecute (188118, f84e4c, 0, 0, 0, 0) + 2c
f657e348 BFOCISstmtExecute (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, f58deccc, 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)
----- 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 kpucallback (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 kpucxecv8 (188118, 5ad2f8, 5ad344, 17790, 0, f74bb544) + 1390
f66682c0 kpucxec (0, 5ad2f8, f85200, 0, 0, 1000) + 1544
f65c5e50 OCISmtExecute (188118, 5ad2f8, 0, 0, 0, 0) + 2c
f657e348 BFOCISmtExecute (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
fel3a894 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
felc969c 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 kpuxexec (f8e9d800, f1492c, 5ad6ac, 0, 0, 18814c) + 2b8
f65c5e50 OCISstmtExecute (188118, f1492c, 0, 0, 0, 0) + 2c
f657e348 BFOCISstmtExecute (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
fel3a894 TM_DBPerformRequest (9e2a20, 1, d03440, 116e990, 1b3ab8, 5) + 41c
feba75b4 SelectKeyed (116e990, 9e2a20, f385eacc, 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 business function which is being invoked by looking top to bottom until you find jdeCallObjectV2() In this example, it is DetermineIfBlanketPOExists.
3. Check for any MEM SAR fix for this Bsfm 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 bugs 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.

(Release 9.2.6.0) 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_rql.c Line: 192
***** Begin OS Data *****
Memory Usage = 4281MB
CPU Usage = 0%
Number of Threads = 10
***** End OS Data *****
***** Begin Detailed Memory Data *****
***** Begin Process Data *****
ENVIRONMENT,Ptr=0000000008C85980,Env=JDV92WN2,PathCode=DV920,Package=D926PKGB,Latest=FALSE
ENVIRONMENT,Ptr=000000000C521FF0,Env=JDV92WN2,PathCode=DV920,Package=D926PKGA,Latest=TRUE
ENVIRONMENT,Ptr=000000000D084730,Env=DV92WN2,PathCode=DV920,Package=D926PKGA,Latest=TRUE
JDBTABLECACHE,Ptr=0000000009F67680,Name=JDB_BV_1631033875JDV92WN2F0004,#Records=2
JDBTABLECACHE,Ptr=0000000009F67940,Name=JDB_BV_1631033875JDV92WN2F0005,#Records=5
JDBTABLECACHE,Ptr=0000000009DFF0E0,Name=JDB_BV_1631033876JDV92WN2F40039,#Records=1
JDBTABLECACHE,Ptr=0000000009DFF3A0,Name=JDB_BV_1631033876JDV92WN2F4009T1,#Records=1
JDBTABLECACHE,Ptr=000000000C159C90,Name=JDB_BV_1631033876JDV92WN2F4009,#Records=1
JDBTABLECACHE,Ptr=000000000C80F3C0,Name=JDB_BV_1631033876JDV92WN2F40205,#Records=1
JDBTABLECACHE,Ptr=000000000C80F7E0,Name=JDB_BV_1631033876JDV92WN2F7306,#Records=1
JDBTABLECACHE,Ptr=000000000C8109C0,Name=JDB_BV_1631033876JDV92WN2F99410,#Records=8
JDBTABLECACHE,Ptr=000000000A034720,Name=JDB_BV_1631034974JDV92WN2F0004,#Records=1
JDBTABLECACHE,Ptr=000000000A0349E0,Name=JDB_BV_1631034974JDV92WN2F0005,#Records=948722
JDBTABLECACHE,Ptr=000000000A01E3C0,Name=JDB_BV_1631034975JDV92WN2F7306,#Records=1
JDBTABLECACHE,Ptr=000000000E4EF8C0,Name=JDB_BV_1631035877DV92WN2F7306,#Records=1
OCIDBCONN,Ptr=000000000A2FD060,DBServer=den60202jems,DBUser=JDE,TNSDB=ems2649,ConnState=AutoInUs
OCIDBCONN,Ptr=000000000A2FE1D0,DBServer=den60202jems,DBUser=TESTDTA,TNSDB=ems2649,ConnState=Auto
OCIDBCONN,Ptr=000000000A300A80,DBServer=den60202jems,DBUser=DV920,TNSDB=ems2649,ConnState=AutoIn
```

```

***** End Process Data *****
***** Begin Session Data *****
SESSION,Ptr=00000000094F82B0,User=USR01,Env=JDV92WN2,Role=*ALL,EnvPtr=0000000008C85980,Machine=den
9/ 7/2021 11:57:56,LastActiveTime= 9/ 7/2021 13:00:09,ThreadedBSFN=0,InlinedBSFN=0
OPENJDBTRANSACTION,Ptr=000000000C417F80,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
TABLE,Ptr=000000000A305C70,Name=F9200,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
TABLE,Ptr=000000000A308670,Name=F9210,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
TABLE,Ptr=000000000D2FAA70,Name=F9203,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
TABLE,Ptr=000000000D2F5370,Name=F9207,CommitStatus=Active,File=jdeddapi.c,Function=CheckAndOpenD
OPENJDBTRANSACTION,Ptr=000000000C41DE70,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000C41EC00,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000C418D10,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000D8C9560,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
JDECACHE,Ptr=000000000E4EE9A0,Name=231DBC09834E4EAE21F1CC9A207A205B918SalesOrderLine,#Cursors=1,
JDECACHE,Ptr=000000000E4ED240,Name=231DBC09834E4EAE21F1CC9A207A205B918SalesOrderHeader,#Cursors=
DATAPOINTER,Ptr=000000000C5E6550,Index=1001,File=b42x0080.c,Function=GetSalesAdvisorDocumentType
SESSION,Ptr=000000000DD6E990,User=JDE,Env=JDV92WN2,Role=*ALL,EnvPtr=000000000C521FF0,Machine=den
9/ 7/2021 12:32:30,LastActiveTime= 9/ 7/2021 12:32:31,ThreadedBSFN=0,InlinedBSFN=0
OPENJDBTRANSACTION,Ptr=000000000D8C4400,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000D8CB080,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000D8C7A40,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
SESSION,Ptr=000000000DA0CFB0,User=JDE,Env=JDV92WN2,Role=*ALL,EnvPtr=000000000C521FF0,Machine=den
9/ 7/2021 12:30:57,LastActiveTime= 9/ 7/2021 12:30:58,ThreadedBSFN=0,InlinedBSFN=0
OPENJDBTRANSACTION,Ptr=000000000C41D0E0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000C41C350,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
OPENJDBTRANSACTION,Ptr=000000000D8C6CB0,CommitMode=Auto,Owner=InitUser,AppName=(UNKNOWN),File=jd
***** 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 led 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.
6. 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.
7. In our example above, the culprit is the JDB Caching of F0005 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:

The screenshot shows a configuration window with several settings. The 'Query Execution Time Threshold' is highlighted in blue and set to the value '1'. Other visible settings include 'Database TCP/IP Port' (1521), 'JDBNET Use' (N), 'Unicode Flag' (Y), and 'Support LOBs' (Y).

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:

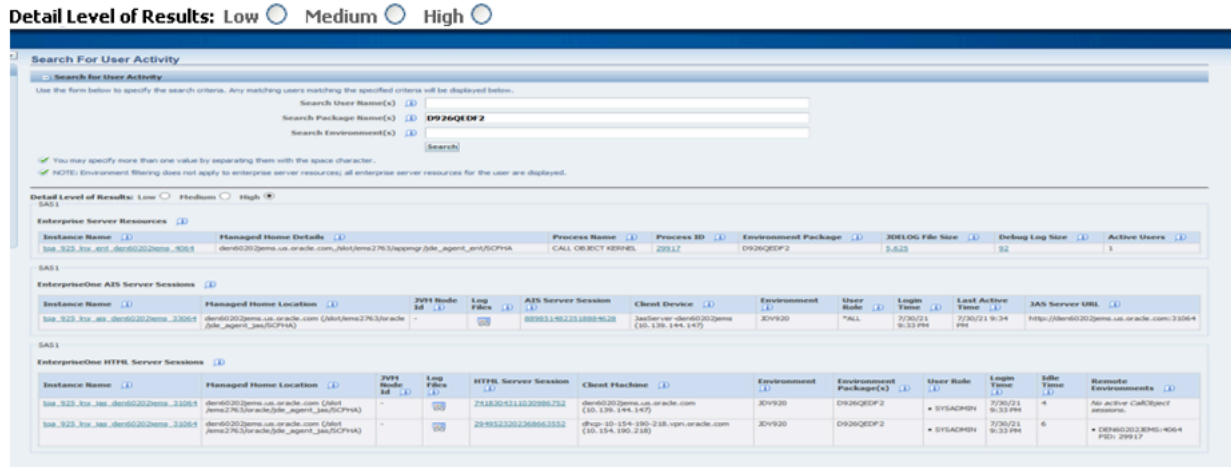
```
859026/452 MAIN_THREAD Tue Apr 20 16:41:26.044816
dbdrv_log.c196
OS400AG016 - doQueryDiagnostics: The following SQL query took 5 seconds which is equal
to or greater than QueryExecutionTimeThreshold (1 seconds) for E1User(JOESMITH) with
DBProxyUser(JDE) .

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 Search For User Activity from the main Management Dashboard page.



The search list can be narrowed by specifying user names, package name, 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.

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: <ul style="list-style-type: none"> • TRUE • FALSE

Field	Description
	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	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - DEBUG</p> <p>INI Entry - logCacheLeaksAtSignoff</p> <p>Default Value - FALSE</p> <p>Allowed Values:</p> <ul style="list-style-type: none"> • TRUE • FALSE <p>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.</p>
Log Data Pointer leaks at signoff	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - DEBUG</p> <p>INI Entry - logDPLeaksAtSignoff</p> <p>Default Value - FALSE</p> <p>Allowed Values:</p> <ul style="list-style-type: none"> • TRUE • FALSE <p>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.</p>

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 an 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.

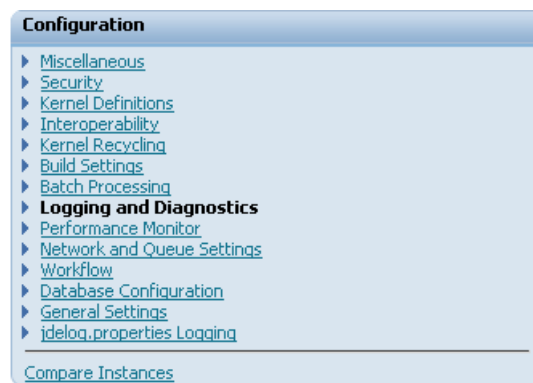
On the other hand, JADE is a finer diagnostic tool and it works by collecting all memory allocations from the given Enterprise Server Kernel.

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.



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 effect 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

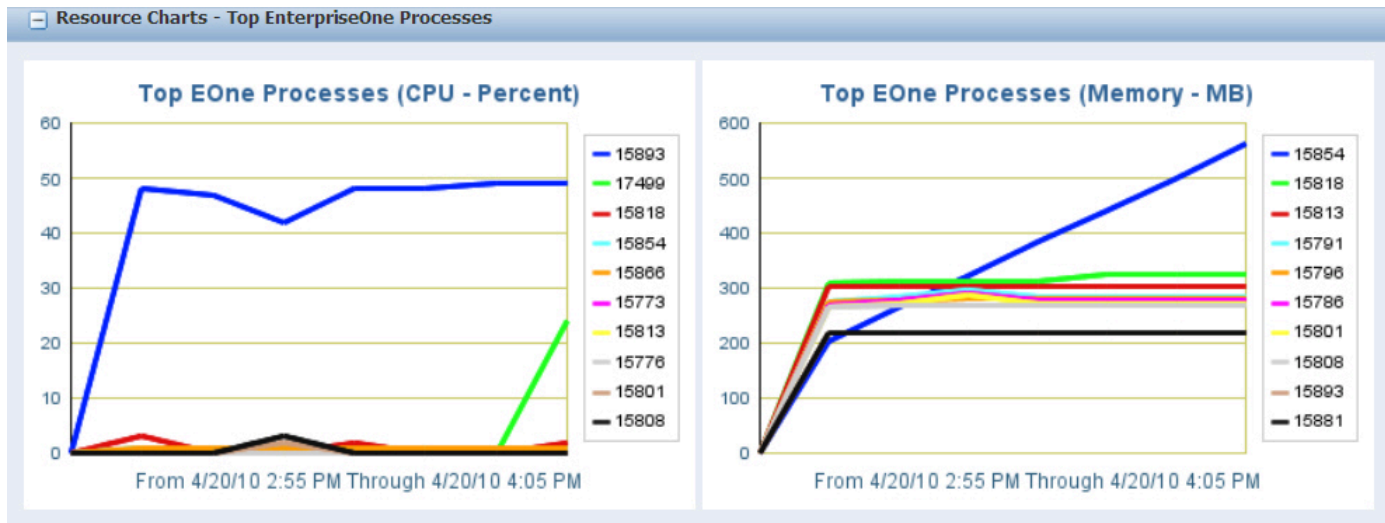
If an Application is known to have a memory leak:

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 "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:



Select [Process]: Remove Zombie

Select All | Select None Previous 1 - 10 Next

<input type="checkbox"/>	Process Name	Process Type	Process ID	Process Status	JDELOG File Size	Debug Log Size	Connected Users	Total Requests	Outstanding Requests	Memory (MB)
<input type="checkbox"/>	R014021 XJDE0001	Runbatch	15854	RUNNING	968	97	0	0	0	561
<input type="checkbox"/>	MANAGEMENT KERNEL	Kernel Process	15818	RUNNING	391	97	0	812	0	326
<input type="checkbox"/>	METADATA KERNEL	Kernel Process	15813	RUNNING	866	97	0	2377	0	303

1. Drill into the known process.

Resource Charts - Process ID: 15854
Return To Top

EOne Process: 15854 (CPU - Percent)

From 4/20/10 2:55 PM Through 4/20/10 4:05 PM

EOne Process: 15854 (Memory - MB)

From 4/20/10 2:55 PM Through 4/20/10 4:05 PM

EOne Process: 15854 (Open JDB Transactions)

From 4/20/10 2:55 PM Through 4/20/10 4:05 PM

EOne Process: 15854 (JDE Cache Count)

From 4/20/10 2:55 PM Through 4/20/10 4:05 PM

Diagnostics
Return To Top

You may collect E1 memory diagnostics, E1 CPU diagnostics or both to the debug log.

After collecting the diagnostics, [click here to view the debug log file.](#)

Jdeheap Advanced Diagnostic Engine (JADE)
 Start button will begin collection of memory usage by JADE
 Stop button will dump current usage and then stop collection
 Dump button will dump current usage, and continue with collection

JADE is Stopped

After dumping or stopping the JADE diagnostics, go to log file to

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.

Log File Viewer [u01/jdedwards/e812/log/jdedebug_16854.log]

Filter Criteria

You may add criterion to narrow the results returned. Criterion will be evaluated in the order displayed.

[Add Another Row](#)

Page Size:

Match Type:

[Apply Filter\(s\)](#) [Save As Favorite](#)

[Download Entire Log File](#) | [Download JADE Parse of Log File](#)

[Previous](#) 1-250 of 951 [Next](#)

Last Modified	4/20/10 4:02 PM	File Size	137,676
Apr 20 14:50:02.167994	DEBUG INIT0	- 16854	**** jdeDebugInit -- output disabled in INI file.
Apr 20 14:50:02.172352	jdemen.c136	- 16854/1	MAIN_THREAD BMD OFF - Not running BSMN MEMORY DIAGNOSTICS v3.98.2.0 level 0
Apr 20 16:01:06.860463	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE processing has been turned on
Apr 20 16:02:46.750190	jdemen.c857	- 16854/1	WRK: Starting jdeCallObject JADE processing has been turned off
Apr 20 16:02:46.751210	jdemen.c894	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, BSMNptrs=, 946, BSMNbytes=, 9977722, Allptrs=, 959, Allbytes=, 9978204, JADE SUM
Apr 20 16:02:46.751274	jdemen.c893	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 0, plafe9ee8, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751315	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 1, p0060a648, 12, gebeEffectiveaddress, jdetool.c412
Apr 20 16:02:46.751354	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 2, plad927b0, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751392	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 3, plac35e08, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751430	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 4, placf8d00, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751469	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 5, pia7070c0, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751506	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 6, plab97358, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751547	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 7, plaa3f3b0, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751585	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 8, pia8e0058, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751623	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 9, pia8886b0, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751660	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 10, plac44e78, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751698	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 11, pia656238, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751845	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 12, plaaeb400, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751892	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 13, pia630900, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751931	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 14, plaa6b98, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.751970	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 15, plae089e8, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.752008	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 16, plada1330, 10000, getan8f0115phone, b0100004.c129
Apr 20 16:02:46.752046	jdemen.c853	- 16854/1	WRK: Starting jdeCallObject JADE2, set=1, 17, plarf4fa58, 10000, getan8f0115phone, b0100004.c129

6. Download JADE Parse 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 p1a666aaa8	10000	getan8f0115phone	b0100004.c129
25	23	02:46.8	set=1		21 p1aafb40	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.

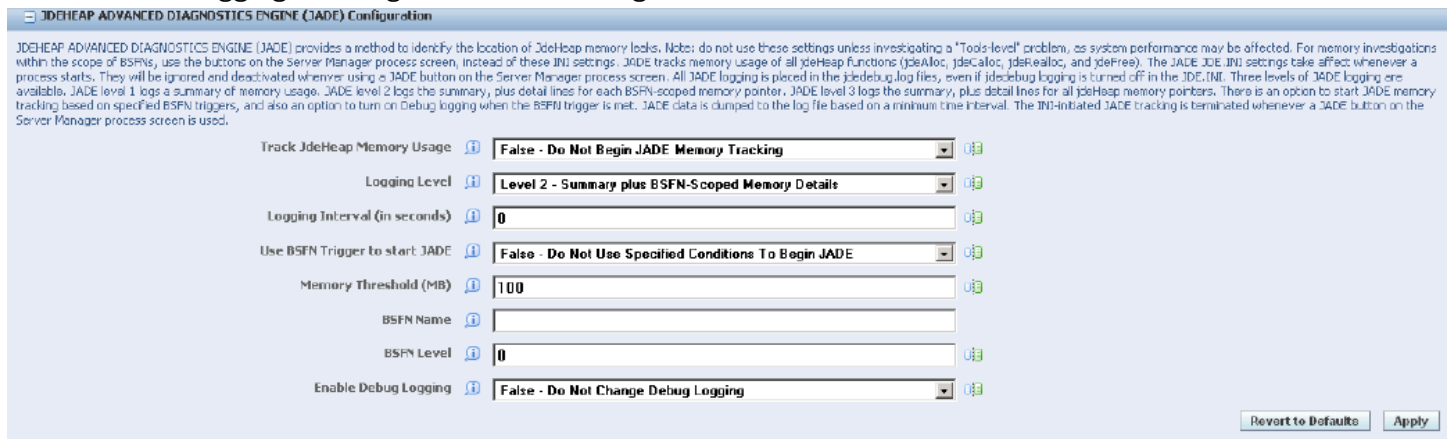
There is an option to start JADE memory tracking based on specified BSFN triggers, and also an option to turn on Debug logging when the BSFN trigger is met. JADE data is dumped to the log file based on a minimum time interval. The INI-initiated JADE tracking is terminated whenever a JADE button on the 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.



The fields used to configure JADE are:

Field	Description
Track JdeHeap Memory Usage	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - trackMemUsage</p> <p>Default Value - 0</p> <p>Allowed Values:</p> <ul style="list-style-type: none"> • False - Do Not Begin JADE Memory Tracking (0) • True - Begin JADE Memory Tracking At Process Start (1) <p>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.</p>
Logging Level	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - logLevel</p> <p>Default Value - 2</p> <p>Allowed Values:</p> <ul style="list-style-type: none"> • 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)

Field	Description
	<p>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.</p>
<p>Logging Level (in seconds)</p>	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - logInterval</p> <p>Default Value - 0</p> <p>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.</p>
<p>Use BSFN Trigger to start JADE</p>	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - bsfnTriggerUseTrigger</p> <p>Default Value - 0</p> <p>Allowed Values;</p> <ul style="list-style-type: none"> • False - Do Not Use Specified Conditions To Begin JADE (0) • True - Use Specified Conditions To Begin JADE (1) <p>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.</p>
<p>Memory Threshold (MB)</p>	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - bsfnTriggerMemThresholdMB</p> <p>Default Value - 100</p> <p>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 tracking starts.</p>
<p>BSFN Name</p>	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - bsfnTriggerBsfName</p> <p>Business Function name. When combined with the BSFN level and memory threshold, triggers the JADE tracking.</p>
<p>BSFN Level</p>	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p>

Field	Description
	<p>INI Entry - bsfnTriggerBsfLevel</p> <p>Default Value - 0</p> <p>Business Function level. When combined with the BSFN name and memory threshold, triggers JADE tracking. Not used when set to zero.</p>
Enable Debug Logging	<p>INI Filename - /u01/jdedwards/e812/ini/JDE.INI</p> <p>INI Section Name - JADE</p> <p>INI Entry - bsfnTriggerEnableDebug</p> <p>Default Value - 0</p> <p>Allowed Values:</p> <ul style="list-style-type: none"> • False - Do Not Change Debug Logging (0) • True - Turn On Debug Logging When JADE Is Triggered (1) <p>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.</p>

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.

BSFN MEMORY DIAGNOSTICS (BMD) Configuration [Return To Top](#)

BSFN MEMORY DIAGNOSTICS (BMD) provides a series of methods to identify the cause of resource usage problems, such as memory leaks. Note: do not use these settings unless investigating a problem, as system performance may be affected. Also, the EnterpriseOne services must be re-started for these JDE.INI settings to take affect. For Runbatch, each new UBE will use any changed BMD values without restarting the services. All BMD logging is placed in the jdedebug.log files, even if jdedebug logging is turned off in the JDE.INI. Three levels of BMD logging are available. BMD level 1 logs CPU and memory usage for all EnterpriseOne server processes. BMD levels 2 and 3 log CPU and memory usage for those processes which run Business Functions (Call-Object kernels and UBE jobs). BMD Level 2 logs only BSFN Level 1 and BSFN Level 2. BMD Level 3 logs all BSFN levels, and also provides an option of enabling Debug logs. Lower BSFN levels include data from all higher BSFN levels that each BSFN calls.

BMD Level

Allocation Frequency

Memory Threshold (MB)

BSFN Name

BSFN Level

Enable Debug Logging

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.

BSFN MEMORY DIAGNOSTICS (BMD) Configuration [Return To Top](#)

BSFN MEMORY DIAGNOSTICS (BMD) provides a series of methods to identify the cause of resource usage problems, such as memory leaks. Note: do not use these settings unless investigating a problem, as system performance may be affected. Also, the EnterpriseOne services must be re-started for these JDE.INI settings to take affect. For Runbatch, each new UBE will use any changed BMD values without restarting the services. All BMD logging is placed in the jdedebug.log files, even if jdedebug logging is turned off in the JDE.INI. Three levels of BMD logging are available. BMD level 1 logs CPU and memory usage for all EnterpriseOne server processes. BMD levels 2 and 3 log CPU and memory usage for those processes which run Business Functions (Call-Object kernels and UBE jobs). BMD Level 2 logs only BSFN Level 1 and BSFN Level 2. BMD Level 3 logs all BSFN levels, and also provides an option of enabling Debug logs. Lower BSFN levels include data from all higher BSFN levels that each BSFN calls.

BMD Level

Allocation Frequency

Memory Threshold (MB)

BSFN Name

BSFN Level

Enable Debug Logging

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).

BSFN MEMORY DIAGNOSTICS (BMD) Configuration [Return To Top](#)

BSFN MEMORY DIAGNOSTICS (BMD) provides a series of methods to identify the cause of resource usage problems, such as memory leaks. Note: do not use these settings unless investigating a problem, as system performance may be affected. Also, the EnterpriseOne services must be re-started for these JDE.INI settings to take affect. For Runbatch, each new UBE will use any changed BMD values without restarting the services. All BMD logging is placed in the jddebug.log files, even if jddebug logging is turned off in the JDE.INI. Three levels of BMD logging are available. BMD level 1 logs CPU and memory usage for all EnterpriseOne server processes. BMD levels 2 and 3 log CPU and memory usage for those processes which run Business Functions (Call-Object kernels and UBE jobs). BMD Level 2 logs only BSFN Level 1 and BSFN Level 2. BMD Level 3 logs all BSFN levels, and also provides an option of enabling Debug logs. Lower BSFN levels include data from all higher BSFN levels that each BSFN calls.

BMD Level **Level 3 - BSFN Memory Logging (A)**

Allocation Frequency **15000**

Memory Threshold (MB) **250**

BSFN Name **CacheProcessUniqueF41021WF**

BSFN Level **2**

Enable Debug Logging **False - Do Not Change Debug Logg**

BSFN MEMORY DIAGNOSTICS (BMD) Configuration [Return To Top](#)

BSFN MEMORY DIAGNOSTICS (BMD) provides a series of methods to identify the cause of resource usage problems, such as memory leaks. Note: do not use these settings unless investigating a problem, as system performance may be affected. Also, the EnterpriseOne services must be re-started for these JDE.INI settings to take affect. For Runbatch, each new UBE will use any changed BMD values without restarting the services. All BMD logging is placed in the jddebug.log files, even if jddebug logging is turned off in the JDE.INI. Three levels of BMD logging are available. BMD level 1 logs CPU and memory usage for all EnterpriseOne server processes. BMD levels 2 and 3 log CPU and memory usage for those processes which run Business Functions (Call-Object kernels and UBE jobs). BMD Level 2 logs only BSFN Level 1 and BSFN Level 2. BMD Level 3 logs all BSFN levels, and also provides an option of enabling Debug logs. Lower BSFN levels include data from all higher BSFN levels that each BSFN calls.

BMD Level **Level 3 - BSFN Memory Logging (A)**

Allocation Frequency **15000**

Memory Threshold (MB) **250**

BSFN Name **CacheProcessUniqueF41021WF**

BSFN Level **2**

Enable Debug Logging **True - Turn On Debug Logging With**

True - Turn On Debug Logging With BMD Level 3

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.

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.

Log File Viewer [u01/jdedwards/e812/log/R014021_XJDE0001_1823_PDF.jdedebug.log]

Filter Criteria

You may add criterion to narrow the results returned. Criterion will be evaluated in the order displayed.

Page Size:

Match Type:

[Download Entire Log File](#) | [Download BMD Parse of Log File](#) | [Download BMD Parse of Log File \(No Zero Deltas\)](#) (BMD results are only available if BMD logging has been turned on) [Previous](#) [Next](#)

Last Modified	10/5/09 2:38 PM	File Size	42,069,220
Sep 23 10:56:48.668445	DEBUG INIT0	- 16739	**** jdeDebugInit -- output disabled in INI file.
Sep 23 10:56:48.672730	jdemem.c87	- 16739/1 MAIN_THREAD	BMD ON - Running BSFN MEMORY DIAGNOSTICS v8.98.2.0
Sep 23 10:56:50.522255	jdeobj.c585	- 16739/1 MAIN_THREAD	BMD level 2. Trigger condition: memory threshold 10
Sep 23 10:56:50.570749	jdeobj.c676	- 16739/1 MAIN_THREAD	BMD2 cpu 3 mem 153224000 up 10048000 Lev:1 jde
Sep 23 10:56:50.860909	jdeobj.c676	- 16739/1 MAIN_THREAD	BMD2 cpu 3 mem 153312000 eq 0 Lev:1 jde
Sep 23 10:57:02.075722	jdeobj.c676	- 16739/1 MAIN_THREAD	BMD2 cpu 2 mem 157920000 up 440000 Lev:1 Get
Sep 23 10:57:03.108529	jdeobj.c676	- 16739/1 MAIN_THREAD	BMD2 cpu 2 mem 171712000 up 13792000 Lev:1 Get
Sep 23 10:57:03.116739	jdeobj.c676	- 16739/1 MAIN_THREAD	BMD2 cpu 2 mem 171712000 eq 0 Lev:1 Get
Sep 23 10:57:03.126824	jdeobj.c676	- 16739/1 MAIN_THREAD	BMD2 cpu 2 mem 171712000 eq 0 Lev:1 Get
Sep 23 10:57:03.152302	jdeobj.c676	- 16739/1 WRK:Starting jdeCallObject	BMD2 cpu 2 mem 171712000 eq 0 Lev:1 Get
Sep 23 10:57:03.157478	jdeobj.c676	- 16739/1 WRK:Starting jdeCallObject	BMD2 cpu 2 mem 171712000 eq 0 Lev:1 Get
Sep 23 10:57:03.159493	jdeobj.c676	- 16739/1 WRK:Starting jdeCallObject	BMD2 cpu 2 mem 171712000 eq 0 Lev:1 Get

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.

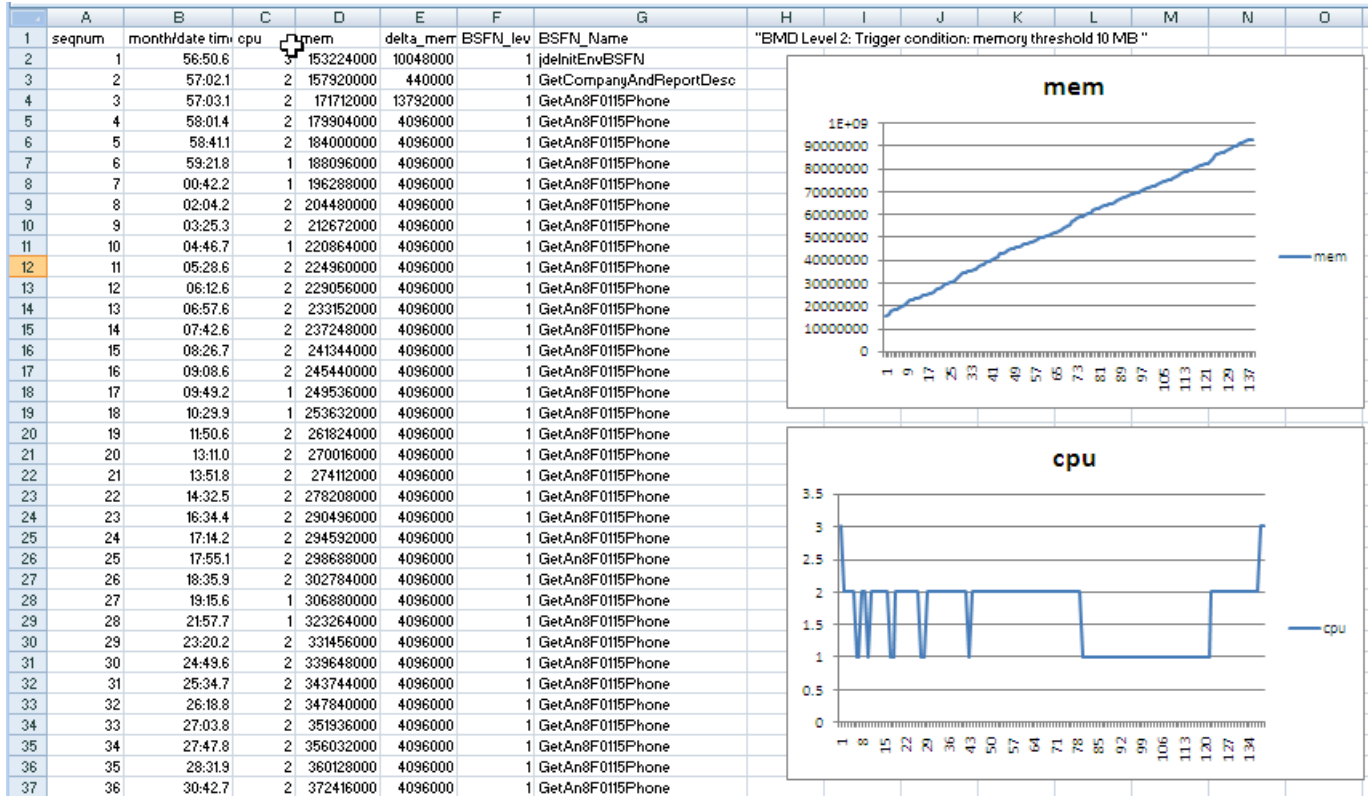
[Download Entire Log File](#) | [Download BMD Parse of Log File](#) | [Download BMD Parse of Log File \(No Zero Deltas\)](#)

Last Modified	10/5/09 2:38 PM	File Size	42,069,220
---------------	-----------------	-----------	------------

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.



Troubleshooting the 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.

Understanding 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 JOBLLOGs 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 PD920
Look for messages similar to these in the JOBLLOGs:
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.

Troubleshooting Enterprise Server Installation

This section explains topics that might create issues during the installation of an *IBM i* enterprise server.

Troubleshooting: Library Installation Verification

Issue	Resolution
You want to verify that the correct libraries and data dictionary items are installed on the <i>IBM i</i> .	See the list of libraries and data dictionary items and descriptions of their contents.

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: <i>IBM i</i> :table configuration problems <i>IBM i</i>	<ul style="list-style-type: none"> • 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.

Issue	Resolution
JDB3300011 - Failed to get location of table F983051 for environment PD920	

Troubleshooting: Setting up the .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, /E920sys/ini/JDE.INI.
You need more information on using the <i>IBM i</i> .INI file	Review the notes and descriptions of .INI settings.

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 /PSFT920/LogFiles, enter this command:

```
MKDIR DIR('/PSFT920') DTAAUT(*RWX) OBJAUT(*ALL)
```

Followed by:

```
MKDIR DIR('/PSFT920/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.

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 Inter-process 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.

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.	<p>The JD Edwards EnterpriseOne network might not be running. Clear the Interprocess Communication (IPC) structures using the JD Edwards EnterpriseOne <i>IBM i</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</p> <p>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.</p>
An error with the security server occurred.	<p>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.</p> <p>If the library list is correct, the problem could be because the activation group under which the job is running on the <i>IBM i</i> may retain some of the information from previous attempts. Log off, log on, and run the PORTTEST program again.</p>
An error with the security server occurred.	<p>The supplied user name or password might not match any names or passwords in the JD Edwards EnterpriseOne security table. Try one of these:</p> <ul style="list-style-type: none"> • 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.
<p>You get these message on the screen:</p> <pre>Invalid parms PORTTEST <USER> <PWD> <ENV></pre>	<p>You might not have included the correct number of arguments in the PORTTEST program. Use these arguments:</p> <p>User - A valid JD Edwards EnterpriseOne user ID.</p> <p>Password - Password for the JD Edwards EnterpriseOne user ID.</p> <p>Environment - A valid JD Edwards EnterpriseOne environment.</p>

Issue	Resolution
Fewer than 99 F0902 records are written to the screen by PORTTEST.	<p>A possible PORTTEST program failure. Examine the log files.</p> <ul style="list-style-type: none"> 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 <i>IBM i</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.

Troubleshooting: Running JDENET

Issue	Resolution
NETWORK dies immediately.	<p>IPCs might not have been cleared before starting JD Edwards EnterpriseOne (that is, starting JDENET using the JD Edwards EnterpriseOne <i>IBM i</i> command STRNET). End JD Edwards EnterpriseOne (ENDNET). Clear IPCs (using the CLRIPC command) and restart JD Edwards EnterpriseOne.</p> <p>The startIPCKeyValue in the .INI file could be used by another version of JD Edwards EnterpriseOne. Try one of these:</p> <ul style="list-style-type: none"> 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 <i>IBM i</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. <p>If not, correct those problems, and then try restarting JD Edwards EnterpriseOne using STRNET.</p> <ul style="list-style-type: none"> 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 <i>IBM i</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 <i>IBM i</i> Configuration guide to determine how to set this up correctly.
An error initializing the environment occurs in the log file.	<ul style="list-style-type: none"> 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.

Troubleshooting: Testing JD Edwards EnterpriseOne by Submitting a Report

Issue	Resolution
<p>You get this message:</p> <p>Communication Failure with <server name></p>	<p>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.</p> <p>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.</p> <p>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.</p> <p>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:</p> <ul style="list-style-type: none"> • If the remote machine is visible, a time-out probably occurred. <p>Rerun the report.</p> <ul style="list-style-type: none"> • 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.
<p>The report does not display any data.</p>	<p>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:</p> <ul style="list-style-type: none"> • 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. <p>If no data is found, it could be because:</p> <ul style="list-style-type: none"> • 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.
<p>The report does not display any data.</p>	<p>An error might have occurred with the report. Review the jddebug.log and jde.log files for errors.</p>

Issue	Resolution
<p>An error initializing the environment occurs in the log file.</p>	<p>The environment might not be set up correctly:</p> <ul style="list-style-type: none"> • 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.
<p>You get this message:</p> <p>Communication Failure with <server name></p> <p>This error occurs sometimes on the workstation</p> <p>Restarting JDENET_N sometimes gets rid of the error</p> <p>You can ping the server from the workstation</p>	<p>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.</p> <p>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.</p>

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.

Troubleshooting: Email and PPAT

Issue	Resolution
<p>The batch application, server package installation, or table conversion log file (in the PrintQueue directory) displays the message:</p> <p>PPAT:troubleshooting <i>IBM i</i></p> <p>XE Email:troubleshooting: <i>IBM i</i></p> <p>DoSendMessage Error: User 5600427 does not exist in the address book file (F0101).</p>	<p>The particular user may not be found in the F0101 table. Add the user to the F0101 table.</p>

Note:

- *Understanding the Library Structure for JD Edwards EnterpriseOne.*
- *Troubleshooting the JDE.INI File.*

Troubleshooting Multiple Release Setup

This table explains how to troubleshoot problems that can occur with multiple releases on the *IBM i*:

Issue	Resolution
<p>When you try to run multiple releases of JD Edwards EnterpriseOne at the same time, conflicts seem to occur between each release.</p>	<p>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:</p> <p>[JDEIPC]</p> <p>startIPCKeyValue</p> <p>[JDENET]</p> <p>serviceNameListen</p> <p>serviceNameConnect</p> <p>Variable names and descriptions:</p> <p>startIPCKeyValue</p> <p>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.</p> <p>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.</p> <p>serviceNameListen</p> <p>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.</p> <p>serviceNameConnect</p> <p>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.</p> <p>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.</p>

Troubleshooting JDBNET

Note: As of Release 9.2.3, JDBNET is no longer supported.

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.
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: <ul style="list-style-type: none"> 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 <i>IBM i</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.

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: <ul style="list-style-type: none"> The PORTTEST program fails. The security server on the <i>IBM i</i> fails. UBE submissions fail. If you activated ipcTrace in the [JDEIPC] section of the server jde.ini file, an error	This could be because the <i>IBM i</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. <ul style="list-style-type: none"> Use the *CNTRL D option to end an <i>IBM i</i> job. Note: You might still have damaged IPC message queues if the <i>IBM i</i> -controlled ending times out. <ul style="list-style-type: none"> Run these program to verify whether a damaged message queue exists. You must have V4R1 PTF# SF45946. CALL QPOZIPCS PARM('-aqE')

Issue	Resolution
<p>similar to this should appear in the JDEDEBUG.log:</p> <pre>IPC2100017 createIPC Msgq (name Port6005) failed, errno=3484: A damaged object was encountered</pre>	<p>This program generates a spool file called IPCS that contains information about message queues on the system. Look for these output:</p> <pre>KEY MODE 0x00000000 ----- 0x00000000 --RW----- 0x00000000 --RW----- 0x00000000 --RW----- 0x00001234 D-RW----RW-</pre> <p>In this example, the message queue 0x00001234 is damaged. To fix, stop, and restart JDENET using these commands:</p> <pre>ENDNET CLRIPC STRNET</pre> <p>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.</p> <p>Note: Some of the message queues might be damaged even if the JDEDEBUG.log file does not indicate that any damage exists.</p>

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 E92OSYS 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.

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.

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 `jddebug_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 `jddebug_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`.

Note: To complete the resolutions provided for these issues, you must sign on to the UNIX enterprise server using an account that has administrative privileges.

Troubleshooting the JDE.INI File

To locate the `JDE.INI` file, search in the `system/bin32` subdirectory. For example, `/u01/JDEdwards/E920/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.

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.

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.

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.

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.

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.

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/E920SYS/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.

Troubleshooting JDBNET Server Not Found

Note: As of Release 9.2.3, JDBNET is no longer supported.

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.

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.

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.

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 `jddebug_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 `jddebug_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 these issues, you must sign on to the Microsoft Windows enterprise server using an account that has administrative privileges.

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.

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.

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.

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.

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\E920\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.

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 this 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 Inter-process 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).

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. 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.

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.

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\E920\ddp\PrintQueue.
- Verify that SaveOutput in the [NETWORK QUEUE SETTINGS] section in the jde.ini file on the workstation is TRUE.

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, see *Overriding Printer Location for Jobs*.
 - 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 `jddebug.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.

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 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 tasks:

- a. From the Microsoft Windows Start menu, select Settings, Printers.
- b. Right-click the desired printer.
- c. Select Document Defaults.
- d. Select the desired default orientation.
- e. Click OK.

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.

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.

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.

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.

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.

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.

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:\E920\internet\jas.log or ;debuglog=d:\E920\internet\jasdebuglog). If the log file paths are not correctly stipulated, the logs may be writing to a file located elsewhere.

If JAS performance response is 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.

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 E920\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 (prior to release 9.2.2) 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.

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 F989999 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.

Troubleshooting Problems Using Log Files (Release 9.2.5.3 Update)

If you need to view logging information for the Java client, open the Java Console in the browser of your choice. 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.

When you enable the Java Console in your browser, turn on 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 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.

7 Working with Servers

Understanding the Work With Servers Program (P986116)

The Work With Servers program (P986116) in Oracle's JD Edwards EnterpriseOne provides a central location from which system administrators can monitor and control:

- Server jobs
- JD Edwards EnterpriseOne subsystems

As a system administrator, you can use the Work With Servers program to print, view, remove, terminate, release, or hold any jobs that currently reside in a queue on any JD Edwards EnterpriseOne server. Similarly, workstation users can control only those jobs submitted by them. This option is generally restricted to only those jobs associated with a specific user ID.

Also, you can use the Work With Servers program to end and to stop JD Edwards EnterpriseOne subsystems, and to view the status of JD Edwards EnterpriseOne subsystems that are running or are waiting to process jobs.

Understanding the Work With Locations and Machines Web Program (Release 9.2.5)

Note: It is recommended to restrict this application to EnterpriseOne administrators.

The Work With Locations and Machines Web (P9654W) program enables you to define or revise machines and locations in your enterprise. In addition, you can use the Work With Locations and Machines Web (P9654W) program to define deployment locations. When you add a new deployment server definition, the system creates a record in the F9650 table for each deployment location. Each server at each deployment location must be defined.

In addition to defining workstations, you can also use the JD Edwards EnterpriseOne Deployment Locations application to enter or revise definitions for these machines:

- Deployment Server
- Enterprise Server
- Virtual Host
- Data Server
- Java Application Server
- Windows Terminal Server
- Business Services Server

You can enter or revise definitions for these machines in multiple locations, including remote locations.

Defining Machines

Access the Work with Locations and Machines form.

1. Highlight the type of machine you would like to create. You can choose from:
 - o Workstations
 - o Deployment Server
 - o Enterprise Server
 - o Virtual Host
 - o Data Server
 - o HTML Server
 - o Business Services Server
2. Click Add.
3. Enter the appropriate values for the type of machine you are creating.

Workstation

Deployment Server Name

Enter the name of the specific server that is being used for deployment.

When you define a secondary deployment server, options on the Form menu enable you to select path codes, data items, foundation modules, and help items. (These options are not available for the primary deployment server.)

Deployment Server

Primary Deployment Server

Specify whether a deployment server is the primary deployment server for a specific location.

If you have set up a primary deployment server, you cannot access the Primary Deployment Server field when you define a new deployment server. You can change the value in this field only when you revise the primary deployment server definition or when you change the primary deployment server to a secondary server. In this case, you can specify a different server as the primary deployment server.

Server Share Path

Enter the shared directory for this path code. The objects that are stored on a file server will be found in this path.

Enterprise Server

Port Number

Identify the port for a given instance of JD Edwards EnterpriseOne. Because the jde.ini file controls the port to which a workstation will connect, for workstations this port number is for reference only.

Logical Machine Name

Enter the logical machine name that is assigned to this unique machine and port. A machine can be a workstation. Because you can have more than one instance of JD Edwards EnterpriseOne running on a given machine, you must assign a logical machine name that identifies the unique physical machine name and port where this instance runs.

The logical machine name should represent the release and purpose of the machine, such as Financial Data Server-E920 or Distribution Logic Server-E920.

Database Type

Enter the type of database.

Server Map Data Source

Enter the name that identifies the data source.

Installation Path

Enter the path on which JD Edwards EnterpriseOne is installed.

Deployment Server Name

Enter the name of the specific server that is being used for deployment.

Server Availability

This field is visible only in Update mode. Use this field to reset the enterprise server status if a package deployment failed.

Virtual Host

Machine Usage

A default value of 21 indicates that it is a virtual host.

Machine Name

Specify a user defined name that identifies the machine. Virtual hosts are not eligible for package deployment, use the individual enterprise servers instead.

Description

Specify a user defined remark that describes the machine.

Release

Specify the release number as defined in the Release Master. Specify the host machine type.

Host Type

Specify the host machine type.

Primary User

Specify the primary user for the listed machine.

Port Number

Identify the port for a given instance of JD Edwards EnterpriseOne. This port number must match with the port number defined in the enterpriser server. Since the jde.ini file controls the port to which a workstation will connect, this port number is only for reference for workstations.

Data Server

Data Source Type

Enter the type of database.

HTML Server

Protocol

Specify the method of communication (for example, http).

Server URL

Enter the URL to the web server.

Http Port

Enter the port number of the web server.

Default Login

Enter the login path.

Installation Path

Enter the path on which JD Edwards EnterpriseOne is installed.

Deployment Server Name

Enter the name of the specific server that is being used for deployment.

Business Services Server

The Business Services Server cannot be added through this application. You must use Server Manager to add a new Business Services Server.

See the [JD Edwards EnterpriseOne Tools Server Manager Guide](#) .

Understanding the Work With Virtual Hosts Program (Release 9.2.5)

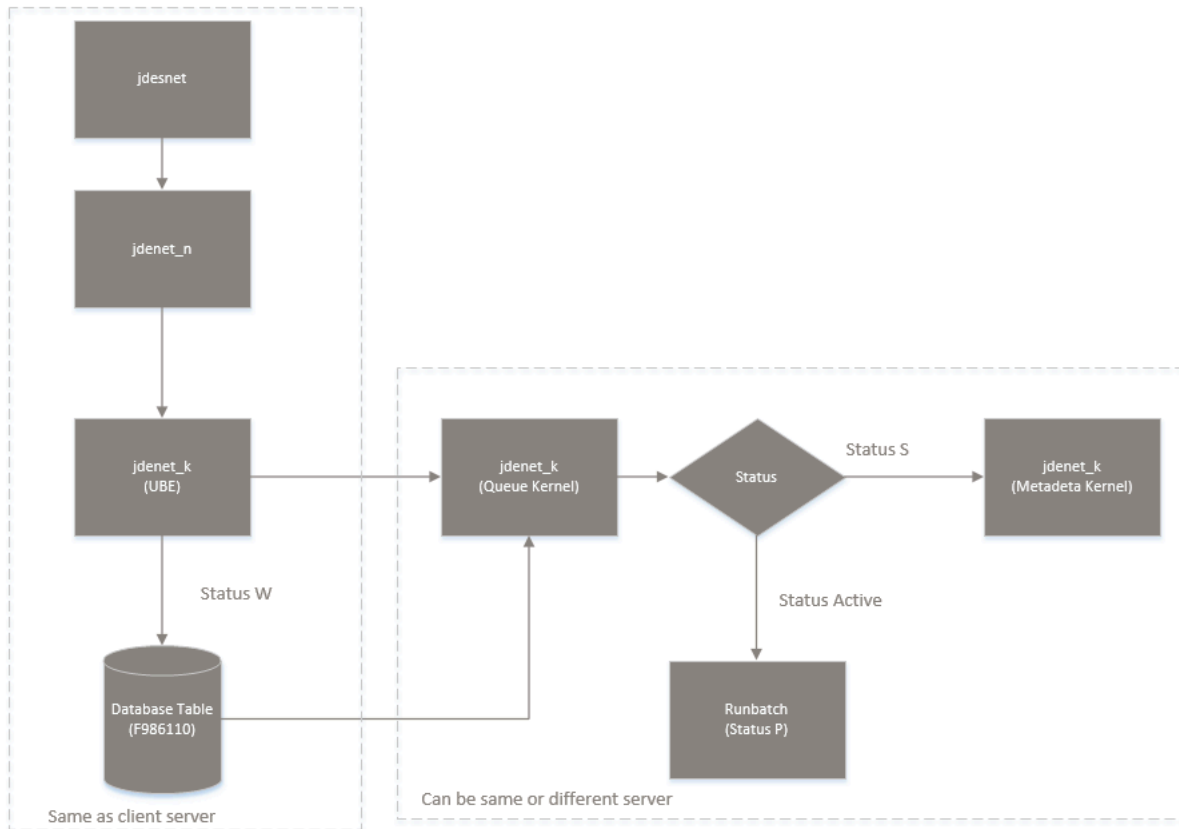
You can use the Work With Virtual Hosts program (P9655) to instruct a queue kernel to stop scheduling jobs on that server, or to start scheduling jobs on the server. A green circle on the Work With Virtual Hosts program indicates that the queue kernel is scheduling on a server using a green circle. A yellow triangle indicates that the queue kernel is going to an inactive state suggesting that no new jobs are being scheduled but some are still running (subsystem UBEs). A red square indicates that the queue kernel on that host is inactive (not scheduling jobs) and that all jobs have stopped running.

Understanding Virtual Hosts Architecture

A job submitted to a queue on an enterprise server could end up being processed by any enterprise server that is defined as part of a virtual host. A virtual host name will be used for the batch cloud.

Servers can be added or removed from the batch cloud virtual host without having to re-route UBEs that are waiting to be processed.

This illustration displays the virtual hosts architecture.



Managing Server Jobs

This section provides an overview of server jobs and discusses how to:

- Set processing option for Work With Servers.
- Check the status of reports.
- Change the priority and the printer for jobs.
- Print jobs.
- View reports online.
- View the logs for a job.
- Terminate jobs.
- Hold and release jobs.

Understanding Server Jobs

By using the Work With Servers application, system administrators can print, view, and delete job records from the job queue. They can also terminate, release, or hold any jobs that currently reside in a queue on any JD Edwards

EnterpriseOne server. Similarly, using the Submitted Job Search form, users can, in general, control only those jobs submitted by them.

You should use JD Edwards EnterpriseOne security to restrict access to the Work With Servers application. Administrators should have access to both the ZJDE0001 or ZJDE0002 version of the program. End-users should be restricted to the ZJDE0001 version, which is known as the Submitted Job Search form. This version of the application restricts users to viewing and modifying only those jobs that were submitted under their user ID initially. Both programs are located on the System Administration Tools menu (GH9011).

Job Status and Priority

After you submit the report, you can check the status of the job in the queue. Depending on the status of the job, you can perform tasks such as printing or deleting the report, viewing the report output online, and holding the report in the queue.

You can also move the priority of the job to a lower or higher status while the job is at the status of W (Waiting).

Overriding Printer Location for Jobs

You can override the location where the job prints. For jobs with a status of D (Done) and E (Error), you can send the job directly to the default printer without viewing the PDF file online. A status of D means that the processing for the job completed successfully. A status of E means that an error occurred during processing. If you print a job with a status of E, you print an error log to aid you when you troubleshoot the report.

Viewing Reports Online

After the job finishes processing on the server, you can view the report output online. For most jobs, the output is in Portable Document Format (PDF), which can be viewed with Adobe Reader. When you view the report output online, the system also creates a PDF file for the report in the following directory on the workstation:

```
\E920SYS\PrintQueue
```

You can attach PDF files to email messages; move or copy the files; and, because most current web browsers can read PDF files, post the reports to a web site. Also, you can copy text from Adobe Reader to the clipboard and paste that text into other applications.

Job Logs

You can view logs that detail the steps taken while the job processed. From the Submitted Job Search form, you can access the `jde.log` and the `jddebug.log` for the report. These logs are helpful if you need to troubleshoot a report that resulted in error. These logs exist on the machine where the job ran.

The `jde.log` is a general-purpose log used to track error messages generated by JD Edwards EnterpriseOne processing. The `jde.log` tracks any fault that might occur within the software, including whether the sign on is successful. 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 `jddebug.log` contains API calls, BSFN logs, and SQL statements, as well as other messages. You can use this log to determine at what time normal execution stopped. The system does not use the `jddebug.log` to track errors; instead, it uses this log to track the timing of processes.

Holding and Releasing Jobs

If a job is at the status of W (waiting), you can hold the job. You might choose to hold a job if the job is large enough to affect the performance of the server on which it processes. You can release a job when server performance is not an issue, such as after regular business hours.

Note: If you want to stop a job that is at a status of P (Processing), you must terminate the job. You cannot restart a job after you terminate it; you must resubmit the job to the server.

Move Jobs (Release 9.2.5)

You can manually move a job from a current host to another or a current queue to another. When you move a job, you can choose jobs that are in Hold status or Wait status.

Resubmit Jobs (Release 9.2.5)

You can manually resubmit a job from a current host to another or a current queue to another. When you resubmit a job, you can choose jobs that are in Done status, Error status or Terminated status.

Terminating Jobs (Release 9.2.6)

You can manually terminate a job that is processing. Prior to Release 9.2.6, the status of the terminated job moved to E (Error) and no details were recorded in the execution details. Starting with Release 9.2.6, the status of the terminated job moves to T (Terminated). When you terminate a job, it is not deleted. With the job at the status of T, you can view a log, and delete or resubmit the job.

You can check the details of terminated jobs. To check the details of a terminated job:

1. Access Submitted Jobs.
2. Select the job with T (Terminated) status.
3. Use the row exit in Submitted Job Search to access Execution Details.
4. In the Submitted Job Execution Detail form, click the Audit Data tab to view the start and end date and the time (time of termination) of the terminated job and the user who terminated the job.

Submitted Reports - Submitted Job Execution Detail

✓ ✕ ⌂ Eoim ⚙ Tools

Debug Logging Status Turn Logging On

Introspection Status Turn Introspection On

Job Status Verify Statuses

Introspection Data **Audit Data** Processing Options Data Selection Data Sequencing Report Interconnect SQL Statement

Host Name	dvcluster4072	Audit Report	R0008P
Job Number	387	Audit Version	XJDE0001
Environment Name	JDV920		
Process ID	16104		
Start Date and Time	20/08/2021 08:54:15	Job Terminated By	JDE
End Date and Time	20/08/2021 08:54:27	Job Process Time	0:00:12
		Rows Processed	0

Forms Used to Manage Server Jobs

Form Name	FormID	Navigation	Usage
Work With Servers	W986116A	System Administration Tools (GH9011), Data Source Management, Work With Servers (P986116).	Select a server in which you want to locate a job.
Submitted Job Search	W986110BA	On the Work With Servers form, from the Row menu, select Server Jobs.	Print, terminate, hold, release, or view a job. Users can manage jobs submitted by their user ID initially. Depending on the security level, you can change the User ID field and the Job Queue field to search for other jobs.
Job Maintenance	W986110BC	On the Submitted Job Search form, select a job with which to work and click Select.	Review information about the batch job, modify the priority of the job, or change the printer on which the job will print.
Printer Selection	W986162B	On the Submitted Job Search form, select a job and then select Print from the Row menu.	Override printer-specific information.
View Logs	W986110BD	On the Submitted Job Search form, select the job for which you want to view a log, and then select View Logs from the Row menu.	View the <code>jde.log</code> and the <code>jddebug.log</code> .
Move Jobs	W986110BH	On the Submitted Job Search form, select Move Jobs Logs from the Form menu.	Manually move a job (or multiple jobs) from a current host to another or a current queue to another or both.
Resubmit Jobs	W986110BH	On the Submitted Job Search form, select Resubmit Jobs Logs from the Form menu.	Manually resubmit a job (or multiple jobs) from a current host to another or a current queue to another or both.

Setting Processing Option for Work with Servers (P986116)

Although processing options are set up during the JD Edwards EnterpriseOne implementation, you can change processing options each time you run a batch application.

1. Security Flag

Use this processing option to specify how submitted jobs can be viewed. Values are:

Blank

No Security

1

Allow users to view jobs by group.

2

Allow users to view only their own jobs.

Setting Processing Options for Job Control Master (P986110B) (Release 9.2.5)

The Job Control Master (P986110B) program has processing options that enables or disables user form controls.

Process

Although processing options are set up during EnterpriseOne implementation, you can change processing options each time you run a program.

1. This value can be left blank. It is not used in the application at all. It is only used so that a version can be created for this application so that it is called correctly from all locations.

2. Allow login user to resubmit other user's jobs

Specify whether the login user can resubmit other user's jobs. Values are:

Blank: Do not allow login user to resubmit other user's jobs.

1: Allow login user to resubmit other user's jobs.

3. Allow other user jobs to be moved by login user.

Specify whether the login user can move other user's jobs. Values are:

Blank: Do not allow login user to resubmit other user's jobs.

1: Allow login user to move other user's jobs.

Checking the Status of Reports

Access the Submitted Job Search form.

User ID

Change the user ID if you want to work with a report submitted by a different user. You can use a wildcard (*) to find a specific user. The default user ID is the user logged on to the current session.

Job Queue

Enter the name of the logical queue on the server for which you want to view jobs.

Status

Click the search button in the Status field to read the UDCs for status codes in the installation.

Changing the Priority and the Printer for Jobs

Access the Work With Servers form.

1. Select a server with which to work and, from the Row menu, select Server Jobs.

By default, the Submitted Job Search form lists jobs for the User ID for the requesting workstation. Depending on the application security level, you can change the User ID field and the Job Queue field to search for other jobs.

Note: A job must be at a status of W (Waiting) to change the priority.

2. Select a job with which to work and click Select.
3. On the Job Maintenance form, modify the information in the Job Priority field and click OK. The value that you enter in this field determines how the job will execute based on this priority. Values 0-9 are valid, where 0 is the highest priority.

Printing Jobs

Access the Work With Servers form.

1. From the Row menu, select Server Jobs.
2. On the Submitted Job Search form, select the job that you want to print, and then choose Print from the Row menu.

The Printer Selection form appears. This form provides printer-specific information as well as information about the format of the report.

3. To print the job, click OK.

Viewing Reports Online

Access the Work With Servers form.

Note: Before you view the report online, verify that you have Adobe Reader installed on the workstation.

1. Select a server from the list and then click Select or select Server Jobs from the Row menu.
2. On the Submitted Job Search form, select the job that you want to view and then select View Job from the Row menu.

Adobe Reader displays an online version of the report output.

Viewing the Logs for a Job

Access the Work With Servers form.

1. Select the server that processed the job that you want to view, and click Select, or select Server Jobs from the Row menu.
2. On the Submitted Job Search form, select the job for which you want to view a log, and then select View Logs from the Row menu.
The View Logs form appears. On this form, you can view the `jde.log` and the `jddebug.log`.
3. Click OK to view the logs.

Note: If you choose both the `jde.log` and the `jddebug.log`, the logs open in the same window. To view the logs separately, you must select the logs separately.

Holding and Releasing Jobs

Access the Work With Servers form.

1. Select a server from the list or use the query by example row to select a specific server.
2. Click Select or select Server Jobs from the Row menu.
The Submitted Job Search form appears.
3. To hold a job, select the job and then select Hold from the Row menu.
4. Click Find to update the detail area.
The status of the job changes to H (Hold).
5. To release a job, select the job and then select Release from the Row menu.
The job must be at the status of H (Hold).
6. Click Find to update the detail area.
The status of the job changes to reflect the position of the job in the queue, for example, W (Waiting), S (In Queue), or P (Processing).

Moving Jobs (Release 9.2.5)

Access the Work With Servers form.

1. Select a server from the list or use the query by example row to select a specific server.
2. Click Select or select Server Jobs from the Row menu. The Submitted Job Search form appears.
3. To move a job, select Move Jobs from the Form menu.
4. On the Submitted Jobs –Move Jobs form, enter the current host and queue from where you want to move the job and the new Host and Queue to which you want to move the job.
5. Click OK.

Resubmitting Jobs (Release 9.2.5)

Access the Work With Servers form.

1. Select a server from the list or use the query by example row to select a specific server.
2. Click Select or select Server Jobs from the Row menu. The Submitted Job Search form appears.
3. To resubmit a job, select Resubmit Jobs from the Form menu.
4. On the Submitted Jobs –Resubmit Jobs form, enter the current host and queue where the job resides and the new Host and Queue from which you want to resubmit the job.
5. Click OK.

Terminating Jobs (Release 9.2.6)

Access the Work With Server Jobs form.

1. Select a server from the list or use the query by example row to select a specific server.
 2. Click Select or select Server Jobs from the Row menu.
- Note:** You can terminate a job only if the job status is P (Processing).
3. On the Submitted Job Search form, select the job to terminate, and then select Terminate from the Row menu.
 4. Click Find to update the detail area.

The status of the job changes to T (Terminated). Prior to Release 9.2.6, the terminated job status changed to E (Error).

Defining Default Logging Levels for Jobs (Release 9.2.6)

To define default logging level, access the Work With Default Output Locations program (P98617).

1. Select Report Logging from the Form exit to access Work With Default Report Logging Options.
2. Click Add to create a new record or select an existing record for which you want to define the default logging level.
3. In the Default Report Logging Option Revisions form, complete the following fields:

Field	Description
<i>User/Role</i>	Define the user for the report logging level.
<i>Report Name</i>	Click the visual assist to select the report for which you want to define the logging level. Enter *ALL to apply the default logging level for all reports. The *ALL entry does not apply to child reports if the reports do not have a specific override setting. They inherit the settings of their respective parent reports.
<i>Version Name</i>	Click the visual assist to select the version of the report for which you want to define the logging level. If the value in the Report Name field is *ALL, the system populates the Version Name field with the default value of *ALL and makes the field unavailable for subsequent modification.

Field	Description
<i>Environment</i>	JD Edwards EnterpriseOne provides a default value for this field based on the environment that you are currently logged onto. Enter *ALL for all environments. You can change this information.
<i>Default Report Logging Option</i>	You can select Logging (JDE.log) or Tracing (JDEDEBUG.log) as default report logging option. If you select Tracing, Logging is automatically selected.
<i>UBE Logging Level</i>	Enter a value from 0 through 6 to indicate the level of detail to be captured in the logs. Alternatively, click the visual assist to open the Select User Defined Code window where you can see the descriptions of the values. Select the appropriate value and click Select.
<i>Host Name</i>	Enter the name of the host server where reports will be processed. The visual assist displays the appropriate host names based on the report logging option you select. To use the default report logging option for all the hosts in your environment, enter *ALL in Host Name.
<i>Object Status</i>	Define the default logging level as active by changing its status to active.

Note: The logging level for a child UBE cannot go lower than the logging level of the parent UBE. However, you can turn up the logging level of the child UBE. After you configure the logging level, it is applicable for all the submission paths.

Using the System Hierarchy to Resolve Default Report Logging Level

The general hierarchy that the system uses to resolve default report logging level that is associated with more than one report or version is illustrated in the following table. The system uses the same hierarchy for each user or role, processing in this order:

1. Username
2. Role
3. *PUBLIC

Report	Version	Environment	Host
report	version	environment	hostname
report	version	environment	*ALL
report	version	*ALL	hostname
report	version	*ALL	*ALL
report	*ALL	environment	hostname
report	*ALL	environment	*ALL
report	*ALL	*ALL	hostname
report	*ALL	*ALL	*ALL
*ALL	*ALL	environment	hostname
*ALL	*ALL	environment	*ALL
*ALL	*ALL	*ALL	hostname
*ALL	*ALL	*ALL	*ALL

Managing Job Queues

This section provides an overview of job queues and discusses how to:

- Add a job queue.
- Copy a job queue.
- Change the status of a job queue.
- Override a job queue.
- Manage virtual job queues.

Understanding Job Queues

Each JD Edwards EnterpriseOne server instance starts a queue kernel process that manages batch processes across operating system platforms. The process keeps track of all jobs that are submitted and controls the order in which the jobs run.

JD Edwards EnterpriseOne uses two tables to maintain queue records:

- Job Control Status Master table (F986110), which maintains records on the status of each job submitted to a queue.
- Queue Control Status Master table (F986130), which stores the names of each queue, such as QBATCH, the name of the server on which the queue runs, the port number for the server instance, the queue status and type, and the maximum number of active jobs allowed.

Note: Since F986130 is a system table, be sure to account for it when you map objects using Object Configuration Manager (OCM).

The following list summarizes how the software, using the queue kernel, manages a UBE that you launch:

- Starts queue kernel when the server instance starts.
- Verifies that a record exists in the F986130 table for the queue to which the job is submitted. If the job is intended for a non-EnterpriseOne queue, verifies that the native queue (for example, *IBM i*) exists.
- Inserts job record into the F986110 table.
- Sends a message to the queue kernel that the new job exists.
- Adds the job to a wait list.
- Schedules the job or submits it to the native queue.
- Starts the job.
- Runs the job.
- Updates the job record in the F986110 table upon receiving a message from the UBE process that the job is complete.
- Removes the job from the list of active jobs.
- Schedules another job.

The queue kernel also follows an algorithm when scheduling jobs. The following list summarizes the algorithm that the queue kernel follows:

- Verifies that jobs in the queue are waiting to be run.
- Verifies that the number of jobs waiting to be run is less than the maximum number of jobs allowed for the queue.
- Takes the highest priority job from the wait list and updates its status to S (Submitted).
- Removes the job from the wait list and adds it to the active list.

Administering Job Queues

Use the Job Queue Maintenance application (P986130) to define and manage job queues. This application enables you to dynamically administer job queues. You can use this application to create, modify, copy, delete, or change the status of job queues, regardless of platform. For example, you can use this application to add a queue record to the Queue Control Status Master table (F986130). You can also revise an existing queue record. For example, you might want to change the maximum number of jobs that can run in a queue.

In addition, when you set up job queues, you can define a default queue in which to submit jobs.

Overriding a Job Queue

When you prepare to submit a batch application, you can change the values of the parameters that define the submission by overriding the job queue. Overriding the job queue means that you change the job queue to which the job is submitted on the server.

To override the job queue for a batch version, you launch the Batch Versions application (P98305), select a batch version, and access the Advanced Version Prompting form (W98305I). The override queue must be one that is available for the server and port.

In working with the Advanced Version Prompting form, you can override the job queue only if the queue kernel is active and if the batch version is mapped to run on the server. If the batch version is mapped to run locally, you cannot override the job queue, even if the queue kernel is active, unless you select the Override Location option.

Note: Overriding the job location means that you change the machine that will run the batch application. For example, a batch application might run locally by default. You can override the processing location to a server, and the batch application will run on the server. Conversely, you can change the processing location from a server to a workstation.

JD Edwards EnterpriseOne displays a Verify Overriding the Job Queue form if the job runs locally and you do not override the processing location.

The status of the queue kernel and the default processing location for the batch application determine the way the Override Job Queue option appears in the Advanced Version Prompting form. The following table summarizes the queue kernel status and processing location combinations that can occur, and the effect each combination has on the Override Job Queue option:

Queue Kernel Status	UBE Processing Location	Status of Job Queue Override Option
Inactive	Local or server	Not visible
Active	Local	Visible but disabled

Queue Kernel Status	UBE Processing Location	Status of Job Queue Override Option
Active	Local, but Override Location option chosen	Enabled
Active	Server	Enabled

Prerequisites

Before you work with job queues, you must activate the queue kernel. To do so, perform these tasks:

- Make sure that the server's jde.ini file contains these settings:

```
[JDENET_KERNEL_DEF14]
krnlName=QUEUE_KERNEL
dispatchDLLName=jdekrnl.dll
dispatchDLLFunction=_DispatchQueueMessage@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1

[DEBUG]
QKLog=0
```

The QKLog controls how much debug information is in the debug log specific to Queue Kernel when debug logging is turned on. A value of 0 means that minimal queue kernel logging is generated. You can change the setting to 1 if you need to generate detailed queue kernel information in your debug logs for troubleshooting purposes.

```
[NETWORK_QUEUE_SETTINGS]
QKActive=1
QKOnIdle=300
```

Where a value of 1 means that the queue kernel is active and a value of 300 sets the queue kernel on idle time to 300 seconds.

- Add the following setting to the client jde.ini file:

```
[NETWORK_QUEUE_SETTINGS]
QKActive=1
```

Forms Used to Manage Job Queues

Form Name	FormID	Navigation	Usage
Work With Job Queues	W986130A	Batch Processing Setup menu (GH9013), Job Queues (P986130).	Add a job queue and change the status of a queue.

Form Name	FormID	Navigation	Usage
Job Queue Revisions	W986130B	On the Work With Job Queues form, click Add.	Add information for a new job queue. Revise or copy a job queue.
Work With Batch Versions - Available Versions	W98305A	Report Management (GH9111), Batch Versions (P98305).	Override a job queue.
Version Prompting	W98305D	On the Work With Batch Versions - Available Versions form, select a version, and then click Select.	Access the Advanced Version Prompting form. Enter data selection or data sequencing.
Advanced Version Prompting	W983051	On Version Prompting, select Advanced from the Form menu.	Select the Override Job Queue option.
Job Queue Search	W986130C	On Version Prompting, click Submit.	Find and select the job queue that you want to override.
Virtual Hosts Queue (Release 9.2.5)	W986130E	On the Work with Job Queues, select Virtual Hosts Queues from the Form menu.	Allow a view of all queues defined on all hosts on the Virtual Server to assist configuration.
Work With Default Report Job Queues (Release 9.2.6)	W986170	From the Batch Processing Setup (GH9013), select Default Output Location (P98167), and then select Job Queue from the Form menu.	Configure options to define default job queues.

Adding a Job Queue

Access the Work With Job Queues form.

1. Click Add.
2. On the Job Queue Revisions form, in the Host field, enter the name of the server on which the queue will run.
3. Enter the name of the queue in the Job Queue field.
4. In the Job Queue Status field, enter **01** if you want the queue to be active, or **02** if you want the queue to be inactive.
5. In the Queue Type field, define whether the queue is a JD Edwards EnterpriseOne queue or a non-EnterpriseOne queue.

A non-EnterpriseOne queue works only on the *IBM i* server.

6. In the Maximum Batch Jobs field, define the maximum number of jobs that can run in the queue.
7. In the Port Number field, specify the port number for the server instance on which the queue will run.
8. In the Default Queue field, check the box for the default queue, or leave it blank for a non-default queue.

Copying a Job Queue

Access the Work With Job Queues form.

1. On the Work With Job Queues form, find the queue that you want to copy and click Copy.
2. On Job Queue Revisions, you can modify any of these fields:
 - o Host
 - o Job Queue
 - o Job Queue Status
 - o Queue Type
 - o Maximum Batch Jobs
 - o Port Number
 - o Default Queue
3. Click OK to complete the copy.

Changing the Status of a Job Queue

Access the Work With Job Queues form.

1. On the Work With Job Queues form, find the queue whose status you want to change.
2. From the Row menu, select Change Status.

JD Edwards EnterpriseOne changes the status of the queue from Active to Inactive or from Inactive to Active, depending on its previous status.

Overriding a Job Queue

Access the Work With Batch Versions - Available Versions form.

1. On the Work With Batch Versions - Available Versions form, find a version of a job that you want to submit and click Select.
2. On the Version Prompting form, select Advanced from the Form menu.
3. On the Advanced Version Prompting form, select the Override Job Queue option and click OK.
4. In the Version Prompting form, select either, both, or neither of these options and click Submit:
 - o Data Selection
 - o Data Sequencing
5. On the Job Queue Search form, find the name of an available queue for the host and port name.
6. Select the queue that you want to override to and click Select.
7. Complete the data selection and sequencing and the processing options required to submit the job and select a printer, if necessary.

Managing Virtual Job Queues (Release 9.2.5)

Note: As a prerequisite, you must add a default queue using the name of the virtual host on the port.

Access the Work With Job Queues form.

1. On the Work With Job Queues form, select Virtual Host Queues from the Form exit.
2. On the Work With Virtual Host Queues form, use the form headers or the query by example to search for Virtual Host Name and Port Number.
3. Select the record to view and manage the job queues available on the selected virtual host.

Defining and Using Default Job Queues (Release 9.2.6)

To define a default job queue, access the Work With Default Output Locations program (P98617).

1. Select Job Queues from the Form exit to access Work With Default Report Job Queues.
2. Click Add to create a new record or select an existing record for which you want to define the default job queue.
3. In the Default Report Job Queue Revisions form, complete the following fields:

Field	Description
<i>User/Role</i>	Define the user for the default report job queue.
<i>Report Name</i>	Click the visual assist to select the report for which you want to define the default report job queue. Enter *ALL to apply the report job queue to all the reports. The *ALL entry does not apply to child reports if they do not have a specific override setting. They inherit the settings of their respective parent reports.
<i>Version Name</i>	Click the visual assist to select the version of the report for which you want to define the default report queue. If the value in the Report Name field is *ALL, the system populates the Version Name field with the default value of *ALL and makes the field unavailable for subsequent modification.
<i>Environment</i>	JD Edwards EnterpriseOne provides a default value for this field based on the environment that you are currently logged onto. Enter *ALL for all the environments. You can change this information.
<i>Default Report Job Queue</i>	Click the visual assist to open Job Queue Search & Select. Select the default queue you want the report to run in.
<i>Host Name</i>	Enter the name of the host server where reports will be processed. The visual assist displays the appropriate host names based on the default report job queue you select. To use this default report job queue for all the hosts in your environment, enter *ALL.
<i>Object Status</i>	Define the default report job queue as active by changing its status to active.

Note: If you call a child job with a synchronous report interconnect, it will run in its parent job queue because a synchronous job runs inside the parent job's process ID. However, if a parent job calls a child job with an asynchronous report interconnect, the child job can be mapped to any queue. After you have configured the default job queue, it is applicable for all submission paths.

Using the System Hierarchy to Resolve Default Job Queue

The general hierarchy that the system uses to resolve a default job queue that are associated with more than one report or version is illustrated in the following table. The system uses the same hierarchy for each user or role, processing in this order:

1. Username
2. Role
3. *PUBLIC

Report	Version	Environment	Host
report	version	environment	hostname
report	version	environment	*ALL
report	version	*ALL	hostname
report	version	*ALL	*ALL
report	*ALL	environment	hostname
report	*ALL	environment	*ALL
report	*ALL	*ALL	hostname
report	*ALL	*ALL	*ALL
*ALL	*ALL	environment	hostname
*ALL	*ALL	environment	*ALL
*ALL	*ALL	*ALL	hostname
*ALL	*ALL	*ALL	*ALL

Managing JD Edwards EnterpriseOne Subsystems

This section provides an overview of JD Edwards EnterpriseOne subsystems and discusses how to:

- Locate subsystems running on a server.
- Review job records for subsystems.
- Terminate subsystems.

Understanding JD Edwards EnterpriseOne Subsystems

Within JD Edwards EnterpriseOne, subsystems are defined as continuously running batch jobs that run independently of, and asynchronously with, JD Edwards EnterpriseOne applications. Subsystem jobs function within the logical process of the operating system or the queue defined for the server platform. You can configure JD Edwards EnterpriseOne to use one or more subsystems.

The term *subsystem* is an industry-wide generic term that usually indicates a system that is a sub-process to an operating system. On *IBM i* server platforms, a subsystem is a logical process that is used to run system jobs, whether

they are JD Edwards EnterpriseOne or other application jobs. For UNIX, JD Edwards EnterpriseOne subsystem is functionally equivalent to a daemon. On UNIX and Windows server platforms, system jobs are processed in queues; these queues are functionally equivalent to subsystems on the *IBM i* platform.

How JD Edwards EnterpriseOne Uses Subsystems

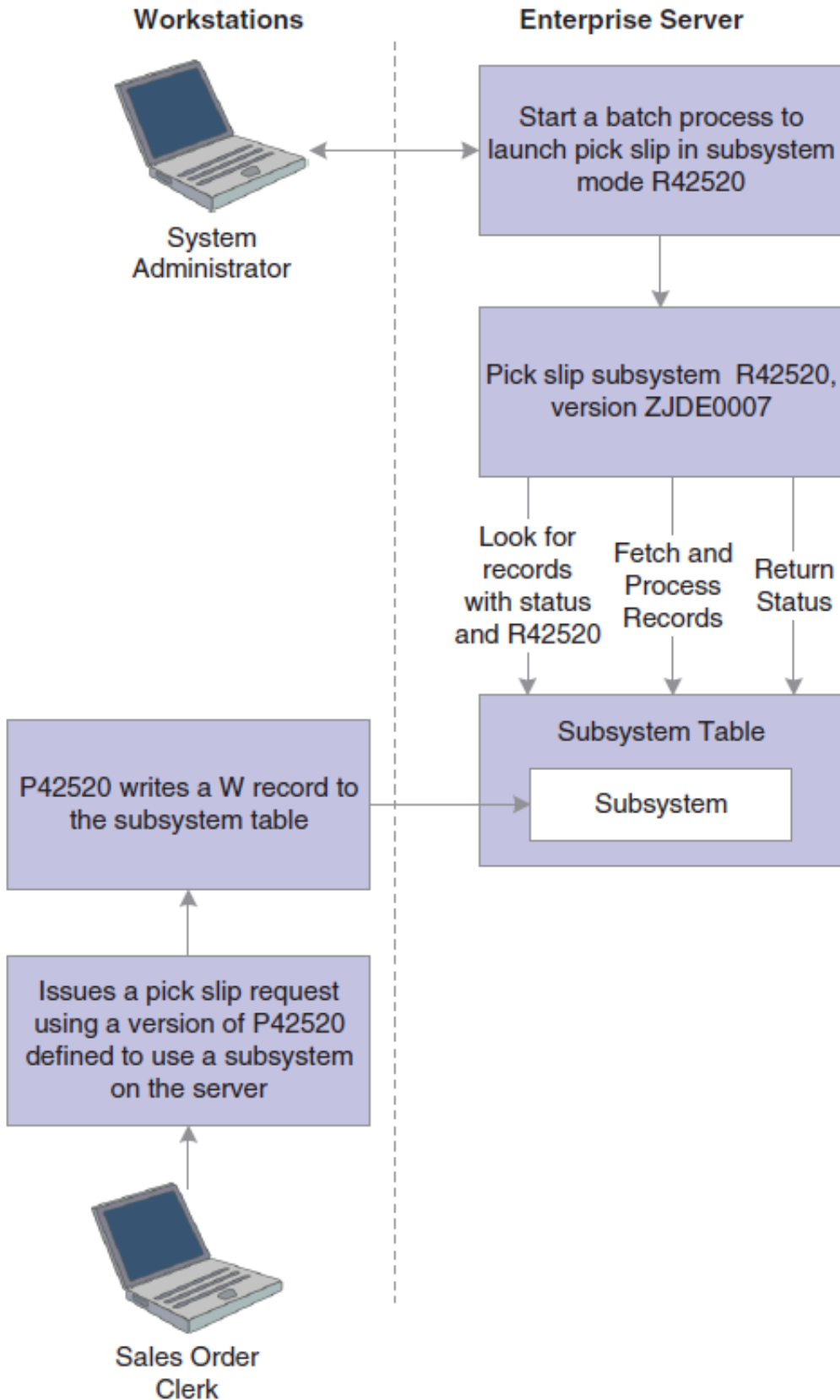
Some JD Edwards EnterpriseOne applications are designed to use subsystems to complete needed work. For example, you can instruct Sales Order Processing to print pick slips through a JD Edwards EnterpriseOne subsystem. You activate a subsystem through the processing options of a batch application. Then you create a specific version of the batch application, using that processing option to run the application in subsystem mode.

You must manually start subsystems to minimize the consumption of system resources. When started, JD Edwards EnterpriseOne subsystems run continuously, looking for and processing requests from JD Edwards EnterpriseOne applications. Subsystems run until you terminate them. Subsystems can have much higher throughput than regular batch jobs (for shorter running jobs) because they initialize their user environment and sessions only once. Also, they need to get report specifications from the metadata kernel only once for the duration of their execution.

Typically, you use subsystem jobs running on the enterprise server to off load processor resources from the workstation. Instead of queuing requests and running them in batches at specified times of the day, you can direct the requests to a subsystem, where they are processed in real-time. For example, you might be running the Sales Order Entry application on a workstation and want to print pick slips. If you are using a version of pick slips that has the Subsystem Job function enabled, the request is executed by a subsystem job. The pick slip request is routed to and processed by the subsystem job on the defined enterprise server. As a result, no additional processing resources are required from the workstation machine to actually print the pick slip.

When an application issues a request for a job to run in a subsystem, it places a record in the Subsystem Job Master table (F986113). These records are identified by subsystem job name and contain status and operational indicators. Embedded in the record is key information that allows the subsystem to process the record without additional interaction with the requesting application. The continuously running subsystem monitors the records in this table. If the subsystem finds a record with its process ID and appropriate status indicators, it processes the record and updates the status accordingly.

This illustration displays the logical sequence of events associated with subsystems:



Enabling Subsystems

To prevent excessive processing overhead during server startup and to prevent unnecessary uses of processor resources for subsystem jobs that might be in use, you must manually start subsystems. Generally, the system administrator or manager-level user is responsible for this task. To manually start subsystems, a version of a JD Edwards EnterpriseOne batch process with a processing option set to enable the use of subsystems is run.

As described, the way that you initially control the creation and start-up of these subsystems and queues depends on the server platform.

Platform (Subsystem or Queue)	Description
IBM i (JDENET)	<p>ERP 8: One <i>IBM i</i> subsystem is used for JD Edwards EnterpriseOne. This subsystem is started automatically when you issue the JD Edwards EnterpriseOne startup command STRNET. The subsystem name is version-specific. For example, for release 9.2, the subsystem name is JDEE920.</p> <p>To process requests that are destined for JD Edwards EnterpriseOne subsystems, you must define a specific job queue running under the JDENET subsystem. For example, a job queue might be named QBATCH.</p> <p>User requests for JD Edwards EnterpriseOne subsystem-defined batch jobs are executed by the job queue that is based on definition in the <i>IBM i</i> user profile.</p> <p>For EnterpriseOne 8.9 and later releases, see the note following this table.</p>
UNIX (jdequeue)	<p>To process requests that are destined for subsystems, you must define one or more queues. For example, a jdequeue might be named QBATCH.</p> <p>User requests for subsystem-defined batch jobs are executed by the job queue, based on the process ID.</p>
NT (jde.ini settings)	<p>ERP 8: One or more queues can exist for JD Edwards EnterpriseOne. These queues must have the same name. You define queues using settings in the <i>jde.ini</i> file.</p> <p>To process requests that are destined for subsystems, you must define the name and number of queues in the [NETWORK QUEUE SETTINGS] section of the <i>jde.ini</i> file. For example, a jdequeue might be named QBATCH.</p> <p>User requests for subsystem-defined batch jobs are executed by the job queue, based on the process ID.</p> <p>For EnterpriseOne 8.9 and later releases, see the note following this table.</p>

Note: To configure batch queues for subsystem jobs, use application P986130 (Work With Job Queues). P986130 allows you to create, modify, copy, delete, or change the status of job queues, regardless of platform. Use P986130 to change the maximum number of jobs that can run in a queue in addition to defining a default queue in which to submit jobs. Details about P986130 can be found in the System Administration Guides for Tools. The System Administration Guides are available on Oracle Technology Network. For 9.2+, the guide name has been renamed to the *JD Edwards EnterpriseOne Tools Runtime Administration Guide*.

System administrators can display all of the subsystems that are running on a server by using the Subsystem Jobs application (P986113). Use this application to:

- Locate a list of subsystems that are running on a server.

- Locate a list of subsystem records that are unprocessed (not available for *IBM i* servers).
- Locate the current record that a subsystem is processing (not available for *IBM i* servers).
- Stop or delete any subsystem.

Subsystem Job Records

Multiple JD Edwards EnterpriseOne processes write records to the Subsystem Job Master table (F986113). Each record has a status code that identifies subsystem request types and operational status. You can use Work With Server Jobs to view the records in this table.

Terminating Subsystems

You can use Work With Server Jobs to terminate subsystems. The following two methods of termination are available:

- Stopping a subsystem job causes it to terminate after it completes processing the current record. Additional unprocessed records in the Subsystem Job Master table (F986113) will not be processed, and no new records can be written. Essentially, the unprocessed records will be lost; that is, the process that initiated the record is not notified that the record was not processed.
- Ending a subsystem job causes it to terminate after processing all of the existing subsystem records. No new records can be written to the Subsystem Job Master table (F986113).

Securing Subsystems

Because subsystem jobs run continuously, sometimes actions from the subsystem jobs are difficult to trace back to the job submitter. The actions that insert triggers for subsystem jobs as well as UBEs should be properly secured. You can use EnterpriseOne application security and action security to grant access to only those users necessary to run certain applications and certain actions and prevent access to other users.

This is an action that needs to be performed by EnterpriseOne customers. This will prevent potentially harmful actions from being taken by an attacker.

Forms Used to Manage JD Edwards EnterpriseOne Subsystems

Form Name	FormID	Navigation	Usage
Work With Servers	W986116A	System Administration Tools (GH9011), Data Source Management, Work With Servers (P986116).	Select a server in which you want to locate a subsystem.
Work With Subsystems	W986113A	On the Work With Servers form, from the Row menu, select Subsystem jobs.	Review the status and type of the subsystem. Stop or end a subsystem.
View Jobs	NA	On the Work With Subsystems form, from the Row menu, select View Jobs.	Review server jobs and job types.

Locating Subsystems Running on a Server

Access the Work With Servers form.

1. On the Work With Servers form, select a server from the list or use the query by example row to select a specific server.
2. From the Row menu, select Subsystem Jobs.
3. On the Work With Subsystems form, select one of these options:

- &Processes

A process is a subsystem that is waiting for work. It is identified by an S (subsystem job) value in the Job Type field.

- &Waiting Jobs

Waiting jobs are report jobs that are queued for a subsystem. They are identified by an R (subsystem record) value in the Job Type field.

All currently running subsystems are displayed. Report number and version identify the running subsystems.

4. Review these fields in the detail area to note the type and status of the subsystem:

Field	Description
Job Type	<p>Indicates the subsystem type. Values are:</p> <ul style="list-style-type: none"> ○ R Subsystem record. ○ S Subsystem job.
Job Status	<p>Indicates the status of the subsystem job or record. Values are:</p> <ul style="list-style-type: none"> ○ W Subsystem record waiting. ○ P Subsystem record processing. ○ E Subsystem record to end the job. ○ R Subsystem job running.

Reviewing Job Records for Subsystems

Access the Work With Subsystems form.

1. After locating a subsystem job, on the Work With Subsystems form, click Find.
2. Select a record in the detail area, and then select View Jobs from the Row menu.
3. On the View Jobs form, click Find.

A list is displayed for all server jobs in the Subsystem Job Master (F986113) with an R (subsystem job running) job type.

Terminating Subsystems

Access the Work With Subsystems form.

1. Select the running subsystem that you want to stop.
2. To stop a subsystem, from the Row menu, select Stop Subsystem.

Note: If you are viewing Waiting Jobs from Work With Server Jobs or if you are viewing subsystem jobs by selecting the View Jobs from Work With Server Jobs, the Stop Subsystem selection is disabled from the Row menu selection.

3. To end a subsystem, from the Row menu, select End Subsystem Job.

Note: If you are viewing Waiting Jobs from Work With Subsystems, the End Subsystem selection is disabled from the Row menu selection.

4. On End Subsystem Job, click OK.

8 Glossary

EnterpriseOne extension

A JDeveloper component (plug-in) specific to EnterpriseOne. A JDeveloper wizard is a specific example of an extension.

JDBNET

A database driver that enables heterogeneous servers to access each other's data. As of Release 9.2.3, JDBNET is no longer supported.

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.

program temporary fix (PTF)

A representation of changes to JD Edwards EnterpriseOne software that your organization receives on magnetic tapes or disks.

replication server

A server that is responsible for replicating central objects to client machines.

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.

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.

Index

A

- accounts
 - Windows [196](#)
- AIX
 - kernel parameter settings
 - maxuproc [71](#)
 - system parameters [72](#)
 - tune parameters [72](#)
 - viewing system parameters [72](#)
- architecture
 - enterprise servers
 - UNIX [47](#)
 - Windows [78](#)
- authorization lists
 - iSeries
 - database security [40](#)
 - samples [35](#)
- automatic start
 - enterprise servers
 - iSeries [9](#)
 - UNIX (HP 9000) [51](#)
 - UNIX (RS/6000) [52](#)
 - UNIX (Sun Solaris) [52](#)

B

- batch output files
 - iSeries [14](#)
 - UNIX [55](#)
 - Windows [94](#)
- batch processes
 - iSeries
 - administering [14](#)
 - encoding passwords [16](#)
 - monitoring processes [15](#)
 - running reports [15](#)
 - scheduling reports [15](#)
 - log files
 - enterprise server [119](#), [125](#), [128](#)
 - UNIX
 - administration [54](#)
 - monitoring processes [56](#)
 - running reports [57](#)
 - scheduling reports [58](#)
 - Windows [93](#), [93](#), [95](#), [96](#)
- batch queues, settings for starting batch queues [82](#)
- BSFNLIB security parameter [23](#)
- business functions
 - security files [98](#)
 - security files for UNIX [61](#)

C

- cleaning up the enterprise server
 - iSeries [11](#), [12](#)
 - iSeries cleanup [12](#)
- CPU binding for AIX [73](#)

D

- data sources
 - troubleshooting the Server Map [131](#)
- databases
 - iSeries
 - security [28](#)
 - setting up security [38](#)
 - UNIX
 - increasing the maxprocess limit [50](#)
 - starting the enterprise server [50](#)
- deployment servers
 - backup requirements [105](#)
- directory structures
 - UNIX [45](#)
- disk striping
 - AIX [73](#)
- domains
 - Windows [86](#)
- DTAPATH datapath security parameter [20](#)

E

- email
 - troubleshooting [205](#)
- enterprise server initialization for iSeries [5](#)
- enterprise servers
 - automatically starting the Linux [52](#)
 - backup requirements [106](#)
 - cleaning up the Windows enterprise server [91](#)
 - creating a separate instance [73](#)
 - debugging [130](#)
 - iSeries [8](#)
 - architecture [3](#)
 - backup requirements [106](#)
 - cleaning up the server [11](#)
 - initialization [5](#)
 - installation problems [180](#)
 - interprocess communication (IPC) [188](#)
 - JDBNET [187](#)
 - jde.ini file [189](#)
 - manual start [8](#)
 - PPAT [186](#)
 - setting up a printer [12](#)
 - shutting down the server [10](#)
- Linux [53](#)
- log files
 - batch processes [125](#)
 - batch processing [119](#)
 - jde.log [119](#)
 - jddebug.log [122](#)
 - logic processing [119](#)
 - viewing files [127](#)
- running multiple instances [73](#)
- Server Map data source
 - troubleshooting [131](#)
- types of problems [119](#)
- UNIX [50](#), [53](#)
 - architecture [47](#)
 - automatic start (HP 9000) [51](#)
 - automatic start (RS/6000) [52](#)
 - automatic start (Sun Solaris) [52](#)

- backup requirements *107*
 - directory structure *45*
 - initialization *49*
 - jde.ini file *191*
 - jde.ini file security *62*
 - manual start *50*
 - setting up a printer *54*
 - Windows *77, 86, 88, 89, 90, 92, 98*
 - backup requirements *107*
 - testing by submitting report *201*
- executable file security
 - UNIX *62*
 - Windows *98*
- F**
- file descriptors
 - kernel parameter settings
 - HP-UX *68*
 - Sun Solaris *68*
- file security
 - UNIX
 - business function files *61*
 - JD Edwards EnterpriseOne executables *62*
 - jde.ini file (enterprise server) *62*
 - specification files *61*
 - Windows *97, 98, 98*
- files
 - restoring a backup file
 - SQL Server *117*
 - viewing batch output files *16*
- H**
- HP 9000
 - enterprise server
 - automatic start *51*
- HP-UX
 - kernel parameter settings
 - file descriptors *68*
 - message queues *65*
 - processes *68*
 - semaphores *66*
 - shared memory *67*
- I**
- INILIB security parameter *20*
- initialization
 - troubleshooting *81*
 - UNIX *49*
- installation
 - JD Edwards EnterpriseOne *52*
 - verifying for Windows *92*
- IPC resources
 - Linux *69*
- IPCS
 - setup for database security
 - iSeries *39*
- iSeries
 - administering servers *3*
 - authorization lists
 - database security *40*
 - samples *35*
 - backing up tables *113*
 - batch output files *14*
 - batch processes *14, 15*
 - administration *14*
 - encoding passwords *16*
 - monitoring processes *15*
 - scheduling reports *15*
 - database security
 - administrators *39*
 - removing administrative authority *39*
 - user profile information *40*
 - enterprise server *8, 11*
 - architecture *3*
 - automatic start *9*
 - backup requirements *106*
 - email *186*
 - initialization *5*
 - installation problems *180*
 - interprocess communication (IPC) *188*
 - JDBNET *187*
 - jde.ini file *189*
 - manual start *8*
 - process flow *3*
 - setting up a printer *12*
 - shutting down the server *10*
 - integrated file system logging support *10*
 - JD Edwards EnterpriseOne database security *28*
 - jde.log *12*
 - jddebug.log *12*
 - library structure *6*
 - restoring a backup file *117*
 - setting up JD Edwards EnterpriseOne database security *38*
- iSeries database security parameters *19*
 - BSFNLIB *23*
 - DTAPATH datapath *20*
 - INILIB *20*
 - JD Edwards EnterpriseOne DB admin profile *22*
 - modifying JDE profile *21*
 - modifying security profile *22*
 - modifying system profile *21*
 - secure log path *23*
- J**
- JD Edwards EnterpriseOne
 - backing up tables *105*
- JD Edwards EnterpriseOne DB admin profile security parameter *22*
- JDBNET
 - troubleshooting
 - iSeries *187*
- jde.ini file *94*
 - troubleshooting
 - iSeries *189*
 - UNIX *191*
 - UNIX *55*
 - security *62*
 - Windows *82, 98*
- jde.log
 - enterprise server *119, 119*
 - setup *127*
 - iSeries *12*
- jddebug.log
 - clearing the file *12*
 - enterprise server *119, 122, 124*
 - disabling *123*
 - enabling *123*
 - naming conventions *124*

- server locations *124*
- setting up *127*

JDENET

- iSeries
 - troubleshooting *184*

- Job Queue Maintenance program (P986130) *224*

job queues

- adding *225*
- changing the status of *226*
- copying *226*
- overriding *226*

jobs

- changing a printer *218*
- changing the priority *218*
- holding *219*
- overriding printer location *214*
- printing jobs *218*
- releasing *219*
- reviewing online *214*
- status and priority *214*
- viewing job logs *214*

K

kernels

- AIX *73*
 - maxuproc *71*
 - parameter settings *70*
 - tune parameters *72*
 - viewing system parameters *72*

HP-UX

- file descriptors *68*
- message queues *65*
- processes *68*
- semaphores *66*
- shared memory *67*

Linux

- parameter settings *69*

Sun Solaris

- file descriptors *68*
- message queues *65*
- processes *68*
- semaphores *66*
- shared memory *67*

L

libraries

- iSeries *6*

Linux

- administering servers *45*
- batch processes
 - scheduling reports *58*
- enterprise server *53*
 - architecture *47*
 - starting automatically *52*
- file limits *70*
- file security *61*
- IPC resources *69*
- kernel parameter settings *69*

log files

- accessing *10*
- clearing jde.log *12*
- clearing jdedebug.log *12*
- enterprise server
 - batch processes *125*

- batch processing logs *119*

- jde.log *119*

- jdedebug.log *122*

- viewing logs *127*

- enterprise server batch process log *119*

- enterprise servers *119*

workstation

- logic processing logs *119*

- viewing server logs *127*

- workstation batch process log *128*

logging

- integrated file system logging support

- iSeries *10*

logic processing logs

- enterprise server *119*

M

manual start

- enterprise servers

- iSeries *8*

- UNIX *50*

- Windows enterprise servers *92*

memory

- kernel parameter settings

- HP-UX *67*

- Sun Solaris *67*

message queues

- kernel parameter settings

- HP-UX *65*

- Sun Solaris *65*

N

network

- services for Windows *89, 90, 90, 90, 91*

O

object owner IDs

- JD Edwards EnterpriseOne tables *108*

Oracle

- backing up tables on UNIX or Windows *115*

- restoring a backup file *116*

output files

- viewing batch output files *16*

OUTQ

- creating for iSeries printer *13*

- starting for iSeries printer *13*

P

P986116 program

- using *216*

P986130 program *224*

parameter settings

- AIX *73*

- maxuproc *71*

- tune parameters *72*

- viewing system parameters *72*

- HP-UX memory *67*

- Sun Solaris memory *67*

passwords

- encoding passwords *16*

PDF

- iSeries output location *14*

performance

- AIX *73*
 - kernel parameter settings *72*

PORTTEST

- iSeries *183*

printers

- iSeries *13*
 - creating the OUTQ *13*
 - printing multiple copies to a remote printer *13*
 - setup *12*

UNIX

- setup *54*

- Windows *86, 86, 86, 87, 88, 88*

- administrators *89*

- setup problems *197*

- printing multiple copies to a remote printer *13*

processes

iSeries

- monitoring batch processes *15*

kernel parameter settings

- HP-UX *68*

- Sun Solaris *68*

- Windows *93*

Q

queue services

- Windows *89, 90, 90, 90, 91*

R

remote printers

- printing multiple copies on the iSeries *13*

reports

- batch processes *95, 96*

- iSeries *15, 15*

- UNIX *57, 58*

- UNIX *59*

Windows

- file location problems *201*

RS/6000

enterprise server

- starting automatically *52*

S

- secure log path security parameter *23*

security

iSeries

- JD Edwards EnterpriseOne databases *28*

- setup for JD Edwards EnterpriseOne databases *38*

- maintaining file security for Windows *97*

- specification file *97*

UNIX

- business function files *61*

- file security *61*

- jde.ini file (enterprise server) *62*

- specification files *61*

- Windows *98*

semaphores

kernel parameter settings

- HP-UX *66*

- Sun Solaris *66*

Server Map data source

- troubleshooting *131*

servers

AIX

- setting value of maxuproc *71*

- backing up tables *112*

- backup requirements *106*

- deployment server *105*

- command line *15, 58*

- batch process reports for iSeries *15*

- batch process reports for UNIX *57*

iSeries *8*

- administration *3*

- cleaning up the enterprise server *11*

- logging support *10*

UNIX

- administration *45*

- jde.ini file security *62*

- setting up a printer *54*

- shutting down the enterprise server *53*

- starting the enterprise server *50*

- Windows *77, 86, 91*

services

- Windows *86, 89*

setting up a printer

- iSeries *13, 13*

- printing multiple copies to a remote printer *13*

Windows

- troubleshooting *197*

shared memory

kernel parameter settings

- HP-UX *67*

- Sun Solaris *67*

SMIT application

- setting the value of maxuproc *71*

Solaris

enterprise server

- automatic start *52*

kernel parameter settings

- file descriptors *68*

- message queues *65*

- processes *68*

- semaphores *66*

- shared memory *67*

specification file security

- UNIX *61*

SQL Server

- backing up tables *116*

- restoring a backup file *117*

Windows

- restoring a backup file *118*

starting the enterprise server

- iSeries *8*

- automatic *9*

- manual *8*

subsystems

- locating subsystems running on a server *233*

- logical sequence of events diagram *229*

- stopping *234*

- viewing job records of *234*

T

tables

- backing up JD Edwards EnterpriseOne tables *105*

- backing up servers *112*

- object owner IDs [108](#)
- testing
 - iSeries
 - PORTTEST [183](#)
 - submitting a report [185](#)
- troubleshooting
 - enterprise servers [119](#), [129](#)
 - batch process log [119](#), [125](#)
 - jde.log [119](#)
 - jddebug.log [122](#)
 - Server Map data source [131](#)
 - UNIX [190](#)
 - Windows [195](#)
 - iSeries enterprise server [180](#), [185](#)
 - architecture [3](#)
 - initialization [5](#)
 - interprocess communication (IPC) [188](#)
 - JDBNET [187](#)
 - jde.ini file [189](#)
 - PPAT [186](#)
 - running JDENET [184](#)
 - UNIX enterprise server
 - architecture [47](#)
 - directory structure [45](#)
 - initialization [49](#)
 - Windows enterprise server [77](#), [78](#), [81](#), [195](#)
 - email [205](#)
 - finding data [203](#)
 - log file location [197](#)
 - report file location [201](#)
 - running JD Edwards EnterpriseOne manually [200](#)
 - setting up a printer [197](#)
 - setting up JD Edwards EnterpriseOne accounts [196](#)
 - stopping JD Edwards EnterpriseOne as run manually [204](#)
 - testing by submitting a report [201](#)
 - using Visual C++ to stop processes [204](#)
 - workstations
 - viewing server logs [127](#)
- troubleshooting JD Edwards EnterpriseOne servers
 - web servers [205](#)
- tune parameters
 - AIX [73](#)
 - kernel parameter settings on AIX [72](#)
 - type security parameter [19](#)
- initialization [49](#)
- jde.ini file [191](#)
- manual start [50](#)
- setting up a printer [54](#)
- file security [61](#)
- JD Edwards EnterpriseOne executable files
 - security [62](#)
- jde.ini file
 - security [62](#)
- kernel parameter settings
 - Linux [69](#)
- restoring an Oracle backup file [116](#)
- running multiple JD Edwards EnterpriseOne enterprise servers [73](#)
- specification files
 - security [61](#)
- verifying the JD Edwards EnterpriseOne installation [52](#)
- user accounts, setting up on Windows [88](#)
- user profiles
 - iSeries database security [39](#)
 - displaying information [40](#)

W

- Windows
 - administering the server [77](#)
 - backing up Oracle tables [115](#)
 - batch [93](#)
 - batch output files [94](#)
 - batch processes [93](#), [95](#), [96](#)
 - business function files [98](#)
 - enterprise server [77](#), [78](#), [81](#), [86](#), [89](#), [90](#), [91](#), [92](#)
 - backup requirements [107](#)
 - file security [97](#)
 - jde.ini file [82](#), [98](#)
 - printer setup [86](#)
 - restoring an Oracle backup file [116](#)
 - specification files [97](#)
 - SQL Server
 - restoring a backup file [118](#)
 - verifying the JD Edwards EnterpriseOne installation [92](#)
- Work With Job Queues form [224](#)
- Work With Servers program (P986116)
 - using [216](#)

U

- UNIX
 - administering servers [45](#)
 - AIX
 - kernel parameter settings [70](#)
 - backing up Oracle tables [115](#)
 - batch output files
 - reviewing [55](#)
 - batch processes
 - administration [54](#)
 - monitoring processes [56](#)
 - running reports [57](#)
 - scheduling reports [58](#)
 - business function files
 - security [61](#)
 - creating a separate instance of the enterprise server [73](#)
 - enterprise server [50](#), [53](#)
 - architecture [47](#)
 - backup requirements [107](#)
 - directory structure [45](#)

