Oracle® Communications MetaSolv Solution

System Administrator's Guide

Release 6.3

E69839-02

February 2017

ORACLE®

Oracle Communications MetaSolv Solution System Administrator's Guide, Release 6.3

E69839-02

Contents

4 Managing the MetaSolv Solution Database

5 Customizing the Application

6 Running Utilities, Local Exchange Routing Gateway, and Background Processor

Preface

This document contains concepts and procedures for installing Oracle Communications MetaSolv Solution (MSS). Information about installing the database, Oracle WebLogic server, clients, Background Processor, utilities, and supporting applications are included.

For additional information about required third-party software, such as WebLogic Server or the database, consult third-party documentation. Information about third-party software products appears in this document only when there are unique facts about the MSS relationship to those products.

Audience

This guide is for individuals responsible for installing or maintaining MSS and ensuring the software is operating as required. This guide assumes the installer has an Oracle DBA and WebLogic administrator background, with a working knowledge of Unix/Linux, Windows and Java JEE.

Related Documents

For more information, see the following documents in the MSS documentation set:

- *MSS Planning Guide:* Describes information you need to consider in planning your MSS environment before installation.
- *MSS Installation Guide*: Describes system requirements and installation procedures for installing MSS.
- *MSS Security Guide:* Provides guidelines and recommendations for setting up MSS in a secure configuration.
- *MSS Database Change Reference:* Provides information on the database changes in MSS releases.
- *MSS Network Grooming User's Guide:* Provides information about the MSS Network Grooming tool.
- *MSS Address Correction Utility User's Guide:* Provides information about the MSS Address Correction utility.
- *MSS Technology Module Guide:* Describes each of the MSS technology modules.
- *MSS Data Selection Tool How-to Guide:* Provides an overview of the Data Selection Tool, and procedures on how it used to migrate the product catalog, equipment specifications, and provisioning plans from one release of your environment to another.

- *MSS CORBA API Developer's Reference:* Describes how MSS APIs work, high-level information about each API, and instructions for using the APIs to perform specific tasks.
- *MSS Custom Extensions Developer's Reference:* Describes how to extend the MSS business logic with custom business logic through the use of custom extensions.
- *MSS Web Services Developer's Guide:* Describes the MSS Web Services and provides information about the MSS Web Service framework that supports web services, the various web services that are available, and how to migrate existing XML API interfaces to web service operations.

For step-by-step instructions for tasks you perform in MetaSolv Solution, log in to the application to see the online Help.

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.

1 Introduction

Oracle Communications MetaSolv Solution (MSS) supports service fulfillment business functions. Each subsystem accesses the MetaSolv Solution database and can be extended to support unique products and services.

The MetaSolv Solution application programming interfaces (API) give third-party applications limited access to parts of the MetaSolv Solution database. Additional accessory software may also be purchased.

Managing MetaSolv Solution Administration

In general, the responsibilities of the MetaSolv Solution administrator role fall into three basic categories, as shown in Table 1–1. These are the tasks described in this document.

Table 1–1 Administrative Responsibilities

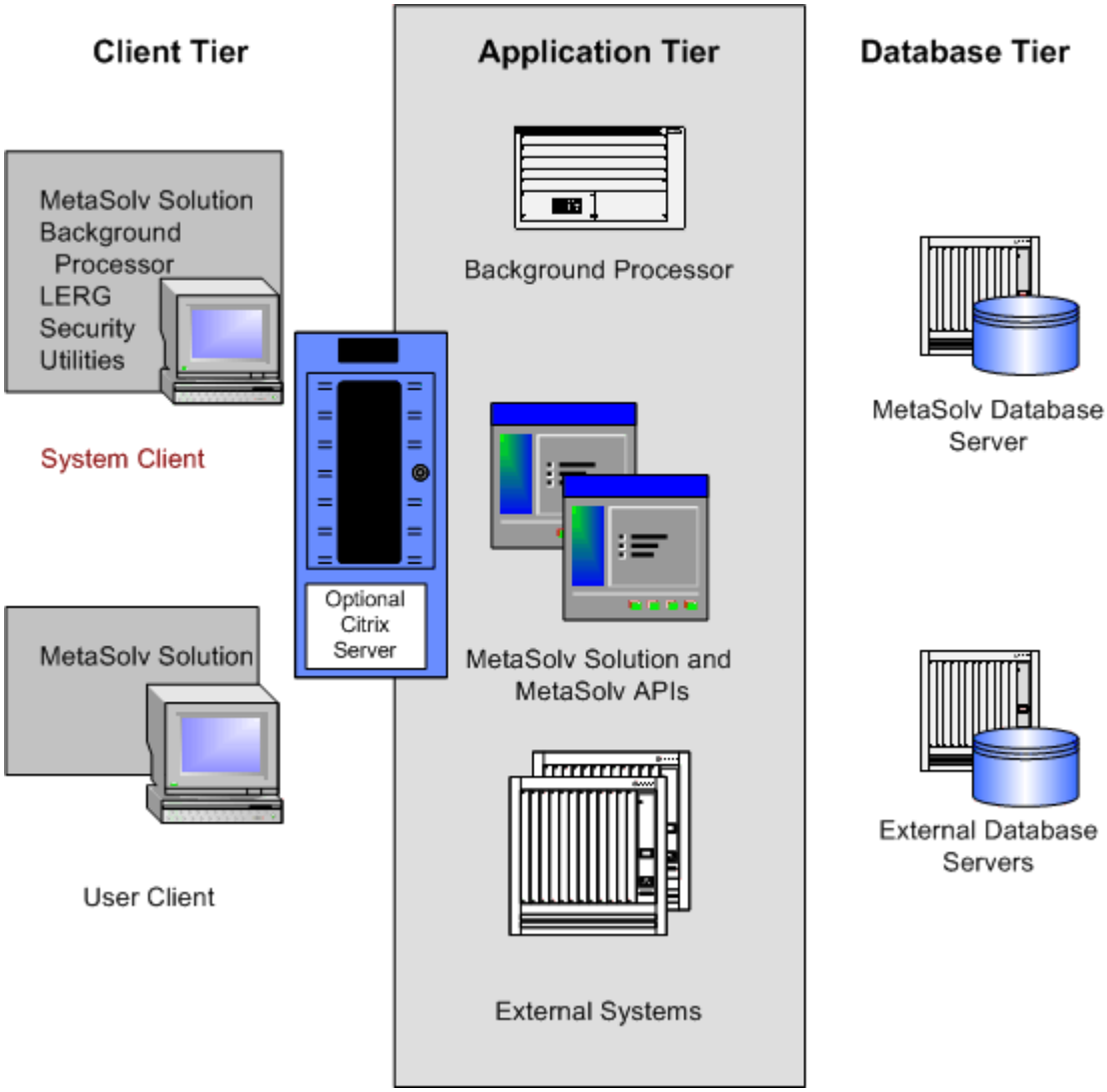
Responsibility	Tasks
Monitoring and maintaining the system	■ Maintaining components ■ Maintaining the configuration files ■ Monitoring log files
Supporting users	■ Providing logon information ■ Assisting end users with application error conditions ■ Resolving API problems ■ Troubleshooting system errors
Managing supporting modules, products, and utilities	■ Managing the Oracle WebLogic Server ■ Managing the Oracle database ■ Managing application and component security ■ Managing the Background Processor ■ Running the LERG ■ Running MetaSolv Solution Utilities ■ Running the NPA Split Utility

Introducing the Architecture

The MetaSolv Solution system consists of three tiers: clients, Oracle WebLogic server and APIs, and database servers. It can also include a Citrix/XenApp terminal that helps manage communication.

Figure 1–1 shows the MetaSolv Solution three-tier architecture:

Figure 1–1 MetaSolv Solution Three-Tier Architecture



MetaSolv Solution is installed on clients and an Oracle WebLogic server. The clients access the Oracle WebLogic server and the Oracle WebLogic server accesses a database. Some accessory products, like the MetaSolv Solution Utilities, access the database directly. See the product documentation for detailed instructions about managing:

- The Oracle WebLogic Server, Oracle WebLogic Server 12.2.1.2
- The Oracle database with an Oracle 12cR1 client

MetaSolv Solution clients are the user machines that run the MetaSolv Solution software. The client machines access the Oracle WebLogic server running in the middle tier. Clients connect to these servers using HTTP or IIOP (both over TCP/IP).

The Oracle WebLogic server connects to the database server using JDBC connection pools. The APIs run on the Oracle WebLogic server and makes direct JDBC connections to the database. JacORB 3.8 is compliant with OMG CORBA 2.3 or higher.

Details about using the Oracle WebLogic server can be found in the server product documentation. This document provides Oracle WebLogic server details only for information unique to the MetaSolv Solution.

The Oracle WebLogic server and the APIs access the MetaSolv Solution database, an Oracle Enterprise Edition 12cR1 database that must reside on a Windows or Unix-based machine.

Installed, the core product consists of many files, most of which require no attention. See "Understanding Configuration Files" for information about the INI and XML configuration files and the EXE files.

Additional files may be installed when MetaSolv Solution is installed, such as the LERG product files. See "Introducing the Accessory Applications" for more information.

See *MetaSolv Solution Planning Guide* for details about various components and their technical requirements

Introducing the Accessory Applications

In addition to the base MetaSolv Solution product, your company may have purchased an accessory package, called a software option. Also, additional products, such as utilities, are delivered with MetaSolv Solution at no extra cost. Some of these applications run separately and are not normally used by end users.

Table 1–2 lists accessory applications that may be loaded on client machines.

Table 1–2 Accessory Options and Products

Product	Description
APIs	MetaSolv Solution offers a robust library of APIs that permit interfacing with third-party systems.
Data Selection Tool	Oracle's Data Selection tool facilitates the selective migration of data between MetaSolv Solution databases of the same version.
MetaSolv Solution Utilities	Purge Utility: Removes old data from the database. Database Health Function: Runs checks to ensure that all database objects (required by MetaSolv Solution) are valid. Structured Formats: Provides a way to enter, define, and store custom formats and valid values for addresses, telephone numbers, switch prefixes, and Local Service Ordering Guide (LSOG) columns. Custom Attributes Tool: Enables you to set up custom attributes to collect and compute data at specific processing points. Migration Tool: Enables you to create and convert network elements and to migrate pre-5.1 connections, Virtual Layout Records, and reconcile design lines to Release 6.3.*x*.

Table 1–2 (Cont.) Accessory Options and Products

Product	Description
Location and Routing Gateway	The Location and Routing Gateway provides the interface between the MetaSolv Solution and the information from Telcordia Technologies Local Exchange Routing Guide (LERG) for North America, automating the loading of the LERG information into the MetaSolv Solution. The automated process provided by this gateway reduces data entry time and ensures that MetaSolv Solution information is accurate and in compliance with Telcordia standards.
NPA/NXX Split Utility Tool	The Number Plan Administration (NPA) NXX Split Utility enables you to update your database whenever an NPA (Area Code) split occurs in your service area. MetaSolv Solution uses NPAs as a part of telephone numbers in several functional areas.

Additional Oracle products may also be in use at your site. Log on to the Oracle software delivery website to access those products.

Securing the MSS Application

MSS application security consists of two processes: authentication and authorization. Authentication is the process of identifying a user with a user ID and password combination. See *MetaSolv Solution Installation Guide* for authentication details. See *MetaSolv Solution Security Guide* for information about authorization, the process of granting or denying access to application functions.

2

Maintaining Configuration

There are many files and preferences that affect the Oracle Communications MetaSolv Solution (MSS) environment. This chapter describes them in the following sections, providing information about:

- Configuration files
- File relationships
- Configuration parameters that may affect product performance
- API server process controls
- Changing configuration files
- Setting system preferences
- Copying encrypted passwords to the **gateway.ini** file
- Gateway events through integration server

Configuration of Oracle WebLogic Server also affects your MetaSolv Solution environment. See "Delivering File Changes" for additional details and refer to the WebLogic Server documentation for instructions on modifying Oracle WebLogic Server settings.

Understanding Configuration Files

MetaSolv Solution is installed with INI and XML configuration files that set the environment for the product, its software options, and the accessory products. Configuration files that you manage can be installed in various locations on the Oracle WebLogic server or the client. You also manage database configuration files that are installed on the database server.

MetaSolv Solution installer setup places the MetaSolv Solution files on the Oracle WebLogic server, on the client, or on the database server. When configuration files are required on the client, a master copy is stored on the Oracle WebLogic server. When the software is updated with a maintenance release or patch, the master copy is automatically updated. When this happens, master copies must be copied to the client machines manually. This process is detailed in *MetaSolv Solution Installation Guide.*

In addition, the MetaSolv Solution configuration files work only when the **Oracle Init.ora** is correctly configured. Table 2–1 describes the configuration files.

Table 2–1 Configuration File Functions

Configuration File Name	Function	Location
gateway.ini	Manages many MetaSolv Solution functions and API access to the database. The **gateway.ini** file is a server-side configuration file and is used by several server processes, including the APIs.	Oracle WebLogic server
init.ora	This Oracle initialization file is described in the Oracle documentation.	Oracle Database server
jmaster.ini	Access to the **jmanager.exe** executable provides administrator access to the Background Processor job queue. Access to the job queue permits cancelling jobs, pausing, and clearing the queue.	client
lerg.ini	The **lerg.exe** file runs the Location and Routing Gateway, which loads data from Telcordia Technologies.	client
loggingconfig.xml	This configuration file produces the**Appserverlog.xml**, **Appserverlog_misc.xml**, and **Appserver_auditlog.xml** log files.	Oracle WebLogic server
npasplit.ini	The executable, **npasplit.exe**, runs the NPA Split Utility. Every user running the split utility must have a copy of **npasplit.ini** in the product folder to designate database connections. The file also contains some user preference information that is specific to the NPA split utility. This file is installed with the split utility.	client
pmat.ini	This configuration file designates the database to be used with the Pre-Migration Analysis Tool (PMAT).	client
tbs.ini	This configuration file is used by the MetaSolv Solution executable, **tbs.exe**. This file is installed on the client when MetaSolv Solution is installed.	client
tbs_util.ini	This configuration file is used by the MetaSolv Solution Utilities executable, **tbs_util.exe**. This file is installed on the client.	client
tnsnames.ora	The MetaSolv Solution installer does not copy or update the **tnsnames.ora** file. You must manually update the **tnsnames.ora** file after you install the Oracle client. **tnsnames.ora** is an Oracle configuration file that contains network addresses mapped to databases. Each client has at least one of these files in an Oracle folder. The client cannot access a database unless it is listed in this file. See the Oracle documentation for details. **tnsnames.ora** resides on the client machine only when one of the accessory products is also installed. It is not required for MetaSolv Solution.	client

MetaSolv Solution Application Components

MetaSolv Solution application components are divided into two groups:

- Components that communicate only with Oracle WebLogic Server (*n*-tier)
- Components that communicate directly with the database (2-tier)

The **gateway.ini** file is a server-side configuration file that is used by several server processes, including the MetaSolv Solution APIs. See "Parameters for gateway.ini File" for details about this file.

Communicating with the Oracle WebLogic Server

Application components that communicate only with the Oracle WebLogic server depend on the Oracle WebLogic Server to communicate directly with the database. Table 2–2 describes system applications that communicate with the Oracle WebLogic server.

Table 2–2 Applications That Communicate With the Oracle WebLogic Server

Application Server Connection INI Files	Executable Name	Product Name
tbs.ini	tbs.exe	MetaSolv Solution
jmaster.ini	jmaster.exe jmanager.exe	Background Processor

These components are managed by configuration files that have identical, or almost identical, content. Each INI file provides the following types of information for the associated EXE:

- An Oracle WebLogic Server profile name

  For example:

  ```
  [Application Servers]
  J2EEProfiles= 'MSS63'
  ```

- An Oracle WebLogic Server port ID

  For example:

  ```
  [J2EEProfile MSS63]
  url=https://server.us.example.com:8630
  ```

Using a tbs_db.ini File

Table 2–3 describes system applications that communicate with the database.

Table 2–3 Applications That Communicate With the Database

Database Connect INI Files	Executable Name	Product Name
lerg.ini	lerg.exe	Location and Routing Gateway
npasplit.ini	npasplit.exe	NPA Split Utility
tbs_util.ini	tbs_util.exe	MetaSolv Solution Utilities

You can create a **tbs_db.ini** file to enable the **tbs_util.exe** and **npasplit.exe** files to communicate with the database through one file, using a shared INI parameter. Doing so allows you to make changes to only the **tbs_db.ini** file, which is shared by the other three files. Following is an example of a **tbs_db.ini** file:

```
[DBMS_Profiles]
Profiles= 'MSSPROD'

[Profile MSSPROD]
DBMS=O84
Database=Oracle
UserId=
DatabasePassword=
LogPassword=
ServerName=M63PROD
LogId=
Lock=
DbParm=DisableBind=0,DelimitIdentifier='No',CommitOnDisconnect='No'
PRomPt=1
AutoCommit=0
url=http://192.0.2.62:9638
```

lerg.ini, **tbs_util.ini**, and **npasplit.ini** include any changes that are in the **tbs_db.ini** file when a **Sharedini=** parameter is added to them, because the parameter indicates that the **tbs_db.ini** file should be referenced. Following is a model **npasplit.ini** file:

```
[DBMS_Profiles]
SharedIni=\\path\TBS_db.ini
```

Where *path* represents the path to the **tbs_db.ini** file.

Following is a sample **tbs_util.ini** file that uses the shared reference:

```
[DBMS_Profiles]
SharedIni=\\srvpldevsrvc1\testing\6.3.x_ini\tbs_db.ini
```

Changing INI Files

When changes are made to the location of the database or the Oracle WebLogic server, changes must be made in the MetaSolv Solution INI files. The INI files provide information about the names you want to appear in the log on database list.

Template copies of these files are copied to the **/appserver/config** directory on the Oracle WebLogic server by the installer. The installer edits these files to include specifications for each specific Oracle WebLogic server.

Pre-Migration Analysis Tool Configuration

Oracle provides a Pre-Migration Analysis Tool (PMAT) that should be used against a pre-M/6 MetaSolv Solution database. This tool provides a detailed analysis of what data may be affected by the Next Generation Migration. The PMAT is a separate executable found on the Oracle software delivery Web site and uses its own .ini file, **Pmat.ini**. To log on and use the PMAT, you must update the **Pmat.ini** file.

Contact Oracle Communications Consulting for guidance on how to obtain the **Pmat.ini** file.

The following is a sample of the file you must update:

```
[DBMS_Profiles]
Profiles='DatabaseProfileName'
```

```
[Profile DatabaseProfileName]
DBMS=O84
Database=Oracle
UserId=
DatabasePassword=
LogPassword=
ServerName=DatabaseName
LogId=
Lock=
PRomPt=1
DbParm=DisableBind=0,DelimitIdentifier='No',CommitOnDisconnect='No'
AutoCommit=0
```

To configure the **Pmat.ini** file, set the *DatabaseProfileName* to name the database profile parameter and set the *DatabaseName* to the name of the database where the migration analysis will be performed.

Setting Oracle WebLogic Server Performance Parameters

Parameters can be set that affect performance of the Oracle WebLogic Server and the database.

For proper performance of the Oracle WebLogic Server, specific MetaSolv Solution configuration settings are recommended. Those settings are discussed in the following sections.

For more information on Oracle WebLogic Server performance and MetaSolv Solution, see *MetaSolv Solution Installation Guide*. For information about Oracle WebLogic Server performance, see the Oracle WebLogic Server documentation.

Database performance also has a strong impact on product performance. To manage database performance, see "Managing Database Performance".

Setting gateway.ini Oracle WebLogic Server Performance Settings

Several **gateway.ini** parameters can have a significant negative impact on Oracle WebLogic server performance. They are:

- CacheRefreshMode [CA]
- DBPollingInterval [Event2]
- DBPollingInterval [Events]
- PrintExportObjects [Gateway]

See "Parameters for gateway.ini File" for details.

Copying Encrypted Passwords to gateway.ini

When you install the application server, the *session_password* and *jndi_password* are encrypted in the [Session] and [JNDI] sections respectively in the **gateway.ini** file. The encrypted passwords are prefixed with {AES}.

For example:

```
[Gateway]
APPJMSUser=app_jms
APPJMSPwd={AES}61b01de259ce47676d3f7b85ca5086f9
[Session]
User=app_api
```

```
Password={AES}1824a67ea02612bcbe3169d0b0d185b4
[JNDI]
User=administrator
Password={AES}1824a67ea02612bcbe3169d0b0d185b4
Factory=weblogic.jndi.WLInitialContextFactory
URL=t3://srvsunpluto:7001
```

After the application server is installed, you may opt to change the passwords for different users of the application server. If you change the user passwords for the application server, you must encrypt your new passwords, and then copy the encrypted passwords in the [Session], [Gateway], and [JNDI] sections of **/appserver/gateway/gateway.ini** file. See "Sample installed gateway.ini File" for more information.

To encrypt the new password using MetaSolv Solution Utilities:

1. Start the MetaSolv Solution Utilities application. See "Using the MetaSolv Solution Utilities" for more information.
2. From the **File** menu, select **Security**, and then select **Encrypt Passwords**.

   The Encrypt Passwords dialog box appears.
3. In the **Password in Text** field, enter your new password.
4. Click **Encrypt**.

   The encrypted password appears in the **Encrypted Word** field.
5. Copy the encrypted password and paste it in the appropriate section of **gateway.ini** file.
6. Restart the application server.

Setting loggingconfig.xml Oracle WebLogic Server Performance Settings

Set all logging levels to error in the **loggingconfig.xml** for normal use. Change the severity only during troubleshooting. For information about the **loggingconfig.xml** file parameters, see "Annotated loggingconfig.xml".

Setting Up API Servers

All of the API servers named in the [Servers] and [ThreadProcs] sections of the **gateway.ini** file are started automatically unless disabled. To disable a server, place a semicolon (;) at the beginning of the line that names the server or during installation, the servers can be disabled. By default, only the Integration Server is enabled.

Some MetaSolv Solution API servers poll the MetaSolv Solution database at designated intervals to determine whether there is any work that the server should perform. These servers have their own unique set of configuration parameters. The servers are:

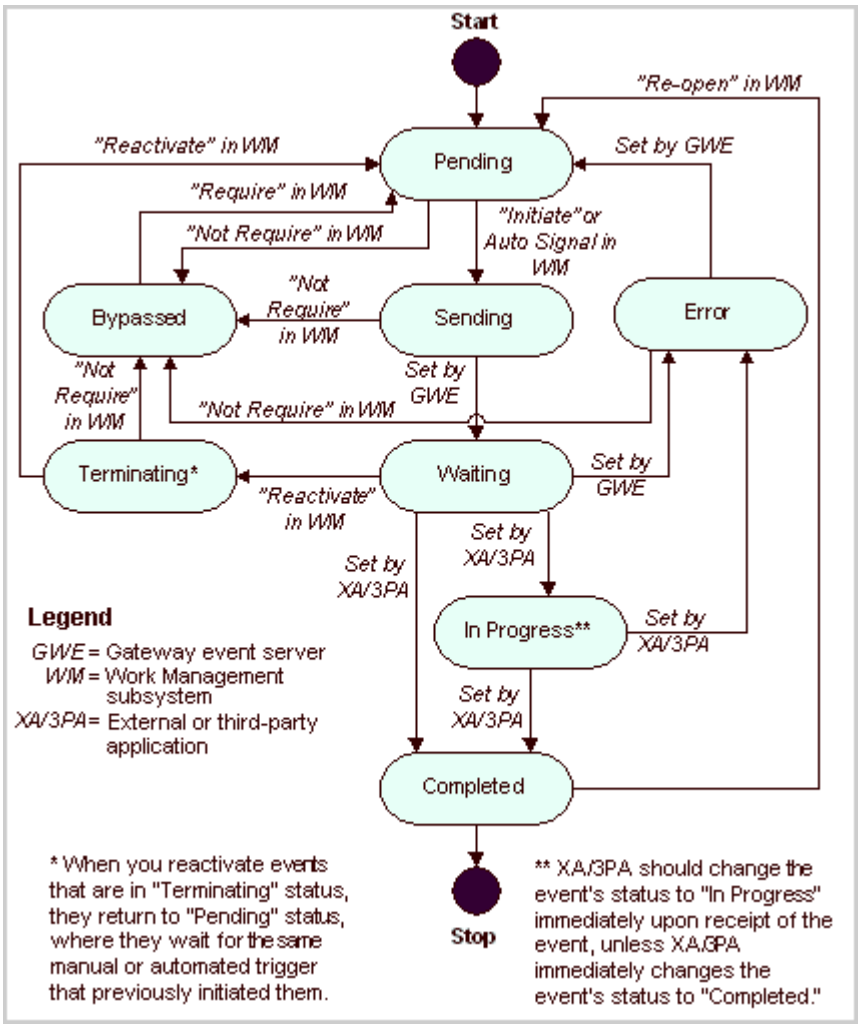
- The Gateway Event server
- Event2 server for PSR End User Billing
- The System Task server
- Integration Server

The Gateway Event server is the engine that handles all aspects of:

- Dispatching internally originating events to the ORB for transmission to third-party applications
- Recording some gateway event reported state changes in the MetaSolv Solution database

Figure 2–1 provides a view of the lifecycle of event statuses from the point of view of the MetaSolv Solution, and illustrates the function of the Gateway Event server.

Figure 2–1 State Transition Diagram for Gateway Events



See "Parameters for gateway.ini File" for descriptions of these parameters. Table 2–4 lists the server processes.

Table 2–4 gateway.ini Server Processes

Server Process	Description
Gateway Event server	The Gateway Event server handles dispatching internal events to the ORB and recording some gateway event state changes in the MetaSolv Solution database
Event2 server for PSR End User Billing	This server performs the same function as the Gateway Event function except for billing gateway events.

Table 2–4 (Cont.) gateway.ini Server Processes

Server Process	Description
System Task server	The System Task server allows system tasks to be automatically completed by MetaSolv Solution.
Integration server	The Integration server enables the interaction of the Oracle WebLogic server, other Oracle products, and third party products.
Signal server proc	Reserved for internal processing.

For more information about the API server processes, see "Servers Parameters".

Updating Configuration Files

Under some conditions, changes must be made to parameters in configuration files. You will make changes when your environment changes, when instructed to make changes by Oracle Global Customer Support, or when using the logging files to track problems.

For details about configuration and deployment at your site, see your site's deployment diagram.

To edit the INI or XML configuration files:

1. Backup the configuration file.
2. Open and edit the file by using any common editor, such as Notepad.
3. Shut down and restart the Oracle WebLogic server unless changing the **loggingconfig.xml** file.

For Oracle WebLogic Server-related changes, check the upper left corner in the WebLogic Server Administration Console where a message is displayed that informs you whether you need to restart the server or not.

Changing Configuration Files When the Environment Changes

Configuration files include specific information about the environment, such as machine names, database IDs, and so on. Use Table 2–5 as a guide to changing your configuration files if environment changes are made.

Table 2–5 Environment Changes Affecting Configuration Files

Situation	Change These Files	Location
Changing a database name	Only 2-tier application INI files	Oracle WebLogic server and client
Adding an API	**gateway.ini**	Oracle WebLogic server
Adding an Oracle WebLogic server	All INI files	client and server
Creating a cluster (Oracle WebLogic task)	See the Oracle Administration Console	Oracle WebLogic server

Table 2–5 (Cont.) Environment Changes Affecting Configuration Files

Situation	Change These Files	Location
Creating a new domain (Oracle WebLogic server task)	See the Oracle Administration Console	Oracle WebLogic server
Moving an Oracle WebLogic server to another machine	All INI files	client and server
Changing ports (HTTP and log ports)	All INI files	client and server
Changing the administrative password for a domain	See the Oracle WebLogic Server documentation.	Oracle WebLogic server
Changing the Oracle database administrator password	Change the **User** parameter in the **gateway.ini** file	Oracle WebLogic server
Activating or deactivating an API	**gateway.ini**	Oracle WebLogic server

Changing System Preferences

System administrators manage the system and global preferences, which affect every user connected to the same MetaSolv Solution database. Users cannot alter system preferences unless given specific security permission. New preferences may be changed or added with service packs or patches.

System and global preferences affect users on one database. The System preferences affect all users of the application. Global preferences affect all users of all applications in the MetaSolv Solution product family. They are represented in the Preferences window by a globe icon. the icon for a system preference is two small heads.

To change a system preference:

1. Log in to MetaSolv Solution.
2. Click **Application Set Up**.
3. Select **Preferences**.
4. Follow instructions in the online Help to make changes.

The ability to change system preferences is a privilege that must be granted through the MetaSolv Solution Security module.

Delivering File Changes

Different configuration files are located on different components and therefore require different delivery methods when settings change.

The **gateway.ini** file is on the Oracle WebLogic server. Changes to it have different follow-up requirements than changes to settings on the UI or in configuration files that reside on a client machine.

Changes to product files, such as those made using service packs or patches, have other requirements.

Caution: You can dynamically change the **loggingconfig.xml** file in your production system. Ensure that you restrict access to this file to prevent unauthorized users from updating it.

Delivering Configuration Changes You Make

Some MetaSolv Solution configuration files reside on the client. They are:

- jmaster.ini
- lerg.ini
- npasplit.ini
- tbs.ini
- tbs_util.ini

Other configuration files reside on the Oracle WebLogic server.

To change any configuration file:

1. Locate the file on the Oracle WebLogic server.
2. Open the file using Notepad or any text editor.
3. Enter your changes and save the file.
4. Deliver the file to the client machines manually.
5. (Applicable only to the **gateway.ini** file) Restart the Oracle WebLogic server.

Note: When you suspend and resume a MetaSolv Solution polling server process, that process does not re-read the **gateway.ini** settings that affect its operation. To put changes into effect, you must shut down and restart the Oracle WebLogic server.

Gateway Events Through Integration Server

Gateway Events through integration server continuously check the MSS database for events that are ready to be sent to an external application. The integration server also monitors the external application for updates to the status of a gateway event. When an external application sends a status update, the integration server records the new status in the MSS database.

In case of a cluster environment, Oracle recommends that you copy the **integration.xml** file (with added integration events) to any CORBA gateway event server (EVENTPROC) appserver to ensure that the CORBA gateway event server does not pick up the integration gateway events.

The following sections describe how to create a gateway event for communication between MSS and an external application.

Creating a Gateway Event

To create a gateway event:

1. Create a new gateway event.

   For example: *xxx_order_event*.

This is done in the Gateway window in MSS. MSS creates a gateway event and assigns an event ID.

2. Query for the event ID associated with the gateway event, using the following SQL:

```
select
gateway_event_id, gateway_event_nm
from
asap.gateway_event ge
where
gateway_event_nm like = <APIGatewayName>
```

In this case, *APIGatewayName* is the name associated with the event.

3. Configure the event by make the following entries in the **integration.xml** file located at *MSLV_Home*/**appserve**r:

```
<handler enabled="true">
<event-id><YourGatewayEventID></event-id>
<class>com.mslv.integration.handlers.DefaultEventHandler</class>
<destination>api</destination>
</handler>
```

where *YourGatewayEventID* is the gateway eventID generated in step 2.

4. Restart the MSS application server.

Outbound Events

All outbound events are published to a JMS queue, **mss.external.event.queue**, using the JMS Factory **MSS.QueueConnectionFactory**. The following list indicates how outbound events can be subscribed to:

- Clients (both, third-party systems not on WebLogic and Oracle clients) can subscribe to the JMS queue externally and receive all events.
- Products using SOA can subscribe to the queue through a channel and use event-name-based filtering to trigger appropriate workflows.

Locating Help Information On Gateway Events

1. Open the Gateway Events window in MSS.
2. On the navigation bar, select **Application Setup**, and then click **Work Management Setup**.

   The Work Management Setup window is displayed.
3. Click **Gateway Events**.

   The Gateway Events window is displayed.
4. Press **F1** for Help.

   The Help window for the Gateway Events window is displayed. There are a number of links on this window that explain gateway events and how to create them.

3

Running the Oracle WebLogic Server

Oracle Communications MetaSolv Solution (MSS) provides several means of monitoring and maintaining the product at its intended capacity. For instance, regularly monitoring log files can assure continued high performance. Other normal maintenance tasks are required when changes or problems occur. This chapter provides detailed instructions about the following:

- Starting and stopping the MetaSolv Solution application server
- Managing the MetaSolv Solution log files
- Monitoring the client logons
- Managing the Oracle WebLogic server
- Resetting the server-to-database connections

When tuning the Oracle WebLogic server, refer to "Oracle WebLogic Server Configuration Settings" for suggested settings.

When changing hardware, refer to the *MetaSolv Solution Planning Guide* for recommendations.

Starting and Stopping the MetaSolv Solution Application Server

The MetaSolv Solution database and servers should always be up and running, ready for access by clients. Shutdown should occur only during normal system maintenance.

Components must be started in the following order:

1. Database
2. Application server
3. MetaSolv Solution client

After the database is up and running, use the MetaSolv Solution start script to start the Oracle WebLogic server. The script sets values for arguments and provides the settings required to make the Oracle WebLogic server start successfully.

The start scripts are located in the domain directory. The script name includes the server name. For example, if a server is named MSLV01, it will have a start script named startMSLV01.sh in the domain directory.

Note: If you change the HTTP port for the server process (using the Oracle WebLogic domain wizard) without reinstalling MSS, you must also change it in the start script for the server process.

To start MetaSolv Solution with a script:

MetaSolv Solution provides scripts to start a server. Use the following command depending on which type of machine you are using:

- For UNIX/Linux:

  `./startMSLVServer.sh <server_name><host_admin_URL>`

- For Windows:

  `startMSLVServer.cmd <server_name><host_admin_URL>.`

Use the following as the admin URL argument:

`http://host_name:admin_port`

Use the following as the admin URL argument when using SSL port:

`https://host_name:admin_sslport`

Note: To start or stop the administration server and managed servers (in a cluster environment) using the SSL port, you must add an **s** after **http** in the ADMIN_URL argument in the startup/stop server scripts for the administration server and for each managed server. For example:

`https://host_name:admin_sslport`

If running in a cluster environment:

After managed servers are started, the Admin server can be shut down to release resources that the process would otherwise consume. However, the Admin server must be running to allow access to the Administration Console or to perform any administrative function.

To stop MetaSolv Solution on a Windows server, choose one of these options:

- Close the application on the server by clicking the **X** box.
- Use the Management console to stop a server: Right-click the server name in the left column and select **Start/stop this server**, then click **Shutdown**.

To stop MetaSolv Solution on a UNIX server, follow these steps:

1. Issue a **ps** command to show all the processes containing a java string in the description.
2. Issue a **kill** command to stop the selected process.

Memory amounts specified, when server process are started, identify servers. For example, to stop a managed server process, you might look for a process with java and Xms512m because the Xms512m identifies the managed server process. The admin server uses Xms32m instead of Xms512m.

To stop MetaSolv Solution with a script:

MetaSolv Solution provides scripts to shut down a server. Use the following command depending on which type of machine you are using:

- For UNIX/Linux:

  `./stopMSLVServer.sh <server_name><host_admin_URL>`

- For Windows:

  `stopMSLVServer.cmd <server_name><host_admin_URL>.`

Use the following as the admin URL argument:

`http://host_name:admin_port`

Use the following as the admin URL argument when using SSL port:

`https://host_name:admin_sslport`

Managing the Oracle WebLogic Server

The Oracle Administration Console is an administrative tool for managing resources, including such tasks as starting and stopping servers, balancing the load on servers or connection pools, tuning and monitoring the configuration of resources, detecting and correcting problems, monitoring and evaluating system performance, and making sure the application is correctly deployed to the target servers or to a cluster.

See the WebLogic Server documentation for more information.

See "Delivering File Changes" for specific configuration and tuning recommendations for MetaSolv Solution and the WebLogic Server. You can access the Oracle WebLogic server using a Web browser at:

`https://host:port/console`

Where:

- *Host* is the name of the administration server machine.
- *Port* is the administration server port number.

> **Note:** Oracle recommends that you access the WebLogic Server using Hypertext Transfer Protocol Secure (HTTPS).

Or you can use the MetaSolv Solution Runtime page link to the WebLogic Administration Console. This link appears on the Runtime Page drop-down. "Getting Oracle WebLogic Server Runtime Information" for details about the Runtime page.

The Administration Console manages the following WebLogic Server tasks:

- Configuring the server
- Tuning the Oracle WebLogic server
- Debugging and monitoring
- Deployments
- Managing WebLogic logs
- Monitoring Oracle WebLogic server performance
- Executing queues
- Managing database connections
- Managing Oracle WebLogic security

Follow directions in the Oracle documentation to perform specific WebLogic tasks. Follow instructions in the *MetaSolv Solution Installation Guide* regarding basic Oracle WebLogic settings before making changes.

Getting Started with the Administration Console

The Administration Console is a web application that uses Java server Pages (JSPs) to access the management resources.

After starting the Administration server, you can start the Administration Console by using the following procedure.

To start the Administration Console:

1. From your browser, type the following URL:

   `http://host_name:port/console`

The value of *host_name* is the DNS name or IP address of the Administration server. The value of *port* is the address of the port on which the Administration server is listening for requests.

When started, the Administration Console prompts for a password. The first time the Administration Console is started, you can use the user name and password under which the Administration server was started. The Administration Console appears.

> **Note:** The Administration server must be running before you can use the Administration Console.

Among the key tasks you will perform using the Administration Console are:

- Managing WebLogic logs
- Executing queues
- Managing database connections
- Monitoring the Oracle WebLogic server through the Runtime page

> **Caution:** See "Delivering File Changes" before making any changes to the number of threads.

Getting Oracle WebLogic Server Runtime Information

The MetaSolv Solution Runtime page provides detailed Oracle WebLogic server information. It also provides links to important administrative tools that provide specialized information.

To view the Runtime Information page:

- Enter the following in your Web browser:

  `http://host_name:port/main`

> **Note:** If you start the Administration server using Secure Socket Layer (SSL), you must add an **s** after **http**. For example:
>
> `https://host_name:port/main`

The Runtime Information page is displayed.

Server Details

The MetaSolv Solution Appserver Runtime page provides a detailed snapshot of an Oracle WebLogic server, providing details about the following:

- Client Information
- Enterprise Archive (ear) Version
- Server Information
- WebLogic Server instance
- URL
- Stored Procedures with Invalid Version (if any)
- JAR file information
- Classpath JARs
- Java System Properties

Links to Administrative Tools

You can also use the Runtime page drop-down to:

- Link to the Administration Console
- Reset connection pools

You can view a runtime page for each server in your MetaSolv Solution configuration. The information in each report should be the same. If it is not, the Oracle WebLogic server in your MetaSolv Solution system are out of sync and you may need to re-install the product.

Managing MetaSolv Solution Log Files

MetaSolv Solution provides three consolidated log files for all modules. When system or user errors occur, you can review a file and discover the source of the problem.

Table 3–1 describes the log files.

Table 3–1 Log Files

File name	Purpose
appsrverlog.xml	System-wide log file.
appserverlog_misc.xml	Miscellaneous catch-all log file. This file traces any activity not specified for `APPSERVERLOG`. It can also be used as an extra log file when troubleshooting, one for which you can set different specifications.
appserver_auditlog.xml	This file contains the records of log on and log off activities, and includes successful and failed client logon attempts. Each attempt to log on is recorded, whether it is successful or not. Since, the audit log file contains the logon and logoff information only, a large archive can be maintained for longer periods of time than when using **appsrverlog.xml.** The **appserver_auditlog.xml** configuration is independently controlled from the main log file's configuration (using the AuditFileApp element of **loggingconfig.xml**).

Using the Log Viewer

You can use the Log Viewer for reviewing MetaSolv Solution log files created for the Oracle WebLogic server. (WebLogic Server log files are accessed through the WebLogic console.) The Log Viewer lets you find the kind of information you want quickly without manually searching through pages of log files. For example, you can use the filter to view error messages only.

To use the Log Viewer, JAVA 8 (with the latest critical patches) must be installed and you must be working on either a Windows or a UNIX workstation.

To open the Log Viewer:

1. Navigate to the following directory:

   MetaSolv_home/*server_name*/**appserver/bin**

2. Start logviewer.
   - For Windows, double-click **logviewer.cmd**.
   - For UNIX, double-click **logviewer.sh**.

For UNIX users, please note that the following information is excerpted from the discussion on installing on UNIX platform in the *MetaSolv Solution Installation Guide*.

The following list contains special tasks required for graphics on a UNIX machine:

- To execute the installation using a graphical user interface:
  - On a workstation, start Hummingbird Exceed or another X-Windows emulator.
  - On the UNIX machine, set the DISPLAY environment variable to send the graphical display to the workstation.

    ```
    $DISPLAY=mymachinename:0.0;export DISPLAY
    ```

- Enable xhost for Oracle WebLogic server that run on a UNIX machine.

  To enable the lookup of graphic settings on the Oracle WebLogic server, you must enable xhost on the machine. Run the following command on the Oracle WebLogic server machine while logged on as root:

  ```
  xhost +
  ```

As you work:

1. Leave the command window open.
2. Select the **logviewer.cmd** for Windows or **logviewer.sh** for UNIX/Linux to see a Log Viewer.
3. Choose a log file from the File menu.
4. Select a message severity level in the **Level** drop-down.

The display will include a list of events that occurred at the level selected and at all higher levels.

If you check the **Exclusive Levels** check box, only events at the level selected will display.

To make your search more specific:

1. Enter criteria in the fields at the top of the window.

2. For detailed information about any event, select it and view the details in the lower part of the window.

The **Exit**, **Clear**, **Pause**, and **Refresh** buttons control the Log Viewer. The **Next**, **Previous**, **First/Newest** and **Last/Oldest** buttons let you change the log file you are viewing from the current log file to one of the ten previous archived log files.

Table 3–2 describes each field of the **appsrverlog.xml** and **appsrverlog_misc.xml** log files.

Table 3–2 MetaSolv Solution Log File Fields

Field	Description
appServerName	Application server where the event occurred.
className	Name of the JAVA class logging the message. This is typically where the problem occurred. If a developer has not specified this field, it appears as an indeterminate "DEF_CLASS." Provide this information if working with Oracle Global Customer Support.
debugCode	A message may simply contain a zero or an arbitrary number for this debug code.
Events	Occurrences of log messages that meet the criteria specified.
Level	Severity of the event. Severity levels are: Alert, Fatal, Error, Warning, Performance, Info, and Debug. (Performance is not currently in use.)
machineName	Identifies the hardware.
Message	Information provided about the event.
messageID	A unique ID for the message that may be useful if working with Oracle Global Customer Support.
moduleName	Specifies the product module in which the event occurred, such as PSR, Security, etc. Audit file logs occur only with the Security module.
productName	MetaSolv Solution.
thread	Debugging aid for internal development.
Time	Time the event occurred.
userName	Identifies the user associated with the event.

Archiving the Log Files

When a MetaSolv Solution log file reaches the maximum size specified in **loggingconfig.xml**, it is automatically archived. The default size is 1024 kB. The ten newest archived log files are maintained and older files are automatically deleted. As each log file is archived, the older files are renamed.

When a log file is archived, MetaSolv Solution appends a number to the file. The lowest number is the newest file. For instance, your company's three newest archived log files are named:

- appsrverlog.xml.1 (Newest archive)
- appsrverlog.xml.2
- appsrverlog.xml.3 (Oldest archive)

Changing the Log File Location

The main logging directory to which log files will be written is specified in the start script **startMSLVmanaged.sh** or **startMSLVsingle.sh** in the domain directory. This value is determined by the **mslv.log.dir** setting specified in this script. By default this value will be set to the **appserver/logs** directory under each server's installation directory.

You can modify the **mslv.log.dir** value in the **startMSLVmanaged.sh** or **startMSLVsingle.sh** script after the installation is completed.

The new value must point to a directory, where the Oracle WebLogic server has permission to create and delete log files.

Setting the loggingconfig.xml Parameters

When troubleshooting, you can change settings in the logging configuration file so that more or less information is captured in the log file for any given module of the software.

To set the **loggingconfig.xml** parameters:

1. Access the **loggingconfig.xml** file in the Oracle WebLogic server config folder.
2. Make a backup copy.
3. Make any necessary parameter changes using any text editor.

   For UNIX/Linux, a vi, emacs or /usr/dt/bin/dtpad editor may be used.

   On the Windows platform, Notepad or WordPad may be used for this purpose.

Most parameters in **appserverlog.xml** should not be changed, and are reserved for error situations that may require assistance from Oracle Global Customer Support. However, as you encounter system or user problems in day-to-day operations, you will want to reset some parameters so you can screen for specific information.

The **loggingconfig.xml** file consists of several sections, listed in Table 3–3.

Table 3–3 loggingconfig.xml Sections

Section	Description and parameters
Message Resourcebundles specification	Defines the resource bundle names and their locations in the JAVA library packages. This section must never be modified unless instructed by Oracle Global Customer Support.
Pluggable interface specification	Implementation classes for the logging framework. This section must never be modified unless instructed by Oracle Global Customer Support.
SQL Tracing configuration	Captures SQL debugging messages when JDBCTrace module's debug value is also set to "on." Only one parameter can be changed: `<param name="debug" value="off"/>`

Table 3–3 (Cont.) loggingconfig.xml Sections

Section	Description and parameters

Row Retrieval Specification	Defines the number of rows that a user is allowed to retrieve and describes the threshold for the "too-many-rows" alert.
	The two parameters are:
	`<param name="query-results-row-limit" value="32000"/>`
	`<param name="too-many-rows-alert-threshold" value=20000"/>`

Logging File specification and configuration	Describes whether log file information is appended or written over, log file size, archiving specifications. See Table 3–4, " Logging File Editable Specifications" for information about editable parameters.
Category	Modules and components for which event levels can be set. Valid levels are **fatal**, **error**, **warn**, **info**, and **debug** in decreasing order of severity. See Table 3–5, " Logging Categories" for a complete list.

Editable Logging File Specifications

The following table describes the editable logging file specifications that can be changed. Locate the listed keyword in the file and edit the parameters shown. Only the following log files can be edited:

- appserverlog.xml
- appserverlog_misc.xml
- appserver_auditlog.xml

The last section of the **loggingconfig.xml** file lets you specify the level at which you want to trace events for each category.

Table 3–4 lists the logging file editable specifications.

Table 3–4 Logging File Editable Specifications

Keywords to search	Editable Logging File Specifications

appender name="XMLFileApp"	`<param name="MaxFileSize" value="10000KB"/>`
	`<param name="append" value="true"/>`
	`<param name="MaxBackupIndex" value="10"/>`
	`<param name="File" value="${mslv.log.dir}/appserverlog.xml"/>`

<appender name="AuditFileApp" class="org.apache.log4j.RollingFileAppender">	`<param name="append" value="true"/>`
	`<param name="MaxFileSize" value="10000KB"/>`
	`<param name="MaxBackupIndex" value="10"/><param name="File" value="${mslv.log.dir}/appserver_auditlog.xml"/>`

<appender name="XMLFileAppNew" class="org.apache.log4j.RollingFileAppender">	`<param name="append" value="true"/>`
	`<param name="MaxFileSize" value="10000KB"/>`
	`<param name="MaxBackupIndex" value="10"/>`
	`<param name="File" value="${mslv.log.dir}/appserverlog_misc.xml"/>`

Table 3–5 lists the available modules for setting trace levels.

Caution: Do not keep event severity levels at Info or Debug level during normal operations. Doing so would cause all events to be logged. Logging all events can have an adverse impact on system performance.

Table 3–5 Logging Categories

Category	Description or Direction
CaCache	Custom attributes (CAs) are characteristics end users can define generically and associate with network templates, element types, connection types, connection specifications, network systems, and allocations of connections or elements within a system.
DLR	Oracle's software supports two types of design layout reports (DLRs): ■ Internal DLRs (also known as on-net, or outbound, DLRs) ■ External DLRs (also known an off-new, or inbound, DLRs) Internal DLRs have always existed for circuits designed in Oracle's software and have been referred to merely as DLRs. The external DLR process begins when a customer sends a service request to a provider requesting the use of part of their network for designing connections. The provider receives the service request and sends the customer a firm order confirmation (FOC) to verify the service request. See the Help for more information.
DLR_Equipment	When users edit equipment or equipment specifications, or move equipment with assigned connections, the design layout records (DLRs) for those connections are reconciled to reflect the change - including pending assignments. See the Help for more information.
DLR_NGN_Activation	References NGN activation. See the Help for more information.
DLR_Circuit	References DLR circuits. See the Help for more information.
EventServer	For MetaSolv Solution internal use only. Use the default settings.
Event2Server	For MetaSolv Solution internal use only. Use the default settings.
Framework	MSS internal framework code that is used/depended on extensively by the other modules. This consists of logging, security, desktop portlet related logging.
Framework.Cache	MSS internal framework code that specifically deals with database query caching.
GLR	GLR is a generic term used to refer to the Graphical Layout Record. The GLR provides information about a specific virtual connection, such as administrative information, design lines, and a graphical-view of the design. The report is generated from Connection Design.
invFromPB	For MetaSolv Solution internal use only. Use the default settings.
Infrastructure	Refers to application startup settings listed on the GUI navigation bar.
Infrastructure_NetworkLocation	Refers to Network Location settings in the GUI.
INTEGRATION_CONVERSION	For MetaSolv Solution internal use only. Use the default settings.
WORK_MANAGEMENT_ CONVERSION	For MetaSolv Solution internal use only. Use the default settings.
INTEGRATION_EVENT_SERVER	For MetaSolv Solution internal use only. Use the default settings.
INTEGRATION_MANAGER	For MetaSolv Solution internal use only. Use the default settings.

Table 3–5 (Cont.) Logging Categories

Category	Description or Direction
JDBCTrace	The module which logs the SQL statements before they are executed for D/B bound operations.
LSR	Local Service Requests are managed by the LSR software and referenced by MetaSolv Solution.
MSLVSessionBean	This exists to log all the entry and exit messages regarding the invocations of various methods in all Stateless Session type Enterprise Java Beans in the MSS application.
NetworkManagement	For MetaSolv Solution internal use only. Use the default settings.
NetProv	For MetaSolv Solution internal use only. Use the default settings.
NetProv_Plant	For MetaSolv Solution internal use only. Use the default settings.
NetProv_Activation	For MetaSolv Solution internal use only. Use the default settings.
NI	For MetaSolv Solution internal use only. Use the default settings.
PSR	PSR provides service request tracking and provisioning functionality for products defined in the product catalog.
E911	E911 functionality lets a user enter and maintain detailed information about a telephone number and its service location.
CNAM	Caller Name identification (CNAM) lets users view the calling name and telephone number of a caller on a caller ID display panel.
LIDB	Line Information Database (LIDB) lets users place restrictions on third-party calls.
PSRAncillary	For MetaSolv Solution internal use only. Use the default settings.
PSREUB	PSR End User Billing extracts billing information from the order to pass to an end user billing interface.
Security	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
SYSTEM	MetaSolv Solution startup and shutdown and DB initialization module related logging.
TEST	For MetaSolv Solution internal use only. Use the default settings.
TimedEventProcess	For MetaSolv Solution internal use only. Use the default settings.
TMS	For MetaSolv Solution internal use only. Use the default settings.
TroubleBatchWorkerThread	For MetaSolv Solution internal use only. Use the default settings.
KeepAliveTroubleBatch	For MetaSolv Solution internal use only. Use the default settings.
WDI	For MetaSolv Solution internal use only. Use the default settings.
WM	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
WM_TskSv	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.

Resolving a Sample Problem Using Log File Messages

The following scenario demonstrates how to use a log file to resolve a common problem.

Error

An incorrect database name, **dNoSuch**, is entered in the Session section of the **gateway.ini** file. There is no such name in the **tnsnames.ora** file.

Presentation

When an attempt is made to start up the system, the WebLogic Server console log displays this message:

```
**********************************
GenericStartup.init() failed.
PLEASE CONSULT appserverlog.xml file in the C:\bea2\user_projects\domains
\mslvhome\appserver\logs directory for detailed message!
**********************************
```

A corresponding message appears in the **appserverlog.xml** file.

Problem Determination

To determine the source of the problem, the administrator opens the **appserverlog.xml** file using the Log Viewer, and selects a level of Error. The administrator locates the event because the Log Viewer displays the following message:

```
<log4j:cause><![CDATA[Could not connect to database.]]></log4j:cause>
<log4j:action><![CDATA[Could not connect to database.]]></log4j:action>
<log4j:throwable><![CDATA[java.sql.SQLException: ORA-01017: invalid
username/password; logon denied
```

The administrator must determine whether such an entry exists in the **tnsnames.ora** file or in the **gateway.ini** file.

Resolution

Assuming that the **gateway.ini** file is incorrect, the administrator follows these steps:

1. Shut down the Oracle WebLogic server.
2. Modify the **gateway.ini** entry and enter a correct database name.
3. Restart the Oracle WebLogic server.

Sample appserverlog.xml Log File Error Section

The **appserverlog.xml** log file contains the following information about this event:

```
<log4j:event logger="cmm.Framework" timestamp="1327054086515" level="ALERT"
thread="kanlddev7110d_gateway_Main" dateTime="Fri Jan 20 02:08:06 PST 2012"
appServerName="DEF_APPSERVER" machineName="tanxdev7110d" messageID="10071"
moduleName="cmm.Framework" userName="User1" className="DEF_CLASS" debugCode="0"
productName="nur">
<log4j:message><![CDATA[Could not connect to database - Problem using Session
information in gateway.ini to access database.]]></log4j:message>
<log4j:cause><![CDATA[Could not connect to database.]]></log4j:cause>
<log4j:action><![CDATA[Could not connect to database.]]></log4j:action>
<log4j:throwable><![CDATA[java.sql.SQLException: ORA-01017: invalid
username/password; logon denied
```

4

Managing the MetaSolv Solution Database

After an installation or an upgrade, and after users are added, Oracle Communications MetaSolv Solution (MSS) requires database adjustments only in response to certain conditions. This chapter provides detailed instructions about the following tasks as they directly relate to MetaSolv Solution:

- Performing general database maintenance
- Managing scalability
- Monitoring and maintaining database performance
- Backing up and recovering the database

The Oracle database provides tools that allow you to monitor its behavior and performance. Refer to the Oracle database documentation for information about the standard monitoring tools.

MetaSolv Solution-specific deployment details that affect the database are documented in *MetaSolv Solution Installation Guide*. To determine the type of deployment in use at your site, review your company's deployment diagram.

MetaSolv Solution DBA Requirements

The MetaSolv Solution DBA must have a thorough understanding of administering, maintaining, and tuning an OLTP system. Refer to the Oracle database documentation for instructions.

Among the MetaSolv Solution DBA's duties are:

- Configuring the database
- Upgrading
- Adding and deleting databases
- Maintaining the database
- Managing database connections
- Managing scalability
- Managing performance
- Backing up and restoring the database
- General administration tasks

The remaining sections in this chapter address how these tasks relate to MetaSolv Solution. The DBA is expected to be proficient in all of these tasks.

Installing, Upgrading, and Configuring the Database

Refer to the Oracle documentation for information about the following items:

- Understanding memory and file structure components
- Understanding **INIT.ORA** parameters
- Implementing Oracle database patches and upgrades
- Understanding platform-specific parameters
- Creating databases as needed for production and testing
- Understanding database file sizing issues
- Implementing security for database access outside of the MetaSolv Solution environment

Maintaining the Database

Oracle expects the DBA to be able to:

- Establish and refresh Oracle **DBMS_STATS** for viewing and modifying optimizer statistics
- Correct table and index sizing issues, including chained rows and number of extents
- Review Oracle system trace and error logs daily for errors
- Rebuild tables, tablespaces, or indexes as needed
- Import or export databases as needed

Oracle recommends that an Analyze Statistics job be scheduled once a week to refresh statistics. If significant data input operations are started on a previously empty table or tables, run statistics more frequently.

Refer to the Oracle database documentation for complete instructions on these maintenance tasks.

Managing Specific MetaSolv Solution Issues

The following sections describe specific Oracle issues that affect MetaSolv Solution.

Adding Database Administrators for MetaSolv Solution

Initial DBA authority is assigned by running a_users.sql during a new install, granting the DBA privilege to the ASAP user. This file is not delivered with service packs because the ASAP user is already set up. See *MetaSolv Solution Installation Guide* for details about **a_users.sql**.

Handling Import and Export Requirements

The MetaSolv Solution database structure includes public synonyms, roles, Oracle users, and privileges granted to roles that must exist to have a fully functional database. These objects are not included when importing or exporting in table or user mode.

When importing a MetaSolv Solution database, MetaSolv Solution does not import the structure separately from the data. The MetaSolv Solution database uses Oracle sequences having a data component which is imported with other Oracle objects. If the

imported data does not coincide with the sequence value, data integrity can be lost and MetaSolv Solution functionality can be affected.

Note: Set the **_grant_secure_role** parameter to TRUE to import roles appropriately.

Note: When testing the upgrade of an existing database, use **Compress=N** to retain the existing table sizes in the new environment.

When creating a copy of the database, always use **Full** mode for both the import and export processes. As for any OLTP system, it is important to export with the **CONSISTENT=Y** option. Exporting a database without using the **CONSISTENT=Y** parameter may cause data inconsistencies, including primary and foreign key errors.

You can use the Oracle export parameter **CONSISTENT=Y**, which allows you to export a database while users are still in the system, while not affecting the sequences.

Note: In a production database, there is no reason for sequences to be out of sync. If you encounter an issue related to sequences, you should contact Oracle Global Customer Support.

Handling Locally Managed Tablespaces

MetaSolv Solution resides on a database defined with tablespaces that are locally managed or dictionary managed.

Maintaining Job Type Table Data

Changes to the Job Type table data can have significant undesirable results; change them only under the direction of Oracle Global Customer Support personnel.

Caution: Check with GCC before making any changes to the Job Type table data.

Handling International Data

For MetaSolv Solution to handle international data, your Oracle database must support an international character set. An international character set provides character support for local scripts and cultural preference information, such as sorting sequences, date and time conventions, and currency notations.

Caution: For some customers, the US7ASCII character set does not support extended characters. Refer to the Oracle documentation for instructions.

To see what character set your Oracle instance is currently using, execute the following SQL query:

```
SELECT USERENV ('LANGUAGE') FROM DUAL;
```

This query returns the current sessions language, territory, and database character set in the following format:

```
<language>_<territory>.<database character set>
```

For information on the **NLS_LANGUAGE** and **NLS_TERRITORY** variables, refer to the Oracle documentation.

You can alter the **NLS_LANGUAGE** and **NLS_TERRITORY** parameters by changing the desired value in the Oracle initialization file and then restarting the instance.

You must change your character set if it does not meet the international standard.

To change your character set:

1. Create a new instance with the desired character set.
2. Export your existing database.

   Note: Test this process first to confirm that the new character set supports previous character sets.

3. Import the existing database into the new instance.

   By importing the existing database into the new instance with the desired character set, your database is capable of supporting international data.

Managing Scalability

Refer to the Oracle database documentation for scalability recommendations. Assistance with managing scalability parameters can be obtained through MetaSolv Solution Professional Services.

Consider the following **init.ora** parameters when managing scalability of the MetaSolv Solution database:

- **processes**

  Increase this parameter as the user count increases.

- **open_cursors**

  Increase this parameter as the user count increases.

- **dml_locks**

  This parameter represents the maximum number of DML locks, one for each table, modified in a transaction. Its value should equal the grand total of locks on tables referenced by all users.

  For example, if three users modify data in one table, three entries are required. If three users modify data in two tables, six entries are required. The number needs to increase as the user count increases.

- **db_block_buffers**

  As the number of users increases, use the STATSPACK report to monitor the data cache hit ratio. A decrease may indicate a need for more buffers.

- **Shared_pool_size**

  As the number of users increases, use the STATSPACK report to monitor the library cache hit ratio. Also, if the pool is set too low, the client application may receive server errors indicating not enough shared pool to load objects.

Managing Database Performance

The following are the **init.ora** parameters that are specific to MetaSolv Solution performance:

- Set the **db_block_size** parameter to 8 KB. A 2 KB block size is not large enough to support the largest index within the block. You must set the block size to 4 KB or larger.
- Set the **optimizer_features_enable** parameter to 12.1.0.2.1.
- Set the **optimizer_adaptive_features** parameter to FALSE.
- Set the **parallel_max_servers** parameter to 1. This parameter is appropriate for a batch job/data warehousing/reporting database rather than an online transaction processing (OLTP) system.

DBA Tasks

To maintain database performance for MetaSolv Solution, the DBA is expected to know how to handle these tasks:

- Establish benchmarks for your site and monitor database performance
- Understand and use STATSPACK reports
- Identify and correct fragmentation issues
- Monitor and correct database bottlenecks
- Adjust memory structures and file placement to improve performance
- Implement MetaSolv Solution database patches and upgrades as needed
- Access Oracle database documentation, or support, or both to resolve any Oracle database errors

Important tools used to monitor the performance of an Oracle database are the STATSPACK reports. For information on STATSPACK reports, go to My Oracle Support and view the documents on system tuning using STATSPACK reports.

> **Note:** Oracle recommends running STATSPACK reports multiple times a day during peak periods.

To successfully monitor the performance of your database, it is important to have a baseline reference. Create the STATSPACK reports during normal use of your system. This information can be stored for reference in report form, printed form, or moved into tables. Books on Oracle database performance tuning are available that describe how to do this.

Identifying Structure and Performance Problems with Scripts

To help identify potential database structure and performance problems, schedule the following scripts to run routinely:

- o_config.sql
- o_analyz.sql
- o_granta, o_grante, o_grantb, o_grantj.sql
- mdl_cmp.sql

- mdl_cur.sql
- mdl_upd_cmp.sql
- mdl_upd_cur.sql

All references to **@@asap.pwd, @@edi.pwd, @@ebond.pwd, @@job.pwd**, and **@@sys.pwd** contained in these scripts can be changed to hard-coded connect strings of **user/password@dbname**;. If the correct logons are maintained and you are logged on at execution time, remove the commands after reviewing the scripts. All references to **@@spooly** can be changed to hard-coded spool **path\\filename** commands. Refer to Oracle database documentation for more information on the spool and connect commands.

Starting in 6.0.2, customers are provided a **mdl_upd** directory in service packs. This directory contains SQL files that include any modifications to the database structure in a service pack. To apply these updates to the reports, copy the contents of this directory into either the **inst_60** or **upg_60** directory. After applying the PRODFIXSQL, you can validate the execution of the scripts by running the **mdl_upd_cur.sql** and/or **mdl_upd_cmp.sql.** These processes provide the reports **mdl_upd_cur.txt** and **mdl_upd_cmp.txt** files (similar to the **mdl_cur.sql** and **mdl_cmp.sql** scripts).

The following sections describes these scripts.

o_config.sql

The **o_config.sql** script identifies table and index objects having:

- Storage parameters below the MetaSolv Solution software standards
- Objects that are not stored in the correct tablespace
- Objects fragmented beyond 100 extents

> **Note:** To run this script outside of the upgrade process for database administration, type **@***path file_name* followed by the command **FORCE**, where *path* is the directory path of the script and *file_name* is the name of the script.
>
> Example:
>
> ```
> @C:\MetaSolvSolution\UPGRADE\O_CONFIG.SQL FORCE;
> ```

The following output files are stored in the Audits subdirectory:

- o_cfggen.sql
- o_cfgrpt.txt

o_cfggen.sql

This script is generated in the Audits subdirectory by the **o_config.sql** script and has the following characteristics:

- Contains SQL that can be applied to your database
- Generated for tables or indexes having storage parameters below the Oracle database standards and for indexes that are not in the indexes tablespace

> **Note:** Review this script before running it. Any large index rebuild might require a significant amount of run time and system resources.

o_cfgrpt.txt

The **o_cfgrpt.txt** file is a report of objects which are currently being stored with storage settings below minimum standards.

The SQL code can be run to correct the storage.

This report is generated in the Audits subdirectory by the **o_config.sql** script and contains a list of tables and indexes having more than 100 extents allocated to them. These items are good candidates to be rebuilt (if indexes).

o_analyz.sql

The **o_analyz.sql** script updates statistics used by the Oracle query optimizer and can affect performance. If the statistics are not updated, the optimizer can incorrectly choose a path to execute a query, resulting in poor performance. Consult the Oracle database documentation for discussions on index statistics and recommended update schedules.

Note: To run this script outside of the upgrade process for database administration, type *@path file_name* followed by the command **FORCE**, where *path* is the directory path of the script and *file_name* is the name of the script.

Example:

```
@ c:\MetaSolvSolution\upgrade\o_analyz.sql FORCE;
```

---

o_granta, o_grante, o_grantb, o_grantj.sql

As with the other maintenance scripts, run these scripts on a scheduled basis to ensure that database permissions are set correctly for all MetaSolv Solution related objects.

mdl_cmp.sql and mdl_upg_cmp.sql

You can run the **mdl_cmp.sql** and **mdl_upg_cmp.sql** scripts (Model Compare Scripts). You can run these independently of an upgrade to help maintain the database and performance levels. Oracle recommends that you run this script at least once a week. This helps maintain the database structure for primary and foreign keys, indexes, and sequences. The **mdl_cmp.sql** script provides the option to fix the problems found or only report the differences. When you run the script you have the option to compare the differences for individual sections or for all sections. The default is to display all sections. The script reports about:

- Extra or missing indexes
- Extra or missing primary keys, foreign keys, and sequences

To run this script outside of the upgrade process for database administration, type

```
@c:\MetaSolvSolution\upgrade\mdl_cmp.sql;
```

If you build additional indexes in the schemas, preface each additional index name with **PFXI_** or **PFXPK_**. This naming convention prevents the extra indexes from being dropped during the upgrade process. This is not required, if you choose not to copy these files and run the updated reports you may see extra database objects in the report.

mdl_cmp.txt and mdl_upd_cmp.txt

The **mdl_cmp.txt** and **mdl_upd_cmp.txt** reports identify differences in the following data model objects for this release:

- Indexes
- Primary keys
- Sequences
- Foreign key exceptions

Reports can be generated on one or more of the above objects. For reports not made on all objects, report summaries for the unselected objects indicate REPORT NOT SELECTED.

Additional details are available in the **\audits\mdl_cmp.txt** file. If the following statement is the only paragraph in the report, your database is 100 percent correct for these checks:

```
PLEASE REVIEW THE ABOVE SUMMARIES WHICH IDENTIFY THE DIFFERENCES FOR INDEXES,
PRIMARY KEYS, and FOREIGN KEYS (and FK exceptions) FROM THE MODEL FOR THIS
RELEASE.
ADDITIONAL DETAILS ARE AVAILABLE WITHIN THE \AUDITS\MDL_CMP.TXT FILE.
IF THIS IS THE ONLY PARAGRAPH IN THE REPORT, YOUR DATABASE IS 100% CORRECT FOR
THESE CHECKS.
```

mdl_cur.sql and mdl_upd_cur.sql

You can run the **mdl_cur.sql** and **mdl_upd_cur.sql** scripts (Model Current Scripts). You can run these independently of the upgrade to generate a report in the **Audits** subdirectory. The resulting **mdl_cur.txt** file contains a report of the current database compared to the MetaSolv Solution data model. This report identifies missing tables or columns, extra tables or columns, and differences with column data types.

mdl_cur.txt and mdl_upd_cur.txt

The **mdl_cur.txt** and **mdl_upd_cur.txt** reports list six areas where the current database structure can differ from the MetaSolv Solution data model. These areas are:

- Change Type: Refers to one of three different types; column change, missing table, or extra table. Column change can be one of three different changes:
  - Extra column
  - Missing column
  - Column difference
- Table: Lists the affected table
- Column: Lists the affected column
- Data Type Column: Lists the affected data
- Null: Lists the affected nullability positions
- Column Position: Lists the affected column positions

Tuning the Init.ora File

Aside from some basic machine configurations and regularly performing analyze statistics on indexes, your biggest gains or losses in performance probably come from changes to the **init.ora** configuration file.

There are many init.ora parameters. This section focuses on parameters that have the biggest impact on MetaSolv Solution performance:

- Major contributors to performance
- Parameters related to scalability adjustments

Some parameters overlap between the two. Almost all of the items can be monitored using the STATSPACK report. See "Managing Database Performance" for more information.

Major Contributors to Performance

The following parameters are major contributors to performance:

db_block_buffers

The value of **db_block_buffers** multiplied by **db_block_size** determines the size of the data buffer. The buffer cache stores tables, indexes, clusters, and rollback segments. The higher the number of block buffers, the less I/O takes place, and the better the system performs.

If excessive paging and swapping activity occurs for user processes or if any paging or swapping of the SGA occurs, reduce **db_block_buffers** to free memory. There is no need to set this above what is necessary.

The key monitoring device for adjustments should be the manually calculated data cache hit ratio in the STATSPACK report.

db_file_multiblock_read_count

Try setting the **db_file_multiblock_read_count** to 8, 16, or higher. This references the number of blocks that Oracle requests the operating system to read when performing a sequential scan. Therefore, if a query needs to read a single record from a table from disk, Oracle database reads not just the block containing the record, but also the number of blocks set by this parameter. This causes additional reads, but the theory is that you will save more disk reads later if queries request other data contained within the additional blocks that were stored in the data cache.

The parameter can be better tuned when matched with the operating system block size. Larger settings can cause a lower cache hit ratio if the additional blocks that were read aren't used by other queries and data has to get flushed to make room for newly cached data. Watch the cache hit ratio and other STATSPACK info.

shared_pool_size

This pool stores shared SQL and PL/SQL blocks, the data dictionary cache, and (if your site is using Multi-threaded server architecture) some session information. There is no need to set this way above what is necessary. Setting this parameter to high might cause Oracle database to store so many queries in the library that it will cause slower performance. This happens when the Oracle database spends more time searching for a cached copy of the query than it would have spent parsing the query. A good indicator that this is happening is improved performance when the shared pool is manually flushed.

sort_area_size

This is the size, in bytes, that a user process has available for sorting. This setting might be dramatically different for each customer, depending in which parts of the MetaSolv Solution are used. Start with at least 300k. Beyond that, watch for the sorts (disk) in the STATSPACK report as an indicator to increase the size. Ideally, there should be no sorts to disk. Be aware that the memory gets pre-allocated per user connection. This can have a significant effect on total system memory used by the Oracle instance.

sort_area_retained_size

This is the size, in bytes, to which Oracle reduces your sort area if sort data is not being referenced. Memory is reduced only after all the rows have been retrieved from the sort space. Sometimes, a number of concurrent sorts may be required, and each is given its own memory allocation, which is determined by this parameter. Match **sort_area_size** per several Oracle white papers' recommendations.

Other parameters to adjust

Depending on the performance analysis from the UTLBSTAT, UTLESTAT, and STATSPACK reports, adjust the following parameters:

- **db_writer**: Increase **db_writer** processes if asynchronous I/O is turned off at your OS level. This parameter is set to **1**. Otherwise, try adjusting it according to your tuning statistics report. In Oracle 8 and above, Oracle introduced **dbwr_io_slaves,** so be sure to read the tuning information on these two parameters and adjust them accordingly.
- **enqueue_resources**: Increase the number of **enqueue_resources** to reduce the **enqueue_waits** value in the STATSPACK report. For more information, refer to the Oracle white paper, *Enqueue_Resources.txt*.
- **sort_write_buffer_size**: Try increasing the **sort_write_buffer_size** parameter. This is the size of each sort write buffer used for direct writes.
- **log_checkpoint_interval**: If fast recovery time is required, keep a value lower than the size of the redo logs and at a size that is a whole fraction of the redo log size. If performance is the more important consideration, make it equal to or greater than the size of the redo logs.
- **log_checkpoint_timeout**: Set this parameter to **zero** (consider making it equal to the redo log.)
- **redo logs**: Make sure the redo logs are not switching too often. Oracle recommends having between 15 to 30 minutes between switches. Increase the size of your redo logs if it is less than 15 minutes and keep increasing it until the time is in the 15 to 30 minute range.

Tuning with Non-init.ora Files

To tune **non-init.ora** files:

1. Use striped disks to improve the I/O throughput. Here is a sample configuration:
   - Raid 0+1 (striped and mirrored) for rollback segments, log files, and temp and index tablespaces
   - Raid 0 (mirrored) for Archive log files
   - If disk cost is not a problem, use Raid 0+1 (striped and mirrored) for everything. Buy more disks to spread your files across separate disks. This helps with I/O performance, and when you feel a bottleneck is occurring you are more easily able to isolate what is causing the bottleneck.

Note: Oracle research has shown that in most cases, Raid 0 performs faster than Raid 5.

2. Separate the Oracle database files from the OS, Oracle database program files, and Oracle Archive log files on different disks.
3. Do not allocate so much memory to the Oracle database and other applications that the OS has to perform memory/disk swapping or paging. A good rule is to not allocate over half the amount of the machine's available physical memory.
4. Limit the server to database-only use. If the OS is being taxed by other resources, the Oracle database runs slow regardless of how well-tuned your instance is.
5. For production, run only one instance of the Oracle database per machine. The more Oracle database instances running on a machine, the harder it is to tune the instance with system resources.

Backing Up and Restoring the System

Whatever backup and recovery solution you choose, test the recovery process on a regular basis. Do not wait for an outage to test your processes and procedures. The DBA should be able to:

- Understand the solutions implemented
- Understand backup archive mode
- Implement a documented, customer-developed backup and recovery plan that meets your company's standards for acceptable down-time and off-site backup storage and retrieval
- Test the database recovery process on a regular basis

For clustering solutions, the MetaSolv Solution database supports an active/passive solution. Any Oracle-supported software vendor that provides an active/passive solution is supported.

Technology management and effectiveness requires specialized experience, skills, and strategies, one of which is the back-up and disaster recovery strategy. It is imperative that a contingency plan be in place to provide continued MetaSolv Solution functionality in the event of server faults, database crashes, other system down-time, as well as scheduled maintenance and updates.

Oracle recommends that the owner of this area come from IT.

Any approach to backup and recovery should, at a minimum, include the following tasks:

1. Identify database environments requiring periodic back-ups
2. Identify regular frequency of back-ups
3. Identify storage area for back-up media
4. Document the back-up procedure and conduct knowledge transfer sessions with IT personnel
5. Document the Disaster Recovery Plan and conduct knowledge transfer with IT personnel
6. Test the Disaster Recovery Plan on a regular basis

5

Customizing the Application

Oracle Communications MetaSolv Solution (MSS) provides three ways for the administrator to customize the application. You can:

- Customize the desktop
- Customize reports
- Code stored procedures in user exits

> **Note:** Coding the user exits is a developer task and usually requires DBA assistance.

Customizing the Desktop

The main MetaSolv Solution desktop is a portal with one default portlet, the Welcome Page. Portlets are the building blocks of a portal. They are a convenient way to insert links to features or applications into the desktop.The user's MetaSolv Solution desktop can include a choice of portlets if you, the administrator, customize the default portlet and/or if you add portlets or permit users to customize their desktops.

In the MSS application, the administrator desktop displays a name for the *System View* and a pen icon. The pen icon appears only on the administrator view. Clicking the pen icon lets the administrator access windows to alter the default desktop.

Oracle ships pre-defined portlets, specified in **metasolv-portlets.xreg** registry file. If you allow it, users can select portlets for their desktops that are already defined or that have been defined by you. To allow users to customize their desktops, set the **Allow users to customize My Desktop** preference to **Y.**

Take note of the System View name and the pen icon at the top of the desktop. If you have not made any changes to the default portal or enabled users to customize their own desktops, the name area does not display System View or the pen icon.

To allow users to customize the default desktop, ensure the **Allow users to customize My Desktop** preference is set to **Y**.

The **Allow users to customize My Desktop** preference is located under **Preferences** > **System Information**.

To change the default desktop seen by users:

1. Log on as the user ID specified in the **gateway.ini DefaultPortalID** parameter, usually known as the ASAP user
2. Click the pen icon beside the System View.

3. Follow directions in the online Help to make selections that set up the default desktop.

   Instructions for customizing the desktop appear in the online Help.

Configuring Portlets

The portlet registry file, **my-portlets.xreg** defines the portlets that you can select when customizing the desktop. If a portlet is to a third-party application and is located on another server, this file can define a logon alias so that users can access the other applications without logging on separately.

Metasolv-portlets.xreg is installed in **/appserver/portal/conf**, within the MetaSolv Solution root directory. It is a system file and should not be modified.

The **My-portlets.xreg** file starts with - <registry> and ends with </registry>. Each portlet is defined between <portlet-entry> and </portlet-entry>, as shown in the following sample. There can be more than one parameter per portlet.

The following is a sample of the **My-portlets.xreg** file:

```
<?xml version="1.0" encoding="UTF-8"?>
- <registry>
- <portlet-entry name="MetaSolvHomePage" hidden="false" type="ref" parent="HTML"
application="false">
- <meta-info>
  <title>Oracle</title>
  <description>Oracle Home Page</description>
  </meta-info>
  <classname>org.apache.jetspeed.portal.portlets.FileServerPortlet</classname>
  <url cachedOnURL="true">http://www.metasolv.com</url>
  <category group="Jetspeed">myGroup</category>
  <parameter name="base" value="http://www.metasolv.com/" type="‟ hidden="false"
cachedOnName="true" cachedOnValue="true" />
  </portlet-entry>
  </registry>
```

The following is a sample file with multiple portlets:

```
<?xml version="1.0" encoding="UTF-8"?>
<registry>
    <portlet-entry name="MetaSolvHomePage" hidden="false" type="ref"
        parent="HTML" application="false">
        <meta-info>
            <title>Oracle</title>
            <description>Oracle Home Page</description>
        </meta-info>

<classname>org.apache.jetspeed.portal.portlets.FileServerPortlet</classname>
        <url cachedOnURL="true">http://www.metasolv.com</url>
        <category group="Jetspeed">myGroup</category>
        <parameter name="base" value="http://www.metasolv.com/" type=""
hidden="false"
            cachedOnName="true" cachedOnValue="true"/>
    </portlet-entry>
    <portlet-entry name="GooglePage" hidden="false" type="ref"
        parent="HTML" application="false">
        <meta-info>
            <title>Google</title>
            <description>Google Search Engine</description>
        </meta-info>
```

```
<classname>org.apache.jetspeed.portal.portlets.FileServerPortlet</classname>
        <url cachedOnURL="true">http://www.google.com</url>
        <category group="Jetspeed">myGroup</category>
        <parameter name="base" value="http://www.google.com/" type=""
hidden="false"
            cachedOnName="true" cachedOnValue="true"/>
    </portlet-entry>
</registry>
```

Table 5–1 lists the available parameters in the **my-portlets.xreg** file.

Table 5–1 Registry File Parameters

Registry File Parameter	Parameter Description
portlet	(Optional) The parameter element has attributes and can be copied and edited for new elements.

Meta-Info	Meta-info has two elements that should be changed to match a custom portlet:
	■ The title will be displayed in a portlet list as well as in the portlet title bar
	■ The description will be displayed in the portlet list.
	Sample:
	`<meta-info>`
	`<title>ABC</title>`
	`<description>ABC Home Page</description>`
	`</meta-info>`

Classname	Classname is the portlet class which retrieves the content from the application, the value should be:
	`org.apache.jetspeed.portal.portlets.FileServerPortlet.`
	`<classname>org.apache.jetspeed.portal.portlets.FileServerPortlet`
	`</classname>`

Table 5–1 (Cont.) Registry File Parameters

Registry File Parameter	Parameter Description
Base parameter	The base parameter helps locate the portlet: If the name is **base**, the application content's base is set to the URL specified in the value attribute. Some applications generate content which requires resources such as images. These resources may be referred to by the relative path to the root of the content. In this case, without the "base" URL set correctly, the icons and images will not be displayed. Most likely the base URL is first segment of the URL. For example, if the content is retrieved from http://www.oracle.com/MSLV/CDA/General/Homepage/ the base is http://www.oracle.com/. If you are not sure about the base URL, ask the application Web Master. Following is a sample base parameter: `<parameter name="base" value="http://www.oracle.com/" type="" hidden="false" cachedOnName="true" cachedOnValue="true"/>`
Parameters sent to the application	If the application requires a specific parameter (such as a stock symbol) you can define the parameter elements so that the parameter name and value attributes are sent to the application as the HTTP Request parameters. You must know what the application expects for the parameter name and define it identically and in the same case in the portlet-entry. For example: `<parameter name="symbol" value="MSLV" type="" hidden="false" cachedOnName="true" cachedOnValue="true"/>` Note that if the hidden attribute is set to **true**, the users will not see this parameter in the portlet customization. If it is set to **false**, this parameter can be customized.
URL	The URL element specifies the location of the application content. The cachedOnURL attribute should be **true.** For example: `<url cachedOnURL="true">http://www.oracle.com</url>`

Adding Portlets

When portlets are enabled at startup of the Oracle WebLogic server, MetaSolv Solution reads all xreg files.

To add portlets to the list of available portlets:

1. The first time you add portlets, copy the sample file, *$MSLV_HOME*/**appserver/portal/examples/my-portlets.xreg** and use it to create another portlet registry file with a unique name in the *$MSLV_HOME*/**appserver/portal/conf**. Make sure the file has an xreg extension.
2. Change only the **name** attribute with a unique portlet name within a <**portlet-entry**></**portlet-entry**>. The portlet-entry tag itself has name, hidden, type, parent, and application attributes.
3. Under the portlet-entry, specify meta-info, classname, parameter (optional), URL, and category as described in Table 5–1, " Registry File Parameters".
4. After a custom portlet is added into the registry, restart the Oracle WebLogic server or redeploy the application (nur.ear) to reload the portlet registry.

Managing Logons within Portlets

When you authenticate a user via the portlet, the user can access an application within the same Oracle WebLogic server or other Oracle WebLogic servers in the same cluster

without logging on separately. However if the application is third party software, or if the application is not in the cluster, logon information must be defined in the portlet if the application requires authentication. There are three options. See the following for sections details.

Defining aliases

You can define the **$PRINCIPAL** and **$PASSWORD** aliases in the portlet-entry's URL as HTTP request parameters, as in this example:

```
<url cachedOnURL="true">https://xyz-site/function?
username=$PRINCIPAL&amp;password=$PASSWORD</url>
```

> **Note:** **&** is used to separate the parameters. You cannot use **&** directly since it is a special character in HTML, thus you must use the escape sequence **&**. Be sure to include the semi-colon.

In the above example, the **?** indicates there are appending parameters. The **username** and **password** are the parameter names. Ask the Web-Master for the real parameter names for the username and password. The **$PRINCIPAL** and **$PASSWORD** aliases are expanded with the real user name and password by the portlet and will be the same one with which the user logged on. Use HTTPS so that the username and password are encrypted.

The advantage of this mechanism is that logging on is automatic. The user does not need to be prompted to enter a username and password.

The limitations of this mechanism are:

- The username and password must be the same on both the MetaSolv Solution portlet and the third party application.
- If the third party application generated content contains hyperlinks, the username and password must be encoded in the links by the third party application.

Defining name and password as portlet-entry parameters

You can define the username and password as portlet-entry parameters. These parameters can be customized by users and are sent to the third party application as HTTP request parameters. The application must be coded to extract the username and password parameters from the HTTP request. After the user is authenticated, the application returns the content. Using HTTPS in the URL parameter encrypts the username and password.

The name is case-sensitive. You must know what the application expects for the parameter name and define it exactly the same in the portlet-entry. For example:

```
<parameter name="username" value="bsmith" type=""
hidden="false" cachedOnName="true" cachedOnValue="true"/>
<parameter name="password" value="mypassword" type=""
hidden="false" cachedOnName="true" cachedOnValue="true"/>
```

> **Note:** If a hidden attribute is set to **true**, then the users will not see this parameter in the portlet customization. If it is set to **false**, then this parameter can be customized.

The advantages of this mechanism are:

- Automatic logon: The user does not need to be prompted to enter username and password.
- The username and password do not need to be the same on both MetaSolv Solution portlet and the third party application.

The limitations of this mechanism are:

- The users need to manually enter the username and password in portlet customization.
- If the third party application generated content contains hyperlinks, the username and password must be encoded in the links by the third party application.

Prompting for the username and password

Instead of sending the username and password embedded in the HTTP request, you can allow the third party application to prompt for the username and password. After the user submits the logon information, a new window displays the third party application content.

The advantages of this mechanism are:

- The third party application does not need to be coded to extract the username and password parameters from the URL.
- The third party application does not need to encode the username and password in the links in the generated content.

The limitation of this mechanism:

- There is no automatic logon.

Restart the Oracle WebLogic server after modifying the portlet registry file.

Third Party Portlet Accesses MetaSolv Solution Applications

MetaSolv Solution Struts Framework extracts the username and password from the HTTP request and performs the authentication. Once the user is authenticated, a session is established, and the following requests from the same user do not need to be authenticated until the user logs off.

Following is the list of HTTP request case-sensitive parameters that can be sent to MetaSolv Solution application:

- UserID - the value of this parameter is the username.
- Password - the value if this parameter is the password.
- TimeZoneID - the time zone id of the user, if not specified the current MetaSolv Solution Portal time zone id is used.

HTTPS is recommended to encode the user name and password, for example:

```
<https://wsplwss01:7002/main/getMyTask.do?UserID=test&Password=test>
```

The session is terminated when it is expired (by default 15 minutes), or when the user logs off:

```
<http://wsplwss01:8001/main/logoff.do>
```

Customizing the Navbar

The **Navbar** tab on the GUI navigation bar contains the standard navigation for MetaSolv Solution. Use the **My Navbar** tab on the navigation bar to display a custom configuration of groups and links. After you customize the **My Navbar** tab, it becomes

the default navigation for all users. You can select the **Navbar** tab on the navigation bar to return to the original navigation.

To allow users to customize the default navigation bare, ensure the **Allow users to customize My Desktop preference** is set to **Y**.

To customize My Navbar:

1. Click **My Desktop** at the top of the navigation bar.
2. Select **Options** > **Customize Navigation Bar** from the menu.
3. To modify the **My Navbar** tab on the navigation bar, do one of the following:
   - Click the **Add a new group** link.

     Enter the group name, select each link you want to add, and click the right arrow button to add.

     To add external links to documents, Web pages, or applications, click the **Add your own links** link at the bottom of the window. After entering the link name and file location, you can click the **Test Link** button before adding it to the list.
   - Click the **Edit a group** link.
   - Click the **Change the order of groups** link.
4. Complete the necessary windows, following instructions in the online Help.

To delete all My Navbar changes:

1. Display My Navbar.
2. Open the **Options** menu.
3. Click **Customize My Navigation Bar**.
4. Click the **Reset My Navbar** button.

Producing Custom Reports

MetaSolv Solution provides preloaded GUI reports through the reporting, trouble management, and security functions. The reporting function provides a variety of reports that can be selected. Instructions for producing these reports appear in the online Help.

When MetaSolv Solution is installed, the installation process deploys a PowerBuilder file named **I_rpt.pbl**, a file that contains predefined MetaSolv Solution reports in the form of datawindows.

You can also create custom reports that you add to the Selected Report list.To create customized reports for MetaSolv Solution, you must purchase and install PowerBuilder or InfoMaker and follow directions in that documentation to create a report and copy it to the reporting pbl.

For information on using PowerBuilder or InfoMaker, investigate the Sybase Web site at:

http://go.sap.com/index.html

Creating a Customized Report

To create a customized report and add it to MetaSolv Solution:

1. Create the report using PowerBuilder or InfoMaker.

See the third-party documentation for instructions. Use PowerBuilder to copy the report datawindow to I_rpt.pbl on the machine where Tbs.exe runs.

2. Log in to MetaSolv Solution.
3. Click the **Reporting** group.
4. Click **Register Reports** and follow directions in the Help to add a new report.

Once registered, the customized report can be selected from the **Selected Report** drop-down on the Reports window. See the online Help topic about generating a custom report for instructions.

Retaining Customized Reports After an Upgrade

When a new release is installed, the installation process deploys a new copy of **I_rpt.pbl.** Pre-existing customized reports must then be manually added to the new I_**rpt.pbl** using PowerBuilder.

To retain customized reports:

1. Copy the existing copy of **I_rpt.pbl** with a different name and save it.
2. Open the copy of **I_rpt.pbl** in PowerBuilder.
3. Copy the datawindows for pre-existing custom reports to the new **I_rpt.pbl**.
4. Manually distribute the updated **I_rpt.pbl** to all client machines.

If you select a report, you may get the following error message:

```
The selected report does not have a datawindow object in I_RPT.BPL.
```

This error message indicates that the datawindow has not been added to the **I_rpt.pbl** file on your machine.

If a data window name for a custom report cannot be found in the drop-down after an upgrade:

1. Unregister the report through the application by deleting the row in the Register Reports window.
2. Register the report again in the new version of MetaSolv Solution.

Using Stored Procedure Exits

MetaSolv Solution provides user exits for the purpose of making database calls. The exits are empty shells of stored procedures that are already part of the application code. To code these exits, you should be familiar with PL/SQL. Because these are empty shells, you are responsible for any impact a user exit has on performance or data. You must optimize your company's code.

> **Note:** Oracle Global Customer Support (GCS) supports user exits as they are written within the delivered code. GCS does not support any customer-developed code that is stored within these shells nor do we support any residual affects to our delivered code or to any data stored in the database caused by customer-derived code.

With these exits, you can have MetaSolv Solution perform tasks that are specific to your business needs, such as:

- Performing custom validation and pre-PSR validation
- Defining dependencies
- Changing optional fields to required
- Defining default fields, such as a Web host user ID
- Generating free-format circuit IDs
- Validate e-mail, ID, and password entries on e-mail service items and Web host items

You cannot use these exits to perform other functions or to call other applications.

Retaining User Exits After an Upgrade

When a new release is installed, the installation process deploys new stored procedures. Pre-existing customized stored procedures (exits) must be saved before the service pack or new release is installed.

To retain existing user exits:

1. Copy the existing stored procedures with different names.
2. Install the service pack or new release.
3. Copy the customized user exits files back to the updated version of the application.

Coding User Exits

These exits are stored in files in the **procs** directory. The **procs** directory is installed at the location where the database installer was run. Contact a DBA to find the location.

The following sections provide information for each user exit, including:

- Description
- Exit name
- File name

To add code to a stored procedure exit:

1. Locate the appropriate file for the exit you want to use.
2. Open the file with Notepad or other basic text editor.
3. Add lines of SQL code to define the task you want the application to perform.
4. Save the file.
5. Run the file, so the proc with the new code can exist on the database.

Generate a Circuit ID

You can add code to this exit to set the format for a custom circuit ID. The exit is only available for OTF and OTS circuit format types. To call this procedure, the user must first set the **Customer Generated Freeformat Circuit ID** preference to **Y**. After setting the preference to **Y**, the user can access the **Connection Identification Maintenance** window for a circuit and see an option called **Generate Circuit ID.**

Table 5–2 describes the circuit ID exit.

Table 5–2 Circuit ID Exit

Identifier	Description
GUI location	Connection Identification Maintenance window
Exit name	pkg_generate_freeformat_ecckt.sp_generate_ecckt
File name	**paecckt.sql**

Generate a Custom Customer Account Number

You can add code to this exit to generate a custom customer account number when creating or selecting a customer for a new Product Service Request. If the user is in Customer Maintenance, the procedure is called by clicking the **Apply** button or moving from the **General** tab. If the user is creating the customer directly from the PSR, the procedure is called by clicking OK after entering all of the customer information or selecting an existing customer.

Table 5–3 describes the customer account number exit.

Table 5–3 Customer Account Number Exit

Identifier	Description
GUI location	Customer Account Maintenance window
Exit name	sp_PSR_cust_acct_number
File name	**custacct.sql**

Perform Custom Validation on Network Locations

You can add code to this exit to create custom validations for network locations. This procedure functions when the user clicks the **OK** button on the **Network Location** window for a new or existing network location.

Table 5–4 describes the custom validation on network locations exit.

Table 5–4 Custom Validation on Network Locations Exit

Identifier	Description
GUI location	Network Location New window
Exit name	sp_NL_custom_validation
File name	**nlcustom.sql**

PSR Custom Pre-Validation

You can add code to this exit to automatically pre-validate a PSR order. This stored procedure exit is called when validating or finishing a PSR order and is called prior to MetaSolv Solution's validation and prior to the other PSR custom validation user exit discussed in "Perform Custom Validation on a PSR Order".

Table 5–5 describes the PSR custom pre-validation exit.

Table 5–5 PSR Custom Pre-Validation Exit

Identifier	Description
GUI location	PSR order

Table 5–5 (Cont.) PSR Custom Pre-Validation Exit

Identifier	Description
Exit name	Sp_psr_custom_pre_validation
File name	**soppreval.sql**

Perform Custom Validation on a PSR Order

You can add code to this exit to perform custom validation on a PSR order. To call this procedure, the user validates or finishes a PSR order.

Table 5–6 describes the PSR custom validation exit.

Table 5–6 PSR Custom Validation Exit

Identifier	Description
GUI location	PSR Order
Exit name	Sp_psr_custom_validation
File name	**socustom.sql**

Perform Custom Validation at Task Completion

You can add code to this exit to perform a custom validation at task completion. This stored procedure is called when a user completes any Work Management task, if the **Perform Custom Validation at Task Completion** preference is set to **Y**.

Table 5–7 describes the custom validation at task completion exit.

Table 5–7 Custom Validation at Task Completion Exit

Identifier	Description
GUI location	Task Completion Maintenance window
Exit name	sp_wm_custom_validation
File name	**wmcustom.sql**

Populate the Internet Dial Up User ID

You can add code to this exit to automatically populate the Internet Dial Up user ID. This stored procedure is called when a user first accesses the **Dial Up Info** tab on a PSR, including an Internet Dial Up service item.

Table 5–8 describes the Internet Dial Up user ID exit.

Table 5–8 Internet Dial Up User ID Exit

Identifier	Description
GUI location	Dial Up Info tab
Exit name	pkg_psr_custom_keys.sp_custom_ dialup_userid
File name	**papsrcky.sql**

Populate the Internet Dial Up Password

You can add code to this exit to automatically populate the Internet Dial Up password. This stored procedure is called when a user first accesses the **Dial Up Info** tab on a PSR, including an Internet Dial Up service item.

Table 5–9 describes the Internet Dial Up password exit.

Table 5–9 Internet Dial Up Password Exit

Identifier	Description
GUI location	Dial Up Info tab
Exit name	pkg_psr_custom_keys.sp_custom_dialup_password
File name	**papsrcky.sql**

Populate the PSR Email Password

You can add code to this exit to automatically populate the PSR email password. This stored procedure is called when a user first accesses an Email item's **Email Info** tab on a PSR order.

Table 5–10 describes the PSR email password exit.

Table 5–10 PSR Email Password Exit

Identifier	Description
GUI location	Email Info tab
Exit name	pkg_psr_custom_keys.sp_custom_email_password
File name	**papsrcky.sql**

Populate the Web Hosting User ID

You can add code to this exit to automatically populate the Web Hosting user ID. This stored procedure is called when a user first accesses the **Web Host Info** tab on a PSR for a Web Hosting item.

Table 5–11 describes the Web Hosting user ID exit.

Table 5–11 Web Hosting User ID Exit

Identifier	Description
GUI location	Web Host Info tab
Exit name	pkg_psr_custom_keys.sp_custom_webhost_userid
File name	**papsrcky.sql**

Populate the Web Hosting Password

You can add code to this exit to automatically populate the Web Hosting password. This stored procedure is called when a user first accesses the **Web Host Info** tab on a PSR for a Web Hosting item.

Table 5–12 describes the web hosting password exit.

Table 5–12 Web Hosting Password Exit

Identifier	Description
GUI location	Web Host Info tab
Exit name	pkg_psr_custom_keys.sp_custom_webhost_password

Table 5–12 (Cont.) Web Hosting Password Exit

Identifier	Description
File name	**papsrcky.sql**

Populate the URL

You can add code to this exit to automatically populate the URL on the Dial Up Info tab. This stored procedure exit is called when a user first accesses the **Dial Up Info** tab on a PSR for an Internet Dial Up service item.

Table 5–13 describes the Dial Up URL exit.

Table 5–13 Dial Up URL Exit

Identifier	Description
GUI location	Dial Up Info tab
Exit name	pkg_psr_custom_keys.sp_custom_url
File name	**papsrcky.sql**

Validate an Email Address

You can add code to this exit to automatically validate the email address entered on an Email service item email Info tab. This stored procedure exit is called on a save, when the user tries to move to another service item or when the user validates.

Table 5–14 describes the email address validation exit.

Table 5–14 Email Address Validation Exit

Identifier	Description
GUI location	Email tab of email service item
Exit name	pkg_psr_custom_keys.sp_custom_email_validation
File name	**papsrsky.sql**

Validate a User ID

You can add code to this exit to automatically validate the user ID entered in the GUI locations listed below. This stored procedure exit is called on a save, when the user tries to move to another service item or when the user validates.

Table 5–15 describes the user ID validation exit.

Table 5–15 User ID Validation Exit

Identifier	Description
GUI location	▪ Dial up Info tab ▪ Web Host Info tab
Exit name	pkg_psr_custom_keys.sp_custom_userid_validation
File name	**papsrcky.sql**

Validate a Password

You can add code to this exit to automatically validate the password entered the GUI locations listed in the table below. This stored procedure exit is called on a save, when the user tries to move to another service item or when the user clicks **validate**.

Table 5–16 describes the password validation exit.

Table 5–16 Password Validation Exit

Identifier	Description
GUI location	■ Email tab of Email Service Item, ■ Dial up tab of Dial up service item ■ Web Host tab of Web Host service item
Exit name	pkg_psr_custom_keys.sp_custom_password_validation
File name	**papsrcky.sql**

Generate an MSAG Validation Audit Trail

You can add code to this exit to generate an MSAG validation audit trail. The stored procedure is called from the End User Location Maintenance window, after the MSAG validation occurs.

Table 5–17 describes the MSAG validation audit trail exit.

Table 5–17 MSAG Validation Audit Trail Exit

Identifier	Description
GUI location	End User Location Maintenance window
Exit name	sp_psr_msag_custom_validation
File name	**somsagcustom.sql**

6

Running Utilities, Local Exchange Routing Gateway, and Background Processor

Several accessory products and utilities can help you run Oracle Communications MetaSolv Solution (MSS). They are:

- Background Processor
- Local Exchange Routing Gateway
- MetaSolv Solution Utilities
  - Purge utility
  - DB Health utility
  - Next Gen Migration utility
  - Custom Attributes utility
  - Geographical Area Types utility
  - Structured Formats utility
  - Address Correction utility
  - NPA Split utility

Note: You must install the Oracle Client on the client machine before using all the utilities except the Background Processor. See *MetaSolv Solution Installation Guide* for details and instructions.

Running Background Processor

The Background Processor processes MetaSolv Solution user work in the background when the option is available and a user selects it.

Configuring and maintaining the Background Processor is a system administrator task, handled partly through a rich client application and partly through system management. Instructions for handling the administrative GUI tasks are in the Help. Conceptual information and instructions for managing the environment are in this chapter.

Understanding the Background Processor

After the Background Processor is installed, three applications are distributed between a remote server and the system administrator's workstation. Each has a unique purpose. Table 6–1 describes the background processor applications.

Table 6–1 Background Processor Applications

Application	Function
jmanager.exe	Job Manager: ■ Adds servers ■ Sets preferences ■ Monitors the Job Queue
jmaster.exe	Job Master spawns Job Workers
Jwkr.exe	Internal Job Workers process specific jobs

The Job Manager and Job Master applications are members of the MetaSolv program group. Only Job Master interacts with Jwkr.exe, so there is no icon.

The Job Manager has an interface for interacting with the Background Processor. The Job Master runs in the background, displaying only an information window when it has been started.

Job Manager

Run Job Manager to define the servers on which Job Master is permitted to run. Part of definition is setting server-specific parameters and global preferences that affect all servers for a given database. With Job Manager, you pause, restart, or clear completed jobs from the entire Job Queue or a specific server's Job Queue.

The Job Manager application is typically executed on an administrator's desktop when administrative functions must be performed.

Use the online Help to get instructions for using Job Manager.

If you clear a canceled job from the Job Queue, notify the user who sent the job. That user must either manually process the job or send it to the Background Processor again.

To prevent access to the job queue, install the Job Manager (**jmanager.exe**) and **jmaster.ini** files on a file server. Access to this executable is obtained through a shortcut. For extra security, keep the server where the EXE and INI files are installed in an inaccessible location so unauthorized users cannot pause or start the queue.

Job Master

Run Job Master on any servers you intend to use to run Background Processor jobs. These servers must have been defined in Job Manager.

To start Job Master, run the **jmaster.exe** file.

If no error messages appear, the Background Processor is up and running. You can verify it by going back to Job Manager and clicking the **Server** button.

If the Server status column is Running, you can start sending tasks to the Background Processor. Each time the queue refreshes it will spawn another Job Worker for a new job.

Use the online Help to get instructions for using Job Master.

The Job Master process:

- The Job Master on each server checks the job queue at intervals for new jobs assigned to their servers. This interval, called the refresh rate, is controlled separately for each Job Master by the settings in the Job Manager.

- When each Job Master refreshes, it also checks for any jobs assigned to other servers that were not started within a designated time, and moves all such jobs into the first available job queue.
- The Job Master then checks the Job Manager for the number of Job Workers it is allowed to run simultaneously.
- If it is running fewer than that number of Job Workers, the Job Master starts additional Job Workers up to the maximum number, assigning each new Job Worker to the oldest job in either the first available queue or that server's queue.

Job Worker

The Job Worker displays an informational window that shows the job being processed and its progress. It is an application that operates behind the scenes, started by Job Master when a job need to be processed.

The Job Worker process:

- After a Job Worker picks up a job for processing, it looks at the Job Type table to get the parameters for the job. The parameters provide more information about the job.
- The Job Worker runs the job (for example, assigns trunks) using the MetaSolv Solution and appropriate APIs.
- Once the job is complete or has failed due to errors, the Job Worker changes the job's status appropriately, logs any messages required, and terminates itself.

> **Caution:** The Job Worker gets parameters for background jobs from the Job Type table. The parameters provide more information about the job. Making changes to Job Type table data can have significant undesirable results; therefore, change the data only under the direction of Oracle Global Customer Support personnel.

Configuring the Background Processor

After installation, the Background Processor configuration may need to be adjusted. You might perform tasks such as editing jmaster.ini to increase the number of servers, or you might change Background Processor preferences after you open jmanager.exe. This section discusses basic Background Processor configuration information.

Setting Up the Background Processor

The system administrator performs several tasks to configure the Background Processor. Some are done with the **jmaster.ini** file, but most are done online in the Job Manager and Job Master applications.

To set up the Background Processor:

1. Create a server for the Background Processor in Job Manager.
2. Setup the preferences within Job Manager.
3. Set up the **jmaster.ini** file.

Locating EXE and INI Files

A typical Background Processor installation puts all of the software for the Background Processor in the same directory as the MetaSolv Solution client executable. The reason for this is that many of the routines that can execute in the

background can also be processed in the foreground. This simplifies change of control between background and foreground software.

Collections of files does not mean that the same file server that stores the Background Processor on disk must execute it. Typically, the Jmaster application is executed on a client machine that is separate from the Oracle WebLogic server.

There are performance considerations when running multiple Background Processors. For details, see "Configuring Multiple Background Processors".

Since the jmanager.exe provides remote administration of the Background Processor by the system administrator using a shortcut, locate the actual files in the common directory on the file server.

Only system administrators should have access to the jmanager executable. A shortcut to jmanager.exe can be installed on a user workstation. However, this gives the user of that workstation system administrator privileges for the Job Queue and enables them to cancel any jobs, pause the queue, and clear the queue.

Configuring Jmaster.ini

The default INI file for Job Manager is **jmaster.ini**.

The Job Workers initiated by the **jmaster.ini** file use the jmaster.ini information to get database information for the Background Processor. If new databases are added after setting up the Background Processor, access to those databases is unavailable until the appropriate INI files are updated.

The only INI file requirement for Job Workers is that the **Jworker.exe** file is installed in the same directory as the **jmaster.ini** file in which the database profile names are defined.

Database profiles

For purge jobs to run in the background, the following profile must be included in **jmaster.ini**:

- PURGEUTIL

If no database profile is defined, the following error message appears:

```
Cannot find database profile necessary to execute this job. Please define the
database profile as described in the application documentation.
```

Sample jmaster.ini file

```
[DBMS_Profiles]

[Application Servers]
J2EEProfiles='D63B(wsplwss01)'

[J2EEProfile D63B(wsplwss01)]
url=http://wsplwss01:8001
database=D63B

[PROFILE PURGEUTIL]
DBMS=O84
Database=Oracle7F
UserId=
DatabasePassword=
LogPassword=
ServerName=D63
```

```
LogId=
Lock=
Prompt=1
DbParm=DisableBind=0
DelimitIdentifier='No'
CommitOnDisconnect='No?
AutoCommit=0
```

Table 6–2 describes the parameters in the **jmaster.ini** file.

Table 6–2 jmaster.ini Parameters

Parameter	Description
Profiles='*xx*'	Insert all names of defined profiles, enclosed in single quotes and separated with commas
[Profile=*xxx*]	Insert the name of a single defined database profile
DBMS=*xxx*	Where *xxx* is a database management system ID
Database=*xxx*	Where *xxx* is the database ID
UserId=	Leave blank
DatabasePassword=	Leave blank
LogPassword=	Leave blank
ServerName=*xxx*	Where *xxx* is the Oracle database SID for the named database
LogId=	Leave blank
Lock=	Leave blank
Prompt=1	Leave blank
DbParm=DisableBind=0	Do not change
DelimitIdentifier='No'	Do not change
CommitOnDisconnect='No'	Do not change
AutoCommit=0	Do not change

Caution: Do not include a SharedIni= path statement in the **jmaster.ini** file. If you do, you will receive an error message stating *999 DBMS is not supported in your current release*.

Setting Preferences

The Job Manager application provides the ability to set Background Processor preferences for all databases accessed. The Background Processor preferences that affect operation of the Job Masters are set using the Job Manager application, as are the other settings that control operation of each server.

Global preferences

- **Job Manager window refresh rate** preference: Adjusts the rate at which the Job Queue is checked for a new job

- **Notification purge lag days:** Controls the number of days trouble notifications with a *complete* or *closed* status remain in MetaSolv Solution before the Purge Trouble Notification job purges them
- **Path of shared directory where reports are stored:** Shows location of reports. The machine running the Background Process must be mapped to this path and must have the permission to write to this folder.
- **Purge Completion Lag Days** preference: Controls how long completed jobs remain in the Job Queue
- **Purge completion lag days for reports** preference: Controls how long reports remain in the Job Queue
- **Trouble Notifications Background Server:** Identifies the Background Processor server that is responsible for sending out notifications produced by the Trouble Management

User preferences

The preferences that affect the Background Processor's interaction with each user are controlled from the MetaSolv Solution application. They are:

- **Background Processor Notifications:** Controls whether the Background Processor notifies the user when their jobs are completed
- **Background Processor Send to Printer:** Controls whether or not reports generated by jobs sent to the Background Processor are printed
- **Default Background Server:** Sets the default server to which the user's jobs are sent. The user can override this default when sending a job to the background by selecting a different server from the drop-down on the Confirm Background Server window. This preference can also be set from the Confirm Background Server window by selecting a server from the drop-down and checking the **Make the selected server my default** check box, then clicking **OK**.
- **Suppress Confirm Background Server Window:** Controls whether or not the Confirm Background Server window is displayed. This preference can also be set to "Yes" from the Confirm Background Server window by checking the **Don't prompt me about this again** check box, then clicking **OK**.

Meeting Performance Requirements

Performance depends on the amount of memory available on the machine where the Job Master resides and on the number of jobs sent to the Job Master from other areas of the MetaSolv Solution. Several Background Processor preferences can be adjusted to modify performance to a further degree.

Configuring Multiple Background Processors

If there are multiple databases or a large volume of background activity, consider configuring multiple Background Processor servers.

Managing Performance and Tuning

If a large number of jobs are frequently sent to the Background Processor, or the turnaround time on a number of jobs is slow, there are several methods you can try to improve performance:

- Increase the number of Job Workers

If a server has a lot of available memory, increase the maximum number of workers to enable more concurrent work. If the refresh rate is also increased, the Job Master checks whether it can start new Job Workers more quickly.

- Add memory

  If a server is memory-bound, a high worker maximum can actually hurt performance because a large amount of swapping to disk-based virtual memory occurs. In this case, cut the maximum number of workers to improve performance.

- Add servers to run additional Job Masters
- Adjust the Background Processor preferences
- Change Background Processor server settings

There are no recommended settings for Background Processor servers. The most efficient settings will depend on your environment. Make small changes to the default to discover the best settings for your site. Changes to the settings take effect immediately after the next refresh, so it is not necessary to restart the Background Processor.

Maintaining the Background Processor

The Background Processor system administrator maintains the functions of the Background Processor, ensuring that it works effectively.

The administrator's tasks fall into one of two categories:

- Managing the Background Processor environment
- Managing the Background Processor job queue

Environment Responsibilities

Background Processor administrator responsibilities include:

- Configuring the Background Processor
- Setting preferences
- Managing the log files
- Managing performance and tuning
- Resolving Background Processor problems

Online Job Queue Responsibilities

Background Processor administrator online responsibilities include:

- Starting and pausing the queue
- Clearing jobs from the queue
- Managing specific jobs
- Scheduling recurring jobs
- Viewing the Job Log

Instructions for day-to-day management of the Job Queue, including windows and procedures, appears in the online Help. From the window help, you can jump to specific information needed to manage the Job Queue.

Responsibilities for End Users

There are no required user-support tasks that the Background Processor administrator must perform. There are no user IDs or passwords to maintain. Users do not log on to the Background Processor itself. Instead they log on to MetaSolv Solution. MetaSolv Solution sends users' jobs to the Background Processor.

If multiple users report that their jobs are consistently failing or if they report that their jobs run too slowly, restart Job Master, tune the system, or research the Job Log.

User job queue

MetaSolv Solution users can view the jobs they have submitted to the Job Queue. From this view, users can monitor their own jobs and can remove them from the queue. Instructions for these tasks are provided in the online Help.

Users can view the job queue in the MetaSolv Solution. This capability creates a virtual individual Job Queue for each user.

The individual Job Queue has most of the same functionality as the Job Manager but is restricted to only that user's jobs.

Reports

If specified by MetaSolv Solution preferences, Service Provisioning jobs running in the background can produce reports. The Background Processor saves the report at the end of the job and produces it with the job output.

These reports can be accessed from all Background Processor windows including the individual Job Queue, the Job Manager, and the Job Master. Typically, system administrators do not work with these reports unless a user is having problems.

The report in the Background Processor is actually a flat text file that pre-populates the format and data at the time.

User notification

To enable notification, a user must set the Background Processor Notification preference to "Yes." Each user must do this to receive notification about jobs. There is no system or global preference to perform this function.

Log Files

The Background Processor automatically produces a number of log files that track basic processing during normal operation. These files are named:

- Error.log
- Worker.log: captures job worker errors
- Manager.log: captures job manager errors
- Master.log: captures job master errors
- Jobname.log: captures errors for a particular job

Read these log files to help track problems occurring in the Background Processor or within the system. They can contain additional helpful information. Use the information available in the Job Parameters window to get job identification information to use while looking in the log files.

If these files are deleted, the Background Processor creates new ones. It is important to note that the Background Processor appends information to these files during

processing and does not overwrite it. Therefore, consider removing old text from the files or deleting the files periodically.

These log files can also be used as an aid to determining what is happening within the Background Processor during operation. Oracle Global Customer Support personnel might ask for copies of the files if problems arise that require their assistance.

Using the Error.log

The Error.log collects error messages for jobs sent to the Background Processor, in the order the messages are generated. This log can give you useful information about specific jobs. For example, if a job has created an error, you can see the messages passed between the client, master, worker, broker, and scheduler. This can help you discover the cause of the error so you or the user can correct it.

Who should fix the problem

You can determine who should fix the problem that caused a Background Processor job to fail by looking at the message in the Job Log window. The Message Originator field shows the originator of the job log message.

If the originator of the message is the Client or Broker (BKR), the MetaSolv Solution user must fix the error and then send the job to the Background Processor again, and you should clear the job that created the error from the job queue.

If the originator of the message Master, Scheduler or Worker, you must fix the error and then restart the job.

Note: Each originator may send multiple events to the job log.

Restarting Background Processor Jobs

If your site regularly schedules a shutdown of all systems, restart the Job Master after each shutdown. Review the server configuration and consider adding a jmaster.exe shortcut to the server's **Start Menu - Programs - StartUp** directory.

Scheduling Automatic Job Submissions

Scheduled jobs are jobs that are automatically submitted to the Background Processor at a scheduled time. Use the Job Manager to set up specific jobs to occur periodically: daily, weekly, or monthly. When a recurring schedule for a specific job type is set up, the Background Processor automatically initiates the job without any prompting from a user.

Recurring schedules are available for the following types of jobs:

- Purging orders

  This job automatically deletes orders with circuits placed *In Service* from MetaSolv Solution prior to the date the job runs.

- Purging tasks

  This job automatically deletes all completed tasks from orders when the last task on the order was completed prior to the date the job runs.

- Deleting expired reservations

  The Reservations function allows user to reserve existing connection positions, equipment port addresses, and cable pairs for future use. The Background

Processor deletes all expired reservations out of the database each time the job runs.

- Purging trouble notifications

  This job automatically purges trouble notifications with a *Complete* or *Closed* status from the MetaSolv Solution database once the *Complete* or *Closed* trouble notifications reach the age specified in the Notification Purge Lag Days preference.

- Deleting jobs from the Job Queue

  This job automatically purges all jobs except those in “Executing" or “Ready to Execute" status. The Background Processor purges completed jobs that do not have reports from the Job Queue once the completed jobs have reached the age specified in the Purge Completion Lag Days preference.

  The Background Processor automatically purges completed jobs that do have reports from the Job Queue once the completed jobs have reached the age specified in the Purge Completion Lag Days For Reports preference.

See the online Help for more information about schedules.

Scheduling purge jobs without the Purge Utility

You can set up recurring schedules for the following job types:

- Deleting Expired Reservations
- Purging Orders
- Purging Tasks
- Purging Trouble Notifications

See the online Help for more information.

Importing Local Exchange Routing Information

The Location and Routing Gateway is the application through which information from Telcordia Technologies’ Local Exchange Routing Guide (LERG) is imported to MetaSolv Solution. A LERG CD from Telcordia provides industry-standard, switch-related telecommunications information (network locations, company codes, telephone numbers, and NPA NXX information).

The Location and Routing Gateway converts the information to a form that MetaSolv Solution can read, and then transfers the information to MetaSolv Solution. Using the Location and Routing Gateway reduces entry time and ensures that MetaSolv Solution contains the most recent Telcordia Technologies information.

Start the Location and Routing Gateway application by running the **lerg.exe** file.

The Location and Routing Gateway is used for two main functions:

- Initial load: initially load data from the LERG.
- Ongoing maintenance: load updated data from the LERG.

The following information is contained in the LERG but is *not* imported to MetaSolv Solution:

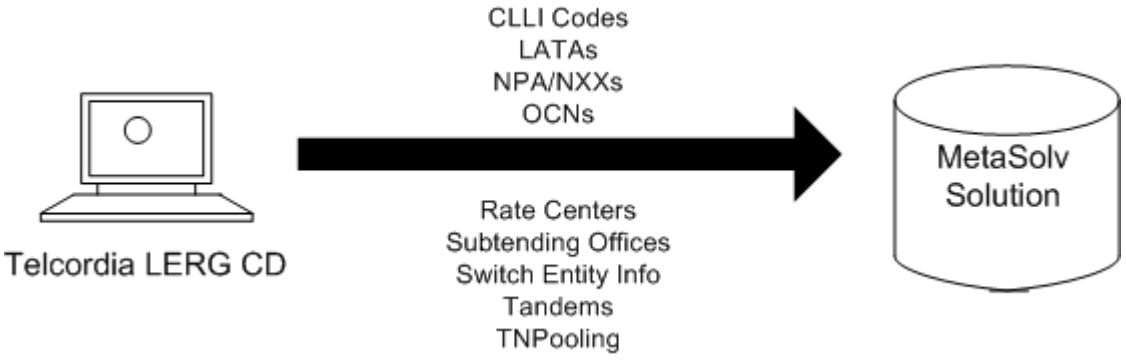
- Access customers
- Country codes
- ICSCs

- NC/NCI codes
- NPA NXX records with a COC type of 700, AIN, BLG, BRD, CTV, ENP, FGB, HVL, INP, LTC, N11, ONA, PRO, RSV, RTG, or UFA.
- Operator Service Access Tandem (ATC) codes
- Operator service codes
- Service types
- Switching entity functions

The Location and Routing Gateway collects and imports only the LERG information that you specify. For example, you can select specific states, and query by NPA/NXX or CLLI code for the records you want to load.

Figure 6–1 shows the LERG information that the Location and Routing Gateway can import to MetaSolv Solution.

Figure 6–1 The Location & Routing Gateway Flow



Starting the Location and Routing Gateway

To start the application:

1. Run the **lerg.exe** file.

   The logon dialog box appears.

2. Enter the same user ID and password that you use for logging in to MetaSolv Solution.
3. Select the database to which the Utilities should connect.

Setting Up the Gateway

Telcordia Technologies periodically changes the format of information fields on the CD and enumerates these changes in the **LERGSPEC.DOC** file (which is on the LERG CD).

To set up the gateway, follow these steps:

1. Make sure the Gateway destination fields match the LERG source fields.

   Within the Location and Routing Gateway application this consists of:

   - Identifying the LERG source directory
   - Reconciling MetaSolv Solution information fields with those on the LERG

Caution: Neglecting to make this reconciliation can cause array boundary, invalid data, and null data errors when attempting to load State information to the Location and Routing Gateway staging tables.

2. Specify which state information you want to load.
3. Load the information from the CD to the Location and Routing Gateway staging tables.

   See the online Help for steps in reconciling the fields

Note: Cities and states are loaded into MetaSolv Solution if a switch record is contained in the LERG. If a new switch is installed after the data is loaded from the LERG, city and state information will not be populated in the database for the new switch. Instead, you can manually enter city and state data through Location and Geography Setup.

Application Setup Fields

These lists contains the names of the LERG fields that are imported into MetaSolv Solution from each LERG file supplied by Telcordia.

Table 6–3 shows the names of the LERG1 fields in the MSS window.

Table 6–3 LERG1

Field	MetaSolv Solution Window Name
Operating Company Name	w_company_list
Operating Company Number	w_company_list

Table 6–4 shows the names of the LERG5 fields in the MSS window.

Table 6–4 LERG5

Field	MetaSolv Solution Window Name
LATA/Boundary	w_lata_list
LATA/Boundary Name	w_lata_list

Table 6–5 shows the names of the LERG6 fields in the MSS window.

Table 6–5 LERG6

Field	MetaSolv Solution Window Name
Record Status	N/A
Record Effective Date	w_npa_nxx_maint
NPA	w_npa_nxx_maint
CO Code (NXX)	w_npa_nxx_maint
CO Code Type	w_switch_detail_maint
Special Service Code (SCC)	w_switch_detail_maint
Dialable Indicator	w_switch_detail_maint

Table 6–5 (Cont.) LERG6

Field	MetaSolv Solution Window Name
Terminating Digits End Office	w_switch_detail_maint
Terminating Digits Access Tandem	w_switch_detail_maint
Portable Indicator	w_npa_nxx_maint
Operating Company Number	w_company_list w_co_exchg_area_maint
Locality Name	w_co_exchg_area_maint
Locality State	w_city_list w_network_location_maint
Line Range - From #	w_npa_nxx_location_maint
Line Range - To #	w_npa_nxx_location_maint
Locality State - Switch (CLLI)	w_co_exchg_area_maint w_network_location_maint
Thousands Block Pooling Indicator	w_npa_nxx_location_maint

Table 6–6 shows the names of the LERG7 fields in the MSS window.

Table 6–6 LERG7

Field	MetaSolv Solution Window Name
Switch	w_co_exchg_area_maint w_network_location_maint
Vertical Coordinates (VC)	w_co_exchg_area_maint w_network_location_maint
Horizontal Coordinates (HC)	w_co_exchg_area_maint w_network_location_maint
International DDD	w_switch_detail_maint
Switch Address - Street	w_network_location_maint
Switch Address - City	w_network_location_maint w_city_list
Switch Address - State	w_network_location_maint
Switch Address - Zip	w_network_location_maint
Assumed Minutes FGA	w_switch_detail_maint
Assumed Minutes FGB	w_switch_detail_maint
LATA	w_lata_list
LATA Name	w_lata_list

Table 6–7 shows the names of the LERG8 fields in the MSS window.

Table 6–7 LERG8

Field	MetaSolv Solution Window Name
LATA/Boundary	w_netarea_definition_maint

Table 6–7 (Cont.) LERG8

Field	MetaSolv Solution Window Name
Status	w_netarea_definition_maint
RC State	w_netarea_definition_maint
Rate Center (RC) Name Abbreviation	w_netarea_definition_maint
RC Type	w_netarea_definition_maint
RC Full Name	w_netarea_definition_maint
Major Vertical Coordinate	w_netarea_definition_maint
Major Horizontal Coordinate	w_netarea_definition_maint
Minor Vertical Coordinate	w_netarea_definition_maint
Minor Horizontal Coordinate	w_netarea_definition_maint

Table 6–8 shows the names of the LERG9 fields in the MSS window.

Table 6–8 LERG9

Field	MetaSolv Solution Window Name
Tandem Data - Switch (Tandem Network Location)	w_tandem_list_query
Tandem Equipment Type (Services)	w_tandem_list_query
Homing Switch Data - Switch (EO Network Location)	w_tandem_list_query
Function(s) Originating	w_tandem_list_query
Function(s) Terminating	w_tandem_list_query

Importing the Data

For specific instructions about importing data, refer to the online Help. The following steps provide a high-level overview of the process.

Restarting a Canceled Import

If you cancel an import and restart it, the importing process resumes importing data where the import stopped. The information you import to MetaSolv Solution is arranged in groups of 50 rows. If an import stopped in the middle of a 50-row group, the process resumes at the beginning of that interrupted 50-row group.

> **Note:** MetaSolv Solution imports rate center information before other information. Please wait until a message appears in the Import in Progress dialog box before restarting the rate center import process.

To restart an import:

1. Open the **Location and Routing Gateway Restart** window.
2. Click the **Restart** button on the toolbar.

Exceptions and messages that occur during a load of LERG information populate the Location and Routing Gateway Audit Log.

Creating a Location and Routing Gateway Report

To produce a Location and Routing Gateway report:

1. Open the **Location and Routing Gateway Reporting** window.
2. Select a range of dates from the **Range** drop-down.

> **Note:** You can also select **Custom** from the **Range** drop-down, and enter dates in the **From Date** and **Through Date** fields.

3. Select the type of report to generate from the **Report** drop-down.
4. Click the **Retrieve** button on the toolbar.

To sort a Location and Routing Gateway report:

1. Generate a report.
2. Take one of these actions to sort the report:
   a. Click a report column heading to sort it alphabetically in either ascending or descending order. (Each time you click the column heading it toggles between ascending and descending order.)
   b. Click the **Sort** button on the toolbar to open the Specify Sort Columns window.
      – Drag an item from the Source Data column to the Columns column to sort the report by that item.
      – Drag a subsequent item to the Columns column to sort the report by the second item within the first.
3. Click **OK**.

To print a Location and Routing Gateway report:

1. Generate a report.
2. Click the **Print Preview** button on the toolbar.
3. Click the **Print Preview** button again to exit the Print Preview mode.
4. Click the **Page Setup** button on the toolbar.
5. Specify the size and source of the paper that reports are to be printed on, their orientation, and the size of their margins.
6. Click **OK**.
7. Click the **Print** button on the toolbar.

Using the MetaSolv Solution Utilities

The MetaSolv Solution utilities are a group of applications that provide additional administrative functionality for MetaSolv Solution.

Starting the Application

The MetaSolv Solution utilities operate from **tbs_util.exe**, managed by a configuration file named **tbs_util.ini**. These files are rarely installed on more than a few client machines. Normally, these utilities are run by DBAs or super users.

To start the utilities application:

1. Run the **tbs_util.exe** file.

   The logon dialog box appears.

2. Enter the same user ID and password that you use for MetaSolv Solution.

3. Select the database to which the Utilities should connect.

A number of utilities are available using this interface. Three have buttons on the primary toolbar and some are only accessed from the **Option** menu. Press F1 from any window for online Help instructions. MetaSolv Solution Utilities has a separate security system, selected from the File Menu, that works exactly like MetaSolv Solution security.

Primary Toolbar Utilities

Three utilities can be accessed from primary toolbar buttons.

- **Purge:** Purges tasks, employees, service requests, extra DLR issues, and expired reservations from the MetaSolv Solution database. Press F1 during any Purge display to for online Help instructions. See "Using the Purge Utility" for more information.
- **DB Health:** Reduces errors in the database, making response time quicker and eliminating problems in processing information. Creates a synonym for each database object, fixes tables, checks objects, adds rows to tables, fixes errors, and checks rules. See "Using the DB Health Utility" for more information.
- **Next Gen Migration:** Creates and converts network elements and migrates pre-5.1 connections, Virtual Layout Records, and reconcile design lines to Release 6.0+. See "Using the Migration Utility" for more information.

Options Menu Utilities

The **Options** menu provides access to the three primary toolbar utilities and to additional utilities which allow you to set specific kinds of setup characteristics for MetaSolv Solution.

- **Custom Attributes:** Custom attributes (CAs) are characteristics you can define generically and associate with network templates, element types, connection types, connection specifications, network systems, and allocations of connections or elements within a system. See "Using the Custom Attributes Utility" for more information.
- **Geographical Area Types:** Geographic areas identify the appropriate format for addresses, telephone numbers and switch prefixes. See "Using the Geographical Area Types Utility" for more information.
- **Structured Formats:** Structured Formats provide a way to enter, define, and store structure and formatting information for addresses, telephone numbers, switch prefixes, and LSOG columns. See "Using the Structured Formats Utility" for information about Structured Formats options.
- **Address Correction Utility:** The Address Correction utility enables you to resolve inconsistencies related to end user locations in the following scenarios:
  - Multiple end user locations having the same address
  - Multiple active addresses of the same address structured format exist at the same end user location
  - Multiple end user locations having the same address in Master Street Address Guide (MSAG) and Ordering and Billing Forum (OBF) structured formats

See "Using the Address Correction Utility" for more information.

Using the Purge Utility

Use the Purge utility to remove old data from the MetaSolv Solution database. This helps maintain or reduce the size of the database.

The following records can be manually purged using the Purge utility, as-needed:

- Tasks
- Employees
- Orders
- Circuits
- Worksheet
- DLR issues
- Reservations
- Server Logs

If you manually purge the records, they can be purged in the foreground or in the background (with the Background Processor). To purge in the background, check the **Run in background mode** check box. The Purge job will go to the Background Processor job queue.

You can also use the Background Processor to automatically purge service requests and/or tasks, as long as a the Purge database profile is added to the **jmaster.ini** file. When Purge jobs are run on the Background Processor, the **jmaster.ini** file must have a PURGEUTIL database profile.

Caution: Be cautious when using the Purge utility. Information you purge is permanently deleted from the MetaSolv Solution database.

Setting the Purge Preferences

There is only one Purge Tasks preference in MetaSolv Solution. However, two DLR issues preference also affect the Purge utility. The preferences are:

- Tasks Enabled for Archive for Task Purge
- Number of Overridden DLR Issues to Keep
- Number of Previous DLR Issues to Keep

Setting the tasks preference

The **Tasks Enabled for Archive** identifies the task types you want to save to the Task Archive file when you purge tasks.

When one of the identified task types is purged, it is written to the TASK_ARCHIVE table on the database. Access to this table is only possible through a customized report or a query using SQL Plus. This table is not accessible through MetaSolv Solutions Utilities application.

Setting the DLR issues preferences

The purge DLR issues function works in conjunction with the following preferences that you set in the Preferences function of MetaSolv's software (not MetaSolv Solutions Utility application preferences). The preferences are:

- Number of Overridden DLR Issues to Keep

Number of overridden design issues to keep for a specific connection/service request combination as a history of a design. When you choose to create a new issue for a service request/connection that already has a pending issue, the pending issue's status changes to "Overridden." The new issue becomes pending. Issues exceeding the number specified are deleted.

- Number of Previous DLR Issues to Keep

Number of previous design issues to keep for a connection as a history of the design. Issues exceeding the number specified are deleted.

Set these preferences to the number of issues you want to keep for specific connections. Older issues are removed

Purging Tasks from Service Requests

This function deletes all tasks from service requests whose last task was completed on or before the calculated purge date. The task purge utility does not delete any tasks from a service request unless all the tasks for the service request were completed on or before the calculated purge date.

To delete tasks from service requests:

1. Click the **Purge** button on the toolbar.
2. Click the **Tasks** button on the toolbar.
3. Enter values in the Year, Month, and Day fields.

   The purge date will be calculated by subtracting these intervals from today's date.
4. Select the **Run in Background Mode** check box if you want to do so.
5. Click the **Start** button on the toolbar.

   Refer to the online Help for field definitions.

> **Caution:** Be cautious when deleting tasks from a service request. Their removal from the MetaSolv Solution database is permanent.

When tasks are purged, MetaSolv Solution clears information from the following tables:

- ASAP.ACTIVATION_COMMAND_PLAN
- ASAP.DLR_LINE_FLOW_THROUGH_OBJECT
- ASAP.EMP_APPT
- ASAP.FLOW_THROUGH_COMMAND
- ASAP.FLOW_THROUGH_ITEM
- ASAP.FLOW_THROUGH_OBJECT
- ASAP.GATEWAY_EVENT_ERROR
- ASAP.INTEGRATION_EVENT_STATUS
- ASAP.PBI_BILL_DISCREPANCY
- ASAP.PBI_BILL_ERROR
- ASAP.PBI_BILL_GROUP

- ASAP.PBI_BILL_TRANASCTION
- ASAP.SERV_REQ_GATEWAY_EVENT
- ASAP.SL_SERVER_LOG
- ASAP.SRC_TASK_COMPLETION
- ASAP.SRSI_GATEWAY_EVENT
- ASAP.TASK
- ASAP.TASKCHK
- ASAP.TASK_JEOP_WHYMISS_ESC
- ASAP.TASK_JEOPARDY_WHYMISS
- ASAP.TASK_SEQUENCE
- ASAP.TASK_SRSI
- ASAP.TASK_STAGE

Purging Employees

This function deletes an employee and their corresponding work queues from the MetaSolv Solution Work Management subsystem. When an employee is deleted, all tasks are transferred from the deleted employee's work queue to another employee's work queue.

Deleting an employee using the Purge utility is the same as deleting an employee using the MetaSolv Solution Work Management subsystem. However, deleting an employee through Work Management requires tasks to be manually transferred to a different work queue.

To purge an employee:

1. Click the **Purge** button on the toolbar.
2. Click the **Employees** button on the toolbar.
3. Select the name of the employee from the **Employee Name** drop-down.
4. Select a work queue from the **Transfer to Work Queue** drop-down. This is the work queue to which the deleted employee's uncompleted tasks will be transferred.
5. Select the **Run in Background Mode** check box if you want to do so.
6. Click the **Start** button on the toolbar.

When employees are purged, MetaSolv Solution clears information from the following tables:

- ASAP.EMP_WORK_QUEUE_AVAILABILITY
- ASAP.EMPLOYEE
- ASAP.WORK_QUEUE
- ASAP.EMP_APPT

Purging Orders

This function deletes orders from MetaSolv Solution.

Purging service requests with active circuits

This function deletes service requests that:

- have active circuits placed "In Service" on or before the calculated purge date.
- are due date complete.

> **Note:** You cannot purge PSRs with E911 information from the MetaSolv Solution database.

To purge service requests with all active circuits:

1. Click the **Purge** button on the toolbar.
2. Click the **Orders** button on the toolbar.
3. Enter values in the Year, Month, and Day fields.

   The purge date will be calculated by subtracting these intervals from today's date.
4. Select the **Run in Background Mode** check box if you want to do so.
5. Click the **Start** button.

When orders are purged using the Orders icon, MetaSolv Solution clears information from the following tables:

- ASAP.ACCESS_PROVIDER_ROLE
- ASAP.ACCESS_SERVICE_REQUEST
- ASAP.ACI_SRSI
- ASAP.ADDITIONAL_CIRCUIT_INFORMATION
- ASAP.ANNOUNCEMENT_WINK_VALIDATION
- ASAP.ASR_BILLING_ADDRESS
- ASAP.ASR_CONTACT_INFORMATION
- ASAP.ASR_CONTACT_TYPE
- ASAP.ASR_ERROR
- ASAP.ASR_RING
- ASAP.ASR_SRSI
- ASAP.ASR_USER_DATA
- ASAP.ASSIGNED_TEL_NUM
- ASAP.ATN_CHAM
- ASAP.ATN_LIDB
- ASAP.ATN_PIC
- ASAP.ATN_PIC_INFO
- ASAP.ATN_RCF_NP_INFO
- ASAP.BAN_USAGE
- ASAP.CAPTION_DETAIL
- ASAP.CAPTION_SET_DETAIL
- ASAP.CIRCUIT_POSITION

- ASAP.CIRCUIT_POSTION_PENDING
- ASAP.CLARIFICATION_REQUEST
- ASAP.CNAM_DISCREPANCY
- ASAP.CNAM_EXTRACT
- ASAP.COMP_NOTICE
- ASAP.CONF_NOTICE
- ASAP.CONTACT
- ASAP.CPE
- ASAP.CUSTOMER_NPA_NXX
- ASAP.DESIGN
- ASAP.DESIGN_DLR_ASSOC
- ASAP.DESIGN_DLR_ISI
- ASAP.DESIGN_NOTE
- ASAP.DESIGN_ORD_SUMM
- ASAP.DIRECTORY_LISTING
- ASAP.DIR_DEL_SRSI
- ASAP.DIR_DEL_DIR_TYPE
- ASAP.DIR_DL_DEL_SRSI
- ASAP.DIR_DSR_SRSI
- ASAP.DIR_HEADER_SRSI
- ASAP.DIR_HEADER_SRSI_REL
- ASAP.DIR_LIST_SI
- ASAP.DIR_LIST_SRSI
- ASAP.DIR_LIST_SRSI_REL
- ASAP.DIR_LIST_TXT_SRSI
- ASAP.DIR_SERV_CONF
- ASAP.DLR_SERV_REQ
- ASAP.DL_DESIGN_LINE_SEQUENCE
- ASAP.DL_NOTE
- ASAP.DSCN_DETAIL
- ASAP.DSR_COMP
- ASAP.DSR_ERROR
- ASAP.DSR_ERROR_CODE
- ASAP.DSR_ERROR_DETAIL
- ASAP.E911_DISCREPANCY
- ASAP.E911_ERROR
- ASAP.E911_EXTRACT

- ASAP.EMP_APPT
- ASAP.END_OFFICE_DETAIL
- ASAP.END_OFFICE_TRAFFIC
- ASAP.END_USER_LOCATION
- ASAP.END_USER_LOCATION_USAGE
- ASAP.EUL_ADDITIONAL_DETAIL
- ASAP.EUL_LSO
- ASAP.FEATURE_GROUP_A
- ASAP.FIRED_EVENTS
- ASAP.FIRM_ORDER_CONFIRMATION
- ASAP.FOC_SWITCHED_CKT_DETAIL
- ASAP.FOC_USER_DATA
- ASAP.FOC_VIRTUAL_CONN
- ASAP.GATEWAY_EVENT_ERROR
- ASAP.GATEWAY_EVENT_NOTES
- ASAP.HUNT_GRP_DETAIL
- ASAP.INTERCEPT_INFO
- ASAP.JACK_INFO
- ASAP.LINE_DETAIL
- ASAP.LOC_SERV_CONF
- ASAP.LOC_SERV_REQ
- ASAP.LOCAL_ORDER_CONTACTS
- ASAP.LOCAL_SERVICE_ORDER_DETAIL
- ASAP.LSC_DETAIL
- ASAP.LSR_CONTACT_INFO
- ASAP.LSR_EUL_ADDL_DATA
- ASAP.LSR_EUL_DETAIL
- ASAP.LSR_QTY_REQ
- ASAP.LSR_SERV_DETAIL
- ASAP.LSR_SRSI
- ASAP.MS_ATTACHMENT_LINK
- ASAP.MSL_CONTACT_TYPE
- ASAP.MSL_SRSI
- ASAP.MULTIPOINT_SERVICE_LEG
- ASAP.N950_ACCESS_NUMBER
- ASAP.NAI_CKT_DETAIL
- ASAP.NAI_CLLI_LOCATION

- ASAP.NAI_INTERMEDIATE_CFA
- ASAP.NETWORK_ASSIGNMENT_INFO
- ASAP.NODE_PORT
- ASAP.NOTES
- ASAP.NOTIF_BKGRD_PROC
- ASAP.NOTIFICATION
- ASAP.NOTIFICATION_PARTY
- ASAP.PAIR
- ASAP.PAIR_PENDING
- ASAP.PARTY_PREPAY
- ASAP.PBI_BILL_DISCREPANCY
- ASAP.PBI_BILL_GROUP
- ASAP.PBI_BILL_ERROR
- ASAP.PBI_BILL_TRANSACTION
- ASAP.POINT_CODE_INFORMATION
- ASAP.PORT_ADDRESS
- ASAP.PORT_ADDRESS_PENDING
- ASAP.PURGE_ERROR
- ASAP.PURGE_ERROR_MESSAGE
- ASAP.PP_PARTY
- ASAP.PSR_USER_DATA
- ASAP.RCONF_ECHO
- ASAP.RCONF_VC_ECHO
- ASAP.REMARK
- ASAP.REQUEST_FORM
- ASAP.REQ_HEADER_TEXT
- ASAP.RING_SRSI
- ASAP.SAALD_LOCATION_TYPE
- ASAP.SERV_ITEM
- ASAP.SERV_ITEM_REL
- ASAP.SERV_ORD
- ASAP.SERV_REQ
- ASAP.SERV_REQ_CONTACT
- ASAP.SERV_REQ_GATEWAY_EVENT
- ASAP.SERV_REQ_SI
- ASAP.SERV_REQ_SI_PRICE
- ASAP.SERV_REQ_SI_USAGE_PRICE

- ASAP.SERV_REQ_SI_VALUE
- ASAP.SERVICE_ITEM_TRBL_TASK
- ASAP.SERVICE_REQUEST_CIRCUIT
- ASAP.SI_XREF
- ASAP.SPEC_ACCESS_AND_LINE_DETAIL
- ASAP.SPLIT_INTERCEPT
- ASAP.SRSI_COMMON_BLOCK
- ASAP.SRSI_LINE
- ASAP.SRSI_PARTY_ROLE
- ASAP.SRSI_SERV_REQ
- ASAP.SRSI_SR_LOC
- ASAP.SRSI_TN_RANGE
- ASAP.SRSI_TREATMENT
- ASAP.SR_EUL
- ASAP.SR_LOC
- ASAP.SR_PARTY_ROLE
- ASAP.SR_PARTY_ROLE_ACCESS
- ASAP.SR_RELATIONSHIP
- ASAP.SR_RELATIONSHIP_TASK
- ASAP.SR_RESP_ORG
- ASAP.SR_SALES_MOD
- ASAP.SR_SERV_REQ_DISC
- ASAP.SR_SERVICE_ITEM
- ASAP.SR_SI_LOCATION
- ASAP.SR_SI_ROLE
- ASAP.SR_STATE
- ASAP.SR_STATUS
- ASAP.SRC_TASK_COMPLETION
- ASAP.SVCREQ_PROVPLAN
- ASAP.TASKCHK
- ASAP.TASK_ARCHIVE
- ASAP.TASK_JEOPARDY_WHYMISS
- ASAP.TASK_RELATIONSHIP
- ASAP.TASK_SCHEDULE
- ASAP.TASK_SEQUENCE
- ASAP.TASK_STAGE
- ASAP.TEL_NUM_INV_LISTING_INFO

- ASAP.TERMINAL_PAIR
- ASAP.TESTING_SERVICE_REQUEST
- ASAP.TESTING_SERVICE_REQUEST_LIMITS
- ASAP.TQ_ADDITIONAL_CIC_INFO
- ASAP.TQ_COMMON_INFORMATION
- ASAP.TQ_FGD_INFORMATION
- ASAP.TQ_SAC_ACTIVITY_INFORMATION
- ASAP.TQ_SAC_NXX
- ASAP.TRANS_CALL_INFO
- ASAP.TRANSLATION_QUESTIONAIRE
- ASAP.TRBL_TASK
- ASAP.TRBL_TASK_SEQUENCE
- ASAP.TRBL_TASK_SR_PR
- ASAP.TRBL_TICKET_ESC_LEVEL
- ASAP.TROUBLE_NOTE
- ASAP.TROUBLE_TICKET
- ASAP.TRUNK_SIDE_DETAIL
- ASAP.USO_CIRCUIT_DETAIL
- ASAP.USO_CIRCUIT_LOC
- ASAP.USO_ORDER_DETAIL
- ASAP.USO_USOCS
- ASAP.VIRTUAL_CONNECTION
- ASAP.VLR
- ASAP.VLR_BANDWIDTH_CKT
- ASAP.WATS_ACCESS
- ASAP.WFP_TASK_PARTICIPATION
- ASAP.WORK_QUEUE_TASK
- ASAP.WQT_WFP_PARTICIPATION

Purging service requests with disconnected circuits

This function deletes service requests (along with circuit information) that have disconnected circuits that were disconnected on or before the calculated purge date.

To purge service requests with disconnected circuits:

1. Click the **Purge** button on the toolbar.
2. Click the **Circuits** button on the toolbar.
3. Enter values in the Year, Month, and Day fields.

   The purge date will be calculated by subtracting these intervals from today's date.
4. Select the **Run in Background Mode** check box if you want to do so.

5. Click the **Start** button.

When orders are purged using the Circuits icon, MetaSolv Solution clears information from the following tables:

- ASAP.ACCESS_SERVICE_REQUEST
- ASAP.ADDITIONAL_CIRCUIT_INFORMATION
- ASAP.ASR_RING
- ASAP.BANDWIDTH_ALLOCATION
- ASAP.BANDWIDTH_CKT
- ASAP.BANDWIDTH_CKT_BIT_RATE
- ASAP.CABLE_PAIR_SET
- ASAP.CIRCUIT
- ASAP.CIRCUIT_LAYOUT_REPORT
- ASAP.CIRCUIT_POSITION
- ASAP.CIRCUIT_POSITION_CONDITION
- ASAP.CIRCUIT_POSITION_PENDING
- ASAP.CIRCUIT_USER_DATA
- ASAP.CIRCUIT_XREF
- ASAP.DESIGN
- ASAP.DESIGN_DLR_ASSOC DDA
- ASAP.DESIGN_DLR_ISI
- ASAP.DESIGN_GLR_ASSGNMT_SEGMENT
- ASAP.DESIGN_LAYOUT_REPORT
- ASAP.DESIGN_NSC_REL_CA_VALUE
- ASAP.DESIGN_ORD_SUMM
- ASAP.DESIGN_SI_CA_MULTI_VALUE
- ASAP.DESIGN_SI_CA_VALUE
- ASAP.DL_DESIGN_LINE_SEQUENCE
- ASAP.DL_NOTE
- ASAP.DLR_CIRCUIT_DESIGN_LINE
- ASAP.DLR_NOTES
- ASAP.DLR_NSC_REL_CA_VALUE
- ASAP.DLR_SI_CA_MULTI_VALUE
- ASAP.DLR_SI_CA_VALUE
- ASAP.DS_DVRSTY_SET_CIRCUIT
- ASAP.DSGN_GLR_ASSGN_SEG_NI_NBR_INV
- ASAP.DV_ASSIGN_LOG
- ASAP.DV_ASSIGN_LOG_DVRSTY_SET

- ASAP.E911_DISCREPANCY
- ASAP.END_USER_LOCATION
- ASAP.FACILITY_CAPABILITY
- ASAP.FIRM_ORDER_CONFIRMATION
- ASAP.FOC_PROVIDER_CIRCUIT
- ASAP.FOC_SWITCHED_CKT_DETAIL
- ASAP.GCD_DLR
- ASAP.GCD_PATH_CIRCUIT
- ASAP.GE_NET_ELEMENT_GATE_EVENT_INST
- ASAP.GE_NETWORK_EVENT_INST_ERROR
- ASAP.GLR_ASSGN_SEG_CKT_POS
- ASAP.NAI_CKT_DETAIL
- ASAP.NOTES
- ASAP.NS_COMP_REL_NS
- ASAP.NS_CON_LABEL_CKT
- ASAP.NS_CON_REL
- ASAP.NS_CON_REL_CA_VALUE
- ASAP.NS_CONNECTION
- ASAP.NS_MULTI_POINT_COMPONENT
- ASAP.NS_NS_CON
- ASAP.PAIR_PENDING
- ASAP.PORT_ADDRESS_PENDING
- ASAP.RESERVATION_CIRCUIT_POSITION
- ASAP.SAALD_LOCATION_TYPE
- ASAP.SERV_REQ
- ASAP.SERVICE_REQUEST_CIRCUIT
- ASAP.SPEC_ACCESS_AND_LINE_DETAIL
- ASAP.SPECIAL_CIRCUIT
- ASAP.SRC_TASK_COMPLETION
- ASAP.TRUNK_SIDE_DETAIL
- ASAP.USO_CIRCUIT_DETAIL
- ASAP.USO_CIRCUIT_LOC
- ASAP.VIRTUAL_CONNECTION

Purging Worksheets

This function deletes worksheets from service requests that were completed on or before the calculated purge date.

To purge worksheets:

1. Click the **Purge** button on the toolbar.
2. Click the **Worksheet** button on the Purge toolbar.
3. Enter values in the Year, Month, and Day fields.

   The purge date will be calculated by subtracting these intervals from today's date.
4. Click the **Start** button.

   When worksheets are purged, information is cleared from the following tables:

   - ASAP.SERV_ITEM
   - ASAP.SERV_REQ
   - ASAP.SERV_REQ_SI
   - ASAP.SR_LOC
   - ASAP.SR_RELATIONSHIP
   - ASAP.SRSI_ORDER_SUPPLEMENT
   - ASAP.SRSI_SR_LOC

Purging DLR Issues

This function allows you to delete extra design layout report issues. You can determine how many different DLR issues exist for a specific circuit by querying for the circuit in MetaSolv Solution.

This function works in conjunction with the DLR Issues preferences regarding the number of issues to keep for specific connections.

To purge DLR issues:

1. Click the **Purge** button on the toolbar.
2. Click the **DLR Issues** button on the Purge toolbar.

This will remove previous and overridden DLRs. Once this done, the data is deleted from the database.

When DLR issues are purged, MetaSolv Solution clears information from the following tables:

- ASAP.ACTIVATION_COMMAND_PLAN
- ASAP.CIRCUIT_LAYOUT_REPORT
- ASAP.DESIGN_DLR_ASSOC
- ASAP.DESIGN_LAYOUT_REPORT
- ASAP.DLR_CIRCUIT_DESIGN_LINE
- ASAP.DLR_ERROR
- ASAP.DLR_LINE_FLOW_THROUGH_OBJECT
- ASAP.DLR_NOTES
- ASAP.GCD_DLR

Purging Expired Reservations

This function allows you to purge expired reservations for equipment. When an expired reservation is purged, the status of the items on the reservation changes from

Reserved to *Unassigned*. This makes the reserved items available for assignment. If you do not purge reservations using this function, the reserved items remain unavailable.

To purge expired reservations:

1. Click the **Purge** button on the toolbar.
2. Click the **Reservations** button on the Purge toolbar.

This will remove expired reservations. Once this done, the data is deleted from the database.

MetaSolv' Solution clears information from the following tables when you purge reservations:

- ASAP.RESERVATION
- ASAP.RESERVATION_CIRCUIT_POSITION
- ASAP.RESERVATION_PAIR
- ASAP.RESERVATION_PORT_ADDRESS

Purging Server Logs

This function allows you to purge server logs.

To purge server logs:

1. Click the **Purge** button on the toolbar.
2. Click the **Server Logs** button on the Purge toolbar.

This will remove server logs. Once this done, the data is deleted from the database.

MetaSolv' Solution clears information from the following tables when you purge server logs:

- SERVER_LOG

Producing a Purge Error Report

To produce a Purge error report:

1. Click the **Purge** button on the toolbar.
2. Click the **Error Report** button on the toolbar.
3. Select a purge function from the **Purge Type** drop-down. The options that appear are those (for example, orders, employees) for which the Purge utility encountered errors.
4. Select a date from the **Purge Date** drop-down. The dates that appear are those dates when the Purge utility encountered errors running the selected purge function.
5. Click the **Retrieve** button on the toolbar.
6. The result displays in the **Purge Error Report Preview** window. From this window, preview the report on the screen or print the report.
7. To print the report, click the **Print** button.

Using the DB Health Utility

Running the database health function checks to make sure that all database objects required by MetaSolv Solution are valid, makes response time quicker, and eliminates

problems in processing information. While DB Health is required only after an installation, maintenance release or patch, you may use it at any time.

You must have DBA authority to use the DB_HEALTH utility. DBA authority is assigned by running a_users.sql during a new install, granting the DBA privilege to the ASAP user. See *MetaSolv Solution Installation Guide* for details.

Managing database health is a simple process:

1. Execute the DB Health utility in every area where errors are found.

Checking Database Health

To check the health of your database:

1. Open the MetaSolv Solutions Utilities application.
2. Click **DB Health**.

   If you are already in **DB Health,** click on the **Execute** button.
3. Click each tab to review the status of the database.

> **Note:** You may see errors after you check DB Health.

The database health function checks six different types of problems. The numbers on each tab represent how many database health errors were found. Table 6–9 describes the tabs.

Table 6–9 DB Health Tabs

Tab	Description
Grant Privileges	Fixes permissions that aren't set up completely due to the necessary rights not being granted for those tables.
Create Public Synonyms	Creates any public synonyms that are missing.
Compile Invalid Objects	Corresponds to stored procedures, triggers, views, and the table changes they reference. The application attempts to recompile the existing definitions of these objects to correct any errors.
Enable Disabled Objects	Displays database objects that are invalid and the DBA must fix the problems on the database, until this tab shows no objects.

Repairing Database Errors

To repair database errors using the DB Health utility:

1. Click each tab.
2. Click **Execute** when errors are listed.
3. Repeat these steps for each tab that lists health errors.

> **Note:** The DB Health Utility cannot correct errors in invalid objects, the DBA must correct these errors. The DB Health Utility will attempt to grant privileges, create public synonyms and enable disabled objects.

Generating the DB Status Report

Use this report to provide Oracle Global Customer Support with database statistics vital to the health of the database.

To run the DB Report:

1. Click **DB Health**.
2. Click **DB Report**.

> **Note:** It will take some time to generate this report.

The following message appears:

```
In an effort to provide the best possible support,
a report of the condition of your database has been created. You will now be
prompted to save the file.
Save the dbstatus.txt and send it to Oracle Global Customer Support.
The file will be about 11 MB and can be compressed.
```

3. Read the message and click **OK**.
4. Save the report as **dbstatus.txt** and send it to Oracle Global Customer Support.

Using the Migration Utility

The Next Generation Migration utility is applicable only if you are upgrading from a pre-6.0.*x* version (for example, MSS 5.2) to 6.3.*x*. If you are moving to 6.3.*x* from a pre-6.0.*x* release, it may be necessary for you to perform a data migration especially if you used the Broadband Network Design module to design and provision ATM/Frame Relay and DSL. The Broadband module is obsolete as of 6.0, and customers who use the Broadband module must migrate using this migration utility or if you need the ability to set up your network elements and associate equipment to them.

Using the Custom Attributes Utility

You can set up custom attributes (CA's) to perform several tasks, including collect and compute data at specific processing points. MetaSolv Solution provides several CAs, which are available after the CA cache refreshes, for you to associate with building blocks.

You can create your own CA's. You can:

- Create a unit of measure for the custom attribute
- Associate a custom attribute with a building block and define how a CA looks and works at a process point
- Create custom attributes in several languages

To create or modify CAs:

1. Access the **Custom Attributes** utility from the Utilities **Options** menu.
2. Follow directions in the online Help.

Using the Geographical Area Types Utility

The Geographical Area Types (GATs) option identifies types of geographic areas related to a specific country and contains rules that define how the geographic areas are related. For example, the United States consists of states and cities.

To create or maintain a Geographical Area Type:

1. Access the **Geographical Area Types** option from the Utilities **Options** menu.
2. Follow directions in the online Help.

Using the Structured Formats Utility

Access the **Structured Formats** utility from the **Options** menu.

Structured formats provide a way to enter, define, and store structure and formatting information for addresses, telephone numbers, and switch prefixes. Since these information structures can vary from country to country, the structured formats functionality uses geographical areas, geographical area types, MetaSolv Solution processes, and industry versions to provide the flexibility needed in setting up the appropriate structures.

Structured formats consist of user-defined components that can define multiple structured formats. Components can be associated to valid values. For example, acceptable values can be indicated for components such as streets, directional prefixes, or house numbers. Also, ineligible characters can be indicated for a component.

To create or maintain a structured format:

1. Access the **Structured Format** option from the Utilities **Options** menu.
2. From the Structured Formats drop-down, select one of these options:
   - **Structured Formats**: Provides the means to create a Structured Format or modify a Structured Format
   - **Valid Values**: Provides the means to create a valid value for a Structured Format
   - **SF Components**: Provides the means to search for a Structured Format component

   Follow instructions in the Help for using these options.

Note: Use the **New From** function when possible rather than creating a brand new structured format each time. When selecting **New From**, the utility copies all the structured format components, valid values, ineligible characters, and country relationships from the selected structured format to the new structured format.

Caution: Because structured formats affect so many areas of MetaSolv Solution, contact MetaSolv Professional Services before setting up a new structured format, or changing an existing one.

For informative conceptual information, see *Structured Formats Best Practices Guide* located under MetaSolv Solution Archived Documentation on the Oracle Help Center.

The guide explains how, when, and why you modify, create, and use structured formats. It also makes recommendations for working with structured formats by following specific sequences and adhering to established guidelines. Included are detailed, step-by-step instructions for certain tasks to clarify the sequence and values involved.

Using the Address Correction Utility

The Address Correction utility enables you to resolve inconsistencies related to end user locations.

To run the Address Correction utility:

1. Start the MetaSolv Solution Utilities. See "Starting the Application" for more information.
2. From the **Options** menu, select **Address Correction Utility**, then **Address Correction**.

   The Address Correction utility is displayed.

   See *MSS Address Correction Utility User's Guide* for more information.

NPA Split Utility

The NPA Split Utility allows users to update the database for area code splits and manage ongoing NPA split administration. It also provides access to historical data that contains NPA split and telephone number inventory information.

> **Caution:** Prior to running the NPA Split Utility, the database administrator should perform a full backup and export of the MetaSolv Solution database.

Once an administrator is notified of upcoming NPA split activity and has received planning information, the administrator must update the NPA tables within the MetaSolv Solution application setup, including the new NPA, affected NXXs, and line ranges.

This can be accomplished either through manual entry of the NPA NXX and association with the end office through NXX assignment in MetaSolv Solution or by using the Location and Routing Gateway. The administrator must also create appropriate number ranges for the new NPA NXX combinations. The ranges created should be the same as the ranges for the original NPA.

The split utility updates:

- Telephone number inventory
- CLT circuit IDs
- Ordered items
- In-service items

The split utility does not update:

- Customer contacts
- User contacts
- Location phone numbers

- LRN (Location routing numbers)
- External billing systems
- LERG information

> **Note:** New NPAs must be created within the MetaSolv Solution Application Setup prior to initiating an NPA split.

Follow these steps to access the NPA Split Utilities:

1. Run the **npasplit.exe** file.
2. Log in to the NPA Split utility as the ASAP user. (The ASAP user is created during installation.)
3. Use the buttons on the main toolbar to perform the tasks described in Table 6–10.

Table 6–10 NPA Split Tasks

Button	Function
NPA Split	Create an NPA split
Split Report	Provides historical data during the permissive dialing period for telephone numbers associated with an NPA split.
Customer Report	After an NPA split is executed by the NPA Split Utility, this report provides information on the customers affected by the split.
Data Report	Lists telephone numbers that do not pass validation, and provides a means to handle duplicate number issues that can occur when executing an NPA split.

7

Troubleshooting and FAQs

This chapter contains information about the following:

- Common system administration problems
- Frequently-asked general questions
- Common Oracle WebLogic server questions
- Common API questions

Cannot Log On or Access Certain Functionality

If users cannot log on, check these possibilities:

- You may need to specify a different Oracle WebLogic server port or server name.
- You may need to correct the database instance entry in the **Tnsnames.ora** file.
- Make sure the Oracle Communications MetaSolv Solution (MSS) Application is deployed on the Oracle Application Server.
- Make sure users are added to the third party security server.
- Make sure users have been added to the MetaSolv Solution.
- Make sure the MetaSolv Solution database resources have been deployed.

The System Seems Slow

If the system seems slow, check these possibilities:

- Verify the Custom Attribute Refresh period in **gateway.ini** file in the [CA] section.
- Make sure MEM_ARGS in startMSLVServer followed the recommended setup guidelines.

  For example, for a machine with 2GB physical memory:

  ```
  Xms size = 2048MB * 0.375 = 768m
  NewSize = 768m / 3 = 256m
  MEM_ARGS="-Xms768m -Xmx768m -XX:NewSize=256m -XX:MaxNewSize=256m"
  ```

- Make sure garbage collection verbose mode (-verbosegc) is not turned on.
- Turn off unnecessary logging for the Oracle WebLogic server using the Administration Console.
- Turn off unnecessary logging in **loggingconfig.xml**.
- Check the amount of memory being used.

- Check the CPU and disk usage on the machine hosting the MetaSolv Solution database.
- Check the database connections.
- Check the execute queue.
- Make sure the **Native IO** is enabled to improve performance. See "Why Do I Have a Core Dump in My Domain Directory?" for special instructions.
- Run the **mdl_cmp.sql** script to help maintain the database structure for primary and foreign keys, indexes, and sequences.
- Run database statistics.

The Oracle WebLogic Server Does Not Start

If the Oracle WebLogic server will not start up, try the following:

- Start the Admin Server using the startMSLVadmin script in the domain directory.
- Start the Oracle WebLogic Application server.
- Run the **startweblogic.cmd** from a cmd prompt. If the Oracle WebLogic license has expired, you will see messages in the Oracle console. If you see messages about an expired or invalid license, you should contact Oracle Global Customer Support.
- If you have a valid license and Oracle WebLogic server will not start up, replace the **config.xml** file in the domain directory with the **config.xml.booted** file. The **config.xml.booted** file represents the contents of the **config.xml** file the last time that Oracle WebLogic server was successfully started. Oracle WebLogic server should successfully start with this file. If it does, you may want to compare its contents with the contents of the **config.xml** used in the unsuccessful startup. The difference in the two files is the cause of the startup failure.

The Database Cannot Be Found

You must be sure that the database configured in the JDBC URL of the connection pools is the same database identified as in the **tnsnames.ora** file found in the Oracle Thin Client installation folder.

I Cannot Access the Administration Console

Check to see whether the Admin server is running. It must be running to allow access to the Administration Console or to perform any administrative function.

A Job Sent to the Background Processor Did Not Complete

If jobs sent to the background fail, look for the following the types of problems:

- End-user errors
- Missing files
- Missing database profiles
- System problems

End users must resolve errors they create in their jobs. Review the error messages that occur in the Job Queue and determine whether or not an error was caused by user input. Check the online Help for assistance with managing the Job Queue or see "Running Background Processor" for more information.

Files that MetaSolv Solution Expects to Find are Missing

MetaSolv Solution expects to find files where they were installed, but they may have been moved or removed altogether, perhaps as the result of reconfiguration efforts.

If files no longer exist or are suspected of having become corrupted or virus-infected, uninstall and then reinstall MetaSolv Solution or contact Oracle Global Customer Support for assistance. For guidance about reinstalling, see *MetaSolv Solution Installation Guide*.

Connections to a Database are Failing

If a log file contains messages that indicate a failure to connect to the MetaSolv Solution database, check the **tbs_util.ini**, **lerg.ini** or **npasplit.ini** file to be sure it contains a profile for the database in question.

I Cannot Run the Log Viewer

Java must be installed and you must have access to the **logviewer.cmd** or **logviewer.sh** on your machine. Located in *MSLV_Home*/*server_name*/**appserver/bin** directory.

Many Errors are Being Logged for the Background Processor

Rarely, the Background Processor can contain a long list of errors for jobs that did not complete or that completed with errors. Pause the queue to allow executing jobs to complete and, if feasible for your business, clear the queue and reboot. This sets the job statuses to **Delete** in the database.

If necessary, the database administrator can access the Oracle database and reset a job status to **Ready** so it can run again, but MetaSolv does not recommend depending on that ability.

Graphics are Not Displayed in the GUI

Check to be sure that the MetaSolv Solution folder on the client contains the **TBSGraphicLoad.exe** file. This file preloads tables with graphic images. Running this file is a manual step that is included in the instructions in the *MetaSolv Solution Installation Guide*. Although the .exe file is loaded to all workstations as part of the client installs, it only has to be run from one workstation once against the database to work for everyone.

Can I Combine My Oracle WebLogic Directory with the MSLV Directory?

You should not combine the MetaSolv Solution directory with other directories, although the product will work if you do. Upgrades target specific directories and will not be applied if you move the files in the MSLV directory.

How Do I Find Out the Names of the JAR files?

Display the Runtime page according to directions in "Getting Oracle WebLogic Server Runtime Information".

Can I Undo a Patch?

Yes. For instructions about uninstalling the product, see *MetaSolv Solution Installation Guide*.

What Do I Do if I See an Error in the Console Log?

If you see an error in the console log (**/appserver/logs/mslv01.mss.log**), run the Log File Viewer to review more detailed logging information that has been written to

appserverlog*.xml files. The server log file name and its contents can be viewed from the WebLogic Server Administration Console.

To view the server log file name:

1. Log in to the WebLogic Server Administration Console.
2. In the **Domain Structure** tree, expand **Environment**, and then click **Servers**.

   The Summary of Servers page appears.
3. Click the *server_name*.

   The Settings for *server_name* page appears.
4. Click the **Logging** tab.
5. In the **Log file name** field, note the name and location of the server log file. The log file is located at the following location:

   MSLV_Home/*m63domain*/**servers**/*server_name*

To view the latest contents of the server log file:

1. Log in to the WebLogic Server Administration Console.
2. In the **Domain Structure** tree, expand **Diagnostics**, and then click **Log Files**.

   The Summary of Log Files page appears.
3. Select the **ServerLog** option and click **View**.

   The Server Log page is displayed, which displays the latest contents of the server log file.

What Do I Need to Check to Be Sure All Passwords Match?

If you change the password for the APP_MSLV user in the Oracle database, you must change it in the WebLogic Server Administration Console for both connection pools by doing the following:

1. Log in to the WebLogic Server Administration Console.
2. In the **Domain Structure** tree, expand **Services**, and then click **Data Sources**.

   The Summary of JDBC Data Sources page appears.
3. Click **mslvDataSource**.

   The Settings for mslvDataSource page appears.
4. Click **Lock & Edit**.
5. Click the **Connection Pool** tab.
6. In the **Password** field, enter the password.
7. In the **Confirm Password** field, enter the password again to confirm it.
8. Click **Save**.
9. Repeat this procedure for the MSLVDbTracePool (mslvDbTraceDataSource).

If you change the password for the APP_API user in the Oracle database, you must change the password in the [Session] section of **/appserver/gateway/gateway.ini**.

```
[Session]
User=app_api
Password=password
```

Note: You must encrypt this password in the **gateway.ini** file using MetaSolv Solution utilities. See "Copying Encrypted Passwords to gateway.ini" for more information.

If you change the password for the administration user for the WebLogic Administration Console, you must also change it in **/appserver/gateway.ini**:

```
[JNDI]
User=administrator
Password=password
Factory=weblogic.jndi.WLInitialContextFactory
URL=http://localhost:9001
```

Note: You must encrypt this password in the **gateway.ini** file using MetaSolv Solution utilities. See "Copying Encrypted Passwords to gateway.ini" for more information.

What are the Default Passwords?

The default passwords for APP_API and APP_MSLV are set by the System Administrator during the database installation process.

When Does Disregard-Logging Start?

The following messages appear in the console log at server startup. Logging will initialize when the server is completely started.

```
log4j:WARN No appenders could be found for logger
(org.apache.struts.util.PropertyMessageResources).
log4j:WARN Please initialize the log4j system properly.
```

What Happens if I Change the HTTP Port for a Server Process?

If you change the HTTP port for the server process (using the Oracle WebLogic domain wizard) without reinstalling MetaSolv Solution, you must also change it in the following places:

- A start script for the server process, for example **/mslvdomain/startMSLV01.sh**
- **/appserver/gateway.ini**:

  ```
  [JNDI]
  User=administrator
  Password=password
  Factory=weblogic.jndi.WLInitialContextFactory
  URL=http://localhost:9001
  ```

- Affected INI files in **/appserver/config** - these must be distributed to client machines manually.

Why Do I Have a Core Dump in My Domain Directory?

This problem is caused by turning on **Use Native IO** using the WebLogic Admin Console. If it is disabled, it will not generate a core file. MetaSolv recommends that the Native IO be enabled to improve performance.

To work around this problem, delete the core dump file in domain directory before server restart; otherwise it will grow and fill up the file system. You should wait for the core dump to complete after shutdown before trying to restart.

See the Oracle WebLogic Server documentation for more information.

How Do I Know Which Database My Oracle WebLogic Server is Connected To?

Display the Runtime page according to directions in "Getting Oracle WebLogic Server Runtime Information".

How Do I Know When to Restart the Server or the Database?

- For a **gateway.ini** change, restarting the Oracle WebLogic server is required.
- Check the upper left corner in the WebLogic Server Administration Console where a message is displayed that informs you whether you need to restart the server or not.

  The icon indicates that changes to this attribute may require the restart of one or more servers. If you alter the value of this attribute, running servers will not reflect the change until they are restarted, and so you may need to restart multiple servers for the new value to take effect.
- After a custom portlet is added into the registry, you must restart the Oracle WebLogic server or redeploy the application (nur.ear) in order to reload the portlet registry.
- Certain key Oracle database initialization parameters require the Oracle database to be shut down and restarted. Refer to the Oracle database documentation for assistance.

Can I Redirect an Existing Oracle WebLogic Server to Another Database?

To redirect an existing Oracle WebLogic server to another database:

1. The connection pool configurations in the Administration Console (jdbc url, db user, password).
2. **Tnsnames.ora** - the **mslvappserver** alias to database.
3. Reset passwords for APP_MSLV, APP_API and APP_INT, if necessary.

How Can I Tell the Version of My Oracle WebLogic Server?

Display the Runtime page according to directions in "Getting Oracle WebLogic Server Runtime Information".

How Can I Force the Oracle WebLogic Server to Run as a UNIX Background Process and What are the Differences When Running This Way?

The Oracle WebLogic server runs in background if you use the MetaSolv Solution start scripts. The console log messages go to the ***.mss.log** file in **/appserver/logs**.

I Cannot Produce All of My Reports After Installing the New Release. What Should I Do?

Custom reports must be manually carried forward from one release to the next. Read "Retaining Customized Reports After an Upgrade" for instructions.

How Can I Customize My Oracle WebLogic Server Startup Script?

Edit the scripts provided by Oracle or create your own. The **startMSLV*** scripts are found in the domain directory. The settings in **startMSLVserver** script are very important and must be configured properly for the Oracle WebLogic server to start and run successfully.

How Can I Uninstall My Oracle WebLogic Server?

Undeploy the **nur.ear** file from the Oracle WebLogic Administration console. Delete the directory where MetaSolv Solution is installed. This will not uninstall Oracle WebLogic server. Run the Oracle WebLogic server uninstall if you want to uninstall the WebLogic software.

Which Environment Variables are Used By My Oracle WebLogic Server?

ORACLE_HOME, BEA_HOME, JAVA_HOME, JACORB_HOME, TNS_ADMIN, CLASSPATH, PATH (others are set inside the start scripts and are effective only for that process).

How Can I Look at the Objects and Object Versions Deployed to My Oracle WebLogic Server?

Display the Oracle WebLogic Administration Console or the MetaSolv Solution Runtime Info page.

How Can I Manage the Max Number of Client Connections, Threads and Other Performance Settings?

See the deployment information in the *MetaSolv Solution Planning Guide* or the Oracle WebLogic documentation. The answer to these questions is specific to the environment, and to what you are trying to do in the application.

How Can I Control the Default Timeout Settings?

Use the Oracle WebLogic Administration Console.

How Can I Allocate More JVM Memory to My Oracle WebLogic Server?

Increase the memory arguments being used when the app server starts. See **/domaindir/startMSLV***server_name***.sh** or **startMSLV***server_name***.cmd**.

What Do I Do if Users Must Access the Oracle WebLogic Server from Outside a Firewall?

You must open the http port to the Oracle WebLogic server.

How Can I Install RSA and DSA Certificates in Order to Implement SSL on My Oracle WebLogic Server?

See the Oracle WebLogic documentation for instructions.

How Can I Debug Oracle WebLogic Server Performance Problems?

See the performance information in the *MetaSolv Solution Installation Guide*. See the Oracle WebLogic documentation Web site. As a rule of thumb, make changes in small increments; change only one thing at a time and evaluate the effects of the change on performance.

How Do I Start the jacORB Name Service?

It is started by the **startMSLVorb** script.

How Can I Control Where My IOR Files are Written?

Use the **IORPATH=** parameter in the **gateway.ini** file.

ADMIN OPTION Not Granted for WOTSTWTWWOO Role After Database Upgrade

If you upgrade your Oracle Database software to 12.1.0.2.0, and then try to create a new user through the Security Users and Groups window in the MSS application, the following error message is displayed:

```
Grant role error. Please contact your system administrator.
ORA-01932: ADMIN option not granted for role 'WOTSTWTWWOO'
```

To resolve this issue, do the following:

> **Note:** Oracle recommends that you perform this procedure in a test environment with a copy of your production data before implementing this procedure in your production environment.

1. In the server parameter file (SPFILE), set the Oracle hidden parameter _grant_secure_role to **true**.
2. Restart the database.
3. Run the **21125221.sql** file located in the *installation_directory*/**prodfixsql** directory, where *installation_directory* is the directory on your database server where you extracted the contents of the **MSS.R6_2_1.b***xxx*_**DBInstall.zip** file; *xxx* is the build number.

   The **21125221.sql** file grants the GRANT ANY ROLE system privilege to ADMIN_ROLE with the ADMIN OPTION.

A

Parameters for gateway.ini File

The **gateway.ini** file is the main configuration file for Oracle Communications MetaSolv Solution and for application programming interface (API) interactions with other applications. The Oracle WebLogic server accesses the **gateway.ini** file and gets the configuration settings for managing these interactions from the **/appserver/gateway** folder in the Oracle WebLogic server.

Using gateway.ini

Some **gateway.ini** parameters can be set during installation. Therefore, the **gateway.ini** file may contain some site-specific values as a result.

In most cases, using the default values is recommended. In some cases, parameter settings are directly tied to internal processing and should never be changed.

Changing Parameters

Valid reasons for changing optional parameters include:

- Improving performance
- Updating a database ID or a password when changes are made in the environment

There may be instances when Oracle Global Customer Support requests that you change a **gateway.ini** parameter for research or debug purposes.

Restarting the Oracle WebLogic Server

When you change a setting in the **gateway.ini** file, that change does not take effect until the next time you start the Oracle WebLogic server. Therefore, whenever you change one or more of the **gateway.ini** parameters, you must shut down and restart the Oracle WebLogic server

When you suspend and resume a MetaSolv Solution server process, that process does not re-read the **gateway.ini** settings that affect its operation. Therefore, changes are ignored unless you shut down and restart the Oracle WebLogic server.

Understanding the gateway.ini File Structure

The **gateway.ini** file parameters are grouped together in the file to make it easier for you to find parameters you may need to change. The section order may be changed.

Following is a list of the parameter groups in the order they appear in the installed **gateway.ini** file:

- Server parameters

- ThreadProcs parameters
- OrbProperties parameter
- Session parameters
- System parameters
- Gateway parameters
- SignalInternetServices
- Events parameters
- Event2 parameters
- WorkManagement parameters
- JNDI parameters
- CA custom attributes parameters
- Portal parameters
- Custom Extension parameters

The following sections provide the following types of information about each section of the **gateway.ini** file:

- Defaults and valid entries
- Parameter descriptions
- Instructions about prohibited changes, where appropriate

Sample installed gateway.ini File

Some lines in the **gateway.ini** file are very long. When viewing the file, some editors may wrap the text, making the file appear to contain extra text. Disable **Word Wrap** to view long lines as they are intended to be used.

```
[Servers]
;This section is modified by the install to contain a list of active servers.
DLRSERVER=MetaSolv.CORBA.WDIDLR.WDIRoot,MetaSolv.WDIDLR.WDIRootImpl
PSRSERVER=MetaSolv.CORBA.WDIPSR.WDIRoot,MetaSolv.WDIPSR.WDIRootImpl
LSRSERVER=MetaSolv.CORBA.WDILSR.WDIRoot,MetaSolv.WDILSR.WDIRootImpl
WMSERVER=MetaSolv.CORBA.WDIWM.WDIRoot,MetaSolv.WDIWM.initialization.WDIRootImpl
NISERVER=MetaSolv.CORBA.WDINI.WDIRoot,MetaSolv.WDINI.WDIRootImpl
PSRANCILLARYSERVER=MetaSolv.CORBA.WDIPSRAncillary.WDIRoot,MetaSolv.PSRAncillary.WDIRootImpl
SOASERVER=MetaSolv.CORBA.WDISOA.WDIRoot,MetaSolv.WDISOA.initialization.WDIRootImpl
TMSSERVER=MetaSolv.CORBA.WDITrouble.WDIRoot,MetaSolv.WDITrouble.initialization.
WDIRootImpl
;ASRSERVER=MetaSolv.CORBA.WDIASR.WDIRoot,MetaSolv.WDIASR.WDIRootImpl
;ASR41SERVER=MetaSolv.CORBA.WDIASR.WDIRoot,MetaSolv.WDIASR.WDIRootImpl
;ASR42SERVER=MetaSolv.CORBA.WDIASR42.WDIRoot,MetaSolv.WDIASR25.WDIRootImpl
;CABSSERVER=MetaSolv.CORBA.WDICABS.WDIRoot,MetaSolv.WDICABS.WDIRootImpl
INFRASTRUCTURESERVER=MetaSolv.CORBA.WDIInfrastructure.WDIRoot,MetaSolv.
WDIInfrastructure.WDIRootImpl
PSREUBServer=MetaSolv.CORBA.WDIPSREUB.WDIRoot,MetaSolv.WDIPSREUB.WDIRootImpl

[ThreadProcs]
;INTEGRATIONSERVER=com.mslv.integration.integrationServer.S3Startup
EVENTPROC=MetaSolv.eventServer.S3Startup
EVENT2PROC=MetaSolv.event2Server.Event2ServerStartup
SYSTEMTASKSERVERPROC=com.mslv.core.api.internal.WM.systemTaskServer.
SystemTaskServer
```

```
SIGNALSERVERPROC=com.metasolv.system.StartServer
INTERNET_SIGNAL_
SERVER=MetaSolv.CORBA.WDIINTERNETSERVICES.WDIRoot,MetaSolv.SignalServer.WDIInternetSignalServerRoot
Impl

[OrbProperties]
ORBPort=2720

[Session]
User=app_api
Password={AES}1824a67ea02612bcbe3169d0b0d185b4

[System]
URLNamingServicePort=15000
StartupMode=server;StartUp Mode - gui, batch, server
Environment=prod;Environment - dev, test, prod
DebugOn=true
LoggingOn=true
LogFileDir=/opt2/metasolv/mslv01/appserver/logs
LogToFile=true
LogToSystemOut=true
AppendToLog=true
IORPath=/opt2/metasolv/mslv01/appserver/ior
TraceLevel=1
;MaxThreads=1000

[Gateway]
PrintExportObjects=true
JobQueueServer=JOB SERVER NAME;Background Processor Logical Name
APPJMSUser=app_jms
APPJMSPwd={AES}61b01de259ce47676d3f7b85ca5086f9
Timeout=601

[Events]
DBPollingInterval=30
DBPollingInterval_Srsi=60
DBPollingInterval_Serv=60
DBPollingInterval_Ge=60
DBPollingInterval_Genet=60
suspendCheckInterval=10
MaxThreads_Srsi=1
MaxThreads_Serv=1
MaxThreads_Ge=1
MaxThreads_Genet=1
maxCapacity_Srsi=500
maxCapacity_Serv=500
maxCapacity_Ge=500
maxCapacity_Genet=500
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEventServer
[Event2]
DBPollingInterval=30
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEvent2Server

[WorkManagement]
DbPollingInterval=120
MaxThreads=5
SystemTaskQueueMaxCapacity=2000
SuspendCheckInterval=5
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendSystemTaskServer
;SystemTasksToInclude=
```

```
;SystemTasksToExclude=

[JNDI]
User=administrator
Password={AES}1824a67ea02612bcbe3169d0b0d185b4
Factory=weblogic.jndi.WLInitialContextFactory
URL=http://srvsunpluto:7001

[CA]
CacheDisable=false
CacheManagerClass=com.mslv.core.api.internal.ca.cache.broker.CacheManager
CacheRefreshActive=true
CacheRefreshMode=daily
CacheRefreshDayOfWeek=6
CacheRefreshHourOfDay=23
CacheRefreshHourlyInterval=1

[Portal]
DefaultPortalId=asap

[Custom]
CLASSPATH=D:/opt2/metasolv/mslv01/appserver/samples/customExtension;
```

Servers Parameters

In the **gateway.ini** file, the term server parameter refers to a server process, which is another term for API. For instance, the **PSRSERVER** parameter name refers to the PSR API.

This section controls which MetaSolv Java API server processes are started automatically when the Oracle WebLogic server is started.

> **Caution:** Do not alter the server names or startup classes.

Some servers may not be listed in the **gateway.ini** initially but are added as separate products are installed.

You can prevent a specific server from starting by placing a preceding semicolon (;) in front of its name.

Sample Servers Parameters

```
[Servers]
;This section is modified by the install to contain a list of active servers.
DLRSERVER=MetaSolv.CORBA.WDIDLR.WDIRoot,MetaSolv.WDIDLR.WDIRootImpl
PSRSERVER=MetaSolv.CORBA.WDIPSR.WDIRoot,MetaSolv.WDIPSR.WDIRootImpl
LSRSERVER=MetaSolv.CORBA.WDILSR.WDIRoot,MetaSolv.WDILSR.WDIRootImpl
WMSERVER=MetaSolv.CORBA.WDIWM.WDIRoot,MetaSolv.WDIWM.initialization.
WDIRootImpl
NISERVER=MetaSolv.CORBA.WDINI.WDIRoot,MetaSolv.WDINI.WDIRootImpl
PSRANCILLARYSERVER=MetaSolv.CORBA.WDIPSRAncillary.WDIRoot,MetaSolv.
PSRAncillary.WDIRootImpl
SOASERVER=MetaSolv.CORBA.WDISOA.WDIRoot,MetaSolv.WDISOA.initialization.
WDIRootImpl
TMSSERVER=MetaSolv.CORBA.WDITrouble.WDIRoot,MetaSolv.WDITrouble.
initialization.
WDIRootImpl
ASR52SERVER=MetaSolv.CORBA.WDIASR52.WDIROOT,MetaSolv.WDIASR52.WDIRootImpl
```

```
ASR53SERVER=MetaSolv.CORBA.WDIASR53.WDIROOT,MetaSolv.WDIASR53.WDIRootImp
LSR10SERVER=MetaSolv.CORBA.WDILSR10.WDIROOT,MetaSolv.WDILSR10.WDIRootImpl
CABSSERVER=MetaSolv.CORBA.WDICABS.WDIRoot,MetaSolv.WDICABS.WDIRootImpl
INFRASTRUCTURESERVER=MetaSolv.CORBA.WDIInfrastructure.WDIRoot,MetaSolv.
WDIInfrastructure.WDIRootImpl
PSREUBServer=MetaSolv.CORBA.WDIPSREUB.WDIRoot,MetaSolv.WDIPSREUB.WDIRootImpl
```

These parameters are modified by the install to contain a list of active servers.

Server Parameter Descriptions

Table A–1 describes the server parameters.

Table A–1 Servers Parameters

Parameter	Description
ASRSERVER, ASRnnSERVER	This parameter starts the ASRnnSERVER for the ASR API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
CABSSERVER	This parameter starts the CABSSERVER and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
DLRSERVER	This parameter starts the DLR server for the Inventory and Capacity Management API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
PSREUBServer	This parameter starts the PSREUBServer for the PSR End User Billing API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
INFRASTRUCTURESERVER	This parameter starts the INFRASTRUCTURESERVER for the Infrastructure API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
JobQueueSERVER	This parameter starts the Job Queue server and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
LSRSERVER	This parameter starts the LSR server for the LSR API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
NISERVER	This parameter starts the NI server for the Number Inventory API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
PSRANCILLIARYSERVER	This parameter starts the PSRANCILLIARYSERVER for the PSR Ancillary API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
PSRSERVER	This parameter starts the PSR server for the PSR API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
SOASERVER	This parameter starts the SOASERVER for the Service Order Activation API and initializes it to the name service. This server is no longer used. To prevent this process from starting, comment out this line with a semicolon.

Table A–1 (Cont.) Servers Parameters

Parameter	Description
TMSSERVER	This parameter starts the TMSSERVER for the Trouble Management API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.
WMSERVER	This parameter starts the WM server for the Work Management API and initializes it to the name service. To prevent this process from starting, comment out this line with a semicolon.

ThreadProcs Parameters

This section includes MetaSolv server processes that function like the APIs. These processes are related to transactions generated by other MetaSolv Solution components, such as gateway events from the MetaSolv Solution client or jobs from the Background Processor. These server processes receive information from other Oracle products instead of third-party applications. These ThreadProcs start when the Oracle WebLogic server starts.

Caution: If the servers in the ThreadProcs section are missing or commented out, the Oracle WebLogic server does not start those servers. This can affect outbound gateway events, TNI Recall, system task processing, and other functions that use these servers. Do not disable a ThreadProcs server unless directed to do so by Oracle Global Customer Support.

Sample ThreadProcs Section

```
[ThreadProcs]
;INTEGRATIONSERVER=com.mslv.integration.integrationServer.S3Startup
EVENTPROC=MetaSolv.eventServer.S3Startup
EVENT2PROC=MetaSolv.event2Server.Event2ServerStartup
SYSTEMTASKSERVERPROC=com.mslv.core.api.internal.WM.systemTaskServer.
SystemTaskServer
SIGNALSERVERPROC=com.metasolv.system.StartServer INTERNET_SIGNAL_SER
VER=MetaSolv.CORBA.WDIINTERNETSERVICES.WDIRoot,MetaSolv.Sig
nalServer.WDIInternetSignalServerRootImpl
```

ThreadProcs Parameter Descriptions

These parameters are modified by the install to contain a list of active servers. See "Setting Up API Servers" for descriptions of active servers.

Table A–2 describes the TheadProcs Parameters.

Table A–2 ThreadProcs Parameters

Parameters	Description
INTEGRATIONSERVER	**Default**: com.mslv.integration.integrationServer.S3Startup **Valid values**: com.mslv.integration.integrationServer.S3Startup **Do not alter or remove this parameter.**

Table A–2 (Cont.) ThreadProcs Parameters

Parameters	Description
EVENTPROC	**Default**: MetaSolv.eventServer.S3Startup **Valid values**: MetaSolv.eventServer.S3Startup This parameter starts the Gateway Event server process. The only acceptable value is **MetaSolv.eventServer.S3Startup**. If this parameter is present, the Oracle WebLogic server starts the Gateway Event server process when the Oracle WebLogic server starts. If this parameter is missing or is commented out, the Oracle WebLogic server does not start the Gateway Event server process and no inbound or outbound gateway events can be processed by the MetaSolv Solution APIs. **Do not alter, remove, or comment out this parameter except as instructed by Oracle Global Customer Support.**
EVENT2PROC	**Default**: MetaSolv.event2Server.Event2ServerStartup **Valid values**: MetaSolv.event2Server.Event2ServerStartup This parameter starts the PSR End User Billing server process. **MetaSolv.event2Server.Event2ServerStartup** is the only acceptable value. If this parameter is present, the Oracle WebLogic server starts the PSR End User Billing server process when the Oracle WebLogic server starts. If this parameter is missing or is commented out, the Oracle WebLogic server does not start the PSR End User Billing server process and no inbound or outbound gateway events can be processed by the MetaSolv Solution APIs. **Do not alter, remove, or comment out this parameter except as instructed by Oracle Global Customer Support.**
SYSTEMTASKSERVERPROC	**Default**: com.mslv.core.ap.internal.WM.systemTaskServer.SystemTaskServer **Valid values**: com.mslv.core.ap.internal.WM.systemTaskServer.SystemTaskServer This parameter starts the System Task server process. The only acceptable entry is **com.mslv.core.ap.in ternal.WM.sys temTaskServer.SystemTaskServer**. If this parameter is present, the Oracle WebLogic server starts the System Task server process when the Oracle WebLogic server starts. If this parameter is missing or is commented out, the Oracle WebLogic server does not start the System Task server process and no system tasks can be automatically completed by the MetaSolv Solution. **Do not alter or remove this parameter.**
SIGNALSERVERPROC	**Default**: com.metasolv.system.StartServerINTERNET_ SIGNAL_SER VER=MetaSolv.CORBA.WDIINTERNETSERVICES.WDIRoot ,MetaSolv.SignalServer.WDIInternetSignalServerRootImpl **Valid values**: com.metasolv.system.StartServer INTERNET_ SIGNAL_ SERVER=MetaSolv.CORBA.WDIINTERNETSERVICES.WDI Root,MetaSolv.SignalServer.WDIInternetSignalServerRootIm pl This parameter starts the SIGNALSERVERPROC server and initializes it to the name service. To prevent this process from starting, comment this line with a semicolon. Also may be referred to as the INTERNET_SIGNAL_SERVER.

OrbProperties Parameter

The OrbProperties section designates the port published in the API servers' CORBA IOR files. If this parameter is not set, jacORB will assign a dynamic port number at startup time.

Sample OrbProperties Parameter

```
[OrbProperties]
OrbPort=2720
```

OrbProperties Parameter Description

Table A–3 describes the OrbProperties parameter.

Table A–3 OrbProperties Parameter

Parameter	Description
ORBPort	**Default**: dynamically assigned **Valid values**: A valid port This parameter designates the port published in the API servers' CORBA IOR files. If the parameter is left blank, the port name will be dynamically assigned by jacORB.

Session Parameters

Session parameters control the JDBC connection between the MetaSolv APIs and the Oracle database.

Sample Session Section

```
[Session]
User=app_api
Password={AES}1824a67ea02612bcbe3169d0b0d185b4
```

Session Parameter Descriptions

Table A–4 describes the session parameters.

Table A–4 Session Parameters

Parameter	Description
User	**Default**: app_api **Valid values**: app_api This is the (database) userid used to make connections to the database for the APIs.
Password	**Default**: none **Valid values**: A valid encrypted password This is the (database) password, for the APP_API user, that is used to make connections to the database for the API's.

System Parameters

The **gateway.ini** system parameters provide general configuration for the MetaSolv APIs.

Sample System Section

```
[System]
;URLNamingServicePort=15000
StartupMode=server;StartUp Mode - gui, batch, server
Environment=prod;Environment - dev, test, prod
DebugOn=false
LoggingOn=true
LogFileDir=D:/opt2/metasolv/mslv01/appserver/logs
LogToFile=false
LogToSystemOut=true
AppendToLog=true
IORPath=/opt2/metasolv/mslv01/appserver/ior
TraceLevel=1
;MaxThreads=1000
```

System Parameter Descriptions

Table A–5 describes the system parameters.

Table A–5 System Parameters

Parameter	Description
URLNamingServicePort	**Default**: 15000 **Valid values**: The port that provides the Naming Service object references using a URL.
StartupMode	**Reserved for internal processing. Do not alter, remove, or comment out this parameter.**
Environment	**Default**: Prod **Valid values**: Prod, dev, test **Reserved for internal processing. Do not alter, remove, or comment out this parameter.**
DebugOn DebugLevel LoggingOn LogFileDir LogToFile LogToSystemOut AppendToLog TraceLevel	Do not enter values for these parameters or remove them from the file. MetaSolv Solution application-specific logging uses a third-party software product (log4j) that is configured by setting properties in the **loggingconfig.xml** file. Oracle WebLogic server logging can be controlled through the WebLogic Server Administration Console. See "Annotated loggingconfig.xml" for more information on the **loggingconfig.xml** file.
IORPATH	**Default**: /opt2/metasolv/mslv01/appserver/ior **Valid values**: A valid directory name The IORPath represents the physical location where the API server CORBA IOR files will be written when the Oracle WebLogic server is started.
MaxThreads	**Default**: 1000 **Valid values**: 1 to a system-specified maximum This specifies the maximum number of threads that the API server will use.

Gateway Parameters

This section specifies miscellaneous and internal configuration parameters.

Sample Gateway Parameters

```
[Gateway]
PrintExportObjects=false
JobQueueServer=JOB SERVER NAME;Background Processor Logical Name
APPJMSUser=app_jms
APPJMSPwd={AES}61b01de259ce47676d3f7b85ca5086f9
Timeout=601
```

Gateway Parameter Descriptions

Table A–6 describes the gateway parameters.

Table A–6 Gateway parameters

Parameter	Description
PrintExportObjects	**Default**: false **Valid values**: true, false **true** causes extra logging. To improve performance, reset this parameter to **false**. This parameter causes the contents of CORBA container objects to be listed in Oracle Global Customer Support API logs, for debugging purposes. **true** specifies that the object state display be prior to client notification when exporting objects from the server to the client. **false** specifies no display of object state.
JobQueueServer	Used for internal processing. Do not alter, remove, or comment out these parameters.
Server	Used for internal processing. Do not alter, remove, or comment out these parameters.
Timeout	Used for internal processing. Do not alter, remove, or comment out these parameters.
APPJMSUser	**Default**: app_jms **Valid value**: WebLogic user with Integration Administrators group
APPJMSPwd	**Default**: {AES}61b01de259ce47676d3f7b85ca5086f9 **Valid value**: A valid encrypted password for the APPJMSUser The (WebLogic) password to establish a connection with the application server for the gateway events.

Events Parameters

The Events parameters control operation of the Gateway Event server.

Sample Events Parameters

The following is how the Events section looks in the **gateway.ini** file:

```
[Events]
DBPollingInterval=30
DBPollingInterval_Srsi=60
DBPollingInterval_Serv=60
DBPollingInterval_Ge=60
DBPollingInterval_Genet=60
suspendCheckInterval=10
MaxThreads_Srsi=1
MaxThreads_Serv=1
MaxThreads_Ge=1
MaxThreads_Genet=1
maxCapacity_Srsi=500
maxCapacity_Serv=500
maxCapacity_Ge=500
maxCapacity_Genet=500
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEventServer
```

Events Parameter Descriptions

Table A–7 describes the Events parameters.

Table A–7 Events Parameters

Parameter	Description
DBPollingInterval	**Default**: 30 (seconds) **Valid values**: number between 1second and 2 million seconds This parameter determines the number of seconds the Gateway event server allows to elapse before it polls the database for events to process. Setting this interval larger than a few minutes may adversely affect overall performance of the application and the API operations that involve gateway events.
SuspendFileName	**Default**: /opt2/metasolv/mslv01/appserver/suspendEventServer **Valid values**: a valid file name Disable this parameter if not needed. Identifies the path and name of a file you can use to signal the Gateway Event server to pause operation. Immediately prior to polling the database, the Gateway Event server checks for the existence of this file. If the file does not exist, the Gateway Event server polls the database, initiates needed work, then waits until the next scheduled poll. If the file exists, the Gateway Event server skips polling the database, then waits until the next scheduled poll. The server repeats this process at each scheduled polling interval as long as the file continues to exist.

Caution: While suspended, the Gateway Event server cannot process any incoming or outgoing gateway events. Suspending the Gateway Event server for more than a few minutes may adversely affect overall performance of MetaSolv Solution and MetaSolv Solution API operations that involve gateway events.

Event2 Parameters

The PSR End User Billing API uses the Event2 server process to send notifications to an external billing system.

Sample Event2 Parameters

```
[Event2]
DBPollingInterval=30
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEvent2Server
```

Event2 Parameter Descriptions

Table A–8 describes the Event2 server parameters.

Table A–8 Event2 Parameters

Parameter	Description
DBPollingInterval	**Default**: 30 (seconds) **Valid values**: a number between 1second and 2 million seconds This parameter determines the number of seconds which the Event2 server should allow to elapse before it polls the database for events to process. The default is 30 seconds. Setting this interval larger than a few minutes may adversely affect overall performance of MetaSolv Solution and MetaSolv Solution API operations that involve gateway events.
SuspendFileName	**Default**: /opt2/metasolv/mslv01/appserver/suspendEvent2Server **Valid values**: a valid file name Disable this parameter if not needed. This parameter identifies the path and filename of a file that you can use to signal the Event2 server that it should pause operation. Immediately prior to polling the database, the Event2 server checks for the existence of this file. The default filename is C:\MetaSolv\APPSERVER\suspendEvent2Server. If the file does not exist, the Event2 server polls the database, initiates any work that needs to be done, then waits until the next scheduled poll. If the file does exist, the Event2 server skips polling the database, then waits until the next scheduled poll. The server repeats this process at each scheduled polling interval as long as the file continues to exist.

Caution: While suspended, the PSR End User Billing server cannot process any gateway events.

WorkManagement Parameters

These parameters control operation of the Work Management system task server.

Sample Work Management Section

```
[WorkManagement]
DbPollingInterval=120
MaxThreads=5
SystemTaskQueueMaxCapacity=2000
SuspendCheckInterval=5
```

```
;SystemTasksToInclude=
;SystemTasksToExclude=
```

Work Management Parameter Descriptions

Table A–9 describes the Work Management system task server parameters.

Table A–9 Work Management Parameters

Parameter	Description
DBPollingInterval	**Default**: 120 (seconds) **Valid value**: number between 1second and 2 million seconds This parameter determines the number of seconds which the System Task server process should allow to elapse before it polls the database for new system tasks that should be processed.
SystemTasksToInclude	**Default**: Not enabled **Valid Value**: Any task name that exists in SYSTEM queue. This parameter determines which tasks should be allowed to process from the SYSTEM queue by the System task server.
SystemTasksToExclude	**Default**: Not enabled **Valid Value**: Any task name that exists in SYSTEM queue. This parameter determines which tasks should not be allowed to process from the SYSTEM queue by the System task server.
MaxThreads	**Default**: 5 **Valid value**: 1 to system-specified maximum This parameter specifies the maximum size of the Thread Pool. When a new thread is requested, the Thread Pool checks its size to see if it is less than the MaxThreads parameter. If it is not less than the MaxThreads parameter, the thread pool will wait for a currently executing thread to finish before it creates a new thread.
SuspendFileName	**Default**: /opt2/metasolv/mslv01/appserver/suspendSystemTaskServer **Valid value**: A valid file name. This parameter identifies the path and filename of a file that you can use to signal the System Task server to pause operation. Immediately prior to polling the database, the System Task server checks for the existence of this file. If the file does not exist, the System Task server polls the database, initiates any work that needs to be done, then waits until the next scheduled poll. If the file does exist, the System Task server skips polling the database, then waits until the next scheduled poll. The server repeats this process at each scheduled polling interval as long as the file continues to exist.

JNDI Parameters

This section specifies the parameters used by the MetaSolv Web applications to connect to the EJB server which uses the Java Naming and Directory InterfaceTM (JNDI) to locate the remote MetaSolv EJB. In most cases, nothing should be changed in this section.

JNDI is a mechanism that provides Java applications with a unified interface to multiple naming and directory services.

Note: If you change the password for the administration user or if you change the HTTP port for the server process, on the Oracle WebLogic server management console, you must also make the same changes in the JNDI section.

Sample JNDI Parameters

```
[JNDI]
User=administrator
Password={AES}1824a67ea02612bcbe3169d0b0d185b4
Factory=weblogic.jndi.WLInitialContextFactory
URL=http://srvsunpluto:7001
```

JNDI Parameter Descriptions

Table A–10 describes the JNDI parameters.

Table A–10 JNDI Parameters

Parameter	Description
User	**Default**: Oracle WebLogic server admin user ID (user input from installer) **Valid values**: Oracle WebLogic server admin user ID (user input from installer) This is the administrator user that is created when running the WebLogic domain wizard to create the domain or single server. Change this parameter only if Oracle Global Customer Support recommends that you split your Web and EJB containers.
Password	**Default**: Oracle WebLogic server admin password (user input from installer) **Valid values**: Oracle WebLogic server admin password (user input from installer) This is the administrator password that is created when running the WebLogic domain wizard to create the domain or single server. The password can be changed from the WebLogic Administration Console. The administration server must be restarted after changing the password to enable the managed servers to connect with the new password.

Table A–10 (Cont.) JNDI Parameters

Parameter	Description
Factory	**Default**: weblogic.jndi.WLInitialContextFactory **Valid values**: weblogic.jndi.WLInitialContextFactory The WebLogic factory class used to create the initial context object for accessing the WebLogic naming service. Change this parameter if you had in-house Java developers who subclassed WLInitialContextFactory to customize it. If you are running on an unsupported Oracle WebLogic server, use their context factory. Both of these are very unlikely scenarios and most users should use the default value.
URL	**Default**: t3://localhost:8001 **Valid values**: A valid URL The URL is used to locate the remote EJB container. Change this parameter if you are running the Oracle WebLogic server T3 service on a port other than 8001 or on a remote machine. A port other than 8001 is a common configuration setting. A machine other than localhost should only be used if MetaSolv recommends that you split your Web and EJB containers.

Custom Attribute (CA) Parameters

The CA cache is used by the MetaSolv Solution application to retrieve, render, validate and update CAs. The CA cache consists of CA definitions and rules set up in the MetaSolv Solution Utilities application. Controlling caching provides better performance than directly accessing the database each time a set of CAs are rendered. CA changes made in MetaSolv Solution Utilities are not reflected in MetaSolv Solution until the CA cache has refreshed.

User-defined custom attributes are read and cached on the Oracle WebLogic server during server startup. This process may take several minutes to complete, depending on the amount of CA data.

The frequency of refreshing and caching is controlled by options in the **gateway.ini** file CA parameters. Depending on the amount of custom attribute data, the frequency of the caching refresh process can put an extra load on the database server and Oracle WebLogic server. You should determine how often the cache needs to be refreshed and set the options accordingly.

> **Note:** Oracle recommends daily updates, using the CacheRefreshHourOfDay parameter.

Caching can be completely disabled by setting CacheDisable=true. If disabled, caching will not occur during Oracle WebLogic server startup and will not be refreshed even if CacheDisable is afterwards changed to false.

If CacheRefreshActive is set to true, the CacheRefreshMode parameter determines the frequency, either an hourly interval, once a day at a specific time, or once a week at a specific time.

Sample CA Parameters

```
[CA]
CacheDisable=false
```

```
CacheManagerClass=com.mslv.core.api.internal.ca.cache.broker.CacheManager
CacheRefreshActive=true
CacheRefreshMode=daily
CacheRefreshDayOfWeek=6
CacheRefreshHourOfDay=23
CacheRefreshHourlyInterval=1
```

Using the default values mentioned above, the Oracle WebLogic server refreshes customer attributes daily at the 23rd hour of the day.

CA Parameter Descriptions

Table A–11 describes the CA parameters.

Table A–11 Custom Attribute Parameters

Parameter	Description
CacheDisable	**Default**: false **Valid values**: true, false Determines whether or not the CA cache is built when the Oracle WebLogic server is started. If set to true, the CA cache is not built when the Oracle Robotic server is started.
Cache Manager Class	**Default**: com.mslv.core.api.internal.ca.cache.broker.CacheManager **Valid values**: com.mslv.core.api.internal.ca.cache.broker.CacheManager Do not change this parameter unless requested to do so by Oracle Global Customer Support. This is a java class name.
Cache Refresh Active	**Default**: true **Valid values**: true, false Determines whether or not the CA cache is refreshed on the Oracle Robotic server on a regular interval. If CacheDisable is false and CacheRefreshActive is true, the Oracle WebLogic server will rebuild the CA cache on an interval defined by the CacheRefreshMode parameter.
CacheRefreshMode	**Default**: hourly **Valid values**: hourly, daily, weekly Determines whether the CA cache is refreshed hourly, daily or weekly. If hourly, the cache is refreshed every x hours, where x is defined by the HourlyInterval parameter (Defaults to every hour). If daily, cache is refreshed every day on the x hour, where x is defined by the HourOfDay parameter. (Defaults to 2:00 p.m.). If weekly, cache is refreshed once a week on the day defined by the DayOfWeek parameter and on the hour defined by the HourOfDay parameter. (Defaults to Friday at 2:00 p.m.)
CacheRefreshDayOfWeek	**Default**: 6 **Valid values**: 1 - 7 The day of the week CA cache is refreshed when the CacheRefreshMode is set to **weekly**. Day 1 is Sunday, Day 2 is Monday... Day 7 is Saturday.

Table A–11 (Cont.) Custom Attribute Parameters

Parameter	Description
CacheRefreshHourOfDay	**Default**:23 **Valid values**: 0 - 23 The hour of day CA cache is refreshed when the CacheRefreshMode is set to **daily** or **weekly**. Hour 0 is 12:00 a.m.
CacheRefreshHourlyInterval	**Default**: 1 **Valid values**: 1 - 23 The hourly interval CA cache is refreshed when the CacheRefreshMode is set to **hourly**. **Note:** For **testing purposes only**, if you want to have the cache refresh more frequently than hourly, you can set the HourlyInterval to a value between 300 and 1800. If set to a value of 300 - 1800, the cache handles the value as seconds. In other words, the cache will refresh anywhere between every 5 minutes to every 30 minutes. **Do not set this value to 300 - 1800 in a production environment as it will impact performance**.

Portal Parameters

Users can customize the MetaSolv Solution desktop to display links to portlets. Portlets provide a convenient way to insert connections to other applications into the interface. A portlet uses pre-configured URL and parameters to retrieve the content from an application, then passes the content to the portlet. The default portal shipped with MetaSolv Solution contains several portlets, with the default portlet being the MetaSolv Welcome Page. When the users first logon or if the Customize User Level Portal preference is disabled, users see the default portlet on their desktops. The System Administrator can change the default portlet.

The initial default MetaSolv Solution ID is **asap**. When the MetaSolv Solution is first installed, only the superuser has access to the product, to Security, and to the Default Portal using the superuser ID and password.

If you have changed the DefaultPortalId and you want to upgrade, you must manually change the DefaultPortalId back to asap after the upgrade.

Sample Portal Parameter

```
[Portal]
DefaultPortalId=asap
```

Portal Parameter Description

Table A–12 describes the portal parameter.

Table A–12 Portal Parameter

Parameter	Description
DefaultPortalId	**Default**: ASAP **Valid value**: an existing User ID This ID is used to retrieve a customized portal to be displayed on the My Desktop page when a user logs on to MetaSolv Solution.

Custom Parameters

A custom extension enables you to extend Oracle Communications MetaSolv Solution functionality with additional business logic specific to your organization. In other words, extensions provide the ability to make calls to external systems and to send e-mail and JMS messages at predefined execution points, over and above the functionality supported by the MetaSolv Solution application and APIs.

Sample Custom Parameter

```
[Custom]
CLASSPATH=D:/opt2/metasolv/mslv01/appserver/samples/customExtension;
```

Custom Parameter Description

Table A–13 describes the custom parameter.

Table A–13 Custom Parameter

Parameter	Description
CLASSPATH	**Default**: /opt2/metasolv/mslv01/appserver/samples/customExtension **Valid value**: A valid file name. This parameter identifies the path for the custom extensions directory.

B

Annotated loggingconfig.xml

The following sample logging configuration file controls the described aspects of configuration.

Table B–1 describes the different sections of **loggingconfig.xml**.

Table B–1 Sections of the loggingconfig.xml File

Section	Description and parameters
Message Resourcebundles specification	Defines the resource bundle names and their locations in the JAVA library packages. This section must never be modified unless instructed by Oracle Global Customer Support.
pluggable interface specification	Implementation classes for the logging framework. This section must never be modified unless instructed by Oracle Global Customer Support.
SQL Tracing configuration	Captures SQL debugging messages when JDBCTrace module's debug value is also set to "on." Only one parameter can be changed: <param name="debug" value="off"/>
JDBC Trace	Defines the maximum number of rows that a user is allowed to retrieve and describes the threshold of the "too-many-rows" alert. Two parameters can be changed: <param name="query-results-row-limit" value="32000"/> <param name="too-many-rows-alert-threshold" value="20000"/>
Logging File specification and configuration	Describes whether log file information is appended or written over, log file size, archiving specifications.
Category	Modules and components for which event levels can be set. Valid levels are "fatal," "error," "warn," "info," and "debug" in decreasing order of severity.

Review the boldfaced comments that precede boldfaced parameters in Example B–1 to understand how to change this file when necessary.

Example B–1 The loggingconfig.xml File Text

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- <!DOCTYPE log4j:configuration SYSTEM "log4j.dtd"> -->
<!DOCTYPE mslvlog:configuration SYSTEM "mslvlog.dtd">
<mslvlog:configuration xmlns:mslvlog="http://www.metasolv.com/logging/">
******* START of Message Resourcebundles specification ****
The following section must never be modified.
<!-- The following defines the resource bundle names and their locations
```

```
       in the java library packages. The resource bundles are used for
       looking up detail, cause and action element values for the given
       message key.
   -->
    <mslv-message-repositories>
<message-bundle category="log">
            <param name="MessageBundleName"
value="com.mslv.core.global.ResourceBundles.error.ErrorDescriptions"/>
        </message-bundle>
<message-bundle category="attr">
            <param name="MessageBundleName" value="MSLVAttrMessages"/>
        </message-bundle>
<message-bundle category="key">
            <param name="ExtensionBundleName" value="MSLVKeyExtensions"/>
        </message-bundle>
</mslv-message-repositories>
******** END of Message Resourcebundles specification.
******* Start of pluggable interface specification ****
******* Never needs to modified by the user ****
<!--
         The following defines various pluggable implementation classes
        for the logging framework. One can replace the default ones
        provided here, with their own implementation classes.
   -->
    <mslv-log-api>
         <param name="logContextFactoryClassName"
value="com.metasolv.common.framework.logging.MSLVLogContextFactoryDefaultImpl"/>
         <param name="logContextClassName"
value="com.metasolv.common.framework.logging.MSLVLogContextDefaultImpl"/>
         <param name="logMessageContextClassName"
value="com.metasolv.common.framework.logging.MSLVMessageContextDefaultImpl"/>
         <param name="logAppContextClassName"
value="com.metasolv.common.framework.logging.MSLVAppContextDefaultImpl"/>
<!--
You can add your own custom severity level here. This is shown as an example. The
values supplied here must not re-specify the values of any levels in
org.apache.log4j.Priority class.
         -->
         <param name="alertLevelIntValue" value="60000"/>
         <param name="perfLevelIntValue" value="25000"/>
    </mslv-log-api>
******* End of pluggable interface specification ****
******** Start of SQL Tracing configuration *******
    <!-- In the future, this could also define a timeout attribute -->
    <mslv-apps>
         <db-server-trace>
               <param name="SCHINTAL" value="off"/>
         </db-server-trace>
         <jdbc-trace>
// on/off. Must also change cmm.JDBCTrace module's debug value to
       "on", when this is "on" to get full SQL debugging messages.
<param name="debug" value="off"/>
<param name="query-results-row-limit" value="32000"/>
<param name="too-many-rows-alert-threshold" value="20000"/>
</jdbc-trace>
    </mslv-apps>
******** end of SQL Tracing configuration

<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
```

```
******* Start of Logging File specification and configuration ****

  <appender name="XMLFileApp" class="org.apache.log4j.RollingFileAppender">
      <param name="ImmediateFlush" value="true"/>
**** true: means when App Server is restarted, the existing files will NOT be
overwritten. The new content will be appended.
If it is false:, each appserver is restarted, the active logging file will be
truncated to zero length (emptied). We recommend leaving this
as 'true.'
      <param name="append" value="true"/>
**** Max file size, before a rollover for archiving takes place.
      <param name="MaxFileSize" value="10000KB"/>
***** How many files to keep in the backup. Keep this at least 1.
      <param name="MaxBackupIndex" value="10"/>
****** The main logging file. Archives will be named with a .1, .2..., .10
appended to this. We don't recommend changing this filename or its location.
      <param name="File" value="${mslv.log.dir}/appserverlog.xml"/>
      <layout
class="com.metasolv.common.framework.logging.api.log4jext.xml.MSLVXMLLayout">
      </layout>
  </appender>

  <appender name="AuditFileApp" class="org.apache.log4j.RollingFileAppender">
      <param name="ImmediateFlush" value="true"/>
      <param name="append" value="true"/>
      <param name="MaxFileSize" value="10000KB"/>
      <param name="MaxBackupIndex" value="10"/>

      <param name="File" value="${mslv.log.dir}/appserver_auditlog.xml"/>
      <layout
class="com.metasolv.common.framework.logging.api.log4jext.xml.MSLVXMLLayout">
      </layout>
  </appender>

  <appender name="ConsoleApp" class="org.apache.log4j.ConsoleAppender">
      <param name="ImmediateFlush" value="true"/>
      <param name="Threshold" value="debug"/>
      <param name="Target" value="System.out"/>
      <layout
class="com.metasolv.common.framework.logging.api.log4jext.xml.MSLVXMLLayout">
      </layout>
  </appender>

  <appender name="MinimalConsoleApp" class="org.apache.log4j.ConsoleAppender">
      <param name="ImmediateFlush" value="true"/>
      <param name="Threshold" value="debug"/>
      <param name="Target" value="System.out"/>
      <layout class="org.apache.log4j.PatternLayout">
           <param name="ConversionPattern" value="%-5p - %-15c - %m%n"/>
      </layout>
  </appender>

  <appender name="WeblogicLogApp"
class="com.metasolv.common.framework.logging.api.log4jext.WeblogicAppender">
      <layout class="org.apache.log4j.PatternLayout">
           <param name="ConversionPattern" value="%-5p - %-15c - %m%n"/>
      </layout>
  </appender>

  <appender name="XMLFileAppNew" class="org.apache.log4j.RollingFileAppender">
```

```
      <param name="ImmediateFlush" value="true"/>
      <param name="append" value="true"/>
      <param name="MaxFileSize" value="10000KB"/>
      <param name="MaxBackupIndex" value="10"/>
      <param name="File" value="${mslv.log.dir}/appserverlog_misc.xml"/>
      <layout
class="com.metasolv.common.framework.logging.api.log4jext.xml.MSLVXMLLayout">
      </layout>
  </appender>

******* End of Logging File Specification and configuration

<!-- Asynchronous appender to receive events, buffer them and log
        them to WebLogic Logging Module in a non-blocking manner.
        If you have multiple slow appenders, then define wrap each
        appender inside another unique async appender to log asynchronously.
   -->
   <appender name="MSLVWLAsyncApp"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVAsyncAppender">
      <appender-ref ref="XMLFileApp"/>
      <appender-ref ref="WeblogicLogApp"/>
   </appender>

  <!--
     This is the single receiver category of all the Logging messages in this JVM.
     Setting severity level on this, will affect what gets logged to the
     variety of destination branching off from this category, such as
     XMLFileApp (appserverlog.xml).
  -->
  <category name="mls"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value="debug"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
      <appender-ref ref="XMLFileApp"/>
    <!-- If you are going to use a slow appender like WeblogicLogApp, then
         wrap all appenders inside a MSLVAsyncApp and use it here.
      <appender-ref ref="MSLVAsyncApp"/>
     -->
</category>

  <!--
     This is an example of how to hierarchically override the super receiver
     category with a child category.
     Setting severity level on this, will affect what gets logged to the
     variety of destination branching off from this category, such as
     XMLFileApp (appserverlog.xml).
  -->
  <category name="mls.cmm.TEST"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
      <appender-ref ref="XMLFileApp"/>
    <!-- If you are going to use a slow appender like WeblogicLogApp, then
         wrap all appenders inside a MSLVAsyncApp and use it here.
      <appender-ref ref="MSLVAsyncApp"/>
     -->
</category>
```

```
  <!--
Category definitions, one for each logging module.
  -->
  <category name="cmm.appserver"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
****** An error level can be re-defined for each module.
****** Valid values are: fatal, error, warn, info, debug, in the decreasing order
of severity. We recommend them leaving these at the shown values. On the other
hand, turning a level down to info/debug level could have adverse impact on the
performance of the system, as verbose logging will take place.
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.ASR"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
  <category name="cmm.CA"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.CaCache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.CONVERSION"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.DLR"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.DLR_Equipment"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.DLR_NGN_Activation"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
```

```
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.DLR_Circuit"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.EventServer"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.Event2Server"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.Framework"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
</category>
<category name="cmm.Framework.Cache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
</category>
<category name="cmm.GLR"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.invFromPB"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.Infrastructure"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
```

```
<category name="cmm.Infrastructure_NetworkLocation"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.INTEGRATION_CONVERSION"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.WORK_MANAGEMENT_CONVERSION"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.INTEGRATION_EVENT_SERVER"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.INTEGRATION_MANAGER"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
***** This is the other parameter that needs to be tuned for **** getting the SQL
Logging going. Change the level to debug to get the logging going, in addition to
changing it at the top of this file, for the parameter named jdbc-trace
  <category name="cmm.JDBCTrace"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.LSR"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.MSLVSessionBean"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.NetworkManagement"
```

```
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.NetProv"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.NetProv_Plant"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.NetProv_Activation"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.NI"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.PSR"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.E911"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.CNAM"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.LIDB"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
```

```
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.PSRAncillary"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.PSREUB"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.Security"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="debug"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="AuditFileApp"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.SYSTEM"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.TEST"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="debug"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.TimedEventProcess"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.TMS"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.TroubleBatchWorkerThread"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.KeepAliveTroubleBatch"
```

```
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.WDI"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.WM"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="cmm.WM_TskSv"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="com.mslv.webapp.context.action.MSLVRequestProcessor"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache.struts.util.PropertyMessageResources"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<!-- Tiles stuff from NavBar -->
  <category name="org.apache.struts.tiles.TilesRequestProcessor"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<!-- JCS stuff -->
  <category name="org.apache.jcs.engine.control.CompositeCacheConfigurator"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
```

```
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache.jcs.engine.memory.lru.LRUMemoryCache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache.jcs.engine.control.CompositeCache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache.jcs.config.OptionConverter"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache.jcs.auxillary.disk.indexed.IndexedDiskCache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<category name="org.apache.jcs.auxillary.javagroups.JavaGroupsCache"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLogger"
additivity="false">
    <level value ="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileApp"/>
  </category>
<root>
    <level value="error"
class="com.metasolv.common.framework.logging.api.log4jext.MSLVLevel"/>
    <appender-ref ref="XMLFileAppNew"/>
  </root>
<!--
  <root>
    <level value="error" class="org.apache.log4j.Level"/>
    <appender-ref ref="XMLFileApp"/>
  </root>
-->
</log4j:configuration>
</mslvlog:configuration>
```

C

Oracle WebLogic Server Configuration Settings

This section contains Oracle WebLogic Server configuration setting recommendations for three sizes of deployment configurations:

- Small (100 users)
- Medium (300 to 400 users)
- Large (1000 users)

These recommendations are based on a set of mixed scenarios.

Use this information when tuning the Oracle WebLogic server. When you make changes, do so in small increments so that you can be sure of the effects without significant negative results.

> **Note:** These recommendations are subject to change pending the results of benchmark testing and feedback on customer usage.

See *MetaSolv Solution Planning Guide* for more information on the recommended hardware for these deployments.

Small Deployment Administration Server

Table C–1 lists the recommended configuration settings for a small deployment administration server.

Table C–1 Small Deployment Administration Server Configuration Settings

Name	Parameter	Value	Parameter	Value
Oracle WebLogic 12.2.1.2	Domain Name	mslvadmin	NA	NA
JDK	Version	8 (with patches)	NA	NA
JVM	-Xms	1024m (32-bit Windows) 1024m (64-bit Windows) 8g (64-bit non-Windows)	-Xmx	1024m (32-bit Windows) 4g (64-bit Windows) 8g (64-bit non-Windows)
JVM	-XX:NewSize	256m	-XX:MaxNewSize	256m
JVM	-server	NA	NA	NA

Table C–1 (Cont.) Small Deployment Administration Server Configuration Settings

Name	Parameter	Value	Parameter	Value
JVM	MAXMETASPACESIZE	386m	NA	NA
Deployment	STARTMODE	TRUE	NA	NA
Application Backlog	50	NA	NA	NA
JDBC (MSLVPool)	Initial Capacity	15	Max Capacity	80
JDBC (MSLVPool)	Capacity Increment	1	Manual Shrinking	NA
JDBC (MSLVDBTracePool)	Initial Capacity	0	Max Capacity	5
JDBC (MSLVDBTracePool)	Capacity Increment	1	Manual Shrinking	NA
JTA	Timeout Seconds	360	NA	NA
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
MSLVDomain Logging	Number Of Files Limited	Check box selected	File Count	10
App Server Logging	Redirect stdout logging enabled	Check box cleared	NA	NA
App Server Logging	Rotation Type	By Size	Rotation File Size	10000
App Server Logging	Limit number of retained files	Check box selected	Files to retain	10
App Server Logging	Domain log broadcaster	Off	HTTP access log file enabled	Check box cleared

Small Deployment Clustered Administration Server

Table C–2 lists the recommended settings for a small deployment clustered administration server.

Table C–2 Small Deployment Clustered Administration Server Settings

Name	Parameter	Value	Parameter	Value
Oracle WebLogic 12.2.1.2	Server Names	Server1, Server2	Server Port	7063, 7063
JDK	Version	8 (with patches)	NA	NA
JVM	-Xms	1024m (32-bit Windows) 1024m (64-bit Windows) 8g (64-bit non-Windows)	-Xmx	1024m (32-bit Windows) 4g (64-bit Windows) 8g (64-bit non-Windows)
JVM	-XX:NewSize	256m	-XX:MaxNewSize	256m
JVM	-server	NA	NA	NA
JVM	MAXMETASPACESIZE	386m	NA	NA
Deployment	STARTMODE	TRUE	NA	NA

Table C–2 (Cont.) Small Deployment Clustered Administration Server Settings

Name	Parameter	Value	Parameter	Value
Application Backlog	200	NA	NA	NA
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
MSLVDomain Logging	Number Of Files Limited	Check box selected	File Count	10
AppServer Logging	Redirect stdout logging enabled	Check box cleared	NA	NA
AppServer Logging	Rotation Type	By Size	Rotation File Size	10000
AppServer Logging	Limit number of retained files	Check box selected	Files to retain	10
AppServer Logging	Domain log broadcaster	Off	HTTP access log file enabled	Check box cleared
Cluster	Name	Cluster280R	MULTICAST IP	239.192.0.0
Cluster	Cluster Address	Server1:7063, Server2:7063	MULTICAST PORT	6060
Cluster	WebLogic Plug-In Enabled	Check box cleared	NA	NA

Medium Deployment Administration Server

Table C–3 lists the recommended configuration settings for a medium deployment administration server.

Table C–3 Medium Deployment Server Configuration Settings

Name	Parameter	Value	Parameter	Value
Oracle WebLogic 12.2.1.2	Domain Name	mslvadomain	Server Name	mslvadmin
JDK	Version	8 (with patches)	NA	NA
JVM	-Xms	1024m (32-bit Windows) 1024m (64-bit Windows) 8g (64-bit non-Windows)	-Xmx	1024m (32-bit Windows) 4g (64-bit Windows) 8g (64-bit non-Windows)
JVM	-XX:NewSize	256m	-XX:MaxNewSize	256m
JVM	-server	NA	NA	NA
JVM	MAXMETASPACESIZE	386m	NA	NA
Deployment	STARTMODE	TRUE	NA	NA
Application Backlog	50	NA	NA	NA
JDBC (MSLVPool)	Initial Capacity	15	Max Capacity	400
JDBC (MSLVPool)	Capacity Increment	1	Manual Shrinking	NA
JDBC (MSLVDBTracePool)	Initial Capacity	0	Max Capacity	5
JDBC (MSLVDBTracePoo	Capacity Increment	1	Manual shrinking	NA
JTA	Timeout Seconds	360	NA	NA

Table C–3 (Cont.) Medium Deployment Server Configuration Settings

Name	Parameter	Value	Parameter	Value
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
MSLVDomain Logging	Number Of Files Limited	Check box selected	File Count	10
App Server Logging	Redirect stdout logging enabled	Check box cleared	NA	NA
App Server Logging	Rotation Type	By Size	Rotation File Size	10000
App Server Logging	Limit number of retained files	Check box selected	Files to retain	10
App Server Logging	Domain log broadcaster	Off	HTTP access log file enabled	Check box cleared

Medium Deployment Clustered Server

Table C–4 lists the setup conventions for a medium deployment clustered server.

Table C–4 Medium Deployment Clustered Server Setup Conventions

Clustered Server Name	Machine Name	Port Number	Replication Group Primary	Replication Group Secondary
Server11	Machine1	7063	G1	G2
Server12	Machine1	9063	G1	G2
Server21	Machine2	7063	G2	G3
Server22	Machine2	9063	G2	G3
Server31	Machine3	7063	G3	G4
Server32	Machine3	9063	G3	G4
Server41	Machine4	7063	G4	G1
Server42	Machine4	9063	G4	G1

Medium Deployment Clustered Server Configuration

Table C–5 lists the recommended configuration settings for a medium deployment clustered server.

Table C–5 Medium Deployment Clustered Server Configuration Settings

Name	Parameter	Value	Parameter	Value
Oracle WebLogic 12.2.1.2	Domain Name	mslvdomain	NA	NA
JDK	Version	8 (with patches)	NA	NA
JVM	-Xms	1024m (32-bit Windows) 1024m (64-bit Windows) 8g (64-bit non-Windows)	-Xmx	1024m (32-bit Windows) 4g (64-bit Windows) 8g (64-bit non-Windows)
JVM	-XX:NewSize	256m	-XX:MaxNewSize	256m
JVM	-server	NA	NA	NA

Table C–5 (Cont.) Medium Deployment Clustered Server Configuration Settings

Name	Parameter	Value	Parameter	Value
JVM	MAXMETASPACESIZE	386m	NA	NA
Deployment	STARTMODE	TRUE	NA	NA
Application Backlog	200	NA	NA	NA
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
MSLVDomain Logging	Number Of Files Limited	Check box selected	File Count	10
AppServer Logging	Redirect stdout logging enabled	Check box cleared	NA	NA
AppServer Logging	Rotation Type	By Size	Rotation File Size	10000
AppServer Logging	Limit number of retained files	Check box selected	Files to retain	10
AppServer Logging	Domain log broadcaster	Off	HTTP access log file enabled	Check box cleared
Cluster	Name	ClusterV480	MULTICAST IP	239.192.0.1
Cluster	Cluster Address	Server11:7063 Server12:9063 Server21:7063 Server22:9063 Server31:7063 Server32:9063 Server41:7063 Server42:9063	MULTICAST PORT	6061
Cluster	WebLogic Plug-In Enabled	Check box cleared	NA	NA

Large Deployment Administration Server

Table C–6 lists the recommended configuration settings for a large deployment administration server.

Table C–6 Large Deployment Administration Server Configuration Settings

Name	Parameter	Value	Parameter	Value
Oracle WebLogic 12.2.1.2	Domain Name	mslvadomain	Server Name	mslvadmin
JDK	Version	8 (with patches)	NA	NA
JVM	-Xms	1024m (32-bit Windows) 1024m (64-bit Windows) 8g (64-bit non-Windows)	-Xmx	1024m (32-bit Windows) 4g (64-bit Windows) 8g (64-bit non-Windows)
JVM	-XX:NewSize	256m	-XX:MaxNewSize	256m
JVM	-server	NA	NA	NA
JVM	MAXMETASPACE SIZE	386m	NA	NA
Deployment	STARTMODE	TRUE	NA	NA
Application Backlog	50	NA	NA	NA
JDBC (MSLVPool)	Initial Capacity	15	Max Capacity	800
JDBC (MSLVPool)	Capacity Increment	1	Manual Shrinking	NA
JDBC (MSLVDBTracePool)	Initial Capacity	0	Max Capacity	5
JDBC (MSLVDBTracePool)	Capacity Increment	1	Manual Shrinking	NA
JTA	Timeout Seconds	360	NA	NA
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
MSLVDomain Logging	Number Of Files Limited	Check box selected	File Count	10
App Server Logging	Redirect stdout logging enabled	Check box cleared	NA	NA
App Server Logging	Rotation Type	By Size	Rotation File Size	10000
App Server Logging	Limit number of retained files	Check box selected	Files to retain	10
App Server Logging	Domain log broadcaster	Off	HTTP access log file enabled	Check box cleared
Web Application	Files Reload Period (for all)	-1	NA	NA

Large Deployment Clustered Server

Table C–7 lists the setup conventions for a large deployment clustered server.

Table C–7 Large deployment clustered server setup conventions

Clustered Server Name	Machine Name	Port Number	Replication Group Primary	Replication Group Secondary
Server11	Machine1	6063	G1	G2
Server12	Machine1	7063	G1	G2
Server13	Machine1	8063	G1	G2
Server14	Machine1	9063	G1	G2
Server21	Machine2	6063	G2	G3
Server22	Machine2	7063	G2	G3
Server23	Machine2	8063	G2	G3
Server24	Machine2	9063	G2	G3
Server31	Machine3	6063	G3	G4
Server32	Machine3	7063	G3	G4
Server33	Machine3	8063	G3	G4
Server34	Machine3	9063	G3	G4
Server41	Machine4	6063	G4	G1
Server42	Machine4	7063	G4	G1
Server43	Machine4	8063	G4	G1
Server44	Machine4	9063	G4	G1

Large Deployment Clustered Server Configuration

Table C–8 lists the recommended configuration settings for a large deployment clustered server.

Table C–8 Large Deployment Clustered Server Configuration Settings

Name	Parameter	Value	Parameter	Value
Oracle WebLogic 12.2.1.2	Domain Name	mslvdomain	NA	NA
JDK	Version	8 (with patches)	NA	NA
JVM	-Xms	1024m (32-bit Windows) 1024m (64-bit Windows) 8g (64-bit non-Windows)	-Xmx	1024m (32-bit Windows) 4g (64-bit Windows) 8g (64-bit non-Windows)
JVM	-XX:NewSize	256m	-XX:MaxNewSize	256m
JVM	-server	NA	NA	NA
JVM	MAXMETASPACESIZE	386m	NA	NA
Deployment	STARTMODE	TRUE	NA	NA
Application Backlog	200	NA	NA	NA
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000

Table C–8 (Cont.) Large Deployment Clustered Server Configuration Settings

Name	Parameter	Value	Parameter	Value
MSLVDomain Logging	Number Of Files Limited	Check box selected	File Count	10
AppServer Logging	Redirect stdout logging enabled	Check box cleared	NA	NA
AppServer Logging	Rotation Type	By Size	Rotation File Size	10000
AppServer Logging	Limit number of retained files	Check box selected	Files to retain	10
AppServer Logging	Domain log broadcaster	Off	HTTP access log file enabled	Check box cleared
Cluster	Name	ClusterV880	MULTICAST IP	239.192.0.2
Cluster	Cluster Address	Server11:6063 Server12:7063 Server13:8063 Server14:9063 Server21:6063 Server22:7063 Server23:8063 Server24:9063 Server31:6063 Server32:7063 Server33:8063 Server34:9063 Server41:6063 Server42:7063 Server43:8063 Server44:9063	MULTICAST PORT	6062
Cluster	WebLogic Plug-In Enabled	Check box cleared	NA	NA

Glossary

The glossary contains a list of terms and their definitions as they relate to Oracle Communications MetaSolv Solution (MSS) documentation:

Access Carrier Name Abbreviation (ACNA)

A three-character abbreviation assigned by Telcordia to each Interexchange Carrier (IXC) and listed in the Local Exchange Routing Guide (LERG).

This abbreviation represents the access customer name to which the exchange carrier renders the access bill.

Access Customer Terminal Location (ACTL)

The COMMON LANGUAGE Location Identifier (CLLI) code of the Inter-Local Access Transport Area (InterLATA) facility terminal location of the access customer providing service.

ACNA (Access Carrier Name Abbreviation)

A three-character abbreviation assigned by Telcordia to each Interexchange Carrier (IXC) and listed in the Local Exchange Routing Guide (LERG).

This abbreviation represents the access customer name to which the exchange carrier renders the access bill.

ACTL (Access Customer Terminal Location)

The COMMON LANGUAGE Location Identifier (CLLI) code of the Inter-Local Access Transport Area (InterLATA) facility terminal location of the access customer providing service.

AID (Access Identifier)

Identifies the port address on a piece of equipment within the network element identified by the target identifier (TID). In the database, the AID information is stored as the concatenated node address for the port address to which the circuit is assigned.

API (Application Programming Interface)

Software that permits other applications to access a specific area of data in the database.

Application Programming Interface (API)

Software that permits other applications to access a specific area of data in the MetaSolv Solution database.

Asynchronous Operations

Operations in which control returns to the invoking application before the operation is acted upon. The invoked application returns the results to the calling application using a callback mechanism after the operation has been completed.

Asynchronous Transfer Mode (ATM)

A high bandwidth, low delay, packet-like switching and multiplexing technique.

ATM (Asynchronous Transfer Mode)

A high bandwidth, low delay, packet-like switching and multiplexing technique.

Backup

The hardware and software resources available to recover data after a degradation or failure of one or more system components.

A copy of computer data on an external storage modem, such as floppy disk or tape.

Bandwidth

A term used in various areas of the telecommunications industry (such as with facilities, SONET, Frame Relay, and ATM). In a channelized environment, (such as with facilities and SONET), the circuit positions used in the MetaSolv Solution act as the discrete means of providing “bandwidth." The term “allocation of bandwidth" is also used in the industry. In the MetaSolv Solution, “bandwidth" refers to a virtual circuit being “allocated" to bandwidth circuits through the Bandwidth Allocation table based on bit rates of each circuit rather than by a specific number of circuit positions (such as channels).

Bandwidth Circuits

In PVC (Permanent Virtual Circuit), bandwidth circuits are circuits that have virtual circuits assigned to them and have allocated capacity based on the digital bit rate as opposed to the method of using a distinct number of circuit positions (channels).

Batch Processing

A mode of computer operation in which a complete program or set of instructions is carried out from start to finish without any intervention from a user. Batch processing is a highly efficient way of using computer resources, but it does not allow for any input while the batch is running, or any corrections in the event of a flaw in the program or a system failure. For these reasons, it is primarily used for CPU-intensive tasks that are well established and can run reliably without supervision, often at night or on weekends when other demands on the system are low.

CAB (Carrier Access Billing)

A system that bills Interexchange Carriers (IXCs) for access time and hardware purchases.

Carrier

A company that provides communications circuits. There are two types of carriers: private and common. Private carriers are not regulated and they can refuse to provide you service. Common carriers are regulated and they cannot refuse to provide you service. Most carriers (for example, MCI, AT&T, and Sprint) are common carriers.

Carrier Access Billing (CAB)

A system that bills Interexchange Carriers (IXCs) for access time and hardware purchases.

CCNA (Customer Carrier Name Abbreviation)

A Telcordia-maintained industry-standard code used to identify access customers (for example, AT&T and MCI).

Cell Relay Service (CRS)

An asynchronous transfer mode (ATM) term; a carrier service which supports the receipt and transmission of ATM cells between end-users in compliance with ATM standards and implementation specifications.

CLEI (Common Language Equipment Identifier)

Codes assigned by Telcordia (formerly Bellcore) to provide a standard method of identifying telecommunications equipment in a uniform, feature-oriented language. The code is a text/barcode label on the front of all equipment installed at Regional Bell Operating Company (RBOC) facilities that facilitates inventory, maintenance, planning, investment tracking, and circuit maintenance processes. Suppliers of telecommunication equipment give Telcordia technical data on their equipment, and Telcordia assigns a CLEI code to that specific product.

CNAM

CNAM is an acronym for:

- Call Name Database (Sprint)
- Calling Name (Caller ID)
- Class Calling Name Delivery (Telcordia)

CBP (Convergent Billing Platform)

Allows for the bundling of services, such as long distance, cellular, paging, and cable, together onto a single monthly invoice.

COM (COMbined file)

A combined file used by the ASR/ISI Gateway for transporting multiple types of files. A COM file may contain various combinations of ASR Response files and ASR Error files.

Commit

The final step in the successful completion of a previously started database change. The commit saves any pending changes to the database.

Common Language Equipment Identifier (CLEI)

Codes assigned by Telcordia (formerly Bellcore) to provide a standard method of identifying telecommunications equipment in a uniform, feature-oriented language. The code is a text/barcode label on the front of all equipment installed at Regional Bell Operating Company (RBOC) facilities that facilitates inventory, maintenance, planning, investment tracking, and circuit maintenance processes. Suppliers of telecommunication equipment give Telcordia technical data on their equipment, and Telcordia assigns a CLEI code to that specific product.

Common Object Request Broker Architecture (CORBA)

A standard architecture that allows different applications to communicate and exchange commands and data.

A central element in CORBA is the Object Request Broker (ORB). An ORB makes it possible for a client object to make a server request without having to know where in a network the server object or component is located and exactly what its interfaces are.

Concatenate

To allocate contiguous bandwidth for transport of a payload associated with a super-rate service. The set of bits in the payload is treated as a single entity, as opposed to being treated as separate bits, bytes or time slots. The payload, therefore, is accepted, multiplexed, switched, transported and delivered as a single, continuous chunk of payload data.

Convergent Billing Platform (CBP)

Allows for the bundling of services, such as long distance, cellular, paging, and cable, together onto a single monthly invoice.

CORBA (Common Object Request Broker Architecture)

A standard architecture that allows different applications to communicate and exchange commands and data.

A central element in CORBA is the Object Request Broker (ORB). An ORB makes it possible for a client object to make a server request without having to know where in a network the server object or component is located and exactly what its interfaces are.

Cross-Connect

A way of connecting two objects together. Cross-connects may be hard-wired or software based. Hard-wired cross-connects are used to connect two pieces of equipment using a physical media. Software cross-connects represent the connections made within a network node. The software cross-connect determines how a circuit is connected through an intelligent network element.

CRS (Cell Relay Service)

An asynchronous transfer mode (ATM) term; a carrier service which supports the receipt and transmission of ATM cells between end-users in compliance with ATM standards and implementation specifications.

Customer Carrier Name Abbreviation (CCNA)

A Telcordia-maintained industry-standard code used to identify access customers (for example, AT&T and MCI).

DACS (Digital Access and Cross-Connect Systems)

AT&T's proprietary digital cross-connect system (DCS) product. DCS is a type of switching/multiplexing equipment that permits per-channel DS0 electronic cross-connects from one T1 transmission facility to another, directly from the DS1 signal. That is, the DCS allows the 24 DS0 channels in one T1 line to be distributed among any of the other T1 lines connected to the DCS, without requiring external cross-connects.

Daemon

A program that runs continuously and exists for the purpose of handling periodic service requests that a computer system expects to receive. The daemon program forwards the requests to other programs (or processes) as appropriate.

Dedicated Plant

Describes a method used to build a telephone company's facilities. It is used when designated equipment, cables, and cable pairs are to be connected specifically to other pieces of equipment or locations. Once those connections are made they are seldom changed.

Design Layout Report (DLR)

A form designed according to the Industry Support Interface (ISI) standard originated by the Ordering and Billing Forum (OBF) committee. This form contains pertinent technical information sent to the access customer for review to ensure that the appropriate design has been provided and for the recording of its contents for future circuit activities. For the MetaSolv Solution system, this entity type and its dependents are used to record when the DLR was issued and to make the necessary changes to defaulted ASR values.

Digital Access and Cross-Connect Systems (DACS)

AT&T's proprietary digital cross-connect system (DCS) product. DCS is a type of switching/multiplexing equipment that permits per-channel DS0 electronic cross-connects from one T1 transmission facility to another, directly from the DS1 signal. That is, the DCS allows the 24 DS0 channels in one T1 line to be distributed among any of the other T1 lines connected to the DCS, without requiring external cross-connects.

DLR (Design Layout Report)

A form designed according to the Industry Support Interface (ISI) standard originated by the Ordering and Billing Forum (OBF) committee. This form contains pertinent technical information sent to the access customer for review to ensure that the appropriate design has been provided and for the recording of its contents for future circuit activities. For the MetaSolv Solution system, this entity type and its dependents are used to record when the DLR was issued and to make the necessary changes to defaulted ASR values.

EC (exchange carrier)

A company providing telecommunication in a licensed area.

ECCKT (Exchange Carrier Circuit Identification)

An AP Circuit ID or multiple circuit Ids.

End User

A customer who uses (rather than provides) telecommunications services.

End User Location

The terminating location of telephone services for residential and business customers.

Equipment Specs

Documents that identify the properties and functionality of a piece of hardware. Equipment Specs are limited to items relevant to the operation of a circuit, such as channel banks, channel units, VF equipment, switches, cards, and so on.

Escalation

The process of elevating a trouble ticket and making the appropriate parties aware that the resolution of the ticket is not progressing as well as expected and that assistance may be needed.

Escalation Method

The type of outage that has prompted a trouble ticket.

Event

In the scope of the APIs, an event represents the occurrence of something in the MetaSolv Solution or in a third-party application that is of significance to the gateway.

Exchange Carrier (EC)

A company providing telecommunication in a licensed area.

Exchange Carrier Circuit Identification (ECCKT)

An AP Circuit ID or multiple circuit Ids.

Facility

Any one of the elements of a physical telephone plant required to provide service (for example, a phone or data line, switching system, or cables and microwave radio transmission systems).

Fault Management

Detects, isolates, and corrects network faults. It is also one of five categories of network management defined by the ISO (International Standards Union).

Fixed Length Records

A set of data records all having the same number of characters.

Flow-Through Provisioning

The automating of the activation process used to remotely communicate with the equipment in the field through Work Management tasks. The MetaSolv Solution itself can act as the Service Management Layer (SML) that sends commands to the Network Management Layer (NML) where the commands are non-vendor specific. The NML then passes these commands and translates them into vendor terms and communicates these to the specific Network Element (NE), which is the actual equipment in the field. Examples of Network Elements are C.O. switch, Digital Loop Carrier (DLC), SONET node, and Digital Cross-connect System (DCS). The MetaSolv Solution may also serve as the NML.

FOC (Form Order Confirmation)

A form the Local Exchange Carrier (LEC) submits to the Interexchange Carrier (IXC) to indicate the date when they are going to install ordered circuits.

Form Order Confirmation (FOC)

A form the Local Exchange Carrier (LEC) submits to the Interexchange Carrier (IXC) to indicate the date when they are going to install ordered circuits.

Frame Relay

A telecommunication service designed for cost-efficient data transmission for intermittent traffic between local area networks (LANs) and between end-points in a wide area network (WAN).

Header Record

The portion of a message containing information that guides the message to the correct destination. The header includes the sender's address, the receiver's address, the precedence level, routing instructions, synchronization pulses, etc.

ICSC (Interexchange Customer Service Center)

The telephone company's primary point of contact for handling the service needs of all long distance carriers. This center is responsible for outlining, configuring, and installing basic service upon customer request.

IDL (Interface Definition Language)

A programming language that helps define interfaces. IDL is inherently object oriented in nature.

IFR (Interface Repository)

A component of ORB that provides persistent storage of the interface definitions, acting as an online database and managing and providing access to a collection of object definitions.

INI file

An application-specific file that contains information about the initial configuration of the application.

Interconnection Interface

Using an API, the MetaSolv Solution can be tightly integrated with a customer's proprietary software using software developed by third-party vendors like TMForum Common Interconnection Gateway Platform (CIGP).

Interexchange Customer Service Center (ICSC)

The telephone company's primary point of contact for handling the service needs of all long distance carriers. This center is responsible for outlining, configuring, and installing basic service upon customer request.

Interface

A mechanical or electrical link connecting two or more pieces of equipment. An interface allows an independent system to interact with the MetaSolv Solution product family.

In this guide, the term interface refers to the CORBA IDL interface that describes the operations the interface object supports in a distributed application. These IDL definitions provide the information needed by clients for accessing objects across a network.

Interface Architecture

The collection of APIs and gateway integration software produced by Oracle to permit access to the database.

Interface Definition Language (IDL)

A programming language that helps define interfaces. IDL is inherently object oriented in nature.

Interface Repository (IFR)

A component of ORB that provides persistent storage of the interface definitions, acting as an online database and managing and providing access to a collection of object definitions.

International Standards Organization (ISO)

An international standards-setting organization.

Internet Service Provider (ISP)

A company that provides individuals and other companies access to the Internet and other related services such as web site building and hosting.

ISO (International Standards Organization)

An international standards-setting organization.

ISP (Internet Service Provider)

A company that provides individuals and other companies access to the Internet and other related services such as web site building and hosting.

Item Types

Predefined types which can be used to build product specifications. Relationships between the item types are also predefined; the item types and relationships together are commonly called the MetaSolv Rules. The MetaSolv Solution only allows product specifications to be built that follow the MetaSolv Rules. These rules allow specific processing to be applied to item types.

Java Database Connectivity (JDBC)

An application program interface (API) specification for connecting programs written in Java to the data in popular databases.

JDBC (Java Database Connectivity)

An application program interface (API) specification for connecting programs written in Java to the data in popular databases.

LATA (Local Access Transport Area)

One of 161 geographical areas in the United States within which a local telephone company may offer local or long distance telecommunications service.

The LATA identifies which exchange carrier or Interexchange Carrier (IXC) may provide service in a defined area.

LIDB (Line Information Database)

A service that provides customers the ability to query Access Provider (AP) databases to determine whether a:

- Caller is the authorized user of a valid AP calling card.
- Particular telephone number can accept collect or third-party billed calls before transmitting any call.

Line Information Database (LIDB)

A service that provides customers the ability to query Access Provider (AP) databases to determine whether a:

- Caller is the authorized user of a valid AP calling card.
- Particular telephone number can accept collect or third-party billed calls before transmitting any call.

LNP (Local Number Portability)

A circuit-switched network capability that allows an end user to change service providers without having to change telephone numbers.

Local Access Transport Area (LATA)

One of 161 geographical areas in the United States within which a local telephone company may offer local or long distance telecommunications service.

The LATA identifies which exchange carrier or Interexchange Carrier (IXC) may provide service in a defined area.

Local Number Portability (LNP)

A circuit-switched network capability that allows an end user to change service providers without having to change telephone numbers.

Local Service Ordering Guidelines (LSOG)

A standardized set of guidelines used for ordering various local services. The local service request (LSR) is the administrative form that must accompany any local service request. This type of service request is used in a local competition environment to order unbundled elements such as loop service, number portability, and loop service with number portability. The local service provider sends a LSR to the network service provider when the local service provider cannot fill the requirements of an end user from owned resources.

Local Service Request (LSR)

The type of service request used in a local competition environment to order unbundled elements such as loop service, number portability, and loop service with number portability. An LSR is sent by the local service provider to the network service provider when the local service provider cannot fill the requirements of an end user from owned resources.

Location

A physical location that is of interest for equipment inventory purposes. This location may have a Telcordia CLLI, a location identifier that is not a CLLI code, or may simply be identified by a street address. Circuit Design creates an entry in network location for End User PRILOCs and SECLOCs if it does not exist. Network location is a supertype of locations. Subtypes of locations include CLLI locations, end user locations, or terminal locations.

LSOG (Local Service Ordering Guidelines)

A standardized set of guidelines used for ordering various local services. The local service request (LSR) is the administrative form that must accompany any local service request. This type of service request is used in a local competition environment to order unbundled elements such as loop service, number portability, and loop service with number portability. The local service provider sends a LSR to the network service provider when the local service provider cannot fill the requirements of an end user from owned resources.

LSR (Local Service Request)

The type of service request used in a local competition environment to order unbundled elements such as loop service, number portability, and loop service with number portability. An LSR is sent by the local service provider to the network service provider when the local service provider cannot fill the requirements of an end user from owned resources.

Mapping

The process of associating each bit transmitted by a service into the SONET payload structure that carries the service. For example, mapping a DS1 service into a SONET VT1.5 associates each bit of the DS1 with a location in the VT1.5.

Network

The interconnection of equipment and outside plant components designed to provide an infrastructure fabric of facilities to support the transport of circuits. Each component of the network (Facilities, Equipment, Plant, and TFC Networks) may stand alone in the individual circuit design/assignment process. Alternatively, the components of the network may be combined to facilitate the designing process by allowing one assignment to encompass many network components together.

Network Element

A system such as a switch or Digital Cross-connect System (DCS) or a single shelf such as an Add-Drop Multiplexer (ADM). Another type of network element is a Digital Loop Carrier (DLC).

Network Node

Maintains information on an intelligent network element that makes up a telecommunications facility network.

NPAC SMS (Number Portability Administration Center and Service Management System)

Assists in administering Local Number Portability (LNP).

OBF (Ordering and Billing Forum)

A subcommittee of the Exchange Carriers Standards Association (ECSA). This forum discusses operational ordering, provisioning, billing, and presubscription.

Object Management Group (OMG)

Formed in 1989 by a group of vendors for the purpose of creating a standard architecture for distributed objects (also known as components) in networks. The architecture that resulted is the Common Object Request Broker Architecture (CORBA).

Object Request Broker (ORB)

The programming that acts as a broker between a client request for a service from a distributed object or component and the completion of that request. Having ORB support in a network means that a client program can request a service without having to understand where the server is in a distributed network or exactly what the interface to the server program looks like. Components can find out about each other and exchange interface information as they are running.

OMG (Object Management Group)

Formed in 1989 by a group of vendors for the purpose of creating a standard architecture for distributed objects (also known as components) in networks. The architecture that resulted is the Common Object Request Broker Architecture (CORBA).

ORB (Object Request Broker)

The programming that acts as a broker between a client request for a service from a distributed object or component and the completion of that request. Having ORB

support in a network means that a client program can request a service without having to understand where the server is in a distributed network or exactly what the interface to the server program looks like. Components can find out about each other and exchange interface information as they are running.

Ordering and Billing Forum (OBF)

A subcommittee of the Exchange Carriers Standards Association (ECSA). This forum discusses operational ordering, provisioning, billing, and presubscription.

Packet Internet Groper (PING)

A program used to test whether a particular network destination on the Internet is online.

Password

A word or string or characters recognized by automatic means, permitting a user access to a place or to protected storage, files, or input/output devices.

Ping (Packet Internet Groper)

A program used to test whether a particular network destination on the Internet is online.

Port Address

Maintains information on an equipment's assignable ports for transmission purposes. These ports can be either physical or virtual as in the relationship with the circuit positions associated with virtual (ST or VT) facilities. Port addresses can be either physical or "enabled" by the physical, as in the relationship with the circuit positions associated with facilities.

The port address can also be identified with a node address used for assignment selection. Other information can be maintained specific to the properties of the port, such as whether the port is line or drop, or identified as east or west.

Product Service Request (PSR)

An order request for end user products provided by a LEC. End user products include local dialtone services such as business lines and residential lines.

Provisioning

The process of accomplishing the physical work necessary to implement the activity requested on an order.

This normally includes the design and the activation processes. For an install of a circuit, this would typically involve Circuit Design in the MetaSolv Solution (making assignments) and activating the circuit.

PSR (Product Service Request)

An order request for end user products provided by a LEC. End user products include local dialtone services such as business lines and residential lines.

Rate Code

Identifies the bit rate associated with a circuit, facility, or equipment. For example, DS0, DS1, or DS3.

Repeat Trouble

Trouble reported on a service item two or more times within a specific period.

Rollback

The undoing of partly completed database changes when a database transaction has failed.

SBO (Send Bill Ord)

A gateway event which must be associated with a task in the provisioning plan assigned to the service request.

Scripts

The APIs use SQL (Structured Query Language) script. A script is a program or sequence of instructions that is interpreted or carried out by another program rather than by the computer processor (as a compiled program is).

Send Bill Ord (SBO)

A gateway event which must be associated with a task in the provisioning plan assigned to the service request.

Service Bureau

A data processing center that does work for others.

Service Category

Identifies the class of cell relay service for the Permanent Virtual Circuit (PVC). This information is identified in both directions of the PVC to support asymmetrical virtual services.

Service Item

A specific instance of a product or service. For example, a telephone line.

Signal

An artifact that communicates information about an event. The point of reference for the API documentation is the MetaSolv Solution product line. Therefore, when reading material about signals, the direction of the signal in relation to the MetaSolv Solution determines whether it is an inbound or outbound signal. When the MetaSolv Solution sends the signal, that signal is called an "outbound signal." When the MetaSolv Solution receives the signal, that signal is called an "inbound signal."

Solicited Message

A message issued the by the MetaSolv Solution acting as a client to another vendor.

SONET (Synchronous Optical NETwork)

An optical interface standard that allows interworking of transmission products from multiple vendors. It is a family of fiber-optic transmission rates from 51.84 Mbps to 13.22 Gbps, created to provide the flexibility needed to transport many digital signals with different capacities, and to provide a standard from which manufacturers can design.

Staging Tables

A set of interim database tables used by the ASR/ISI gateway when processing access service request (ASR) files.

Synchronous Operation

An operation in which the invoking application gets the results of the operation immediately upon the return of the call. The receiver of the operation acts upon that operation and returns the results. No callback mechanism is used.

Synchronous Optical Network (SONET)

An optical interface standard that allows interworking of transmission products from multiple vendors. It is a family of fiber-optic transmission rates from 51.84 Mbps to 13.22 Gbps, created to provide the flexibility needed to transport many digital signals with different capacities, and to provide a standard from which manufacturers can design.

Target Identifier (TID)

Identifies a group of equipment associated as part of a system or network element. In the MetaSolv Solution the TID information is maintained on the Node tab of the Network Element Properties window.

Third Party

Companies that write customized and interconnection interfaces to the MetaSolv Solution APIs, allowing you to access data.

TID (Target Identifier)

Identifies a group of equipment associated as part of a system or network element. In the MetaSolv Solution, the TID information is maintained on the Node tab of the Network Element Properties window.

Transmission Rate

The bit rates associated with a circuit, facility, or equipment. For example, DS0, DS1, DS3, N/A, and so on.

Trouble

Any cause that may lead to or contribute to an end-user perceiving a failure or degradation on the quality of service of a telecommunications service.

VCI (Virtual Circuit Identifier)

The part of the logical connection address on the ATM switch port where the physical NNI or UNI circuit terminates. The PVC may be assigned one VCI per physical circuit. The VCI accompanies the virtual path identifier (VPI) if the PVC Connection Type is "Channel"; it is not used if the type is "Path." In a combined identification, the two are displayed as VPI or VCI.

Virtual

A term that has been used in various areas of the telecommunications industry such as with SONET, Frame Relay, and ATM. In a SONET environment, the MetaSolv Solution uses "virtual" facilities the identify SONET auto-built ST and VT facilities as virtual facilities because the Virtual Indicator on the Transmission Facility Circuit table. In the MetaSolv Solution SONET application, the "virtual" facilities are used to transport other signals such as DS3 and DS1 circuits. In Frame Relay and ATM, the MetaSolv Solution has used the "virtual" term for the permanent virtual circuit (PVC). Therefore, a "Virtual Facility" is used in the realm of SONET auto-built STS and VT facilities and "Virtual Circuit" is used when referring to the Frame Relay or ATM PVC.

Virtual Circuit Identifier (VCI)

The part of the logical connection address on the ATM switch port where the physical NNI or UNI circuit terminates. The permanent virtual circuit (PVC) may be assigned one VCI per physical circuit. The VCI accompanies the virtual path identifier (VPI) if the PVC Connection Type is "Channel"; it is not used if the type is "Path." In a combined identification, the two are displayed as VPI or VCI.

Virtual Path Identifier (VPI)

The logical connection address on the ATM switch port where the physical NNI or UNI circuit terminates. The permanent virtual circuit (PVC) may be assigned one VPI per physical circuit. The VPI is accompanied by the virtual circuit identifier (VCI) if the PVC Connection Type is "Channel"; the VPI alone is used if the type is "Path," In a combined identification, the two are displayed as VPI or VCI.

VPI (Virtual Path Identifier)

The logical connection address on the ATM switch port where the physical NNI or UNI circuit terminates. The permanent virtual circuit (PVC) may be assigned one VPI per physical circuit. The VPI is accompanied by the virtual circuit identifier (VCI) if the PVC Connection Type is "Channel"; the VPI alone is used if the type is "Path." In a combined identification, the two are displayed as VPI or VCI.

Work Queue

A collection place for tasks associated with a service request. There are two types of work queues: child (individual) and parent (group). A child work queue is, typically, set up for one person. A parent work queue is most often set up for a group, department, or someone responsible for managing task assignments.