

MetaSolv Solution TM 6.0

Administration Guide

Eighth Edition
June 2007



Copyright and Trademark Information

Copyright © 2007, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

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

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Document History

Edition	Date	Reason
First	June 2004	General Availability
Second	December 2004	<ul style="list-style-type: none">◆ TBS_DB.INI file to be created by administrator◆ Additional user exits◆ Migration tool button on Utilities toolbar
Third	April 2005	Changes to JMaster.ini file
Fourth	June 2005	Changes to Gateway.ini parameters
Fifth	December 2005	Corrections on using the admin console and the log viewer
Sixth	September 2006	Added stored procedure exit for generating an MSAG validation audit trail.
Seventh	December 2006	Removed references to setting alarms.
Eighth	June 2007	<ul style="list-style-type: none">◆ Copyrights, About this Guide, customer portal updated with Oracle.◆ Chapter 4—Added info on using the log viewer with UNIX.◆ Chapter 7—Added info on tables affected by the purge utility.◆ Appendix A—Added info on Custom Attribute default settings.◆ Removed references to Field Operations Portal.

Contents

About this guide	v
Audience	vi
Additional information and help	vi
Oracle Support	vi
MetaSolv Solution documentation set	vii
1. Introduction	1
Managing MetaSolv Solution administration	1
Introducing the architecture	2
Introducing the accessory applications	3
2. Securing the application	7
Planning application authorization	7
Learning how the application works	8
Designing the security model	8
Implementing application security	11
Adding new users and groups	12
Authorizing system administrators	13
Validating API logons	14
Tracking logons	14
Managing application passwords	15
Creating additional security administrators	16
Assigning MetaSolv Solution permissions	16
Scanning windows and controls into the database	20
Creating MetaSolv Solution security reports	21
Managing utilities security	21
3. Maintaining configuration	23
Understanding configuration files	23
Understanding INI file relationships	26
Communicating with the application server	26
Using a Tbs_db.ini file	27
Pre-Migration Analysis Tool configuration	28
Setting application server performance parameters	29
Setting Gateway.ini application server performance settings	29
Setting Loggingconfig.xml application server performance settings	30
Setting up API servers	30
Updating configuration files	32
Changing configuration files when the environment changes	33
Changing system preferences	34
Delivering file changes	35
Delivering configuration changes you make	35
Enabling users to access accessory products	36
Using the Zero Admin Client (ZAC) to deliver changes	37

4. Running the application server system	39
Starting and stopping MetaSolv Solution	39
Managing the application server	41
Getting started with the Administration Console	42
Getting application server runtime information	43
Managing MetaSolv Solution log files	45
Using the Log Viewer	46
Archiving the log files	49
Changing the log file location	49
Setting the Loggingconfig.xml parameters	50
Resolving a sample problem using log file messages	55
Resetting server-to-database connections	58
5. Managing the MetaSolv Solution database	59
MetaSolv Solution DBA requirements	59
Maintaining the database	60
Managing specific MetaSolv Solution issues	61
Managing scalability	63
Managing database performance	64
DBA tasks	64
Identifying structure and performance problems	64
Tuning the init.ora file	68
Tuning with non-init.ora files	71
Backing up and restoring the system	71
Sample database reports	72
Sample mdl_cur.txt or mdl_upd_cur.txt reports	72
Sample mdl_cmp.txt or mdl_upd_cmp.txt reports	74
6. Customizing the application	75
Customizing the desktop	75
Configuring portlets	77
Customizing the Navbar	83
Producing custom reports	85
Creating a customized report	87
Retaining customized reports after an upgrade	88
Using stored procedure exits	88
Retaining user exits after an upgrade	89
Coding user exits	89
7. Running utilities and accessories	97
Running Background Processor	97
Understanding the Background Processor	97
Configuring the Background Processor	102
Maintaining the Background Processor	110
Importing local exchange routing information	114
Starting the Location and Routing Gateway	115
Setting up the gateway	116
Creating a Location and Routing Gateway report	121

Using the MetaSolv Solution Utilities	122
Starting the application	122
Using the Purge utility	124
Using the DB Health utility	137
Using the Migration utility	141
Using the Custom Attributes utility	141
Using the Geographical Area Types utility	142
Using the Structured Formats utility	142
NPA Split utility	143
8. Troubleshooting and FAQs	147
Appendix A: Gateway.ini file parameters	157
Using Gateway.ini	157
Changing parameters	157
Restarting the application server	157
Understanding the Gateway.ini file structure	158
Sample installed Gateway.ini file	159
Servers parameters	163
ThreadProcs parameters	166
OrbProperties parameter	169
Session parameters	170
System parameters	172
Gateway parameters	174
SignalInternetServices parameters	175
Events parameters	176
Event2 parameters	177
WorkManagement parameters	178
JNDI parameters	180
CA (custom attributes) parameters	183
Portal parameters	186
Appendix B: Annotated loggingconfig.xml	187
Appendix C: Glossary	205
Appendix D: BEA Application Server configuration settings	221
Small deployment administration server	222
Small deployment clustered administration server	224
Medium deployment administration server	226
Medium deployment clustered server	228
Medium deployment clustered server configuration	228
Large deployment administration server	231
Large deployment clustered server	233
Large deployment clustered server configuration	234
Index	237

About this guide

This document contains concepts and procedures for maintaining an established MetaSolv Solution system. The system includes the database, application servers, clients, Background Processor, utilities, supporting applications, and application security.

MetaSolv Solution application details appear in this document in separate chapters covering these topics:

- ◆ Security
- ◆ System management, including chapters on these topics
 - ◆ Configuration maintenance
 - ◆ Maintenance of the system
 - ◆ Database management
- ◆ Troubleshooting tips

Several applications support MetaSolv Solution. See [“Running utilities and accessories”](#) on page 97 for information about those applications. They are:

- ◆ Background Processor
- ◆ MetaSolv Solution Utilities
- ◆ Location and Routing Gateway
- ◆ NPA NXX Utility

Appendices provide technical references for two of the most important administrative files:

- ◆ Gateway.INI
- ◆ Loggingconfig.xml

For additional information about required third-party software, such as the BEA application server or the database, consult the documentation provided by the manufacturer. Information about third-party software products appears in this document only when there are unique facts about the MetaSolv Solution relationship to those products.

Audience

This guide is for individuals responsible for installing or maintaining MetaSolv Solution and ensuring the software is operating as required. This guide assumes the installer has an Oracle DBA and BEA administrator background, with a working knowledge of Windows 2000 and Java J2EE.

Additional information and help

To get additional information or help for MetaSolv Solution, refer to the following resources:

- ◆ Oracle E-Delivery—Provides access to product software and documentation.
 - Visit the E-Delivery Web site at <http://edelivery.oracle.com>.
 - Software and product documentation are contained in the Oracle Communications MetaSolv Solution 6.0 Media Pack.
 - Developer documentation is contained in the Oracle Communications MetaSolv Solution Developer Documentation Pack. Access to developer documentation requires a password.
- ◆ Oracle MetaLink—Provides access to software patches and a searchable Knowledge Base.
 - Visit the MetaLink Web site at <https://metalink.oracle.com/>, and log on using your User Name and Password.
 - Click the Patches & Updates tab to search for patches (efixes).
 - Click the Knowledge tab to search for technical bulletins, fixed issues, and additional product information. To narrow your search, click the Communication Apps link under Product Categories on the left side of the page.

Oracle Support

The preferred method of reporting service requests (SRs) is through MetaLink. MetaLink is available 24 hours a day, 7 days a week.

Although it is Oracle's preference that you use MetaLink to log SRs electronically, you can also contact Support by telephone. If you choose to contact Support by phone, a support engineer will gather all the information regarding your technical issue into a new SR. After the SR is assigned to a technical engineer, that person will contact you.

For urgent, Severity 1 technical issues, you can either use MetaLink or you can call Support. Oracle Support can be reached locally in each country. To find the contact information for your country, go to <http://www.oracle.com/support/contact.html>.

MetaSolv Solution documentation set

This guide is part of a set of documentation about the MetaSolv Solution. For this product, system and API information appears in PDFs such as this one. End-user documentation for this release is in the online Help.

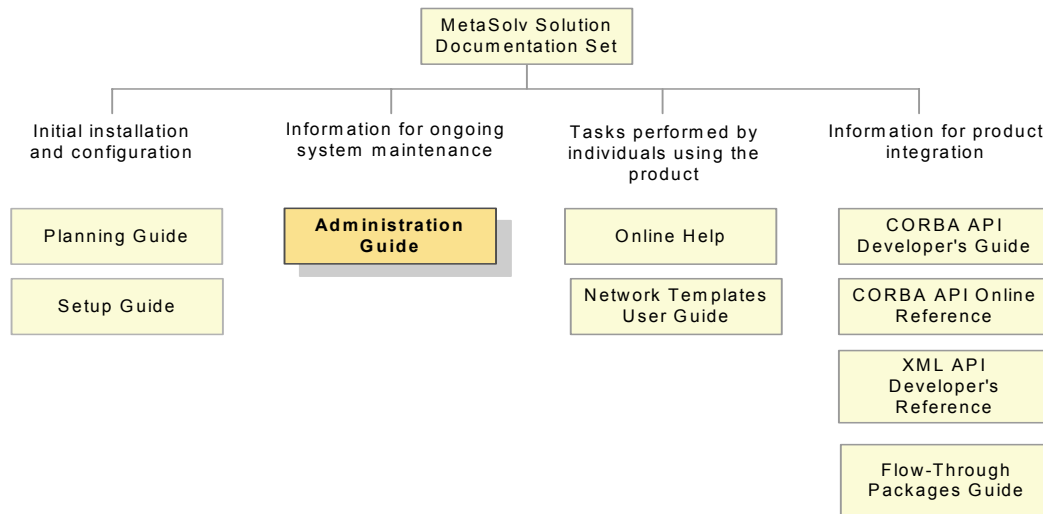


Figure 1: MetaSolv Solution documentation set

MetaSolv Solution books are delivered in Portable Document Format (PDF). You can view a book online using Adobe Acrobat Reader.

To view a document

Locate the document on the Oracle E-Delivery or Oracle MetaLink Web site and do one of the following:

- ◆ Right-click the PDF file and select **Open** from the pop-up menu.
- ◆ Double-click the PDF file.

This action starts Acrobat Reader and opens the PDF document you selected. The following figure shows how a document appears in Acrobat Reader:

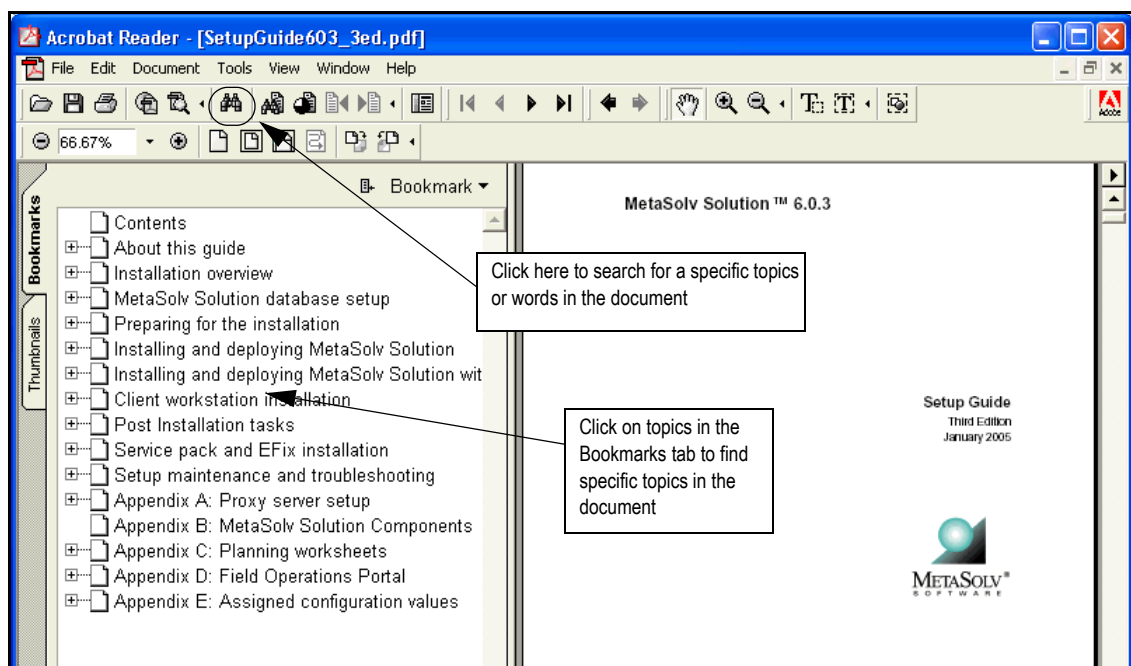


Figure 2: Finding information in a PDF document

Introduction

MetaSolv Solution supports service fulfillment business functions. Each subsystem accesses the MetaSolv Solution database and can be extended to support unique products and services. Users access MetaSolv Solution through a customizable interface.

The MetaSolv Solution APIs 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 the following table. These are the tasks described in this document.

Table 1: Administrative Responsibilities

Responsibility	Tasks
Monitoring and maintaining the system	<ul style="list-style-type: none">◆ Maintaining components◆ Maintaining the configuration files◆ Monitoring log files
Supporting users	<ul style="list-style-type: none">◆ Providing logon information◆ Assisting end users with application error conditions◆ Resolving API problems◆ System error troubleshooting
Managing supporting modules, products, and utilities	<ul style="list-style-type: none">◆ Managing the BEA 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, application servers and APIs, and database servers. It can also include a Citrix/MetaFrame terminal that helps manage communication.

The following illustration shows the MetaSolv Solution three-tier architecture:

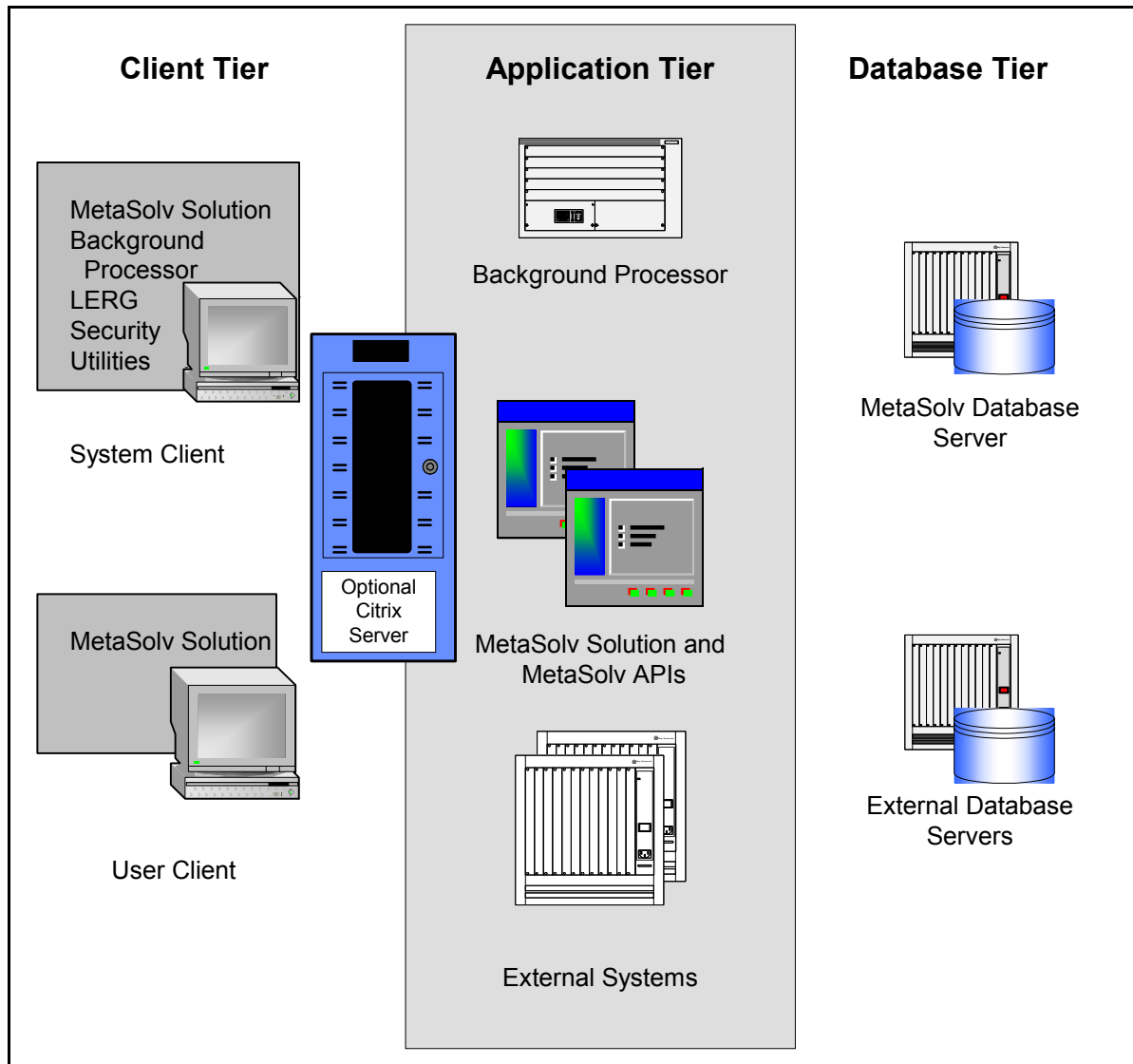


Figure 1: MetaSolv Solution three-tier architecture

MetaSolv Solution is installed on clients and an application server. The clients access the application server and the application server accesses a database. Some accessory products,

like the MetaSolv Solution Utilities, access the database directly. Because the application server and database components are supplied by other companies, you should see that product documentation for detailed instructions about managing:

- ◆ The BEA WebLogic Application Server, BEA WebLogic Server 7 SP4
- ◆ The Oracle database with a Oracle 9.2.0.1 client

In version 6.0.3, BEA WebLogic Server 8.1 SP3 and the Oracle client 9.2.0.3 are required. In release 6.0.5, the Oracle client is no longer required for the application server to communicate with the database.

MetaSolv Solution clients are the user machines that run the MetaSolv Solution software. The client machines access the application server running in the middle tier. Clients connect to these servers using HTTP or IIOP (both over TCP/IP). The application server connects to the database server using JDBC connection pools. The APIs run on the application servers and make direct JDBC connections to the database. The application server is compliant with OMG CORBA 2.4 and J2EE 1.2 specifications.

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

The application server and the APIs access the MetaSolv Solution database, an Oracle Enterprise Edition 9.2.0 database that must reside on a high-performance MicroSoft, HP, or Sun server separate from all other MetaSolv Solution components.

Installed, the core product consists of many files, most of which require no attention. For information about the INI and XML configuration files and the EXE files, see [“Understanding configuration files”](#) on page 23.

Additional files may be installed when MetaSolv Solution is installed, such as the LERG product files. See [“Introducing the accessory applications”](#) for more information.

For details about various components and their technical requirements, see the *MetaSolv Solution Planning Guide*.

Introducing the accessory applications

In addition to the base MetaSolv Solution product, your company may have purchased an extra-cost 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. You can change a client Start page to permit access to an accessory product by following instructions in [“Enabling users to access accessory products”](#) on page 36.

The following table lists accessory applications that may be loaded on client machines.

Table 2: Accessory options and products

Product	Description
APIs	MetaSolv offers a robust library of APIs that permit interfacing with third-party systems.
Data Selection Tool	MetaSolv's Data Selection tool facilitates the selective migration of data between MetaSolv databases of the same version.
MetaSolv Solution Utilities	<p>Purge Utility – This utility removes old data from MetaSolv's database.</p> <p>Database Health Function – This tool allows updates to the database.</p> <p>Structured Formats – These tools provide 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.</p> <p>Custom Attributes tool – This tool lets you set up custom attributes to collect and compute data at specific processing points.</p> <p>Migration tool – This tool allows you to create and convert network elements and to migrate pre-5.1 connections, Virtual Layout Records, and reconcile design lines to Release 6.0.3.</p>
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 information is accurate and in compliance with Telcordia standards.

Table 2: Accessory options and products

Product	Description
MSAG Import Utility	The Master Address Street Guide (MSAG) Import Utility populates the MetaSolv Solution database with MSAG information received from a third-party vendor. With the information in MetaSolv Solution, you can validate end user addresses with MSAG addresses and catch errors that improve the accuracy of E911 reporting.
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 uses NPAs as a part of telephone numbers in several functional areas.
Operational Reports	MetaSolv's Operational Reports allow you to report on data in the MetaSolv Solution, facilitating the integration of business processes and information across functional areas of your organization.

Additional MetaSolv Software products may also be in use at your site. Log on to the Oracle E-Delivery Web site to access those products.

Securing the application

MetaSolv Solution 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 the *MetaSolv Solution Setup Guide* for authentication details. This chapter provides details about authorization, the process of granting or denying access to application functions.



Security instructions provided in this chapter assume that you have chosen Oracle, the default security provider, .

Planning application authorization

MetaSolv Solution security provides controlled, group-based access to specified parts of the MetaSolv Solution application and data. Users can be associated with groups, and users and groups can be restricted from specific areas of the software.

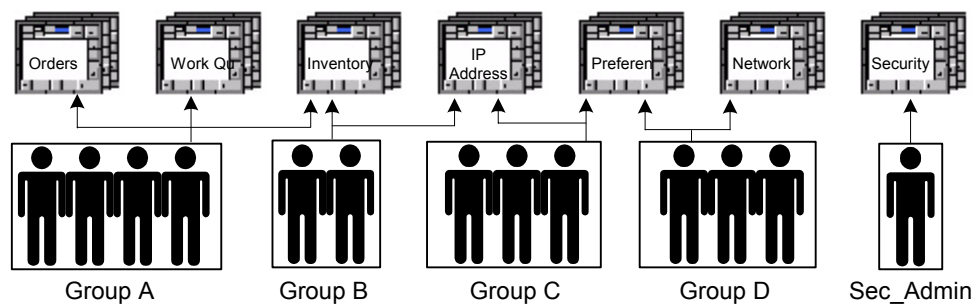


Figure 2: Group-based security

Security is active when the application is installed. It cannot be deactivated. Only one user, the security administrator, has authorization to sign on. Only two groups are active at installation: Default and Sec_Admin (security).

Every layer of the architecture has predefined superusers. A superuser has full control of all functions of the architecture layer. For MetaSolv Solution, the superuser is *ASAP*, the only user with security permissions (the only initial member of Sec_Admin).

Learning how the application works

Before planning the security model and creating users and groups, learn how MetaSolv Solution works at a high level so that you understand the main functions of the product. Next, meet with representatives from each department to define areas of the product they will be using. You will assign access to the users and groups you create based on this information.

Pre-implementation checklist

Gather the following information before beginning to set up security for MetaSolv Solution:

- ◆ Person/group responsible for setting up new user IDs, maintaining general security features/permissions
- ◆ Network user ID naming conventions
- ◆ Company-wide standard on password change intervals
- ◆ Policy for establishing temporary user IDs
- ◆ Any existing group structure in legacy systems that might be leveraged for this implementation

Designing the security model

A security plan must be implemented after the product is installed and before it can be used. This section describes how to make decisions about the security model and provides tools for planning the security implementation.

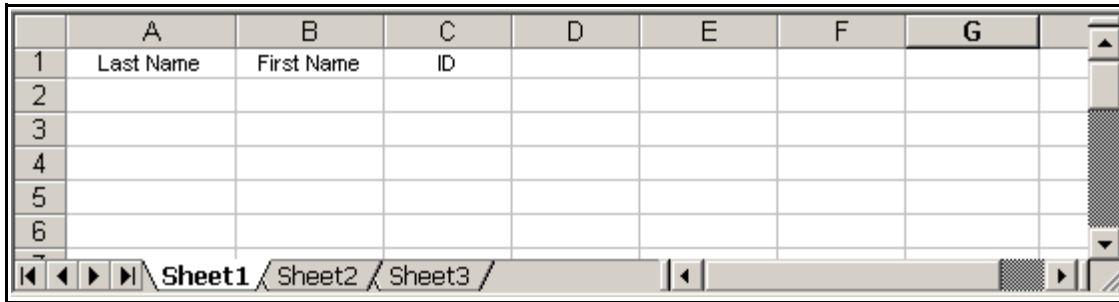
Designing the security model for MetaSolv Solution involves:

- ◆ Identifying the most logical groups of users
- ◆ Creating matrixes for listing relevant users, groups, and processes
- ◆ Completing the security planning matrixes

Use planning matrixes like the examples in the following sections. These matrixes give you a view of groups and permissions. Patterns become obvious and adjustments can be made before implementation.

Identifying MetaSolv Solution users

When planning security, begin by listing all MetaSolv Solution users in the left-most column of an Employee/Group matrix like the sample below. Include those users who may never enter data and only need to find information. The user name should match the Oracle user ID.



	A	B	C	D	E	F	G
1	Last Name	First Name	ID				
2							
3							
4							
5							
6							

Figure 3: Security matrix, Employees

Understanding groups and permissions

When users are added they are automatically members of the Default group and their security permissions are **Not Set**. This permission allows all users to access to all objects except security. Therefore, it is important to assign users to groups with restrictions and permissions.

Organizing users into groups eases security maintenance by reducing the number of permissions for individual users. For example, you might set up a Provisioner group, an Ordering group, and a Marketing group. If every user fits into one of those three groups, you only need to set and maintain permissions for three groups instead of for many individuals.

For a new installation, new MetaSolv Solution users are automatically added one at a time to the Default group and then assigned to one or more groups. The groups are given permissions to access different aspects of the product.

When a user belongs to a group, the user receives all the permissions of the group. If a user belongs to multiple groups, the least restrictive group permissions apply. A permission that is directly granted to a user overrides any group permission levels. Therefore, some users can have more or less restrictive permissions than other users in the same group if they are also restricted as individuals.

MetaSolv recommends always using groups to set permissions, even if it means a group might have only one member for a period of time. If your groups are well-planned, other users will undoubtedly be added. To routinely set permissions without using groups would require extensive setup time for implementation and ongoing maintenance for each individual. If you do not use groups, setting individual security may also require scanning too many windows and controls into the database. If you must scan objects into the database for each user, the number of records increases, increasing the possibility of negative performance impacts. This will be explained in more detail, later in this chapter.

Identifying logical groups of users

A quick way to designate groups within MetaSolv Solution is to identify the departments that use the product, and use those departments as the highest-level group names. If your departments are large or have diverse responsibilities, you can identify subsets of those departments as groups.

Another way to designate groups is:

- ◆ Identify processes
- ◆ Group processes into like functions
- ◆ Map users to those groups

After determining the basic groups at your company, list them across the top of the Employee/Group matrix you used to list all user names, as shown in the following example.

	A	B	C	D	E	F	G	
1	Last Name	First Name	ID	Security Admin	IT Operations	Order Entry	Engineering	Network
2	Boutell	Jennifer	Jboute					
3	Robbins	Bill	Brobbi					
4	Smith	John	Jsmith					
5	Wilson	Mary	Mwilso					
6	Yates	Joan	Jyates					

Figure 4: Security matrix, employees and groups

Associating users with groups

With each user listed down the left side and each group listed across the top, fill in the matrix, associating each user with at least one group.

	A	B	C	D	E	F	G	
1	Last Name	First Name	ID	Security Admin	IT Operations	Order Entry	Engineering	Network
2	Boutell	Jennifer	Jboute	x	x	x	x	
3	Robbins	Bill	Brobbi		x			
4	Smith	John	Jsmith			x		
5	Wilson	Mary	Mwilso			x		
6	Yates	Joan	Jyates				x	x

Figure 5: Security matrix, users and groups

Implementing application security

When planning is complete, the initial process for establishing security is straightforward and sequential, as shown in the next figure.

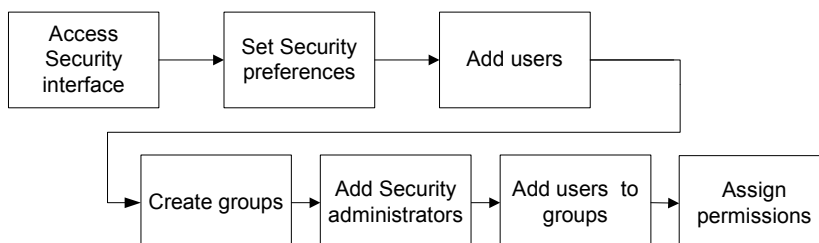


Figure 6: First security implementation process

Before you begin implementing security in MetaSolv Solution, be sure you have completed the following tasks:

- ◆ Be sure there are no user restrictions on the servers where MetaSolv Solution resides.
- ◆ Make sure the MetaSolv Solution server names are in the Gateway.ini file using the parameters in [“Gateway parameters”](#) on page 174.
- ◆ Assign users read-only access to the network folders containing the application files.

To access MetaSolv Solution security, begin by logging on to MetaSolv Solution. When the product opens, an interface like the following example appears:

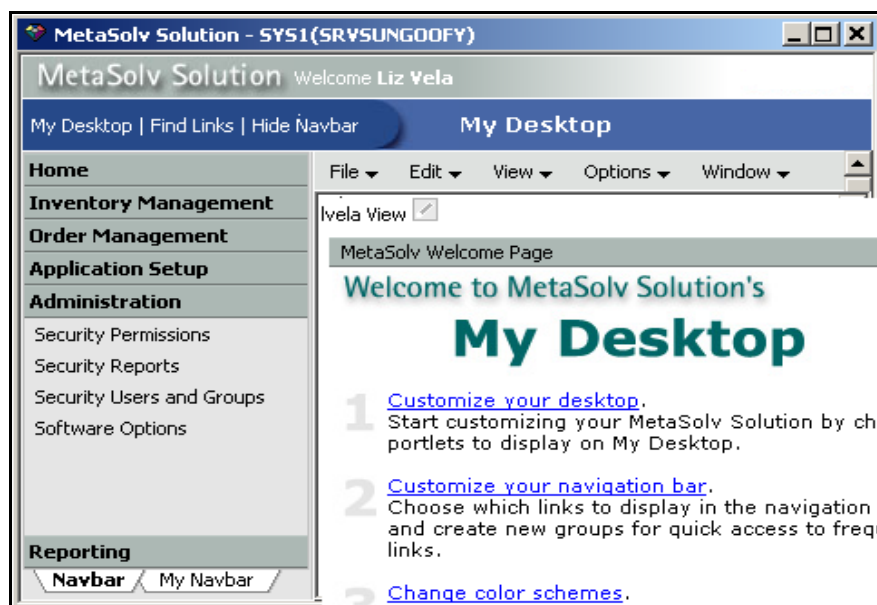


Figure 7: Security options

Select **Administration** on the navigation bar to add users, assign MetaSolv Solution permissions, and to maintain groups. The rest of this chapter and the online Help provide details about procedures and descriptions of window and fields.

Adding new users and groups

All users are initially members of the Default group and can remain in that group even when assigned to other groups. The Default group is initially restricted only from accessing security.

Both users and groups can be members of groups. A user or a group can be unassigned from any group with which it is associated. Make sure that the unassigned user or group does not need the permissions being removed when it is disassociated from the parent group.

To add users or groups:

1. Log on to MetaSolv Solution using the ASAP ID and password.
2. Click **Administration**.
3. Click **Security Users and Groups**.
4. Click **Users** or **Groups** in the left pane.
5. Right-click with the cursor in the right pane.
6. Click **New** or **New From** to open the dialog for adding a new user.
7. Accept the 90-day default password expiration date by taking no action, or change it based on your business practices

Follow operation instructions in the Help for adding new users or new groups.



If using a non-Oracle security scheme for authentication, add the user using the application provided by the third party.

Authorizing system administrators

To authorize users to have administration privileges, use the following table.

Table 3: How to assign administrator privileges

Authorization	How to authorize
DBA identification	Scripts are run during installation that provide the ASAP user with DBA authority. Use the ASAP user ID to perform MetaSolv Solution DBA tasks.
Security administrator	Add the user's ID to the Sec_Admin group or to another group that has some or all security permissions. See “Creating additional security administrators” on page 16.
Access to configuration files	Master configuration files reside on the application server. These files can be edited by any user unless password-protected by the IT department. When distributed to client machines, configuration changes can be made.
Customize default desktop	Log on with the ID specified in the DefaultPortalId parameter of Gateway.ini file.
Customize default portlets	Log on with the ID specified in the DefaultPortalId parameter of Gateway.ini file.
Customize the navigation bar (Navbar)	The user identified in the Gateway.ini User parameter in the JNDI section can set the Allow Users to Customize My Desktop preference in MetaSolv Solution. This default determines whether users can customize their desktop Navbar, according to the instructions in the Help.
Set system and global preferences	A user can be assigned permission within application security to set these preferences that affect user setup in the MetaSolv Solution database. Instructions are in the online Help.
Manage BEA Application Server	Refer to the BEA documentation to learn how to authorize BEA administrators.

Validating API logons

Some MetaSolv Solution API Servers, such as PSR, LSR, Work Management, have security features that can be enabled using optional parameters in the Gateway.ini file. The parameters are:

To turn on Login validation for an API server, add the following parameters to this section of the gateway.ini file, as in the following sample lines:

```
validateUserAndPassword=true  
securexxxxxxxx=true
```

Where xxxxxxxxxxx is a recognized server name. See [“OrbProperties parameter”](#) on page 169 for details.

These parameters work only if you are sending a ConnectReq CORBA object to the WDIRootImpl object. This does not apply to all APIs, because some, such as the ASR API, do their own transaction management.

Refer to the CORBA API Developer’s Guide for details about coding the APIs. See [“OrbProperties parameter”](#) on page 169 to find out how to set up API logon validation.

Tracking logons

The log file appserver_auditlog.xml can tell you when users log on and off and when there are failed attempts. This capability can be used to identify unauthorized access attempts. Review the material in [“Setting the Loggingconfig.xml parameters”](#) on page 50 and the sample configuration file in [“Appendix B: Annotated loggingconfig.xml”](#) for instructions on setting the configuration and for a sample configuration file. [“Managing MetaSolv Solution log files”](#) on page 45 provides instruction for viewing the log file.

The following example shows the different types of audit messages recorded in this log file for logon/logout actions from a fictitious userid named 'SCHINTAL'.

a) Authentication failure:

Message: Login attempt failed for SCHINTAL. Exception: ORA-01017: invalid username/ password; logon denied

Cause: Invalid username and/or incorrect password.

Action: Please supply correct user/password.

b) Authenticated (successful):

Message: Login detected for SCHINTAL.

Cause: User has signed on.

Action: None.

c) User has signed off:

Message: Logout detected for SCHINTAL.

Cause: Logout by user or Client disconnected.

Action: None.

Managing application passwords

The security administrator manages the following application password tasks:

- ◆ Setting the password preference
- ◆ Specifying password expiration dates
- ◆ Resetting passwords
- ◆ With Oracle authentication, creating a new Oracle user ID for APIs to use when accessing the database

Setting the password preference

To set the password preference:

1. Log on to MetaSolv Solution
2. Click **Administration**.
3. Click **Preferences**.
4. Double-click the Security folder.
5. Double-click the **Change password upon initial logon** preference. Otherwise, user passwords match IDs until users choose to change them.
6. Check the **Change password upon initial logon** setting to require a password change at the first logon. Otherwise, leave blank.

Specifying a password expiration date

To specify a password expiration date when adding a user:

1. Logon to MetaSolv Solution
2. Click **Administration** on the Navbar
3. Click **Security Users and Groups** on the Navbar
4. Add a user and accept the 90-day default password expiration date by taking no action, or change it based on your business practices

At any time after a user has been added, you can set a specific password expiration date or specify that the password does not expire.

Maintaining user passwords

To assign or change a password expiration date for an existing user:

1. Click **Administration**

2. Click **Security Users and Groups**.
3. Double-click the user whose password expiration date you want to specify.
4. On the Edit User window, click the **Password Expires On** field to display the current calendar.
5. Select a date, or clear the field to indicate no expiration.
6. Click **OK**.

Creating additional security administrators

To create additional administrators:

1. Logon to MetaSolv Solution.
2. Click **Administration**.
3. Click **Security Users and Groups**.
4. Add the user to the Sec_Admin group or create a new sub-group for Sec_Admin.

You can create sub-groups with subsets of permissions, such as:

- ◆ Access to the Security Permissions Window
- ◆ Access to Security Reports
- ◆ Access to the Users and Groups Window
- ◆ Access to Work Management Task Editing
- ◆ Access to the Security System Preference
- ◆ Users to Change Global Preferences

Assigning MetaSolv Solution permissions

Permissions are controlled through the MetaSolv Solution interface (GUI). Permissions control what users and groups can access in the application. Permissions (inclusive or restrictive) are assigned as a means of controlling feature access and on-screen displays. Features and windows can be blocked and fields can be made invisible.

To access permissions:

1. Log on to MetaSolv Solution.
2. Click **Administration**.
3. Click **Security Permissions**.

A window like the following appears.

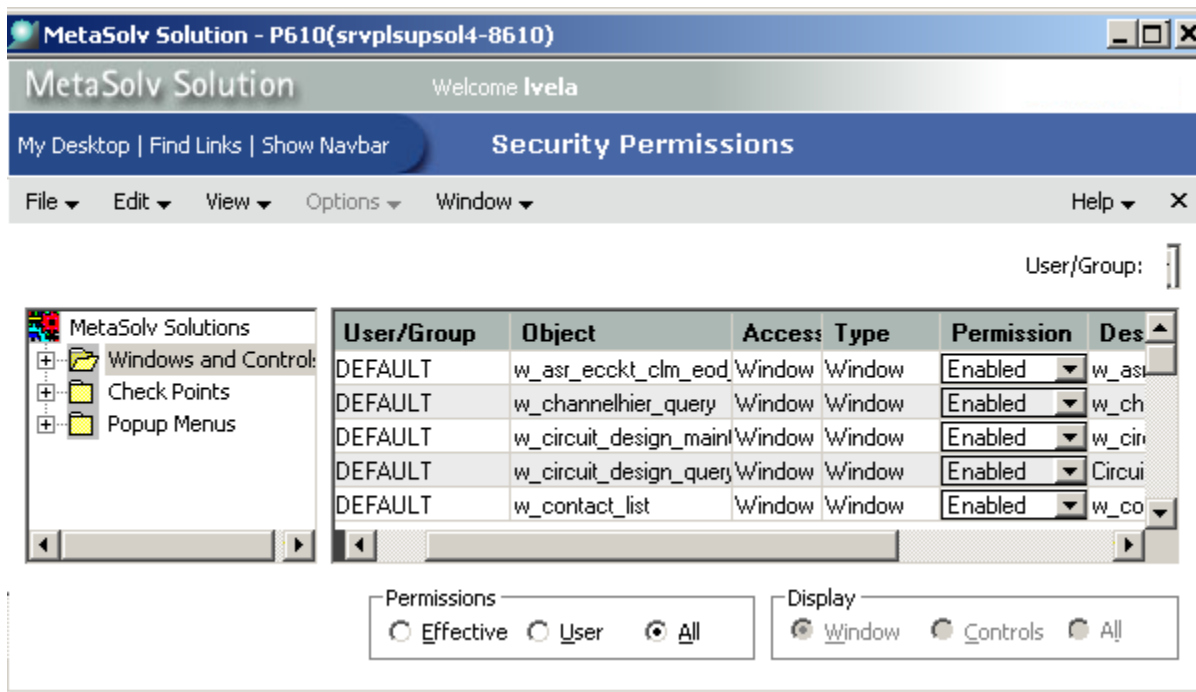


Figure 8: Permissions window

The left-most pane displays a long list of MetaSolv Solution windows. The right-most pane displays a list of permissions for the user or group displayed in the User/Group drop-down. Follow directions in the online Help to assign permissions.

Understanding permissions

Permissions determine what a user can do and which items can be seen within the application. Permissions can be assigned to a group or to a specific user.

The rules for determining access include the following:

- ◆ When a user is assigned to multiple groups, the least restrictive group permissions apply.
- ◆ **Not Set** allows access to objects as a default.

When an individual user is assigned a permission, it overrides group permissions.

Different objects are associated with different types of permissions. The next sections describe the permissions for each type of object.

Window permissions

There are four levels of permissions that you can assign to a window:

Read Only—Users can see and access the window but cannot make changes.

Enabled—Users have explicit permission to use the window and make changes.

No Access—Users cannot see or use the window.

Not Set—Users can access the window. It is similar to **Enabled**, but it does not override other permissions when the system determines the least restrictive permissions for a user.

Control permissions

Objects such as drop-down lists, treeview buttons, tabs, columns, checkboxes, and fields are known as controls. Control names are not preloaded into the MetaSolv Solution database.

If you want to secure a control and the name does not appear in Security, it does not exist in the MetaSolv Solution database. To add the control to the database, refer to [“Scanning windows and controls into the database”](#) on page 20.

Controls are associated with the following permissions:

Enabled—Users have explicit permission to access the control and make changes.

Disabled—The control is non-functional.

Invisible—The control is grayed out.

Not Set—Users can access the control. It is similar to **Enabled**, but it does not override other permissions when the system determines the least restrictive permissions for a user.

Pop-up menu permissions

You can secure an entire popup menu or a specific item on a popup menu. Popup menus are preloaded into Security and cannot be added or customized other than setting permissions.

Pop-up menus are associated with the following permissions:

Disabled—The pop-up menu is non-functional.

Enabled—Users have explicit permission to use the pop-up menu and make changes.

Invisible—The pop-up menu does not appear.

Not Set—Users can access the pop-up menu. It is similar to **Enabled**, but it does not override other permissions when the system determines the least restrictive permissions for a user.

Check point permissions

Check points secure certain logical functions in the MetaSolv Solution that do not correspond to window objects. Because these functions can have far-reaching impacts, check points are preloaded into Security and cannot be added or customized. The Sec_Admin group can access all security check points.

The following processes are protected by check points:

- ◆ Cascade Reconcile
- ◆ Mass DLR Reconcile
- ◆ w_row_in_use
- ◆ Assign Permissions
- ◆ Users and Groups
- ◆ Security Reports
- ◆ Preferences
- ◆ Software Options
- ◆ PBDatabaseTrace
- ◆ Active Ticket Queue
- ◆ Edit Tasks
- ◆ Exception Queue Access
- ◆ System Queue Access



At installation, only the security administrator has the ability to change a completion date on the **Work Management Work Queue Manger window - Task Detail tab**. The security administrator can assign that access to groups or users using the **Edit Tasks** check point.

Permissions that can be assigned to check points are:

Enabled—Users have explicit permission to pass the check point.

Not Set—Users can access the window. It is similar to **Enabled**, but it does not override other permissions when the system determines the least restrictive permissions for a user.

Password— User must enter a password to pass the check point.

No user outside of the Sec_Admin group can pass a check point without the security administrator providing explicit permission to access the protected object or function and providing a special password for the pop-up that appears when trying to access the area. The Sec_Admin group can access all security check points.

Scanning windows and controls into the database

The list of windows in the Security Permissions window uses development window names and not the names that appear on the GUI. The list contains all of the JSP windows in the application and many PowerBuilder windows. Not all PowerBuilder windows are listed. If a window you need is not listed, you can scan the controls on the window into the MetaSolv Solution database and then assign permissions to those controls.

To scan a window:

1. Open the window containing controls you want to secure, and press F2.
2. Make a note of the highlighted window name when the dialog appears.

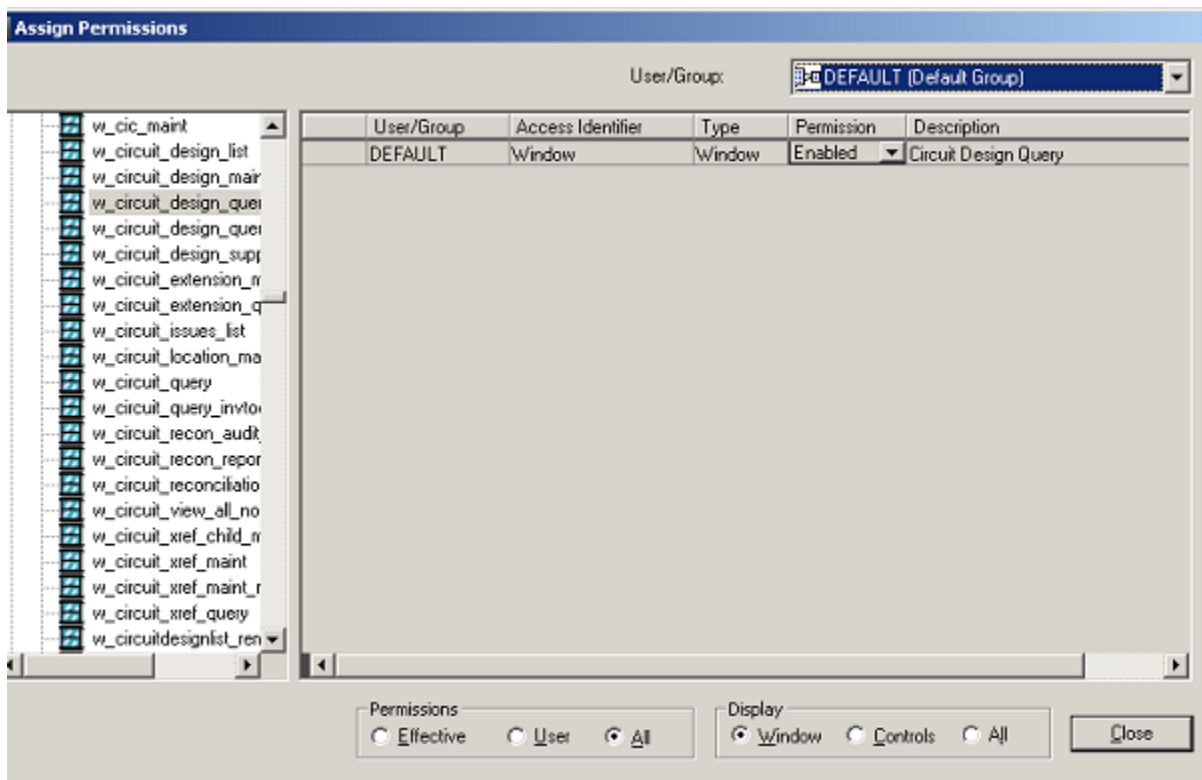


Figure 9: Assign Permissions window

3. Assign permissions to the controls on that window using the Assign Permissions window

Creating MetaSolv Solution security reports

Security reports provide information on the following aspects of MetaSolv Solution system security:

Table 4: Security reports

Report	Description
Individual Detail	Lists controls and statuses for the selected user or group
Hierarchy Detail	Lists controls and statuses for the Ancestor Hierarchy of the selected user
Ancestor Hierarchy	Shows the parent groups of the selected group/user, recursively
Descendant Hierarchy	Shows the groups/users who are assigned to the selected group
User Exception	Shows users who are not members of the Default group

For detailed instructions on creating security reports, see the Help.

Managing utilities security

A separate authorization is required to access security in Tbs_util.exe, the MetaSolv Solution utilities. It is very important to secure these utilities because if you leave them unsecured an authorized user can purge database records. See the Help for instructions on using the MetaSolv Solution utilities security feature.

Maintaining configuration

There are many files and preferences that affect the MetaSolv Solution 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
- ◆ Delivery of configuration and software updates by using the Zero Admin Client (ZAC)

Configuration of the BEA WebLogic Server also affects your MetaSolv Solution environment. See the [“Delivering file changes”](#) on page 35 for additional details and refer to your BEA documentation for instructions on modifying BEA 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 application server or the client. You also manage database configuration files that are installed on the database server.

MetaSolv Solution setup places the MetaSolv files in the appropriate locations and also maintains master copies on the application server. When the software is updated with a service pack or patch, the application server versions are updated. Those copies are then pushed out to the clients when client updates occur.

Various configuration files can be found on the application server, on the client, or on the database server. When configuration files are required on the client, a master copy is stored on the application server. The master copy is automatically updated with service packs or patches. When this happens, master copies must be copied to the client machines, either manually or using the Zero Admin Client. This process is detailed in the *MetaSolv Solution Setup Guide*.

In addition, the MetaSolv Solution configuration files only work when the Oracle Init.ora and Tnsnames.ora files are correctly configured. The following table describes configuration files.

Table 5: Configuration File Functions

Configuration file name	Function	Location
Gateway.ini	Manages many MetaSolv Solution functions and API access to the database. The Gateway.ini is a server-side configuration file and is used by several server processes, including the APIs.	application server
Init.ora	This Oracle initialization file is described in the Oracle documentation.	database server
Jmaster.ini	Jmanager.exe Access to this 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	Lerg.exe The Location and Routing Gateway loads data from Telcordia Technologies.	client
Loggingconfig.xml	This configuration file produces the Appsrverlog.xml, Appserverlog_misc.xml, and Appserver_auditlog.xml log files.	application server
Npasplit.ini	Npasplit.exe This executable 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	Tbs.exe This executable runs MetaSolv Solution. This file is installed on the client when MetaSolv Solution is installed.	client

Table 5: Configuration File Functions

Configuration file name	Function	Location
Tbs_util.ini	Tbs_util.exe This executable runs MetaSolv Solution utilities. It is installed on the client.	client
Tnsnames.ora	This 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* and application server
Zac_config_MSS.xml Zac_config_lerg.xml Zac_config_utility.xml Zac_config_npasplit.xml Zac_config_jobmgr.xml Zac_config_jobmstr.xml	These configuration files manage individual products upgraded by the Zero Admin Client (ZAC). ZAC manages upgrades and other changes to client files.	client* and application server *Reside on the client temporarily for version comparisons.

Understanding INI file relationships

MetaSolv Solution application components are divided into two groups:

- ◆ Components that communicate only with the application server
- ◆ Components that communicate directly with the database

The Gateway.ini file is a server-side configuration file that is used by several server processes, including the MetaSolv Solution APIs. For details about this file, see [“Appendix A: Gateway.ini file parameters”](#) on page 157.

Communicating with the application server

Application components that communicate only with the application server depend on the application server to communicate directly with the database.

Table 6: Applications that communicate with the application 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 application server profile name
- ◆ An application server port ID
- ◆ A database name

Using a Tbs_db.ini file

The following table describes system applications that communicate with the database.

Table 7: 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 Lerg.exe, Tbs_util.exe, and Npasplit.exe 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= 'D53B'

[Profile D53B]
DBMS=O73
Database=Oracle7
UserId=
DatabasePassword=
LogPassword=
ServerName=@D53B
LogId=
Lock=
PRomPt=1
DbParm=DisableBind=0,DelimitIdentifier='No',CommitOnDisconnect='No'
AutoCommit=0
URL=http://wsplwss01:8001
```

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=\\xxxxxxxxx\TBS_db.ini
```

Where xxxxxxxx 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=\\srvpldevsrvcl\testing\6.0.x_ini\tbs_db.iniChanging INI
files
```

When changes are made to the location of the database or application 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 drop-down.

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

Pre-Migration Analysis Tool configuration

MetaSolv Software 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 E-Delivery Web site and uses its own .ini file, labeled Pmat.ini. To log on and use PMAT, you must update the Pmat.ini file.

The Pmat.ini file is downloaded from the Oracle E-Delivery Web site at the same time as the Pre-Migration Analysis Tool executable. Following is a sample of the file you must update:


```
[DBMS_Profiles]
Profiles= 'DatabaseName'

[Profile DatabaseName]
DBMS=O84
Database=Oracle7
UserId=
DatabasePassword=
LogPassword=
ServerName=DatabaseName
LogId=
Lock=
PRomPt=1
DbParm=DisableBind=0, DelimitIdentifier='No', CommitOnDisconnect='No'
AutoCommit=0
```

To configure the Pmat.ini file

To configure the Pmat.ini file, set the Profiles parameter to the name of the database where the migration analysis will be performed.

For procedures and conceptual information about migration, see the *Migration Guide*

Setting application server performance parameters

Parameters can be set that affect performance of the application server and the database.

For proper performance of the application server, specific MetaSolv Solution configuration settings are recommended. Those settings are discussed in the following sections.

For more information on application server performance and MetaSolv Solution, see the *MetaSolv Solution Setup Guide*. For information about BEA application server performance, see the BEA documentation.

Database performance also has a strong impact on product performance. To manage database performance, see [“Managing database performance”](#) on page 64.

Setting Gateway.ini application server performance settings

Several Gateway.ini parameters can have a significant negative impact on application server performance. They are:

- ◆ CacheRefreshMode [CA]
- ◆ DBPollingInterval [Event2]

- ◆ DBPollingInterval [Events]
- ◆ PrintExportObjects [Gateway]

See “[Appendix A: Gateway.ini file parameters](#)” on page 157 for details.

Setting Loggingconfig.xml application server performance settings

Set all logging levels to error in loggingconfig.xml for normal use. Change the severity only during problem resolution. For information about the Loggingconfig.xml file parameters, see “[Appendix B: Annotated loggingconfig.xml](#)” on page 187.

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.

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

The following figure 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.

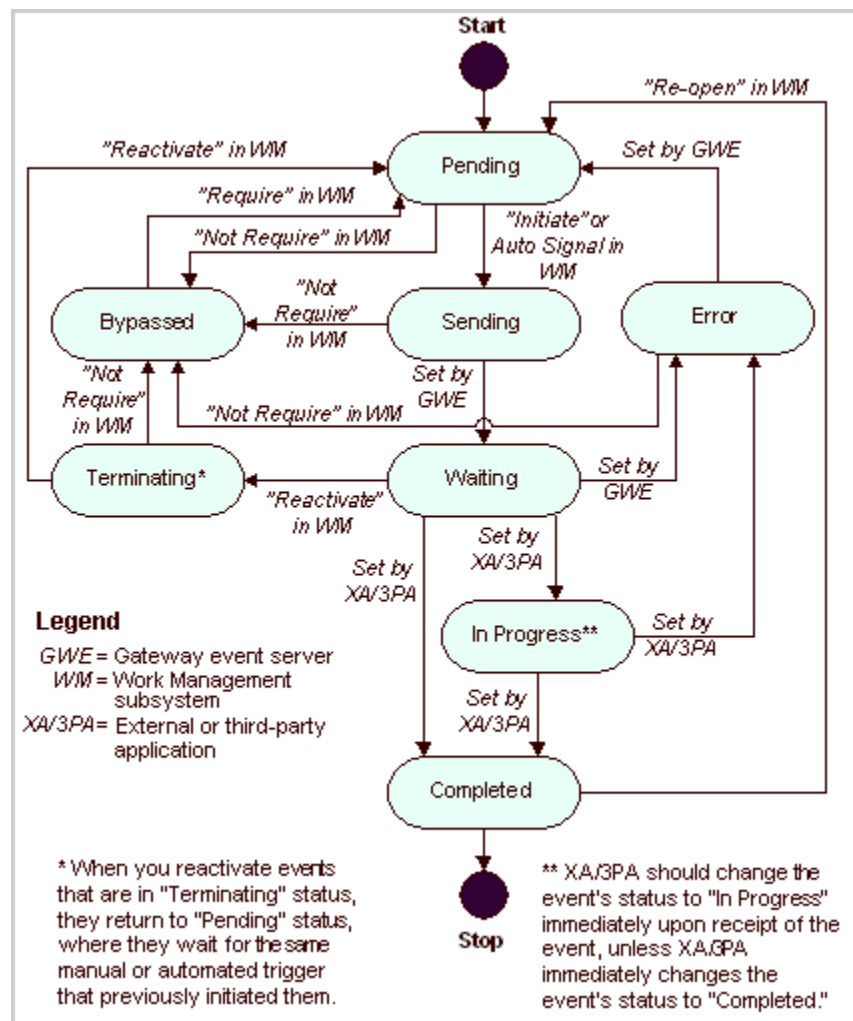


Figure 10: State Transition Diagram for Gateway Events

See [“Appendix A: Gateway.ini file parameters”](#) on page 157 for descriptions of these parameters. The server processes are:

Table 8: 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.
System Task server	The System Task server process allows system tasks to be automatically completed by MetaSolv Solution.
Integration server	The integration server is a process that enables the interaction of the application server to other MetaSolv products.
Signal server proc	Reserved for internal processing.

Look for more information about the API server processes in [“Servers parameters”](#) on page 163.

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 Global Customer Care, 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:

4. Make a backup file prior to editing configuration files.
5. Use any common editor, such as Notepad, to open and edit the file.
6. Shut down and restart the application server unless changing the loggingconfig.xml file.

For BEA-related changes, look for a yellow triangle sign in front of the question mark on the Administration Console. If it is blinking, that means you need to restart the server to make the change take effect.

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 this table as a guide to changing your configuration files if environment changes are made.

Table 9: Environment changes affecting configuration files

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

The following is a sample of the Preferences window:

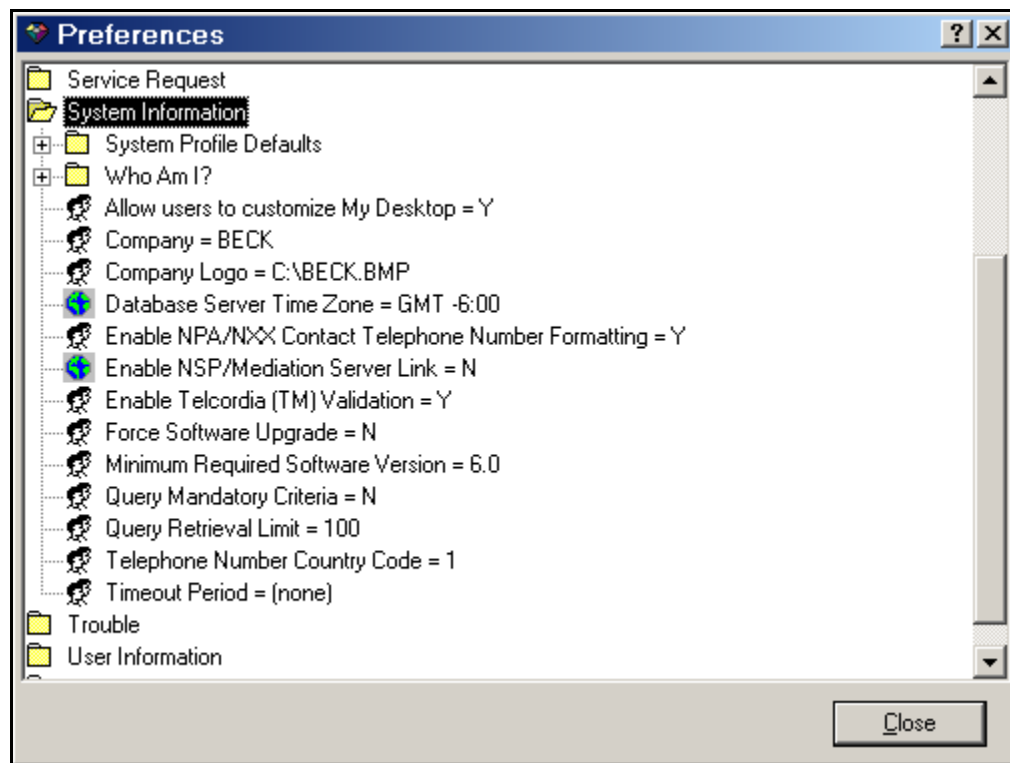


Figure 11: System Preferences

To change a system preference:

1. Log on to MetaSolv Solution.
2. Click **Application Set Up**.

3. Select **Preferences**.
4. Follow instructions in the Help to make changes.

The ability to change system preferences is a privilege which 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. When clients use the Start page to access MetaSolv Solution or an accessory product, ZAC automatically updates the client files. If they do not use the Start page, you will devise other means of delivering file changes or upgrades.

The Gateway.ini file is on the application server. Changes to it have different follow-up requirements than changes to settings on the GUI 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.



Warning! Two configuration files can be changed dynamically in your production system. These files are the Zac configuration files and Loggingconfig.xml. The ability to change them dynamically makes them valuable system tools, but you may want to take precautions to prevent unauthorized or untimely changes.

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 application server, but the Zac configuration files are also copied to the client as a temporary file.

To change any configuration file:

1. Locate the file on the application server.
2. Open the file using Notepad or another editor.
3. Enter your changes and save the file.
4. Deliver the file to the client machines manually or follow directions in [“Using the Zero Admin Client \(ZAC\) to deliver changes”](#) on page 37.

When the change is to the Gateway.ini file, also follow this step:

5. Restart the application server.



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 application server.

Enabling users to access accessory products

You can let users access accessory products from the Start page by changing the parameters in the Zac configuration files.

Application	Config File and Parameter Names	Default
MetaSolv Solution	Zac_config_mss.xml MSS	Default=true
	Zac_config_mss.xml MSSINI	Default=true Downloads Tbs.ini configuration file changes.
	Zac_config_MSS.xml PBVM	Default=true
Background Processor	Zac_config_jobmstr.xml JMASTERINI	Default=false Downloads Jmaster.ini changes.
	Zac_config_jobmgr.xml JOBMGR	Default=false Downloads Jmanager.ini changes.
	Zac_config_jobmstr.xml JOBMSTR	Default=false Required to download Job Master.
Location and Routing Gateway	Zac_config_lerg.xml LERG	Default=false
	Zac_config_lerg.xml LERGINI	Default=false Downloads Lerg.ini changes.
NPA Split Utility	Zac_config_npasplit.xml NPASPLIT	Default=false
	Zac_config_npasplit.xml NPASPLITINI	Default=false Downloads Npasplit.ini changes.

Application	Config File and Parameter Names	Default
MetaSolv Solution Utilities	Zac_config_utility.xml UTILITY	Default=false
	Zac_config_utility.xml UTILINI	Default=false Downloads Tbs_util.ini changes.

Using the Zero Admin Client (ZAC) to deliver changes

MetaSolv Software provides a feature called the Zero Admin Client for automatically updating client files. If you change a configuration file, you should also change the Zac configuration files.

To maintain changes in the Zac configuration files:

1. Open the appropriate Zac configuration file.
2. Record the changes.

When the client machine signs on using the ZAC link or other option, the differences are noted, and the new version is downloaded.

The following table shows the how the ZAC configuration files use parameter names to relate to the action INI file names.

Zac configuration files parameter	File name and function
JMASTERINI	Jmaster.ini Job Manager (Background Processor)
LERGINI	Lerg.ini Location and Routing Gateway
MSSINI	Tbs.exe file MetaSolv Solution application
NPASPLITINI	Npasplit.ini NPA Split Utility
UTILINI	Tbs_util.ini MetaSolv Solution Utilities

Users may occasionally see this error message:



When this message appears it means that the local version and the server version of the software are different. The user must use ZAC to update the version of the software or find another way to upgrade. See the *MetaSolv Solution Setup Guide* for options on delivering files to users.

To distribute changes when not using the ZAC Start Page:

1. Periodically distribute a link to the ZAC Start Page to users.
2. Request that users use the distributed link to update their files.

Running the application server system

MetaSolv Solution 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 MetaSolv Solution
- ◆ Managing MetaSolv Solution log files
- ◆ Monitoring client logons
- ◆ Managing the application server
- ◆ Resetting server-to-database connections

When tuning the BEA application server, refer to [“Appendix D: BEA Application Server configuration settings”](#) for suggested settings.

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

Starting and stopping MetaSolv Solution

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

After the database is up and running, use the MetaSolv Solution start script to start the application server. The script sets values for arguments and provides the settings required to make the application 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. If it is a standalone server, the start script will be named `startMSLV01Single.sh`.



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

To start MetaSolv Solution, run these scripts:

1. `startMSLVorb.sh` (to start jbroker ORB)

If running in a domain, start the Admin server with `startMSLVadmin.sh` before starting `MSLV01`.

2. `start<servername>.sh`

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 512. Using this example, the following displays in the Telnet window when you stop MetaSolv Solution.:

```

rplwss01:/opt2/m602/server01/m6domain/server01 > ps -ef|grep java

mslvbea 10457  1 0  Jun 24 ?      0:10
rplwss01:/opt2/bea/jdk131_08/jre/bin/..bin/sparc/native_threads/java -Dorg.omg.CORBA.R

mslvbea 13131 13113 0 10:14:28 pts/6  9:22
rplwss01:/opt2/bea/jdk131_08/bin/..bin/sparc/native_threads/java -server -Xms512m -Xm5

mslvbea 13217 12725 0 10:57:29 pts/6  0:00 grep java

rplwss01:/opt2/m602/server01/m6domain/server01 > kill 13131

```

Figure 12: Stopping MetaSolv Solution on UNIX

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:

- ◆ UNIX: `./stopMSLVServer.sh <server name><host admin URL>`
- ◆ Windows: `stopMSLVServer.cmd <server name><host admin URL>`. Use "t3://hostname:adminport" as the admin URL argument.

Managing the application server

The BEA 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.

Refer to *WebLogic Server Performance and Tuning* and the *WebLogic Server Admin Guide*, for details on using the BEA WebLogic Server. These documents can be downloaded from the BEA documentation Web site.

For specific configuration and tuning recommendations for MetaSolv Solution and the WebLogic Application Server, see [“Delivering file changes”](#) on page 35.

You can access the application server using a Web browser, at `http://host:port/console/` or you can use the MetaSolv Solution Runtime page link to the WebLogic Administration Console. This link appears on the Runtime Page drop-down. See [“Getting application server runtime information”](#) on page 43 for details about the Runtime page.

The Administration Console manages the following WebLogic Server tasks:

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

Follow directions in the BEA documentation to perform specific WebLogic tasks. Follow instructions in the *MetaSolv Solution Setup Guide* regarding basic BEA 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

From your browser, type the following URL:

```
http://hostname:port/console
```

The value of hostname is the DNS name or IP address of the Administration server. 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.



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 application server through the Runtime page



Warning: Refer to the “[Delivering file changes](#)” on page 35 before making any changes to the number of threads.

Getting application server runtime information

The MetaSolv Solution Runtime page provides detailed application server information. It also provides links to important administrative tools that provide specialized information. See the *MetaSolv Solution Setup Guide* for instructions on accessing the ZAC Start page links.

To use the Runtime Page:

1. Display the Start page.
2. Do one of the following:
 - ◆ Select MSS Runtime Information from the drop-down.
 - ◆ Enter the Runtime Page URL in your browser address field as in the following sample:

```
http://hostname:httpport/main/get_runtime_info
```

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

If you start the Administration server using Secure Socket Layer (SSL), you must add an "s" after http. For example,

```
https://hostname:port/console
```

The Runtime page looks like the following sample:

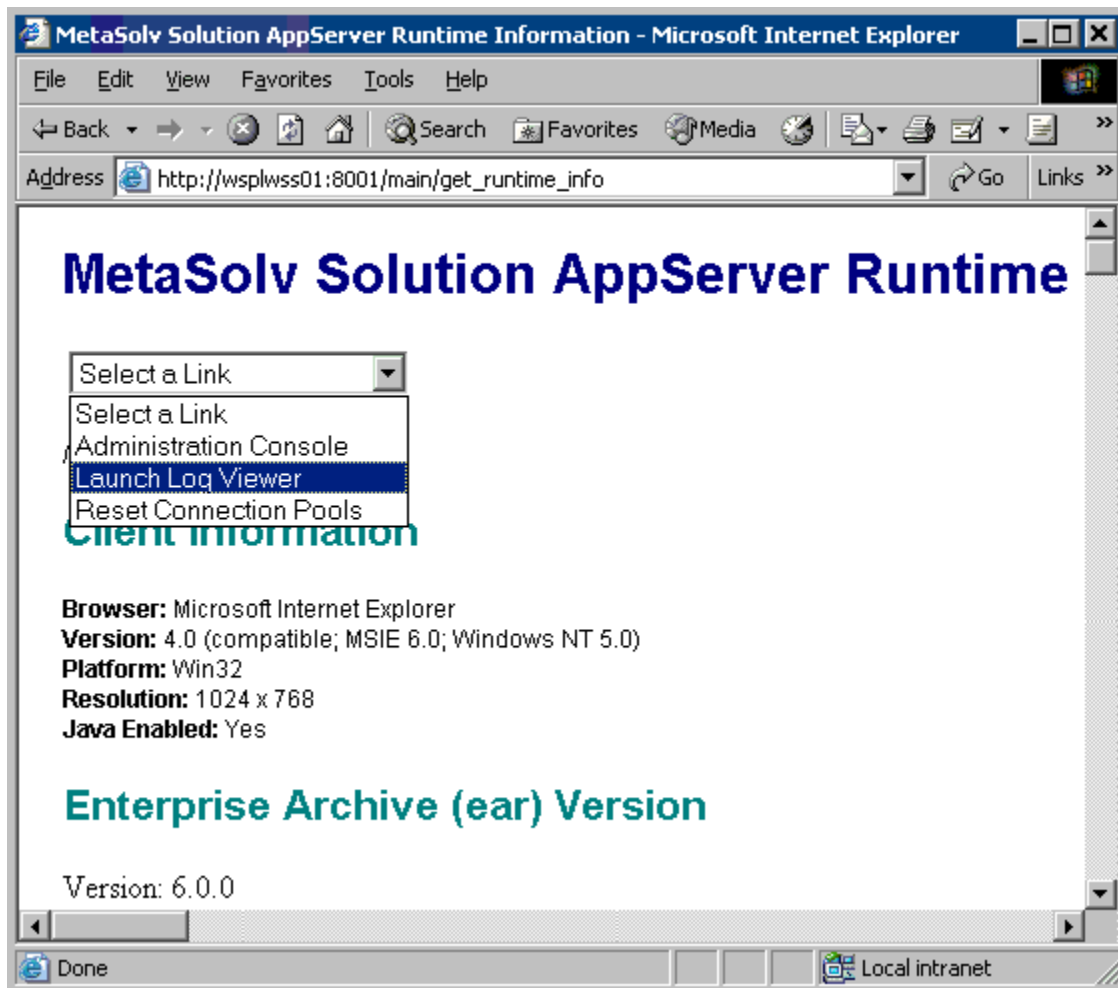


Figure 13: Runtime Page

Server details

The MetaSolv Solution Appserver Runtime page provides a detailed snapshot of an application 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
- ◆ View MetaSolv Solution log files
- ◆ 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 application servers 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. The log files are:

Table 10: Log Files

File name	Purpose
appsrvrverlog.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 the Appsrverlog.xml. The Appserver_auditlog.xml configuration is independently controlled from the main log file's configuration (using the AuditFileApp element of the loggingconfig.xml).

Using the Log Viewer

You can use the Log Viewer for reviewing MetaSolv Solution log files created for the application 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 1.3.1 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 the following information which is an excerpt from the Setup Guide under the section heading entitled "If you are installing on a UNIX platform".

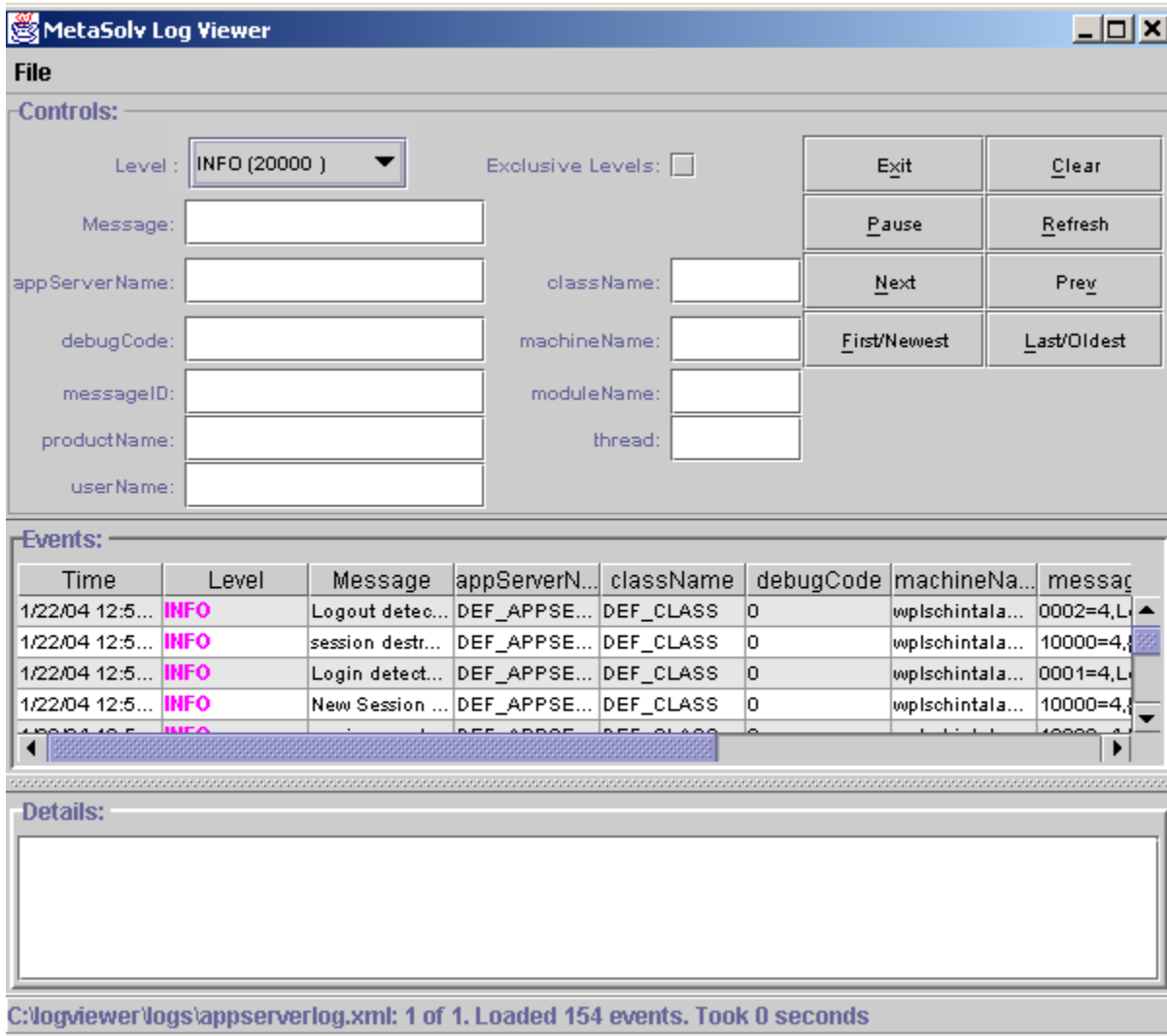
The following list contains special tasks required for graphics on a UNIX machine:

- ◆ 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 application servers that run on a UNIX machine.

To enable the lookup of graphic settings on the application server, you must enable xhost on the machine. Execute "xhost +" on the application server machine while logged on as "root".

As you work:

1. Leave the command window open.
2. Select the **logviewer.cmd** to see a Log Viewer, like the following sample:

**Figure 14: MetaSolv Solution 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** checkbox, 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.

The following table describes each field of the appsrverlog.xml and appsrverlog_misc.xml log files.

Table 11: 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 Global Customer Care
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 Global Customer Care.
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.

Table 11: MetaSolv Solution Log File Fields

Field	Description
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 10 MB. 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:

- ◆ `appsrvrlog.xml.1` (Newest archive)
- ◆ `appsrvrlog.xml.2`
- ◆ `appsrvrlog.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 application 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 application server Config folder.
2. Make a backup copy.
3. Make any necessary parameter changes using any text editor.

On Solaris, 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 Appsrverlog.xml should not be changed, and are reserved for error situations that may require assistance from Global Customer Care. 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 below.

Table 12: 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 Global Customer Care.
Pluggable interface specification	Implementation classes for the logging framework. This section must never be modified unless instructed by Global Customer Care.
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"/>
Logging File specification and configuration	Describes whether log file information is appended or written over, log file size, archiving specifications. See the next table for a description of the editable parameters. See Table 13 on page 51 for 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 14 on page 52 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 13: Logging File Editable Specifications

Keywords to search	Editable Logging File Specifications
appender name="XMLFileApp"	<pre><param name="MaxFileSize" value="10000KB"/> <param name="append" value="true"/> <param name="MaxBackupIndex" value="10"/> <param name="File" value="\${mslv.log.dir}/ appserverlog.xml"/></pre>
<pre><appender name="AuditFileApp p" class="org.apache .log4j.RollingFil eAppender"></pre>	<pre><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"/></pre>
<pre><appender name="XMLFileAppN ew" class="org.apache .log4j.RollingFil eAppender"></pre>	<pre><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"/></pre>

The following table lists the available modules for setting trace levels.



Warning: 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 14: 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	<p>MetaSolv's software supports two types of design layout reports (DLRs):</p> <ul style="list-style-type: none"> ◆ Internal DLRs (also known as on-net, or outbound, DLRs) ◆ External DLRs (also known as off-net, or inbound, DLRs) <p>Internal DLRs have always existed for circuits designed in MetaSolv'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.</p> <p>See the Help for more information</p>
DLR_Equipment	<p>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.</p> <p>See the Help for more information</p>
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, if requested by Global Customer care. Use the default settings.
Event2Server	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.

Table 14: Logging Categories

Category	Description or Direction
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, if requested by Global Customer care. 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, if requested by Global Customer care. Use the default settings.
WORK_MANAGEMENT_CONVERSION	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
INTEGRATION_EVENT_SERVER	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
INTEGRATION_MANAGER	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
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, if requested by Global Customer care. Use the default settings.

Table 14: Logging Categories

Category	Description or Direction
NetProv	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
NetProv_Plant	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
NetProv_Activation	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
NI	For MetaSolv Solution internal use only, if requested by Global Customer care. 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, if requested by Global Customer care. 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, if requested by Global Customer care. Use the default settings.
TimedEventProcess	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
TMS	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.

Table 14: Logging Categories

Category	Description or Direction
TroubleBatchWorkerThread	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
KeepAliveTroubleBatch	For MetaSolv Solution internal use only, if requested by Global Customer care. Use the default settings.
WDI	For MetaSolv Solution internal use only, if requested by Global Customer care. 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:

```
message><![CDATA[The server encountered the following Exception: Could not
connect to database java.sql.SQLException: Database alias dNoSuch not found
in file C:\Oracle\Ora81
```

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 application server.
2. Modify the Gateway.ini entry and enter a correct database name.
3. Restart the application server.

Sample appserverlog.xml log file error section

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

```
>>>>
<log4j:event logger="cmm.SYSTEM" timestamp="1076339625342" level="ERROR"
thread="wplsSchintalapat_gateway_Main" dateTime="Mon Feb 09 09:13:45 CST 2004"
userName="SystemManager" moduleName="cmm.SYSTEM"
appServerName="DEF_APPSERVER" messageID="10119"
productName="DEF_PRODUCT" className="DEF_CLASS" debugCode="0"
machineName="wplsSchintalapat">
<log4j:message><![CDATA[The server encountered the following Exception: Could not
connect to database java.sql.SQLException: Database alias dNoSuch not found in file
C:\Oracle\Ora81
at com.metasolv.jdbc.TNSNamesReader.getEntry(TNSNamesReader.java:95)
at com.metasolv.jdbc.TNSNamesReader.getHost(TNSNamesReader.java:66)
at com.metasolv.jdbc.TNSNamesReader.getTcpString(TNSNamesReader.java:59)
at com.metasolv.jdbc.TNSNamesReader.getJdbcUrl(TNSNamesReader.java:77)
at
com.metasolv.system.TBSCORBAAPIServer.startSystemManager(TBSCORBAAPIServer.jav
a:231)
at com.metasolv.system.TBSCORBAAPIServer.init(TBSCORBAAPIServer.java:86)
at com.metasolv.system.GenericStartup.init(GenericStartup.java:54)
at com.mslv.webapp.context.listener.MSLVContextListener.contextInitialized(Unknown
Source)
at
weblogic.servlet.internal.WebAppServletContext$FireContextListenerAction.run(WebAppServ
letContext.java:6376)
at weblogic.security.acl.internal.AuthenticatedSubject.doAs(AuthenticatedSubject.java:317)
at weblogic.security.service.SecurityManager.runAs(SecurityManager.java:118)
at
weblogic.servlet.internal.WebAppServletContext.notifyCreated(WebAppServletContext.java:1
671)
at
weblogic.servlet.internal.WebAppServletContext.preloadResources(WebAppServletContext.j
ava:3160)
at weblogic.servlet.internal.HttpServer.preloadResources(HttpServer.java:654)
```

```
at weblogic.servlet.internal.WebService.preloadResources(WebService.java:483)
at weblogic.servlet.internal.ServletInitService.resume(ServletInitService.java:30)
at weblogic.t3.srvr.SubsystemManager.resume(SubsystemManager.java:131)
at weblogic.t3.srvr.T3Srvr.resume(T3Srvr.java:964)
at weblogic.t3.srvr.T3Srvr.run(T3Srvr.java:359)
at weblogic.Server.main(Server.java:32)
.
at com.mslv.core.global.MSLVException.createMSLVException(MSLVException.java:163)
at com.mslv.core.global.MSLVException.throwException(MSLVException.java:89)
at com.metasolv.system.SystemManager.beginSession(SystemManager.java:348)
at com.metasolv.system.SystemManager.beginHeadless(SystemManager.java:411)
at com.metasolv.system.SystemManager.startup(SystemManager.java:210)
at com.metasolv.system.SystemManager.startup(SystemManager.java:228)
at
com.metasolv.system.TBSCORBAAPIServer.startSystemManager(TBSCORBAAPIServer.java:231)
at com.metasolv.system.TBSCORBAAPIServer.init(TBSCORBAAPIServer.java:86)
at com.metasolv.system.GenericStartup.init(GenericStartup.java:54)
at com.mslv.webapp.context.listener.MSLVContextListener.contextInitialized(Unknown
Source)
at
weblogic.servlet.internal.WebAppServletContext$FireContextListenerAction.run(WebAppServ
letContext.java:6376)
at weblogic.security.acl.internal.AuthenticatedSubject.doAs(AuthenticatedSubject.java:317)
at weblogic.security.service.SecurityManager.runAs(SecurityManager.java:118)
at
weblogic.servlet.internal.WebAppServletContext.notifyCreated(WebAppServletContext.java:1
671)
at
weblogic.servlet.internal.WebAppServletContext.preloadResources(WebAppServletContext.j
ava:3160)
at weblogic.servlet.internal.HttpServer.preloadResources(HttpServer.java:654)
at weblogic.servlet.internal.WebService.preloadResources(WebService.java:483)
at weblogic.servlet.internal.ServletInitService.resume(ServletInitService.java:30)
at weblogic.t3.srvr.SubsystemManager.resume(SubsystemManager.java:131)
at weblogic.t3.srvr.T3Srvr.resume(T3Srvr.java:964)
at weblogic.t3.srvr.T3Srvr.run(T3Srvr.java:359)
at weblogic.Server.main(Server.java:32)
]]></log4j:message>
</log4j:event>

>>>
```

Resetting server-to-database connections

The application server accesses the database using connection pools, which are managed by the WebLogic Application Server. If the WebLogic Application Server is disconnected from the database server due to network problems or database shutdown, the connection pools of the WebLogic server need to be reset after problems are resolved. The administrator can either restart the application server directly or use the Runtime page drop-down to reset the connection pools.

Either way, follow directions in the BEA documentation to reset connection pools.

Managing the MetaSolv Solution database

After an installation or an upgrade, and after users are added, MetaSolv Solution 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 documentation for information about the standard monitoring tools.

MetaSolv Solution-specific deployment details that affect the database are documented in the *MetaSolv Solution Setup 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 database is an OLTP design that includes the following features:

- ◆ Sequences to support simultaneous inserts
- ◆ Wide use of bind variables to get re-use out of the *shared_pool*
- ◆ Indexes supporting foreign keys
- ◆ Tuned SQL statements that attempt to minimize full table scans

The MetaSolv Solution DBA must have a thorough understanding of administering, maintaining, and tuning an OLTP system. Refer to the Oracle 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 DBA must be qualified to perform these tasks to provide continuous support for MetaSolv Software. 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 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

MetaSolv Software 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

MetaSolv Software 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 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 the *MetaSolv Solution Setup Guide* for details about `a_users.sql`.

Handling import and export requirements

The MetaSolv 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 database, MetaSolv does not import the structure separately from the data. The MetaSolv 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.



To test an upgrade of an existing database, **Compress=N** should be used so that existing table sizes are retained 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.



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 MetaSolv Global Customer Care.

Handling locally managed tablespaces

MetaSolv Solution resides on a database defined with tablespaces that are locally managed or dictionary managed. Locally managed tablespaces are supported in release 6.0.

Maintaining Job Type table data

Changes to the WarningsJob Type table data can have significant undesirable results; change them only under the direction of MetaSolv's Global Customer Care personnel.



Warning: 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.



Warning: For some customers, the WarningsUS7ASCII 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 Administrator's Guide*.

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.



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 documentation and the Oracle Metalink knowledge base 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.

MTS-related parameters

MTS-related parameters allow more users without increasing memory allocations. This is achieved by sharing a pool of dedicated server processes.

MetaSolv Software tested MTS in an NT environment and found a noticeable cost of CPU time to manage the sharing of these server processes between the active sessions.

Also noted was consistent performance in dedicated server mode for up to 500 users. However as much memory was used in dedicated server mode for 500 as for 1000 users in MTS mode. At 500 users the system was memory bound, but at 1000 users it was CPU bound. MetaSolv Software was also using a version of the MetaSolv Solution software that was not using the bind variables client option. If your machine size and number of users does not warrant MTS, use a dedicated connection.

Managing database performance

Following are the init.ora parameters that are specific to MetaSolv Solution performance:

The recommended **db_block_size** parameter is 8kb. A 2kb block size is not large enough to support the largest index within the block. The block size must be set at 4kb or larger.

The **optimizer_features_enable** parameter must be set to 9.2.0. The 6.0 release of MetaSolv Solution supports only the 9.2.0 optimizer.

The **parallel_max_servers** parameter should be set to 1. This parameter is more appropriate for a batch job/data warehousing/reporting database than an OLTP system. There have been significant performance issues reported as a result of setting this parameter above 1.

Set the **optimizer_mode** parameter to **CHOOSE** (`optimizer_mode=choose`). MetaSolv Solution SQL is tuned against Oracle's **COST BASED** optimizer. Using **RULE** for the **optimizer_mode** parameter is not supported.

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 documentation, or support, or both to resolve any Oracle errors

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



MetaSolv Software 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 performance tuning are available that describe how to do this.

Identifying structure and performance problems

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
- ◆ mdl_upd_cur.txt

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 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 scouter 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 identical to the mdl_cur.sql and mdl_cmp.sql

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 FILENAME* followed by the command *FORCE*, where *PATH* is the directory path of the script and *FILENAME* 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_cfgprt.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 MetaSolv software standards and for indexes that are not in the indexes tablespace



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 documentation for discussions on index statistics and recommended update schedules.



To run this script outside of the upgrade process for database administration, type *@(PATH) (FILENAME)* followed by the command *FORCE*, where *PATH* is the directory path of the script and *FILENAME* 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

You can run the *mdl_cmp.sql* and *mdl_upg.cmp* scripts independently of an upgrade to help maintain the database and performance levels. MetaSolv 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 .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.com.txt

The `mdl_cmp.txt` and `mdl_upd.com.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 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 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 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

See [“Sample mdl_cur.txt or mdl_upd_cur.txt reports” on page 72](#) for an example of the *mdl_cur.txt* report.

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 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” on page 64](#).

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.

If you are using NT up to version 4.0, be aware that there is a 2GB standard memory limit. Since all memory allocations are requested from the *Oraclednn.exe* program in NT, setting the data buffer unnecessarily high consumes memory that could be used for increasing the sort area size buffer which might also improve performance. 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 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 to store so many queries in the library that it will cause slower performance. This happens when Oracle 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

1. Using striped disks improves 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.



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 program files, and Oracle Archive log files on different disks.
3. Do not allocate so much memory to Oracle 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, Oracle runs slow no regardless of how well-tuned your instance is.
5. For production, run only one instance of Oracle per machine. The more Oracle 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.

MetaSolv 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

Sample database reports

Sample mdl_cur.txt or mdl_upd_cur.txt reports

```
=====
Report Date: 02/09/200x 10:42AM
Company Name:ACME TELCO
MetaSolv Version:6.0
Oracle Version:9.2.0.3
Database Name: METASOLV SOLUTION PROD
=====
```

Change Type	Table	Column	Datatype	Null
Missing Column:	ASAP.ASR_ERROR	CURRENT_VALUE	VARCHAR2(80)	NULL
Missing Column:		DOCUMENT_NUMBER	NUMBER(9)	NOT NULL
Column Difference:		LAST_MODIFIED_USERID	VARCHAR2(8)	NOT NULL
Column Difference:		LAST_MODIFIED_DATE	DATE	NOT NULL
Extra Column:	ASAP.ASSIGNED_TEL_NUM	SPECIAL_FIELD1	NUMBER(9)	NULL
Extra Column:		SPECIAL_FIELD2	NUMBER(9)	NULL
Column Difference:	ASAP.BANK_CODE	TRANSACTION_ORIGIN	VARCHAR2(10)	
(should be varchar2(8)	NULL			
Column Difference:	ASAP.BAN_USAGE	LAST_MODIFIED_DATE	DATE	NULL
(should be not null)				

Change Type	Table	Column	Datatype	Null
Column Difference:	ASAP.DLR_NOTES	DLR_NOTES_TEXT	VARCHAR2(49)	NOT NULL
Column Difference:		NOTE_IND	CHAR(1)	NOT NULL
Missing Table:	ASAP.APPLICATION	APPLICATION	VARCHAR2(40)	NOT NULL
Missing Table:		STATUS	CHAR(1)	NOT NULL
Missing Table:		AUDIT_CREATE_DATE	DATE	NOT NULL
Missing Table:		AUDIT_CREATE_USERID	VARCHAR2(7)	NOT NULL
Missing Table:		AUDIT_UPDATE_DATE	DATE	NULL
Missing Table:		AUDIT_UPDATE_USERID	VARCHAR2(7)	NULL
Missing Table:		DESCRIPTION	VARCHAR2(254)	NULL
Missing Table:		SECURITY_METHOD	NUMBER(1)	NULL
Missing Table:		TRANSACTION_ORIGIN	VARCHAR2(8)	NULL
Extra Table:	ASAP.tmp_APPLICATION	APPLICATION	VARCHAR2(40)	NOT NULL
Extra Table:		STATUS	CHAR(1)	NOT NULL
Extra Table:		AUDIT_CREATE_DATE	DATE	NOT NULL
Extra Table:		AUDIT_CREATE_USERID	VARCHAR2(7)	NOT NULL
Extra Table:		AUDIT_UPDATE_DATE	DATE	NULL
Extra Table:		AUDIT_UPDATE_USERID	VARCHAR2(7)	NULL
Extra Table:		DESCRIPTION	VARCHAR2(254)	NULL
Extra Table:		SECURITY_METHOD	NUMBER(1)	NULL
Extra Table:		TRANSACTION_ORIGIN	VARCHAR2(8)	NULL

Sample mdl_cmp.txt or mdl_upd_cmp.txt reports

```
=====
Report Date: 02/09/200x 10:42AM
Company Name:ACME TELCO
MetaSolv Version:6.0
Oracle Version:9.2.0.3
Database Name: METASOLV SOLUTION PROD
=====

/*****
**INDEX COMPARE REPORT  **
*****/

IIDX_CAT          TABLE_NAME          INDEX_NAME          FIELD_LIST
extra indexes     ASAP.ACCESS_CUSTOMER      ACCESS_CUSTOMER     CUSTOMER_NAME
extra indexes     ASAP.TRBL_TASK_EXT_REF     FK_TTER_PARTY       PARTY_ID_REF_TO
missing indexes   ASAP.ACCESS_CUST           FKIDX_ACCESS_CUST   PARTY_ID, PARTY_ROLE
missing indexes   ASAP.ACCESS_PROVIDER       FKIDX_AC_PR_SE_CE   OPERATING_COMPANY

/*****
**PRIMARY KEY COMPARE REPORT  **
*****/

PK_CAT          TABLE_NAME          CONSTRAINT_NAME          FIELD_LIST
extra pks       ASAP.ACCESS_CUSTMER   PK_ACCESS_CUSTOMER      CUSTOMER_NAME
extra pks       ASAP.ACCESS_CUSTMER   PK_ACCESS_CUSTOMER
missing pks     ASAP.ACCESS_CUST      PK_ACCESS_CUST          PARTY_ID, PARTY_ROLE
missing pks     ASAP.ACTIVATION        PK_ACTIVATION_COMAND    ACTIVATION_COMAND
missing pks     ASAP.ADDRESS           PK_ADDRESS              ADDRESS_ID

/*****
**FOREIGN KEY COMPAREREPORT  **
*****/

FK_CAT          TABLE_NAME          CONSTRAINT_NAME          FIELD_LIST
extra fks       ASAP.ACCESS_CUSTOMER  FK_ACBA_CITY            CITY_NAME, STATE_CODE
extra fks       ASAP.ACCESS_CUSTOMER_ FK_AC_ACNA              ACNA REFERENCES
missing fks     ASAP.ACCESS_CUST      FK_AC_CU__PA_RO        PARTY_ID,
missing fks     ASAP.ACCESS_PROVIDER  FK_APR_ACCESS_SERVICE   DOCUMENT_NUMBER
missing fks     ASAP.ACCESS_PROVIDER  FK_AC_PR_SE_CE__CO      OPERATING_COMPANY
```

Customizing the application

MetaSolv Solution provides three ways for the administrator to customize the application. You can:

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

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.

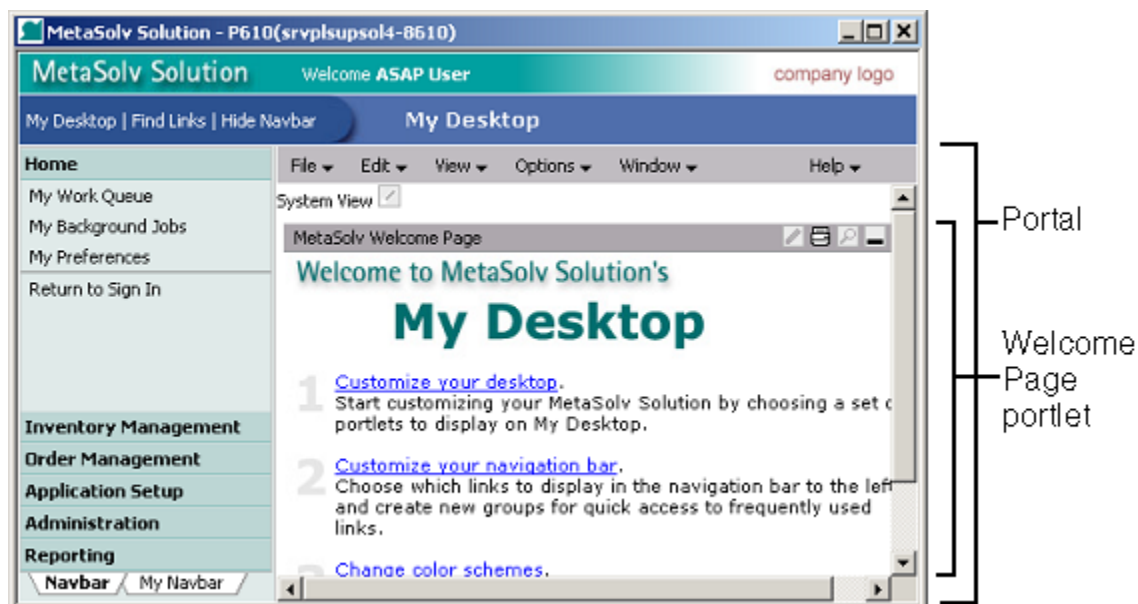


Figure 15: Administrator basic default desktop

The following sample desktop shows the administrator (ASAP user) desktop with a name of *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.

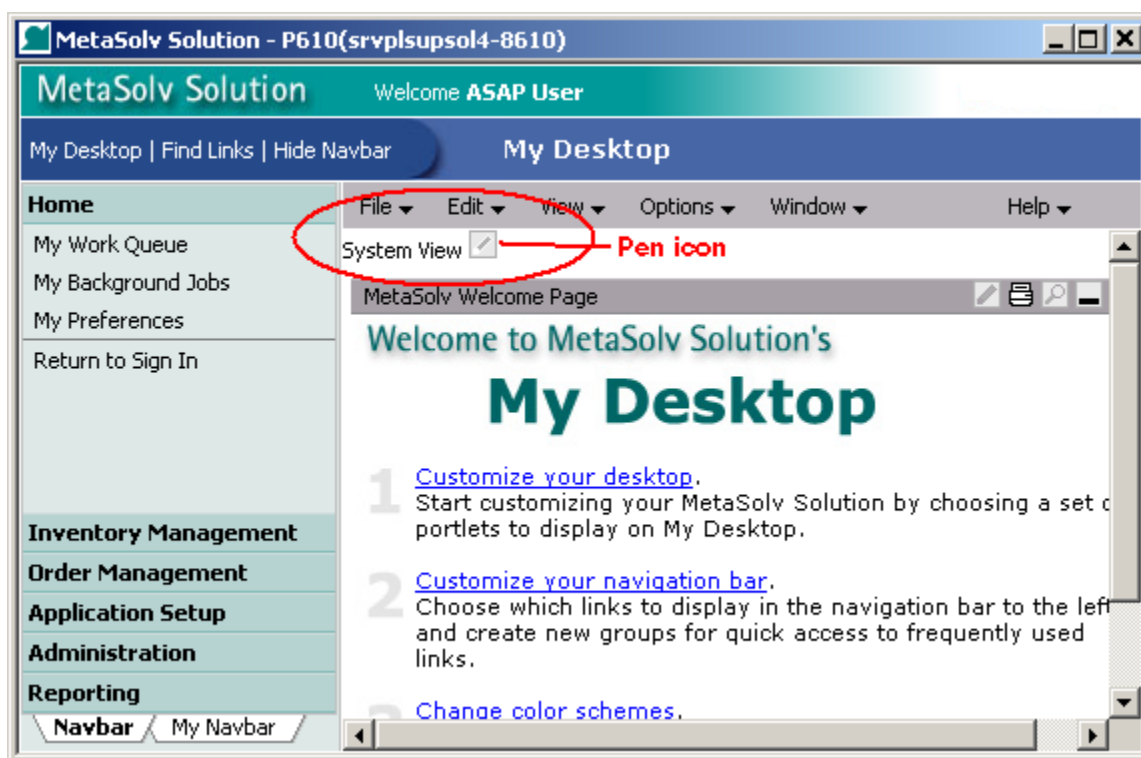


Figure 16: Accessing desktop customization

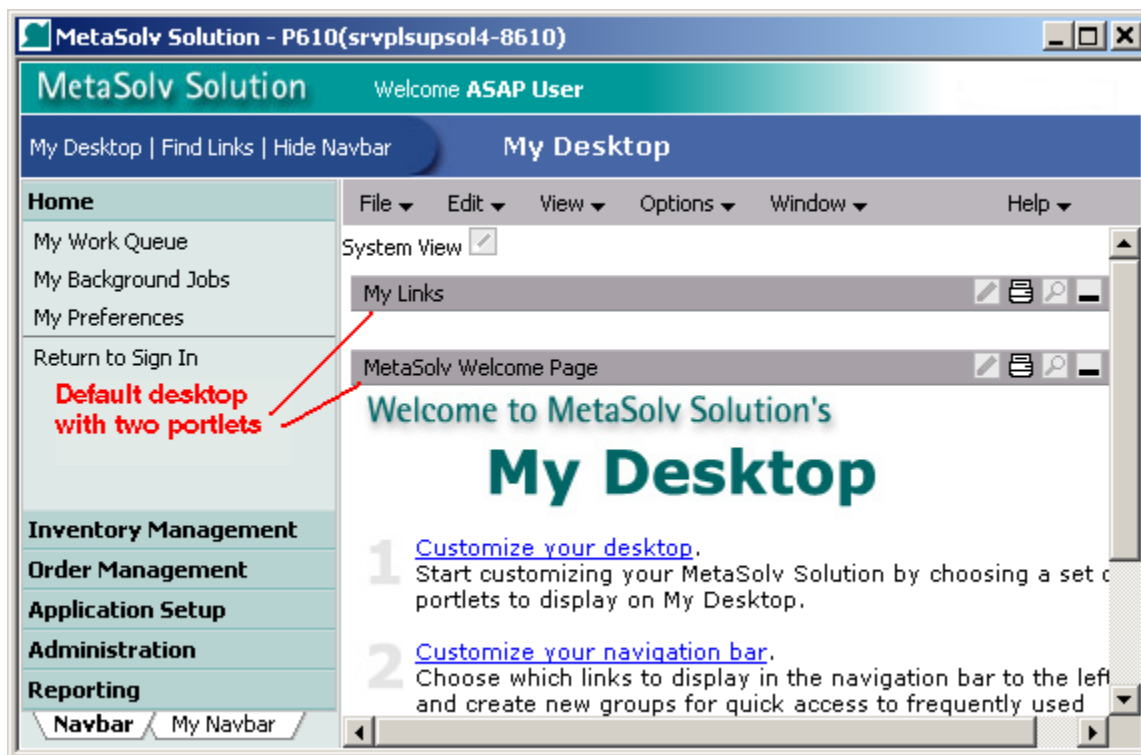
MetaSolv Software ships pre-defined portlets, specified in `metasolv-portlets.xreg` registry file. If you allow it, users can select portlets for their desktops that are already defined or that have been defined by you. To allow users to customize their desktops, set the **Allow users to customize My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

Note 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."

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 next to the words *System View*
3. Follow directions in the online Help to make selections that set up the default desktop.

**Figure 17: Default desktop with two portlets**

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/portlet/conf`, within the MetaSolv Solution root directory. It is a system file and should not be modified.

Following is a sample *My-portlets.xreg* file. The 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.

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

Figure 18: My-portlets.xreg file

The following shows 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>MetaSolv Software</title>
      <description>MetaSolv Software 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>
```

```

</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>

```

Figure 19: Sample file with multiple portlets

Following are descriptions of the available *My-portlets.xreg* file parameters:

Table 15: 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	<p>Meta-info has two elements that should be changed to match a custom portlet:</p> <ul style="list-style-type: none"> ◆ 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. <p>Sample:</p> <pre> <meta-info> <title>ABC</title> <description>ABC Home Page</description> </meta-info> </pre>

Table 15: Registry file parameters

Registry file parameter	Parameter description
Classname	<p>Classname is the portlet class which retrieves the content from the application, the value should be:</p> <p>org.apache.jetspeed.portal.portlets.FileServerPortlet.</p> <pre><classname>org.apache.jetspeed.portal.portlets.FileServerPortlet </classname></pre>
Base parameter	<p>The base parameter helps locate the portlet:</p> <p>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.</p> <p>Most likely the base URL is first segment of the URL. For example, if the content is retrieved from http://www.metasolv.com/MSLV/CDA/General/Homepage/, the base is http://www.metasolv.com/. If you are not sure about the base URL, ask the application Web Master. Following is a sample base parameter:</p> <pre><parameter name="base" value="http://www.metasolv.com/" type="" hidden="false" cachedOnName="true" cachedOnValue="true"/></pre>
Parameters sent to the application	<p>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:</p> <pre><parameter name="symbol" value="MSLV" type="" hidden="false" cachedOnName="true" cachedOnValue="true"/></pre> <p>Note that if the hidden attribute is set to <i>true</i>, the users will not see this parameter in the portlet customization. If it is set to <i>false</i>, this parameter can be customized.</p>
URL	<p>The URL element specifies the location of the application content. The cachedOnURL attribute should be "true". For example:</p> <pre><url cachedOnURL="true">http://www.metasolv.com</url></pre>

Adding portlets

When portlets are enabled at startup of the application 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 15](#) on page 79.
4. After a custom portlet is added into the registry, restart the application 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 application server or other application 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=$PRINCIPAL&password=$PASSWORD</url>
```



& 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 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"/>
```



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 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 application server after modifying the portlet registry file.

Third Party Portlet Accesses MetaSolv Applications

MetaSolv 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 applications:

- ◆ UserID - the value of this parameter is the username.
- ◆ Password - the value if this parameter is the password.
- ◆ TimeZoneID - the timezone id of the user, if not specified the current MetaSolv Portal timezone 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.

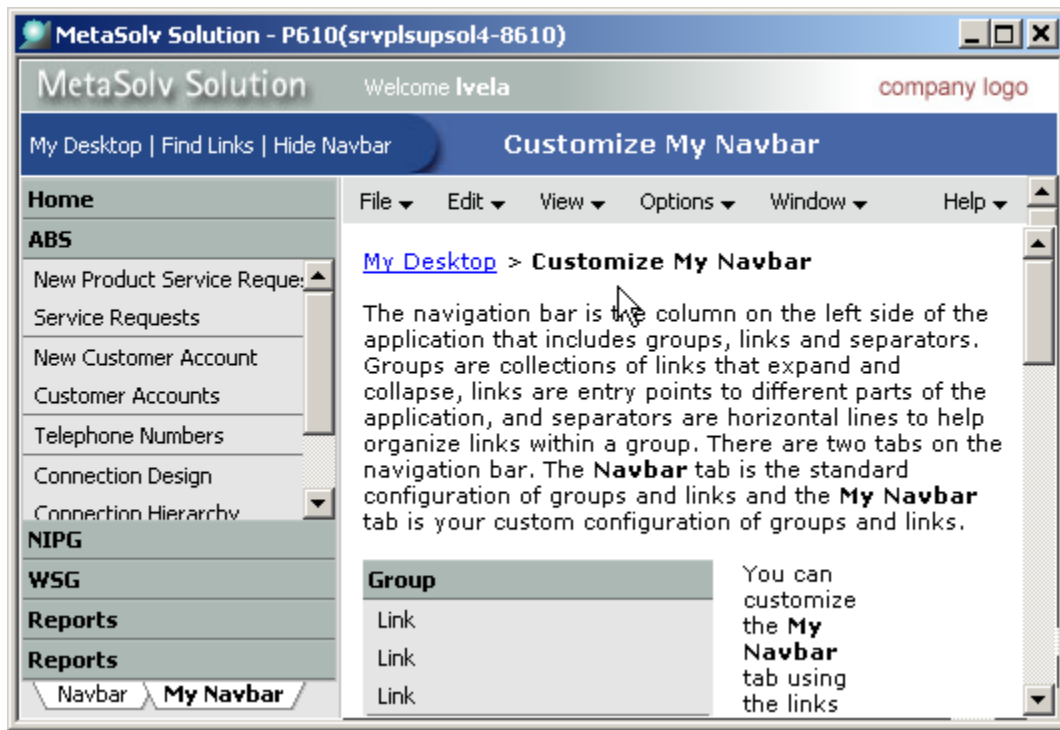


Figure 20: Customizing My Navbar

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. Then, 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.

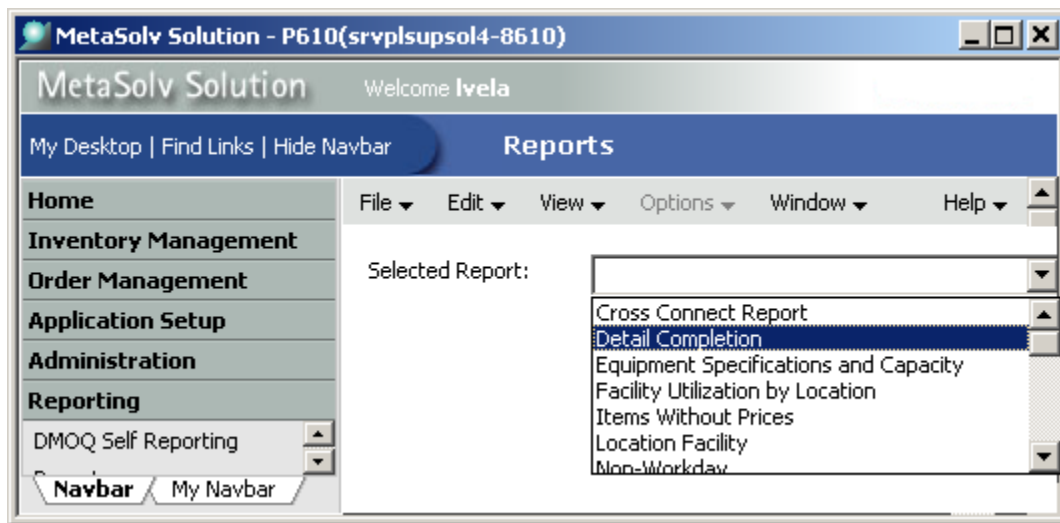


Figure 21: MetaSolv Solution Reports

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.

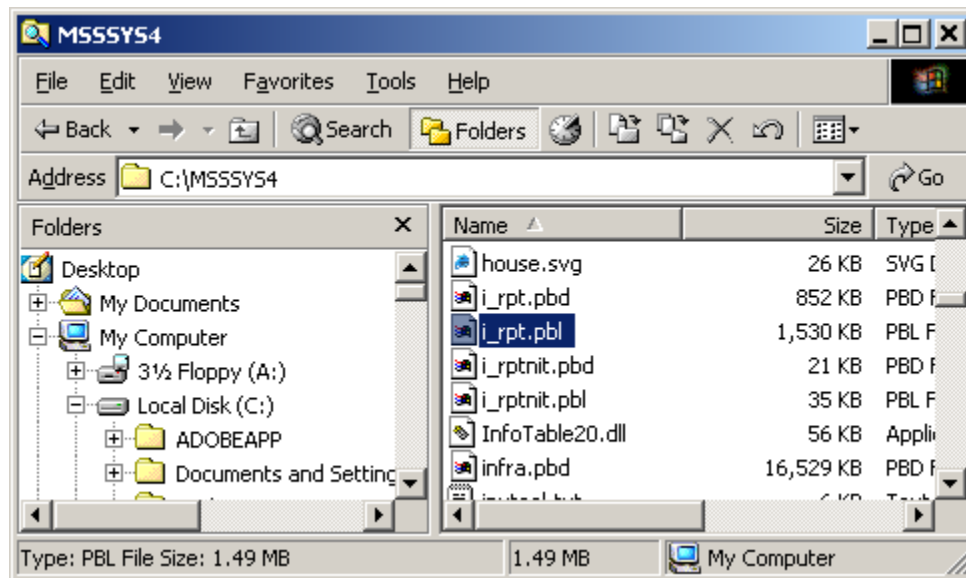


Figure 22: i_rpt.pbl

You can also create custom reports that you add to the Selected Report drop-down pictured in [Table 15](#). 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://www.sybase.com/products/internetappdevttools>

Following is a PowerBuilder sample custom report for MetaSolv Solution:

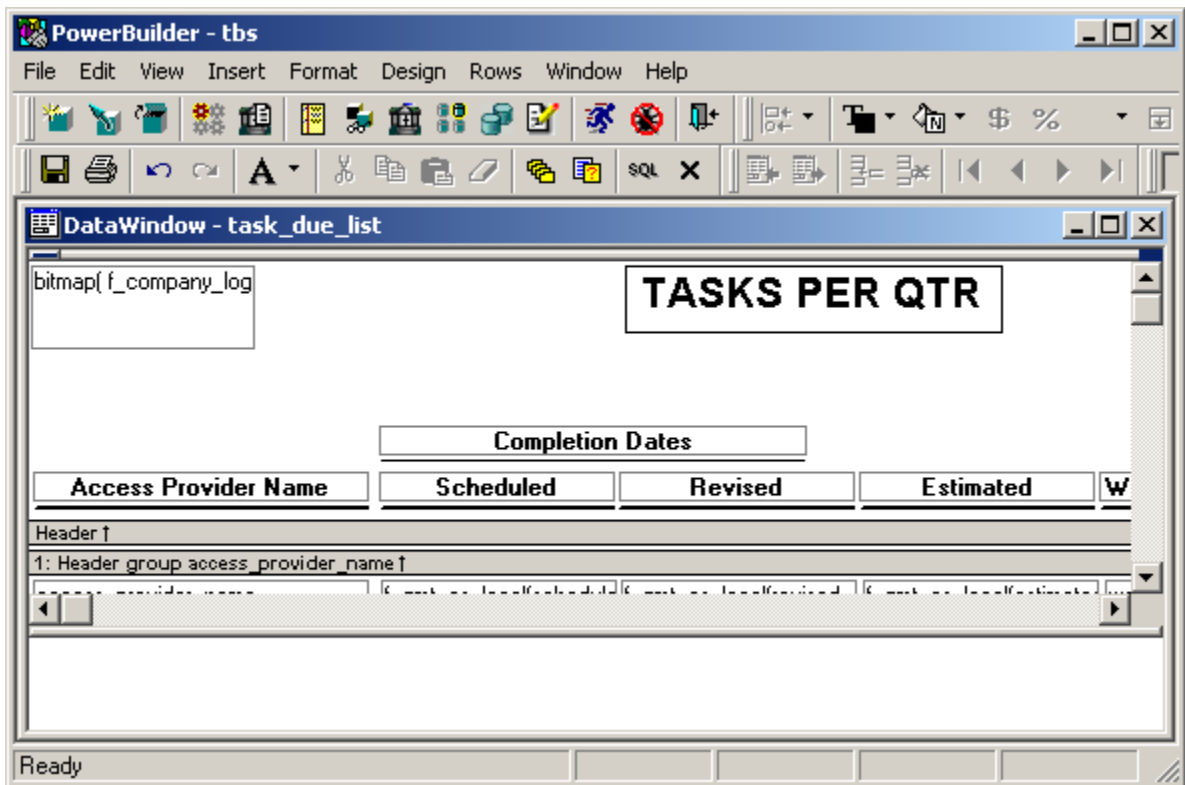


Figure 23: MetaSolv Solution custom report in PowerBuilder

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 on to MetaSolv Solution.
3. Click the **Reporting** group on the navigation bar.
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 a user selects a report and gets the following dialog, it means the datawindow has not been added to I_rpt.pbl on that user's machine.

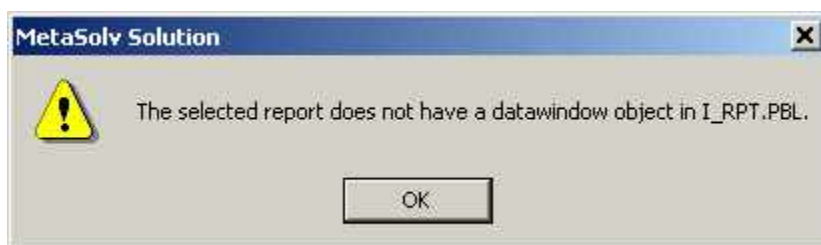


Figure 24: Error message for missing datawindow

If a datawindow 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.



Global Customer Care (GCC) supports user exits as they are written within the delivered code. GCC 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.

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 **Generating Circuit ID**.

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.

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.

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

PSR custom pre-validation

This is a new exit for Release 6.0.3. 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's validation and prior to the other PSR custom validation user exit discussed in [“Perform custom validation on a PSR order”](#).

GUI location	PSR order
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.

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."

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.

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.

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.

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.

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.

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

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.

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.

GUI location	<ul style="list-style-type: none">◆ 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**.

GUI location	<ul style="list-style-type: none">◆ 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.

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

Running utilities and accessories

Several accessory products and utilities can help you run MetaSolv Solution. They are:

- ◆ Background Processor
- ◆ Local Exchange Routing Gateway
- ◆ MetaSolv Solution Utilities
 - ◆ Purge utility
 - ◆ DB Health utility
 - ◆ Migration tool utility (Release 6.0.3)
 - ◆ Custom Attributes utility
 - ◆ Geographical Area Types utility
 - ◆ Structured Formats utility
- ◆ NPA Split utility

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 an online 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.:

Application	Function
Jmanager.exe	Job Manager: <ul style="list-style-type: none">◆ Adds servers◆ Sets preferences◆ Monitors the Job Queue

Application	Function
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.

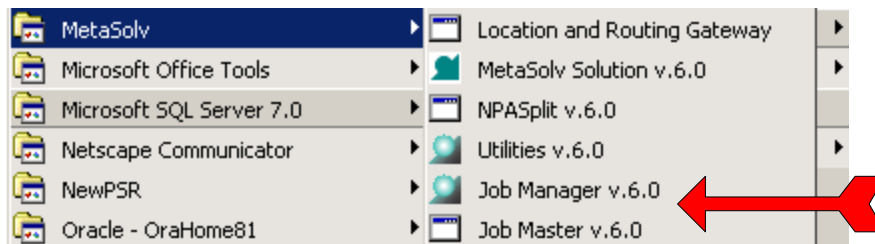


Figure 25: MetaSolv icon group

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.

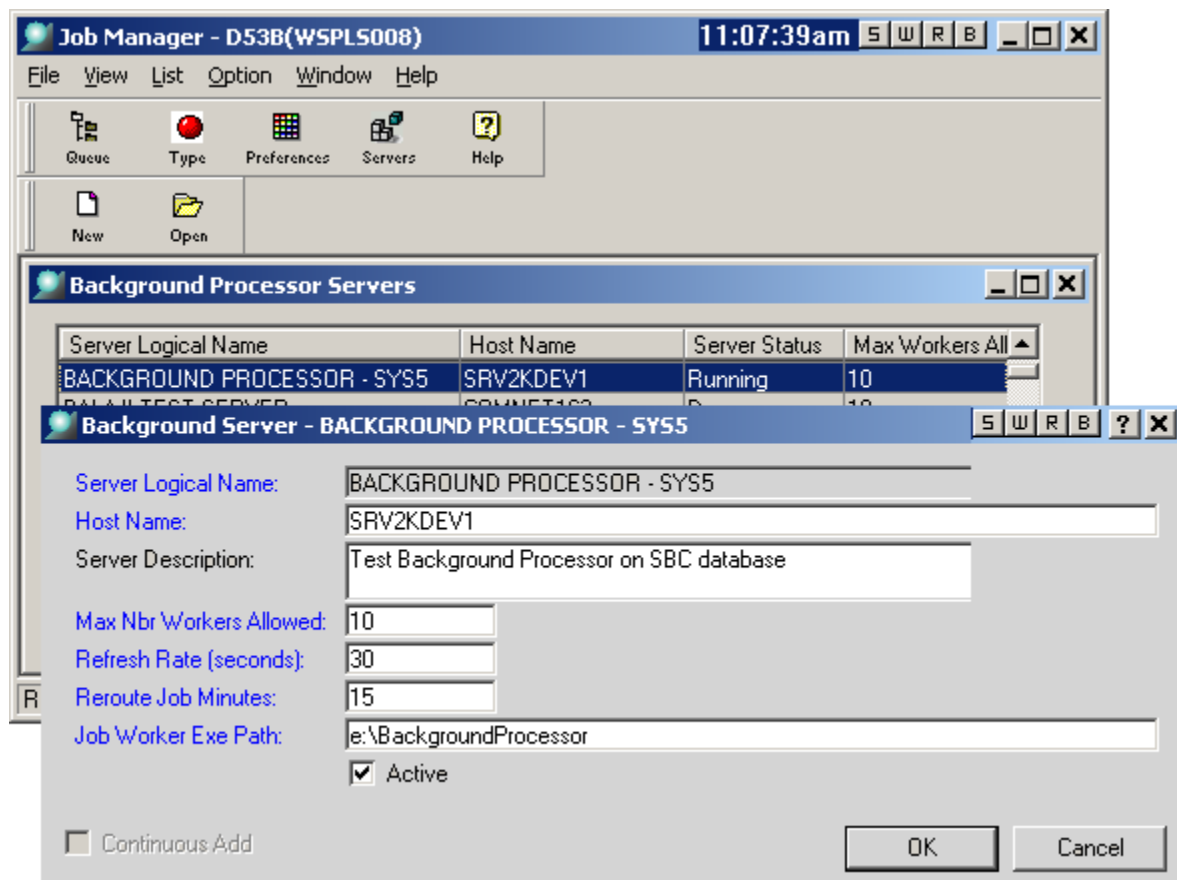


Figure 26: Job Manager GUI

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:

1. Click the **Start** menu.
2. Select **MetaSolv**.
3. Click **Job Master v.6.0**.

Following is the Job Master notification window that appears after you click OK.



Figure 27: Job Master notification window

Once you see this window and 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:

1. 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.
2. 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.

3. The Job Master then checks the Job Manager for the number of Job Workers it is allowed to run simultaneously.
4. 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.

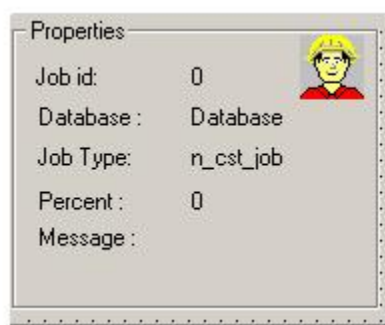


Figure 28: Job Worker informational window

The Job Worker process:

1. 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.
2. The Job Worker runs the job (for example, assigns trunks) using the MetaSolv Solution and appropriate APIs.
3. 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.



Warning! 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 MetaSolv's Global Customer Care 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 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.

Collocations 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 an application server that is separate from the file server. The application server executes both Job Master and Job Worker.

There are performance considerations when running multiple Background Processors. For details, see “Configuring multiple Background Processors” on page 107.

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

At installation, the Jmaster.ini and Tbs.ini files are identical. The Jmaster.ini file must be located with the Jmaster.exe so that Job Master can use it to initiate Job Workers. Job Worker depends on finding the actual profile in the Jmaster.ini file so that it can log on to the system. All database profiles that the Background Processor must access must be defined in the Jmaster.ini. If the database profiles are not defined, the Job Worker cannot process the job.

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 profiles must be included in Jmaster.ini:

- ◆ PURGEUTIL
- ◆ WorkerDB

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.
```

Starting with the 6.0.4 version, these profiles are added automatically during a full installation. If you have an earlier release, you must add the profiles manually. See the following sample Jmaster.ini file to see what the profiles include.

Sample Jmaster.ini file

```
[DBMS_Profiles]
[Application Servers]
J2EEProfiles='D53B(wsplwss01) '
[J2EEProfile D53B(wsplwss01)]
url=http://wsplwss01:8001
database=D53B

[PROFILE PURGEUTIL]
DBMS=O84
Database=Oracle7F
UserId=

DatabasePassword=
LogPassword=
ServerName=d60
LogId=
Lock=
Prompt=1 DbParm=DisableBind=0,DelimitIdentifier='No',CommitOnDisconnect='No'
AutoCommit=0

[PROFILE WorkerDB]
DBMS=O84
Database=Oracle7F
UserId=
DatabasePassword=
LogPassword=
ServerName=d60
LogId=
Lock=
Prompt=1 DbParm=DisableBind=0,DelimitIdentifier='No',CommitOnDisconnect='No'
AutoCommit=0
```

The following table describes the parameters in the Jmaster.ini file.

Table 16: 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 application server ID 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



Warning! 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.

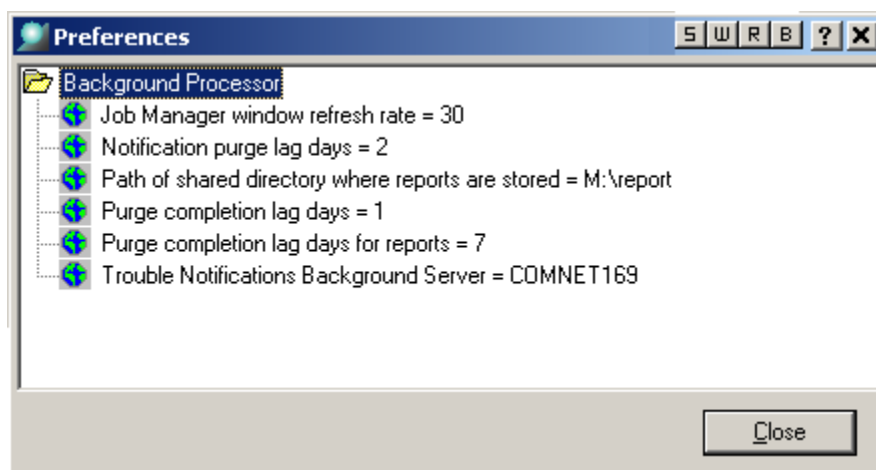


Figure 29: Background Processor Preferences

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 the MetaSolv software 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.
- ◆ **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 server 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.

Multiple databases and multiple Job Masters

Each Background Processor server has its own queue within the database's Job Queue. There is also a first available queue that holds jobs for the first available Background Processor server. At least one Job Master runs on a Background Processor server. At least one Job Master must be connected to each database so the jobs for that database can be processed. Each Job Master processes jobs from one database. Connecting a Job Master to a specific database is done when you start the Job Master and select the desired database. The following illustration shows a configuration with multiple databases.

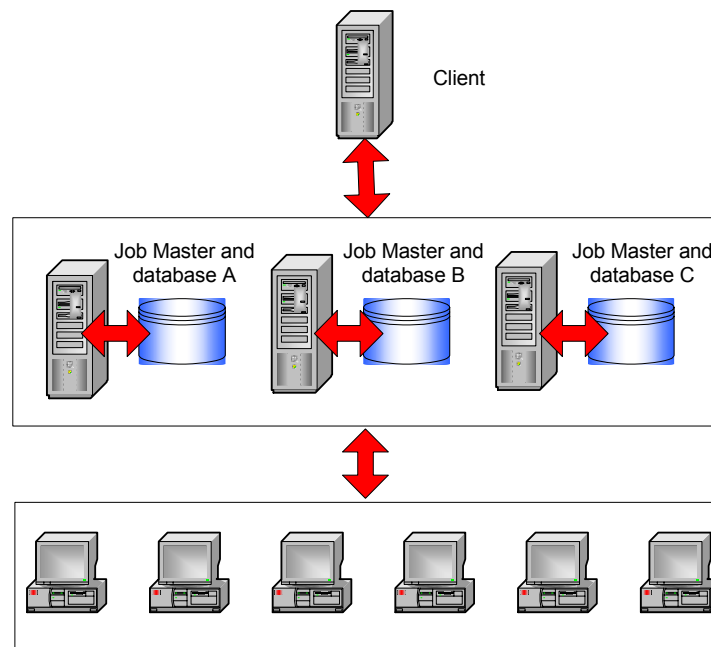


Figure 30: Multiple Job Masters and Multiple Databases

Running multiple Job Masters

To run multiple Job Masters on the same computer, start another copy of Jmaster.exe, keeping these points in mind:

- ◆ Make sure the Jmaster.ini file lists the DBMS profile name for the database that the Job Master runs against.
- ◆ Log on to the database you want when starting the Job Master.

Multiple Job Masters and one database

Multiple Job Masters might be required to process the background load when a single server cannot handle the required number of Job Workers needed to handle the workload in an

acceptable time frame. In this case, multiple Job Masters can be started, each managing its own workers.

Use this architecture only when multiple servers are used. If only a single server is available, increase the number of Job Workers allowed for that server.

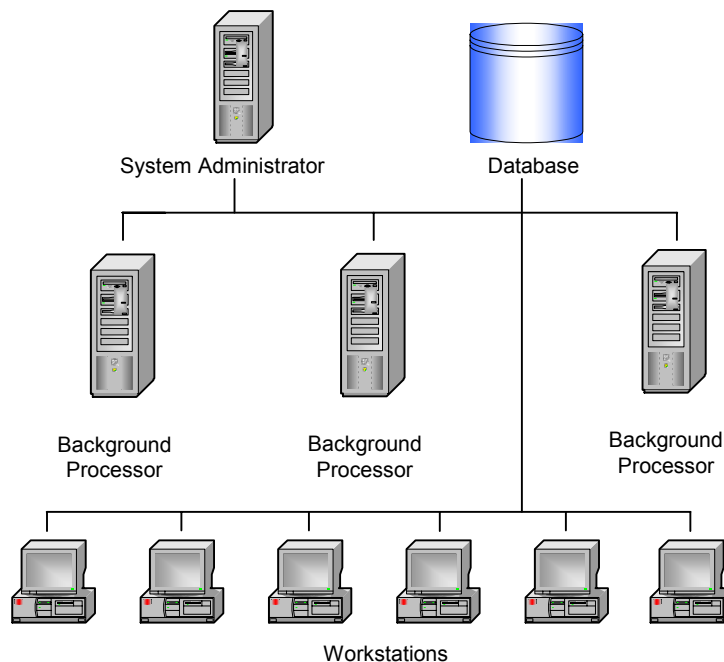


Figure 31: Multiple Job Masters and One Database

Each Job Master and Job Worker maintains a separate database connection. Likewise, the Job Manager maintains a separate connection while it is operating. Each connection counts as a separate database user. If you plan to have a large number of Job Masters and Job Workers accessing the same database, ensure that the database handles the increased number of users.

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 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.

MetaSolv Solution users can view the job queue in the MetaSolv software. This capability creates a virtual individual Job Queue for each user. The individual Job Queue is also available in the Work Management subsystem and the ASR/ISI Gateway's Scheduler.

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 job name errors

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. MetaSolv's Global Customer Care 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.



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

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

- ◆ Purging tasks

The Background Processor 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

The Background Processor 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

The Background Processor 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 online Help topics about schedules for instructions.

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 specific instructions.

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 takes 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.

Log on to the application by selecting **MetaSolv>Location and Routing Gateway** from the **Start** menu.

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. The Location and Routing Gateway can import the following LERG information to MetaSolv Solution:

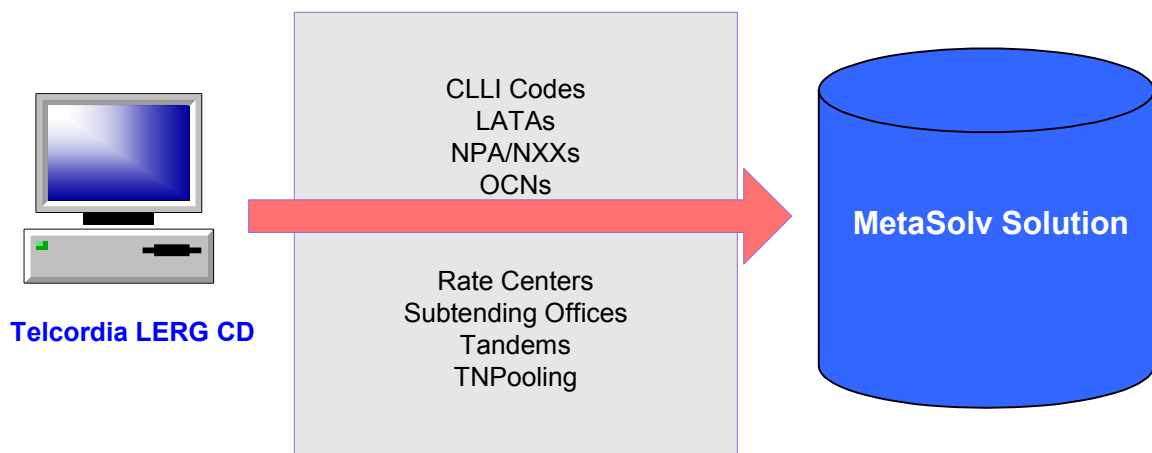


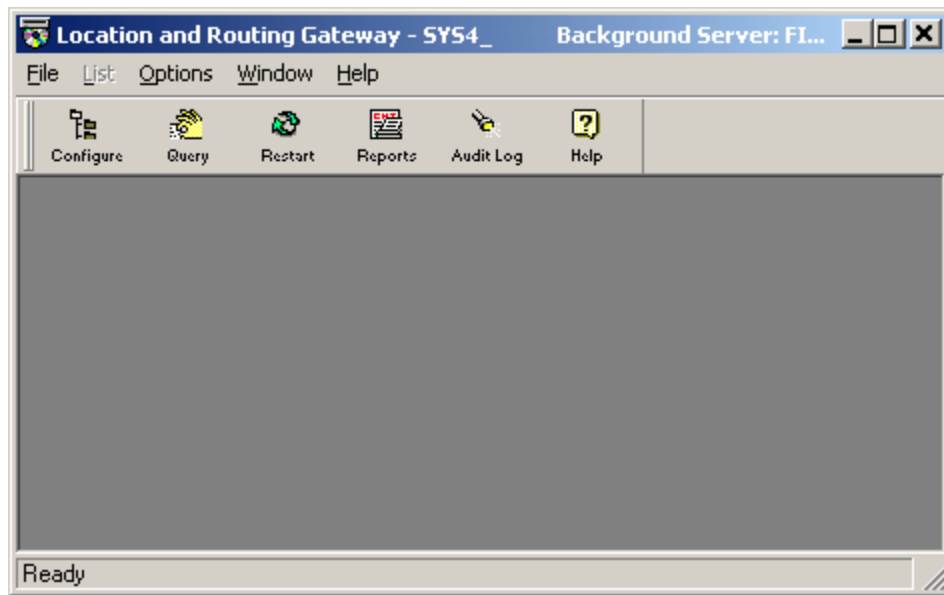
Figure 32: The Location & Routing Gateway flow

Starting the Location and Routing Gateway

To start the application:

1. Click the **Start** menu and click **MetaSolv**.
2. Click **Location and Routing Gateway** in the MetaSolv program group. (See the *MetaSolv Solution Setup Guide* for instructions on using ZAC Start page links.)
3. When the logon dialog appears, use the same ID and password that you use for MetaSolv Solution.
4. Select the database to which the Utilities should connect.

Following is the interface for the Location and Routing Gateway:



Setting up the gateway


Telcordia Technologies periodically changes the format of information fields on the CD and enumerates these changes in the Lergspec.txt 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

 **Warning:** Neglecting to make this Warnings 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 MetaSolv Solution online Help for steps in reconciling the fields.



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 17: LERG1

Field	MetaSolv Solution window name
Operating Company Name	w_company_list
Operating Company Number	w_company_list

Table 18: LERG5

Field	MetaSolv Solution window name
LATA/Boundary	w_lata_list
LATA/Boundary Name	w_lata_list

Table 19: 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
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

Table 19: LERG6

Field	MetaSolv Solution window name
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 20: 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
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

Table 20: LERG7

Field	MetaSolv Solution window name
Switch Address - City	w_network_location_maint w_city_list
Switch Address - State	w_network_location_maint

Table 21: LERG8

Field	MetaSolv Solution window name
LATA/Boundary	w_netarea_definition_maint
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 22: 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

Table 23: LERG13

Field	MetaSolv Solution Access
Block Status	List>Network Location>NXX Assign
Block Effective Date	List>Location>NPA NXX
NPA	List>Location>NPA NXX
NXX	List>Location>NPA NXX
Line Range for Block	List>Network Location>NXX Assign
Operating Company Number	Order Management Setup>Operating Company Location & Geography Setup>CO Exchange Area
Administrative Operating Company Number	N/A
Switch CLI Code	Location & Geography Setup>CO Exchange Area Location & Geography Setup>Network Location

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.



MetaSolv software 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.



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 **Preview Print** button on the toolbar.
3. Click the **Preview Print** 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. Click the **Start** menu and click **MetaSolv**.
1. Click **Utilities 6.0** in the MetaSolv program group. (See the *MetaSolv Solution Setup Guide* for instructions on using ZAC Start page links.)
2. When the logon dialog appears, use the same ID and password that you use for MetaSolv Solution.
3. Select the database to which the Utilities should connect.

Following is the interface for the MetaSolv Solution Utilities:

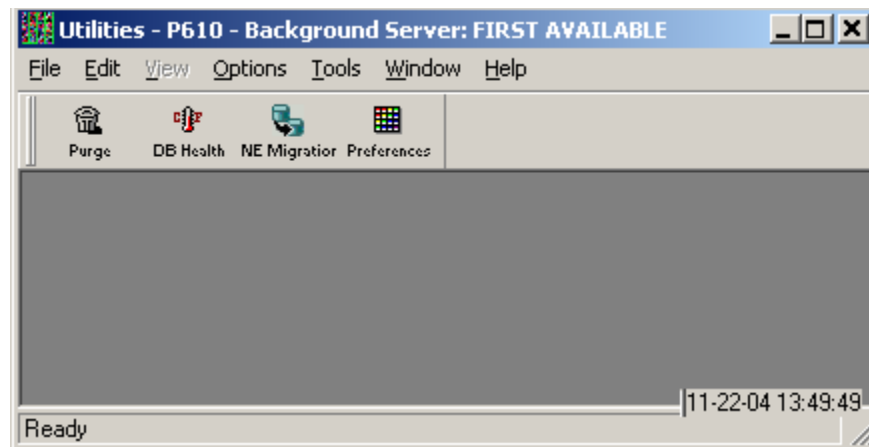


Figure 33: MetaSolv Solution Utilities

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 Utility**—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.
- ◆ **DB Health Utility**—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, finds the fastest access path, fixes errors, and checks rules.



The Oracle Client must be installed on the client machine before the Purge Utility can be used. See the *MetaSolv Solution Setup Guide* for details and instructions.

- ◆ **Network Element Migration**—The Next Generation Migration (NGM) tool allows you to create and convert network elements and to migrate pre-5.1 connections, Virtual Layout Records, and reconcile design lines to Release 6.0.3.

Options Menu utilities:

The **Options** menu provides access to the two 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.
- ◆ **Geographical Area Types**—Geographic areas identify the appropriate format for addresses, telephone numbers and switch prefixes.
- ◆ **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”](#) on page 142 for information about Structured Formats options.
- ◆ **Migration tool utility**—The Migration tool utility provides navigation for the migration process.

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.



Warning: Be cautious when using the WarningsPurge 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
- ◆ 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** checkbox if you wish to do so.
5. Click the **Start** button on the toolbar.

Refer to the online Help for field definitions.



Warning: 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:

- ◆ ACTIVATION_COMMAND_PLAN

- ◆ ASAP.EMP_APPT
- ◆ ASAP.SRC_TASK_COMPLETION
- ◆ DLR_LINE_FLOW_THROUGH_OBJECT
- ◆ FLOW_THROUGH_COMMAND
- ◆ FLOW_THROUGH_ITEM
- ◆ FLOW_THROUGH_OBJECT
- ◆ GATEWAY_EVENT_ERROR
- ◆ GATEWAY_EVENT_ERROR
- ◆ SERV_REQ_GATEWAY_EVENT
- ◆ SRSI_GATEWAY_EVENT
- ◆ TASK
- ◆ TASK_JEOP_WHYMISS_ESC
- ◆ TASK_JEOPARDY_WHYMISS
- ◆ TASK_SEQUENCE
- ◆ TASKCHK

Purging employees

This function deletes an employee and their corresponding work queue(s) 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** checkbox if you wish 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

Purging orders

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.



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** checkbox if you wish to do so.
5. Click the **Start** button.

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

- ◆ ACCESS_PROVIDER_ROLE
- ◆ ACCESS_SERVICE_REQUEST
- ◆ ACI_SRSI
- ◆ ADDITIONAL_CIRCUIT_INFORMATION
- ◆ ANNOUNCEMENT_WINK_VALIDATION
- ◆ ASR_BILLING_ADDRESS
- ◆ ASR_CONTACT_INFORMATION
- ◆ ASR_CONTACT_TYPE
- ◆ ASR_ERROR
- ◆ ASR_RING
- ◆ ASR_SRSI
- ◆ ASR_USER_DATA
- ◆ ASSIGNED_TEL_NUM
- ◆ ATN_CHAM
- ◆ ATN_LIDB
- ◆ ATN_PIC

- ◆ ATN_PIC_INFO
- ◆ ATN_RCF_NP_INFO
- ◆ BAN_USAGE
- ◆ CAPTION_DETAIL
- ◆ CAPTION_SET_DETAIL
- ◆ CIRCUIT_POSITION
- ◆ CIRCUIT_POSTION_PENDING
- ◆ CLARIFICATION_REQUEST
- ◆ CNAM_DISCREPANCY
- ◆ CNAM_EXTRACT
- ◆ COMP_NOTICE
- ◆ CONF_NOTICE
- ◆ CONTACT
- ◆ CPE
- ◆ CUSTOMER_NPA_NXX
- ◆ DESIGN
- ◆ DESIGN_DLR_ASSOC
- ◆ DESIGN_DLR_ISI
- ◆ DESIGN_NOTE
- ◆ DESIGN_ORD_SUMM
- ◆ DIRECTORY_LISTING
- ◆ DIR_DEL_SRSI
- ◆ DIR_DEL_DIR_TYPE
- ◆ DIR_DL_DEL_SRSI
- ◆ DIR_DSR_SRSI
- ◆ DIR_HEADER_SRSI
- ◆ DIR_HEADER_SRSI_REL
- ◆ DIR_LIST_SI
- ◆ DIR_LIST_SRSI
- ◆ DIR_LIST_SRSI_REL
- ◆ DIR_LIST_TXT_SRSI
- ◆ DIR_SERV_CONF
- ◆ DLR_SERV_REQ
- ◆ DL_DESIGN_LINE_SEQUENCE
- ◆ DL_NOTE
- ◆ DSCN_DETAIL

- ◆ DSR_COMP
- ◆ DSR_ERROR
- ◆ DSR_ERROR_CODE
- ◆ DSR_ERROR_DETAIL
- ◆ E911_DISCREPANCY
- ◆ E911_ERROR
- ◆ E911_EXTRACT
- ◆ EMP_APPT
- ◆ END_OFFICE_DETAIL
- ◆ END_OFFICE_TRAFFIC
- ◆ END_USER_LOCATION
- ◆ END_USER_LOCATION_USAGE
- ◆ EUL_ADDITIONAL_DETAIL
- ◆ EUL_LSO
- ◆ FEATURE_GROUP_A
- ◆ FIRED_EVENTS
- ◆ FIRM_ORDER_CONFIRMATION
- ◆ FOC_SWITCHED_CKT_DETAIL
- ◆ FOC_USER_DATA
- ◆ FOC_VIRTUAL_CONN
- ◆ GATEWAY_EVENT_ERROR
- ◆ GATEWAY_EVENT_NOTES
- ◆ HUNT_GRP_DETAIL
- ◆ INTERCEPT_INFO
- ◆ JACK_INFO
- ◆ LINE_DETAIL
- ◆ LOC_SERV_CONF
- ◆ LOC_SERV_REQ
- ◆ LOCAL_ORDER_CONTACTS
- ◆ LOCAL_SERVICE_ORDER_DETAIL
- ◆ LSC_DETAIL
- ◆ LSR_CONTACT_INFO
- ◆ LSR_EUL_ADDL_DATA
- ◆ LSR_EUL_DETAIL
- ◆ LSR_QTY_REQ
- ◆ LSR_SERV_DETAIL

- ◆ LSR_SRSI
- ◆ MS_ATTACHMENT_LINK
- ◆ MSL_CONTACT_TYPE
- ◆ MSL_SRSI
- ◆ MULTIPOINT_SERVICE_LEG
- ◆ N950_ACCESS_NUMBER
- ◆ NAI_CKT_DETAIL
- ◆ NAI_CLLI_LOCATION
- ◆ NAI_INTERMEDIATE_CFA
- ◆ NETWORK_ASSIGNMENT_INFO
- ◆ NODE_PORT
- ◆ NOTES
- ◆ NOTIF_BKGRD_PROC
- ◆ NOTIFICATION
- ◆ NOTIFICATION_PARTY
- ◆ PAIR
- ◆ PAIR_PENDING
- ◆ PARTY_PREPAY
- ◆ PBI_BILL_DISCREPANCY
- ◆ PBI_BILL_GROUP
- ◆ PBI_BILL_ERROR
- ◆ PBI_BILL_TRANSACTION
- ◆ POINT_CODE_INFORMATION
- ◆ PORT_ADDRESS
- ◆ PORT_ADDRESS_PENDING
- ◆ PURGE_ERROR
- ◆ PURGE_ERROR_MESSAGE
- ◆ PP_PARTY
- ◆ PSR_USER_DATA
- ◆ RCONF_ECHO
- ◆ RCONF_VC_ECHO
- ◆ REMARK
- ◆ REQUEST_FORM
- ◆ REQ_HEADER_TEXT
- ◆ RING_SRSI
- ◆ SAALD_LOCATION_TYPE

- ◆ SERV_ITEM
- ◆ SERV_ITEM_REL
- ◆ SERV_ORD
- ◆ SERV_REQ
- ◆ SERV_REQ_CONTACT
- ◆ SERV_REQ_GATEWAY_EVENT
- ◆ SERV_REQ_SI
- ◆ SERV_REQ_SI_PRICE
- ◆ SERV_REQ_SI_USAGE_PRICE
- ◆ SERV_REQ_SI_VALUE
- ◆ SERVICE_ITEM_TRBL_TASK
- ◆ SERVICE_REQUEST_CIRCUIT
- ◆ SI_XREF
- ◆ SPEC_ACCESS_AND_LINE_DETAIL
- ◆ SPLIT_INTERCEPT
- ◆ SRSI_COMMON_BLOCK
- ◆ SRSI_LINE
- ◆ SRSI_PARTY_ROLE
- ◆ SRSI_SERV_REQ
- ◆ SRSI_SR_LOC
- ◆ SRSI_TN_RANGE
- ◆ SRSI_TREATMENT
- ◆ SR_EUL
- ◆ SR_LOC
- ◆ SR_PARTY_ROLE
- ◆ SR_PARTY_ROLE_ACCESS
- ◆ SR_RELATIONSHIP
- ◆ SR_RELATIONSHIP_TASK
- ◆ SR_RESP_ORG
- ◆ SR_SALES_MOD
- ◆ SR_SERV_REQ_DISC
- ◆ SR_SERVICE_ITEM
- ◆ SR_SI_LOCATION
- ◆ SR_SI_ROLE
- ◆ SR_STATE
- ◆ SR_STATUS

- ◆ SRC_TASK_COMPLETION
- ◆ SVCREQ_PROVPLAN
- ◆ TASKCHK
- ◆ TASK_ARCHIVE
- ◆ TASK_JEOPARDY_WHYMISS
- ◆ TASK_RELATIONSHIP
- ◆ TASK_SCHEDULE
- ◆ TASK_SEQUENCE
- ◆ TASK_STAGE
- ◆ TEL_NUM_INV_LISTING_INFO
- ◆ TERMINAL_PAIR
- ◆ TESTING_SERVICE_REQUEST
- ◆ TESTING_SERVICE_REQUEST_LIMITS
- ◆ TQ_ADDITIONAL_CIC_INFO
- ◆ TQ_COMMON_INFORMATION
- ◆ TQ_FGD_INFORMATION
- ◆ TQ_SAC_ACTIVITY_INFORMATION
- ◆ TQ_SAC_NXX
- ◆ TRANS_CALL_INFO
- ◆ TRANSLATION_QUESTIONNAIRE
- ◆ TRBL_TASK
- ◆ TRBL_TASK_SEQUENCE
- ◆ TRBL_TASK_SR_PR
- ◆ TRBL_TICKET_ESC_LEVEL
- ◆ TROUBLE_NOTE
- ◆ TROUBLE_TICKET
- ◆ TRUNK_SIDE_DETAIL
- ◆ USO_CIRCUIT_DETAIL
- ◆ USO_CIRCUIT_LOC
- ◆ USO_ORDER_DETAIL
- ◆ USO_USOCS
- ◆ VIRTUAL_CONNECTION
- ◆ VLR
- ◆ VLR_BANDWIDTH_CKT
- ◆ WATS_ACCESS
- ◆ WFP_TASK_PARTICIPATION

- ◆ WORK_QUEUE_TASK
- ◆ 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** checkbox if you wish to do so.
5. Click the **Start** button.

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

- ◆ ACCESS_SERVICE_REQUEST
- ◆ ADDITIONAL_CIRCUIT_INFORMATION
- ◆ ASR_RING
- ◆ BANDWIDTH_ALLOCATION
- ◆ BANDWIDTH_CKT
- ◆ BANDWIDTH_CKT_BIT_RATE
- ◆ CABLE_PAIR_SET
- ◆ CIRCUIT
- ◆ CIRCUIT_LAYOUT_REPORT
- ◆ CIRCUIT_POSITION
- ◆ CIRCUIT_POSITION_CONDITION
- ◆ CIRCUIT_POSITION_PENDING
- ◆ CIRCUIT_USER_DATA
- ◆ CIRCUIT_XREF
- ◆ DESIGN
- ◆ DESIGN_DLR_ASSOC DDA
- ◆ DESIGN_DLR_ISI
- ◆ DESIGN_GLR_ASSGNMT_SEGMENT
- ◆ DESIGN_LAYOUT_REPORT

- ◆ DESIGN_NSC_REL_CA_VALUE
- ◆ DESIGN_ORD_SUMM
- ◆ DESIGN_SI_CA_MULTI_VALUE
- ◆ DESIGN_SI_CA_VALUE
- ◆ DL_DESIGN_LINE_SEQUENCE
- ◆ DL_NOTE
- ◆ DLR_CIRCUIT_DESIGN_LINE
- ◆ DLR_NOTES
- ◆ DLR_NSC_REL_CA_VALUE
- ◆ DLR_SI_CA_MULTI_VALUE
- ◆ DLR_SI_CA_VALUE
- ◆ DS_DVRSTY_SET_CIRCUIT
- ◆ DSGN_GLR_ASSGN_SEG_NI_NBR_INV
- ◆ DV_ASSIGN_LOG
- ◆ DV_ASSIGN_LOG_DVRSTY_SET
- ◆ E911_DISCREPANCY
- ◆ END_USER_LOCATION
- ◆ FACILITY_CAPABILITY
- ◆ FIRM_ORDER_CONFIRMATION
- ◆ FOC_PROVIDER_CIRCUIT
- ◆ FOC_SWITCHED_CKT_DETAIL
- ◆ GCD_DLR
- ◆ GCD_PATH_CIRCUIT
- ◆ GE_NET_ELEMENT_GATE_EVENT_INST
- ◆ GE_NETWORK_EVENT_INST_ERROR
- ◆ GLR_ASSGN_SEG_CKT_POS
- ◆ NAI_CKT_DETAIL
- ◆ NOTES
- ◆ NS_COMP_REL_NS
- ◆ NS_CON_LABEL_CKT
- ◆ NS_CON_REL
- ◆ NS_CON_REL_CA_VALUE
- ◆ NS_CONNECTION
- ◆ NS_MULTI_POINT_COMPONENT
- ◆ NS_NS_CON
- ◆ PAIR_PENDING

- ◆ PORT_ADDRESS_PENDING
- ◆ RESERVATION_CIRCUIT_POSITION
- ◆ SAALD_LOCATION_TYPE
- ◆ SERV_REQ
- ◆ SERVICE_REQUEST_CIRCUIT
- ◆ SPEC_ACCESS_AND_LINE_DETAIL
- ◆ SPECIAL_CIRCUIT
- ◆ SRC_TASK_COMPLETION
- ◆ TRUNK_SIDE_DETAIL
- ◆ USO_CIRCUIT_DETAIL
- ◆ USO_CIRCUIT_LOC
- ◆ 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:

- ◆ ACTIVATION_COMMAND_PLAN
- ◆ CIRCUIT_LAYOUT_REPORT
- ◆ DESIGN_DLR_ASSOC
- ◆ DESIGN_LAYOUT_REPORT
- ◆ DLR_CIRCUIT_DESIGN_LINE
- ◆ DLR_ERROR
- ◆ DLR_LINE_FLOW_THROUGH_OBJECT
- ◆ DLR_NOTES
- ◆ 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 item(s) 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:

- ◆ RESERVATION

- ◆ RESERVATION_CIRCUIT_POSITION
- ◆ RESERVATION_PAIR
- ◆ 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 updates the database and helps reduce errors, makes response time quicker, and eliminates problems in processing information. While DB Health is required only after an installation, service pack 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 the *MetaSolv Solution Setup Guide* for details.

Managing database health is a simple, two-step process:

1. Check DB Health.
2. 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 the **DB Health** button.
3. Click each tab to review the status of the database.

The database health function checks six different types of problems. The numbers on each tab represent how many database health errors were found. The tabs are:

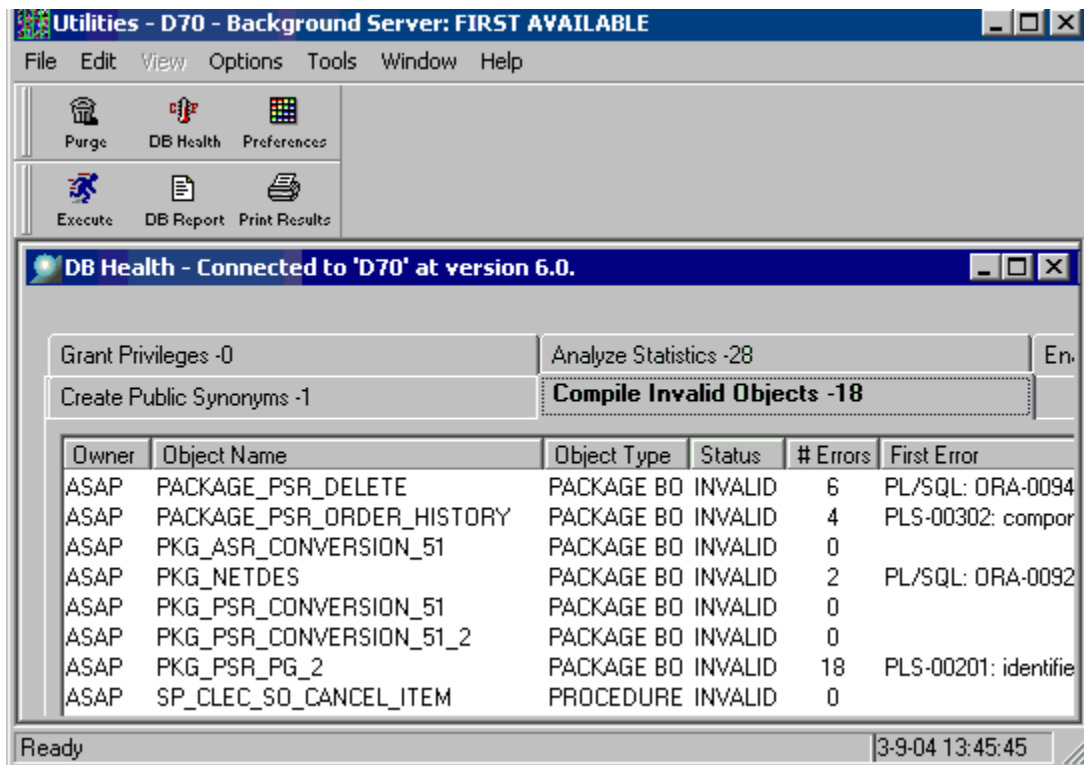
Table 24: DB Health tabs

Tab	Description
Grant Privileges	Fixes tables that aren't set up completely due to the necessary rights not being granted for those tables.
Create Public Synonyms	Supports specific code requiring synonyms. MetaSolv Solution creates a synonym for each database object.
Analyze Statistics	Provides information on the fastest path to access information in the database. It shows tables that do not have access information. Also, when data is added to the database over a period of time, the information on the quickest access to the information becomes obsolete.
Invalid Sequences	Performs a database object check and adds new rows to tables within the database. It also verifies that the sequence has the appropriate values for the next sequence statement. This function can take some time to perform, so schedule this procedure at a convenient time.
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.

Table 24: DB Health tabs

Tab	Description
Enable Disabled Objects	Attempts to correct any disabled foreign keys and checks constraint data enforcement rules. Data can conflict with the rule requiring the data to be fixed before enabling the constraint.

The following sample shows errors that might display after you check DB Health:



Repairing database errors

To repair database errors using the DB Health utility:

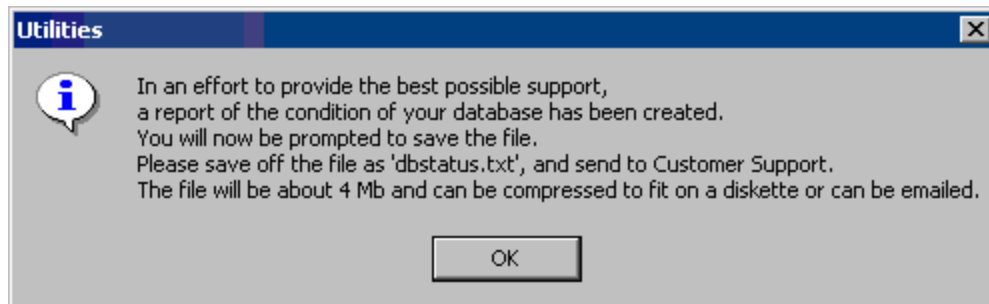
1. Click each tab.
2. Click the **Execute** button when errors are listed.
3. Repeat these steps for each tab that lists health errors.

Generating the DB Status report

Use this report to provide MetaSolv's Global Customer Care with database statistics vital to the health of the database.

To run the DB Report:

1. Click the **DB Health** button.
2. Click the **DB Report** button.
3. The following message: appears:



4. Read the message box and click **OK**.
5. Save the report as **dbstatus.txt** and send to MetaSolv Global Customer Care.

Using the Migration utility

The Next Generation Migration utility provides navigation for the migration process. Menu items are listed in the order in which tasks should be performed. For procedures and conceptual information about the migration, see the *Migration Guide*.

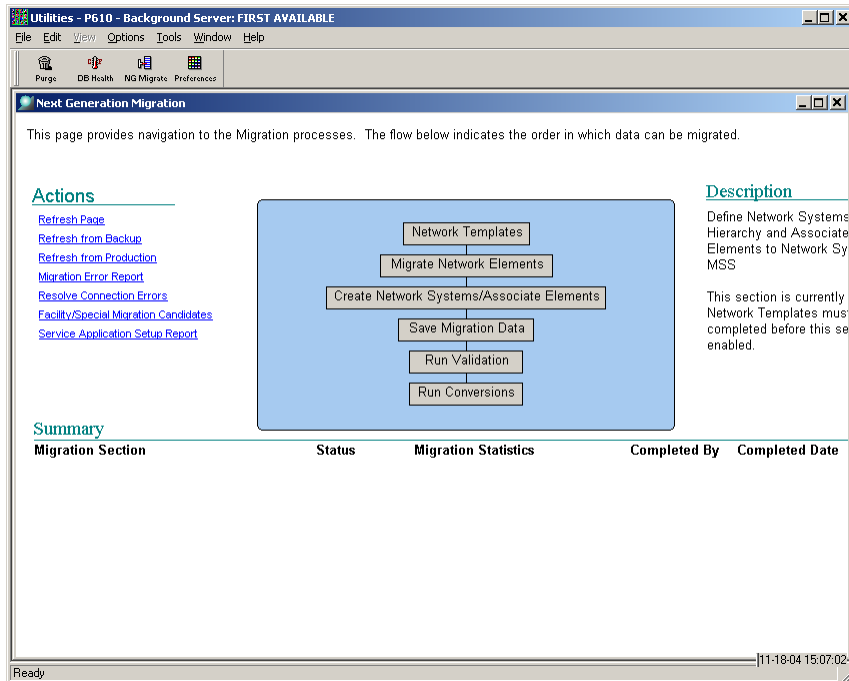


Figure 34: Migration Utility

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 immediately available 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 CA's:

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.



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.



Warning: Because Warningsstructured 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 the *Structured Formats Best Practice Guide* located on the Oracle E-Delivery Web site.

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. While some steps may have changed for Release 6.0, the conceptual information remains valid.

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.



Warning: 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



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. Log on to the NPA Split utility as the ASAP user. (The ASAP user is created during installation.)
2. Open the Start page.
3. Click the NPA NXX utility link.

You can also double-click the npasplit.exe in the client directory. Refer to the NPA Split Utilities Help for instructions about operating the interface.

The main NPA Split Utility window looks like the following sample:

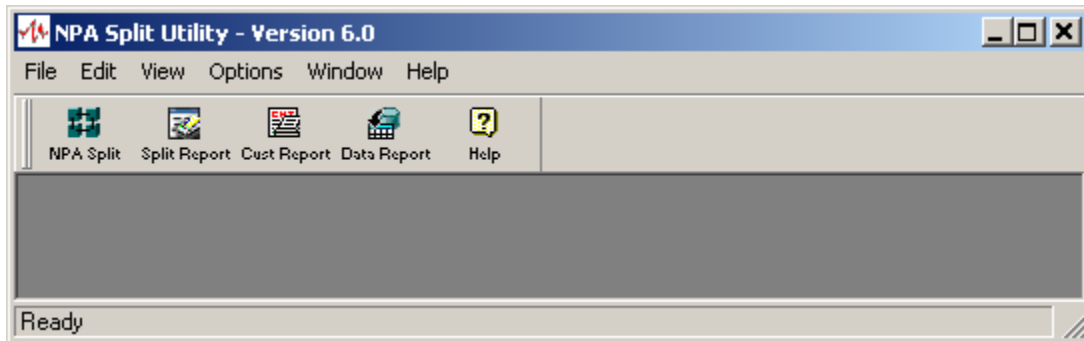


Figure 35: NPA Split Utility window

Use the buttons on the main toolbar to perform the tasks described below:

Table 25: 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.

Troubleshooting and FAQs

This chapter contains information about the following:

- ◆ Common system administration problems
- ◆ Frequently-asked general questions
- ◆ Common application 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 application server port or server name
- ◆ You may need to correct the database instance entry in Tnsnames.ora
- ◆ Make sure the MetaSolv Solution Application is deployed on the BEA 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 [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 BEA server using the Administration Console.

- ◆ Turn off unnecessary logging in Loggingconfig.xml.
- ◆ Make sure Server Name -> Configuration -> File -> Reload Period is set to -1.
- ◆ 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?”](#) on page 152 for special instructions.
- ◆ Run the mdl_cmp.sql script to help maintain the database structure for primary and foreign keys, indexes, and sequences.

The application server does not start

If the BEA application server will not start up, try the following:

- ◆ Start the Admin Server using the startMSLVadmin script in the domain directory.
- ◆ Start the BEA server.
- ◆ Run the startweblogic.cmd from a cmd prompt. If the BEA license has expired, you will see messages in the BEA console. If you see messages about an expired or invalid license, you should contact MetaSolv Global Customer Care.
- ◆ If you have a valid license and BEA 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 BEA was successfully started. BEA 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 the mslvappserver alias in the Tnsnames.ora file found in the location specified by the following property in /appserver/gateway/gateway.ini:

OracleThinTNSNamesLocation=/lpt/app/oracle/product/9.2.0/network/admin/tnsnames.ora.

A link on the Start page is not working

If a blank window with a logo appears, check the following:

- ◆ Are the browser settings correct? See the MetaSolv Solution Setup Guide for instructions.
- ◆ Is the parameter granting access to the EXE set to true in the Zac_config.xml file? See the *MetaSolv Solution Setup Guide* for details.

-
- ◆ Find out if an error message appeared during the download. If there was a message, ask the user to send you a copy of the ZAC.log file in the MetaSolv Solution desktop Config folder. This log file will contain copies of any download error messages.
 - ◆ Perhaps the option or software is an extra-cost feature that your company has not purchased.

I can't 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 didn't 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”](#) on page 97 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 the MetaSolv software or call MetaSolv Global Customer Care for assistance. For guidance about reinstalling, see the *MetaSolv Solution Setup 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.ini, Jmaster.ini, Tbs_util.ini, or Lerg.ini file to be sure it contains a profile for the database in question.

I can't run the Log Viewer

Java must be installed and you must have access to the logviewer.cmd on your machine.

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 Setup 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 BEA 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 enable all of the links on the Start page?

Set the parameters for those products to **True** in the Zac_config.xml file on the application server. See the *MetaSolv Solution Setup Guide* for details.

How do I find out the names of the JAR files?

Display the Runtime page according to directions in [“Getting application server runtime information”](#) on page 43.

Can I "un-do" a patch?

Yes. For instructions about uninstalling the product, see the *MetaSolv Solution Setup 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. WebLogic-generated log errors can be viewed from the

Administration Console at **Servers > servername > General >**. Click the link for **View server log** at the bottom of the page.

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 Administration Console for both connection pools. At JDBC > Connection Pools > MSLVPool > General >, click **Change** next to **Password** > **Enter new password**. Repeat for MSLVDbTracePool.

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=mss6api
```

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=metasolv
Factory=weblogic.jndi.WLInitialContextFactory
URL=t3://localhost:9001
Local=true
```

What are the default passwords?

- ◆ The default password for APP_API is **mss6api**
- ◆ The default password for APP_MSLV is **mss6mslv**

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 BEA 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=metasolv
Factory=weblogic.jndi.WLInitialContextFactory
URL=t3://localhost:9001
Local=true
```
- ◆ Affected ini files in /appserver/config - these must be distributed to client machines manually or using ZAC. Refer [“Using the Zero Admin Client \(ZAC\) to deliver changes”](#) on page 37 for help.

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.

For detailed answers to common BEA questions, or for any other BEA question, see the BEA WebLogic Server documentation at <http://e-docs.bea.com/>.

How do I know which database my application server is connected to?

Display the Runtime page according to directions in [“Getting application server runtime information”](#) on page 43.

How do I know when to restart the server or the database?

- ◆ For a Gateway.ini change, restarting the application server is required.
- ◆ In the BEA Administration Console, any parameter that requires a restart after changing it has a yellow warning icon next to it. The warning icon pops up when you click the parameter.

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 application server or redeploy the application (nur.ear) in order to reload the portlet registry.
 - ◆ If the application server is disconnected from the database server due to network problems or database shutdown, the connection pools of the WebLogic server need to be reset after problems are resolved. The administrator can either restart the application server directly or use the Runtime page drop-down to reset the connection pools.
 - ◆ Certain key Oracle initialization parameters require Oracle to be shut down and restarted. Refer to the Oracle documentation for assistance.

Can I redirect an existing application server to another database?

Yes. To do so, you must change:

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 and APP_API if necessary.

How can I tell the version of my application server?

Display the Runtime page according to directions in [“Getting application server runtime information”](#) on page 43.

How do I reconnect the application server if the database goes down?

Use the RESET_POOL and ENABLE_POOL commands to reset the WebLogic connection pools without shutting down the application server. Following are the commands:

```
java weblogic.Admin -url t3://srvplbeta1:11011 -username weblogic -password weblogic RESET_POOL MSLVPool
java weblogic.Admin -url t3://srvplbeta1:11011 -username weblogic -password weblogic ENABLE_POOL MSLVPool
```

How can I force the application server to run as a UNIX background process, and what differences are there when running this way?

The application 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”](#) on page 88 for instructions.

How can I customize my application server startup script?

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

How can I uninstall my application server?

Undeploy the nur.ear from the BEA Administration console. Delete the /metasolv directory. This will not uninstall BEA. Run the BEA uninstall if you want to uninstall the WebLogic software.

Which environment variables are used by my application server?

ORACLE_HOME, BEA_HOME, JAVA_HOME, JWBROKER_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 application server?

Display the BEA 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 BEA 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 BEA Administration Console.

How can I allocate more JVM memory to my application server?

Increase the memory arguments being used when the app server starts. See `/domaindir/startMSLV01.sh` and `startMSLVserver.sh`.

What do I do if users must access the application server from outside a firewall?

You must open the http port to the application server.

How can I install RSA and DSA certificates in order to implement SSL on my application server?

See the BEA documentation for instructions.

How can I debug application server performance problems?

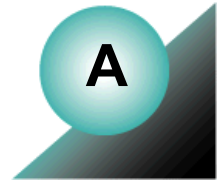
See the performance information in the *MetaSolv Solution Setup Guide*. See the BEA documentation Web site. As a rule of thumb, make changes in small increments; change only one thing at a time and evaluate the effects of the change on performance.

How do I start the JBroker Name Service?

It is started by the `startMSLVorb` script.

How can I control where my IOR files are written?

Use the `Gateway.ini` file `IORPATH=` parameter.



Appendix A: Gateway.ini file parameters

The Gateway.ini file is the main configuration file for MetaSolv Solution and for API interactions with other applications. The application server accesses the Gateway.ini file and gets the configuration settings for managing these interactions from the application server /appserver/config folder.

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 MetaSolv Global Customer Care requests that you change a gateway.ini parameter for research or debug purposes.

Restarting the application server

When you change a setting in the Gateway.ini file, that change does not take effect until the next time you start the application server. Therefore, whenever you change one or more of the Gateway.ini parameters, you must shut down and restart the application 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 application server.

Understanding the Gateway.ini file structure

Gateway.ini 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

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
WMSEVER=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
;ASR24SERVER=MetaSolv.CORBA.WDIASR.WDIRoot,MetaSolv.WDIASR.WDIRootImpl
;ASR25SERVER=MetaSolv.CORBA.WDIASR25.WDIRoot,MetaSolv.WDIASR25.WDIRootImpl
;ASR26SERVER=MetaSolv.CORBA.WDIASR26.WDIRoot,MetaSolv.WDIASR26.WDIRootImpl
;ASR27SERVER=MetaSolv.CORBA.WDIASR27.WDIRoot,MetaSolv.WDIASR27.WDIRootImpl
;ASR28SERVER=MetaSolv.CORBA.WDIASR28.WDIRoot,MetaSolv.WDIASR28.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.WDIInternetSignalServerRootImpl
```

[OrbProperties]

ORBPort=

[Session]

(Note: Starting with release 6.0.5, the parameters DatabaseName, JDBCDriver, and OracleThinTnsNamesLocation are not included.)

DatabaseName=mslvappserver

User=app_api

Password=mss6api

JDBCDriver=oracle

OracleThinTnsNamesLocation=/opt/u01/app/oracle/product/9.2.0/network/admin/
tnsnames.ora

[System]

URLNamingServicePort=15000

StartupMode=server;Startup Mode - gui, batch, server

Environment=prod;Environment - dev, test, prod

DebugOn=true

DebugLevel=1

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

Server=JOB SERVER NAME;Background Processor Logical Name

PSRServerHost=[bind type];[optional: user name];[optional: password];
[ior location];[optional: naming service name]

WMServerHost=[bind type];[optional: user name];[optional: password];
[ior location];[optional: naming service name]

ResidentialSvcCatgId=<ResidentialSvcCatgId>

BusinessSvcCatgId=<BusinessSvcCatgId>

Timeout=601

[SignalInternetServices]

```
;ConnectionManager.WDICreditCard=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIAccount=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIMail=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIDomain=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIWebHosting=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIPSR=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDINI=[bind type];[optional: user name]
    ;[optional: password];[ior location];[optional: naming service name]
;ActivationIniPath=/opt2/metasolv/mslv01/appserver/gateway/activation.ini
```

[Events]

```
DBPollingInterval=30
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEventServer
```

[Event2]

```
DBPollingInterval=30
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEvent2Server
```

[WorkManagement]

```
DbPollingInterval=120
MaxThreads=1
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendSystemTaskServer
```

[JNDI]

```
User=administrator
Password=metasolv
Factory=weblogic.jndi.WLInitialContextFactory
URL=t3://srvsunpluto:7001
Local=true
```

[**CA**]

CacheDisable=false

CacheManagerClass=com.mslv.core.api.internal.ca.cache.broker.CacheManager

CacheRefreshActive=true

CacheRefreshMode=hourly

CacheRefreshDayOfWeek=6

CacheRefreshHourOfDay=14

CacheRefreshHourlyInterval=1

[**Portal**]

DefaultPortalId=asap

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 application server is started.



Warning: Do not alter the server names or startup classes.

Some servers may not be listed in the Gateway.ini initially but are added a separate products are installed.

You can prevent a specific server from starting by removing the preceeding semicolon (;) from 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
WMSEVER=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
;ASR24SERVER=MetaSolv.CORBA.WDIASR.WDIRoot,MetaSolv.WDIASR.WDIRootImpl
;ASR25SERVER=MetaSolv.CORBA.WDIASR25.WDIRoot,MetaSolv.WDIASR25.WDIRootImpl
;ASR26SERVER=MetaSolv.CORBA.WDIASR26.WDIRoot,MetaSolv.WDIASR26.WDIRootImpl
;ASR27SERVER=MetaSolv.CORBA.WDIASR27.WDIRoot,MetaSolv.WDIASR27.WDIRootImpl
;ASR28SERVER=MetaSolv.CORBA.WDIASR28.WDIRoot,MetaSolv.WDIASR28.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.

Table 26: 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.
BillingServer	This parameter starts the BillingServer 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.

Table 26: Servers Parameters

Parameter	Description
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.
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.
WMSEVER	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 components, such as gateway events from the MetaSolv client or jobs from the Background Processor. These server processes receive information from other MetaSolv Software products instead of third-party applications. These ThreadProcs start when the MetaSolv Solution application server starts.



Warning: If the servers in the ThreadProcs section are missing or commented out, the application 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 Global Customer Care.

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
```

These parameters are modified by the install to contain a list of active servers. For descriptions of active servers, see [“Setting up API servers”](#) on page 30.

Table 27: 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 27: Threadprocs parameters

Parameters	Description
EVENTPROC	<p>Default: MetaSolv.eventServer.S3Startup</p> <p>Valid values: MetaSolv.eventServer.S3Startup</p> <p>This parameter starts the Gateway Event server process. The only acceptable value is <i>MetaSolv.eventServer.S3Startup</i>. If this parameter is present, the application server starts the Gateway Event server process when the application server starts. If this parameter is missing or is commented out, the application server does not start the Gateway Event server process and no inbound or outbound gateway events can be processed by the MetaSolv Solution APIs.</p> <p>Do not alter, remove, or comment out this parameter except as instructed by Global Customer Care.</p>
EVENT2PROC	<p>Default: MetaSolv.event2Server.Event2ServerStartup</p> <p>Valid values: MetaSolv.event2Server.Event2ServerStartup</p> <p>This parameter starts the PSR End User Billing server process. <i>MetaSolv.event2Server.Event2ServerStartup</i> is the only acceptable value. If this parameter is present, the Application Server starts the PSR End User Billing server process when the Application Server starts. If this parameter is missing or is commented out, the Application 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.</p> <p>Do not alter, remove, or comment out this parameter except as instructed by Global Customer Care.</p>

Table 27: Threadprocs parameters

Parameters	Description
SYSTEMTASKSERVERPROC	<p>Default: com.mslv.core.ap.internal.WM.systemTaskServer.SystemTaskServer</p> <p>Valid values: com.mslv.core.ap.internal.WM.systemTaskServer.SystemTaskServer</p> <p>This parameter starts the System Task server process. The only acceptable entry is <i>com.mslv.core.ap.internal.WM.systemTaskServer.SystemTaskServer</i>. If this parameter is present, the application server starts the System Task server process when the application server starts. If this parameter is missing or is commented out, the application server does not start the System Task server process and no system tasks can be automatically completed by the MetaSolv Solution.</p> <p>Do not alter or remove this parameter.</p>
SIGNALSERVERPROC	<p>Default: com.metasolv.system.StartServerINTERNET_SIGNAL_SERVER=MetaSolv.CORBA.WDIINTERNETSERVICES.WDIRoot,MetaSolv.SignalServer.WDIInternetSignalServerRootImpl</p> <p>Valid values: com.metasolv.system.StartServer INTERNET_SIGNAL_SERVER=MetaSolv.CORBA.WDIINTERNETSERVICES.WDIRoot,MetaSolv.SignalServer.WDIInternetSignalServerRootImpl</p> <p>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.</p>

OrbProperties parameter

The OrbProperties section designates the port published in the API servers' CORBA IOR files. If this parameter is not set, JBroker will assign a dynamic port number at startup time".

Sample OrbProperties parameter

```
[OrbProperties]
```

```
ORBPort=
```

Table 28: OrbProperties parameter

Parameter	Description
ORBPort	<p>Default: dynamically assigned</p> <p>Valid values: A valid port</p> <p>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 JBroker.</p>

Session parameters

Session parameters control the JDBC connection between the MetaSolv APIs and the Oracle database.

Sample Session section

```
[Session]
DatabaseName=mslvappserver
User=app_api
Password=mss6api
JDBCDriver=oracle
OracleThinTNSNamesLocation=/opt/u01/app/oracle/product/9.2.0/network/admin/tnsnames.ora
```

Table 29: Session Parameters

Parameter	Description
DatabaseName	This refers to the database alias parsed from Tnsnames.ora. Do not alter, remove, or comment out this parameter. Starting with release 6.0.5, this parameter is no longer included in the Session section of Gateway.ini.
User	Default: app_api Valid values: app_api This is the (database) userid is used to make connections to the database for the APIs. This is not a UNIX user.
Password	Default: mss6api Valid values: A valid password This is the (database) password that is used to make connections to the database for the API's. This is not a UNIX user password.
JDBC driver	A keyword for the type of JDBC driver used to make database connections. Do not alter, remove, or comment out this parameter. Starting with release 6.0.5, this parameter is no longer included in the Session section of Gateway.ini.

Table 29: Session Parameters

Parameter	Description
OracleThinTNSNamesLocation	<p>Default: Populated at installation</p> <p>Valid values: The location of the client Tnsnames.ora file.</p> <p>The file is parsed at server startup to build the JDBC connect string.</p> <p>Example:</p> <p>/opt/u01/app/oracle/product/9.2.0/network/admin/tnsnames.ora</p> <p>Starting with release 6.0.5, this parameter is no longer included in the Session section of Gateway.ini.</p>
validateUserAndPassword (optional parameter)	<p>Default: none</p> <p>Valid values: true, false</p> <p>Add this parameter and one or more server name parameters to turn on logon validation for a particular API server. See the example below.</p> <ul style="list-style-type: none"> ◆ securePSRServer ◆ secureDLRServer ◆ secureInfrastructureServer ◆ secureJobQueueServer ◆ secureLSRServer ◆ secureNIServer ◆ securePSREUBServer ◆ securePSRAncillaryServer ◆ securePSRServer ◆ secureSOAServer ◆ secureTMServer ◆ secureWMServer <p>Example:</p> <pre>validateUserAndPassword=true securePSRServer=true</pre> <p>Starting with release 6.0.5, this parameter is no longer included in the Session section of Gateway.ini.</p>

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=
DebugLevel=
LoggingOn=
LogFileDir=
LogToFile=
LogToSystemOut=
AppendToLog=
IORPath=/opt2/metasolv/mslv01/appserver/ior
TraceLevel=
;MaxThreads=1000
```

Table 30: 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.

Table 30: System parameters

Parameter	Description
DebugOn DebugLevel LoggingOn LogFileDir LogToFile LogToSystemOut AppendToLog TraceLevel	<p>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. BEA logging can be controlled through the WebLogic Server Administration Console.</p> <p>See “Appendix B: Annotated loggingconfig.xml” for more information on the loggingconfig.xml file.</p>
IORPATH	<p>Default: /opt2/metasolv/mslv01/appserver/ior</p> <p>Valid values: A valid directory name</p> <p>The IORPath represents the physical location where the API server CORBA IOR files will be written when the application server is started.</p>
MaxThreads	<p>Default: 1000</p> <p>Valid values: 1 to a system-specified maximum</p> <p>This specifies the maximum number of threads that the API server will use.</p>

Gateway parameters

This section specifies miscellaneous and internal configuration parameters.

Sample Gateway parameters

```
[Gateway]
PrintExportObjects=true
JobQueueServer=JOB SERVER NAME;Background Processor Logical Name
Server=JOB SERVER NAME;Background Processor Logical Name
PSRServerHost=[bind type];[optional: user name];[optional: password];
[ior location];[optional: naming service name]
WMServerHost=[bind type];[optional: user name];[optional: password];
[ior location];[optional: naming service name]
ResidentialSvcCatgId=<ResidentialSvcCatgId>
BusinessSvcCatgId=<BusinessSvcCatgId>
Timeout=601
```

Table 31: Gateway parameters

Parameter	Description
PrintExportObjects	Default: True Valid values: True, false <i>True</i> causes extra logging. To improve performance, reset this parameter to <i>False</i> . This parameter causes the contents of CORBA container objects to be listed in MetaSolv Global Customer Care API logs, for debugging purposes. <i>True</i> specifies that the object state display be prior to client notification when exporting objects from the server to the client. <i>False</i> specifies no display of object state.
JobQueueServer Server PSRServerHost WMServerHost ResidentialSvcCatgId BusinessSvcCatgId Timeout	Used for internal processing. Do not alter, remove, or comment out these parameters.

SignalInternetServices parameters

This section of the Gateway.ini file is reserved for internal processing. Do not alter or remove these parameters.

```
[SignalInternetServices]
;ConnectionManager.WDICreditCard=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIAccount=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIMail=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIDomain=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIWebHosting=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDIPSR=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ConnectionManager.WDINI=[bind type];[optional: user name]
;[optional: password];[ior location];[optional: naming service name]
;ActivationIniPath=/opt2/metasolv/mslv01/appserver/gateway/activation.ini
```

Parameter descriptions

Table 32: SignalInternetServices parameter

Parameter	Description
All	Reserved for internal processing. Do not alter or remove these parameters.

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
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendEventServer
```

Table 33: Events parameters

Parameter	Description
DBPollingInterval	<p>Default: 30 (seconds)</p> <p>Valid values: number between 1second and 2 million seconds</p> <p>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.</p>
SuspendFileName	<p>Default: /opt2/metasolv/mslv01/appserver/suspendEventServer</p> <p>Valid values: a valid file name</p> <p>Disable this parameter if not needed.</p> <p>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.</p>



Warning: 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
```

Parameter descriptions

Table 34: Event2 parameters

DBPollingInterval	<p>Default: 30 (seconds)</p> <p>Valid values: a number between 1second and 2 million seconds</p> <p>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.</p>
SuspendFileName	<p>Default: /opt2/metasolv/mslv01/appserver/suspendEvent2Server</p> <p>Valid values: a valid file name</p> <p>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.</p>



Warning: 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=1
SuspendFileName=/opt2/metasolv/mslv01/appserver/suspendSystemTaskServer
```

Table 35: WorkManagement Parameters

Parameter	Description
DBPollingInterval	<p>Default: 120 (seconds)</p> <p>Valid value: number between 1second and 2 million seconds</p> <p>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.</p>
MaxThreads	<p>Default: 1</p> <p>Valid value: 1 to system-specified maximum</p> <p>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.</p>

Table 35: WorkManagement Parameters

Parameter	Description
SuspendFileName	<p>Default: /opt2/metasolv/mslv01/appserver/suspendSystemTaskServer</p> <p>Valid value: A valid file name.</p> <p>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.</p> <p>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.</p>

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 Interface™ (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.



If you change the password for the administration user or if you change the HTTP port for the server process, on the BEA management console, you must also make the same changes in the JNDI section.

Sample JNDI parameters

```
[JNDI]
User=administrator
Password=metasolv
Factory=weblogic.jndi.WLInitialContextFactory
URL=t3://srvsunpluto:7001
Local=true
```


Table 36: JNDI parameters

Parameter	Description
User	<p>Default: BEA admin user ID (user input from installer)</p> <p>Valid values: BEA admin user ID (user input from installer)</p> <p>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 MetaSolv Global Customer Care recommends that you split your Web and EJB containers.</p>
Password	<p>Default: BEA admin password (user input from installer)</p> <p>Valid values: BEA admin password (user input from installer)</p> <p>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 Management Console.</p> <p>The administration server must be restarted after changing the password to enable the managed servers to connect with the new password.</p>
Factory	<p>Default: <code>weblogic.jndi.WLInitialContextFactory</code></p> <p>Valid values: <code>weblogic.jndi.WLInitialContextFactory</code></p> <p>The WebLogic factory class used to create the initial context object for accessing the WebLogic naming service.</p> <p>Change this parameter if you had in-house Java developers who subclassed <code>WLInitialContextFactory</code> to customize it. If you are running on an unsupported application server, use their context factory. Both of these are very unlikely scenarios and most users should use the default value.</p>
URL	<p>Default: <code>t3://localhost:8001</code></p> <p>Valid values: A valid URL</p> <p>The URL is used to locate the remote EJB container. Change this parameter if you are running the BEA 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.</p>

Table 36: JNDI parameters

Parameter	Description
Local	<p>Default: true</p> <p>Valid values: true, false</p> <p>Instructs the EJB framework to ignore other settings and just use the local EJB container.</p> <p>Change this parameter to false if MetaSolv recommends that you split your Web and EJB containers. Otherwise it should remain "true".</p> <p>This parameter is used in building up the URL to the EJB container.</p>

CA (custom attributes) 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 application 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 application server. You should determine how often the cache needs to be refreshed and set the options accordingly.



MetaSolv Software recommends daily updates, using the CacheRefreshHourOfDay parameter.

Caching can be completely disabled by setting CacheDisable=true. If disabled, caching will not occur during application 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=hourly
CacheRefreshDayOfWeek=6
CacheRefreshHourOfDay=14
CacheRefreshHourlyInterval=1
```

Using the default values above, the application server will refresh customer attributes every hour of every day.

Table 37: Custom attribute parameters

Parameter	Description
CacheDisable	Default: false Valid values: true, false Determines whether or not the CA cache is built when the application server is started. If set to true, the CA cache is not built when the application server is started.
CacheManagerClass	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 Global Customer Care. This is a java class name.
CacheRefreshActive	Default: true Valid values: true, false Determines whether or not the CA cache is refreshed on the application server on a regular interval. If CacheDisable is false and CacheRefreshActive is true, the application 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..)

Table 37: Custom attribute parameters

Parameter	Description
CacheRefreshDayOfWeek	<p>Default: 6</p> <p>Valid values: 1 - 7</p> <p>The day of the week CA cache is refreshed when the CacheRefreshMode is set to <i>weekly</i>. Day 1 is Sunday, Day 2 is Monday... Day 7 is Saturday.</p>
CacheRefreshHourOfDay	<p>Default: 14</p> <p>Valid values: 0 - 23</p> <p>The hour of day CA cache is refreshed when the CacheRefreshMode is set to <i>daily</i> or <i>weekly</i>. Hour 0 is 12:00 a.m.</p>
CacheRefreshHourlyInterval	<p>Default: 1</p> <p>Valid values: 1 - 23</p> <p>The hourly interval CA cache is refreshed when the CacheRefreshMode is set to <i>hourly</i>.</p> <p>Note: For testing purposes only, if you wish 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.</p>

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 one portlet: 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.

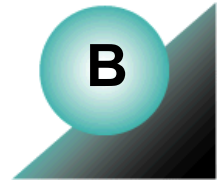
Sample Portal parameter

[Portal]

DefaultPortalId=asap

Table 38: 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.



Appendix B: Annotated loggingconfig.xml

The following sample logging configuration file controls the described aspects of configuration.

Table 39: 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 Global Customer Care.
pluggable interface specification	Implementation classes for the logging framework. This section must never be modified unless instructed by Global Customer Care.
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 " />
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 preceed boldfaced parameters in this sample to understand how to change this file when necessary.

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/">
***** BEGIN 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.
***** Begin of pluggable interface specification ****
***** Never needs to modified by the user ****

    <!--

        The following defines various pluggable implemenation 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.MSLVLogContextFactoryDefaultImp
1"/>
        <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 ****
```

```
***** Begin 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"/>
```

```
    </jdbc-trace>
```

```
    </mslv-apps>
```

```
***** end of SQL Tracing configuration
```

```
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
```

```
***** Begin 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 recieve 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 hierarchially 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 recommmed 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"/>

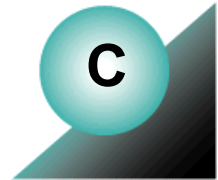
```

```

        <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>
```



Appendix C: Glossary

The glossary contains a list of terms and their definitions as they relate to MetaSolv Solution 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 medium, 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:

- n Call Name Database (Sprint)
- n Calling Name (Caller ID)
- n 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, contiguous "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 MetaSolv Software 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

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 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 of 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 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 operations

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 MetaSolv customers 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 etc.

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 to 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.

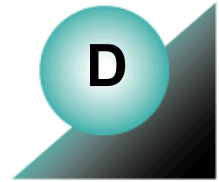
VPI (Virtual Path Identifier)

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.



Appendix D: BEA Application Server configuration settings

This section contains BEA WebLogic Application 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 that do not include the XML API option.

Use this information when tuning the application server. When you make changes, do so in small increments so that you can be sure of the effects without significant negative results.



These recommendations are subject to change pending the results of benchmark testing and feedback on customer usage.

To see recommended hardware for these deployments, see the *MetaSolv Solution Planning Guide*.

Small deployment administration server

Table 40: Small deployment administration server configuration settings

Parameter		Value	Parameter		Value
BEA WebLogic 8.1 SP3	Domain Name	mslvadmin			
Sun JDK	Version	142_04			
JVM	-Xms	768m	-Xmx		768m
	-XX:NewSize	256m	-XX:MaxNewSize		256m
	-server				
Deployment	STARTMODE	TRUE			
Application Backlog	50				
JDBC (MSLVPool)	Initial Capacity	15	Max Capacity		80
	Capacity Increment	1	Allowed Shrinking		Check box selected
JDBC (MSLVDBTracePool)	Initial Capacity	0	Max Capacity		5
	Capacity Increment	1	Allowed Shrinking		Check box selected
ExecuteQueue	Queue Length	256	Queue Length Threshold Percent		5
	Thread Count	15	Threads Increase		2
	Threads Max	100	Threads Minimum		5
	Thread Priority	5			

Table 40: Small deployment administration server configuration settings

	Parameter	Value	Parameter	Value
Memory	Low Memory GCThreshold	20	Low Memory Granularity Level	5
	Low Memory Sample Size	10	Low Memory Time Interval	600
JMS	JMS Thread Pool Size	0		
Bridge	Messaging Bridge Thread Pool Size	1		
JTA	Timeout Seconds	360		
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10
App Server Logging	Log to Stdout	Check box cleared	Debug to Stdout	Check box cleared
	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10
	Log to Domain Logfile	Check box cleared	HTTP Enable Logging	Checkbooks cleared
	Rotation Type	Size	ENABLE JDBC LOGGING	Checkbooks cleared
Web Application	Files Reload Period (for all)	-1		

Small deployment clustered administration server

Table 41: Small deployment clustered server settings

	Parameter	Value	Parameter	Value
BEA WebLogic 8.1 SP3	Server Names	Server1, Server2	Server Port	7063, 7063
Sun JDK	Version	142_04		
JVM	-Xms	768m	-Xmx	768m
	-XX:NewSize	256m	-XX:MaxNewSize	256m
	-server			
Deployment	STARTMODE	TRUE		
Application Backlog	200			
ExecuteQueue	Queue Length	256	Queue Length Threshold Percent	5
	Thread Count	15	Threads Increase	2
	Threads Max	400	Threads Minimum	5
	Thread Priority	5		
Memory	Low Memory GCThreshold	20	Low Memory Granularity Level	5
	Low Memory Sample Size	10	Low Memory Time Interval	600
JMS	JMS Thread Pool Size	0		
Bridge	Messaging Bridge Thread Pool Size	1		
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10

Table 41: Small deployment clustered server settings

	Parameter	Value	Parameter	Value
AppServer Logging	Log to Stdout	Check box cleared	Debug to Stdout	Check box cleared
	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10
	Log to Domain Logfile	Check box cleared	HTTP Enable Logging	Check box cleared
	Rotation Type	Size	Enable JDBC Logging	Check box cleared
Web Application	Files Reload Period (for all)	-1		
Cluster	Name	Cluster280R	MULTICAST IP	237.5.5.5
	Cluster Address	Server1:7063, Server2:7063	MULTICAST PORT	6060
	WebLogic Plug-In Enabled	Check box cleared		

Medium deployment administration server

Table 42: Medium deployment administration server configuration settings

	Parameter	Value	Parameter	Value
BEA WebLogic 8.1 SP3	Domain Name	mslvadomain	Server Name	mslvadmin
Sun JDK	Version	142_04		
JVM	-Xms	768m	-Xmx	768m
	-XX:NewSize	256m	-XX:MaxNewSize	256m
	-server			
Deployment	STARTMODE	TRUE		
Application Backlog	50			
JDBC (MSLVPool)	Initial Capacity	15	Max Capacity	400
	Capacity Increment	1	Allowed Shrinking	Checked
JDBC (MSLVDBTracePool)	Initial Capacity	0	Max Capacity	5
	Capacity Increment	1	Allowed Shrinking	Checked
ExecuteQueue	Queue Length	256	Queue Length Threshold Percent	5
	Thread Count	15	Threads Increase	2
	Threads Max	100	Threads Minimum	5
	Thread Priority	5		
Memory	Low Memory GCThreshold	20	Low Memory Granularity Level	5
	Low Memory Sample Size	10	Low Memory Time Interval	600
JMS	JMS Thread Pool Size	0		

Table 42: Medium deployment administration server configuration settings

Parameter		Value	Parameter		Value
Bridge	Messaging Bridge Thread Pool Size	1			
JTA	Timeout Seconds	360			
MSLVDomain Logging	Rotation Type	By Size	File Min Size		10000
	Number Of Files Limited	Check box selected	File Count		10
App Server Logging	Log to Stdout	Check box cleared	Debug to Stdout		Check box cleared
	Rotation Type	By Size	File Min Size		10000
	Number Of Files Limited	Check box selected	File Count		10
	Log to Domain Logfile	Check box cleared	HTTP Enable Logging		Check box cleared
	Rotation Type	Size	ENABLE JDBC LOGGING		Check box cleared
Web Application	Files Reload Period (for all)	-1			

Medium deployment clustered server

Table 43: 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 44: Medium deployment clustered server configuration settings

Parameter		Value	Parameter		Value
BEA WebLogic 8.1 SP3	Domain Name	mslvdomain			
Sun JDK	Version	142_04			
JVM	-Xms	768m	-Xmx		768m
	-XX:NewSize	256m	-XX:MaxNewSize		256m
	-server				
Deployment	STARTMODE	TRUE			
Application Backlog	200				

Table 44: Medium deployment clustered server configuration settings

	Parameter	Value	Parameter	Value
ExecuteQueue	Queue Length	256	Queue Length Threshold Percent	5
	Thread Count	15	Threads Increase	2
	Threads Max	400	Threads Minimum	5
	Thread Priority	5		
Memory	Low Memory GCThreshold	20	Low Memory Granularity Level	5
	Low Memory Sample Size	10	Low Memory Time Interval	600
JMS	JMS Thread Pool Size	0		
Bridge	Messaging Bridge Thread Pool Size	1		
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10
AppServer Logging	Log to Stdout	Check box cleared	Debug to Stdout	Check box cleared
	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10
	Log to Domain Logfile	Check box cleared	HTTP Enable Logging	Check box cleared
	Rotation Type	Size	Enable JDBC Logging	Check box cleared

Table 44: Medium deployment clustered server configuration settings

Parameter		Value	Parameter	Value
Web Application	Files Reload Period (for all)	-1		
Cluster	Name	ClusterV480	MULTICAST IP	237.5.5.6
	Cluster Address	Server11:70 63 Server12:90 63 Server21:70 63 Server22:90 63 Server31:70 63 Server32:90 63 Server41:70 63 Server42:90 63	MULTICAST PORT	6061
	WebLogic Plug-In Enabled	Check box cleared		

Large deployment administration server

Table 45: Large deployment administration server configuration settings

	Parameter	Value	Parameter	Value
BEA WebLogic 8.1 SP3	Domain Name	mslvadomain	Server Name	mslvadmin
Sun JDK	Version	142_04		
JVM	-Xms	768m	-Xmx	768m
	-XX:NewSize	256m	-XX:MaxNewSize	256m
	-server			
Deployment	STARTMODE	TRUE		
Application Backlog	50			
JDBC (MSLVPool)	Initial Capacity	15	Max Capacity	800
	Capacity Increment	1	Allowed Shrinking	Check box selected
JDBC (MSLVDBTracePool)	Initial Capacity	0	Max Capacity	5
	Capacity Increment	1	Allowed Shrinking	Check box selected
ExecuteQueue	Queue Length	256	Queue Length Threshold Percent	5
	Thread Count	15	Threads Increase	2
	Threads Max	100	Threads Minimum	5
	Thread Priority	5		
Memory	Low Memory GCThreshold	20	Low Memory Granularity Level	5
	Low Memory Sample Size	10	Low Memory Time Interval	600
JMS	JMS Thread Pool Size	0		

Table 45: Large deployment administration server configuration settings

Parameter		Value	Parameter		Value
Bridge	Messaging Bridge Thread Pool Size	1			
JTA	Timeout Seconds	360			
MSLVDomain Logging	Rotation Type	By Size	File Min Size		10000
	Number Of Files Limited	Check box selected	File Count		10
App Server Logging	Log to Stdout	Check box cleared	Debug to Stdout		Check box cleared
	Rotation Type	By Size	File Min Size		10000
	Number Of Files Limited	Check box selected	File Count		10
	Log to Domain Logfile	Check box cleared	HTTP Enable Logging		Check box cleared
	Rotation Type	Size	ENABLE JDBC LOGGING		Check box cleared
Web Application	Files Reload Period (for all)	-1			

Large deployment clustered server

Table 46: 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 47: Large deployment clustered server configuration settings

	Parameter	Value	Parameter	Value
BEA WebLogic 8.1 SP3	Domain Name	mslvdomain		
Sun JDK	Version	142_04		
JVM	-Xms	768m	-Xmx	768m
	-XX:NewSize	256m	-XX:MaxNewSize	256m
	-server			
Deployment	STARTMODE	TRUE		
Application Backlog	200			
ExecuteQueue	Queue Length	256	Queue Length Threshold Percent	5
	Thread Count	15	Threads Increase	2
	Threads Max	400	Threads Minimum	5
	Thread Priority	5		
Memory	Low Memory GCThreshold	20	Low Memory Granularity Level	5
	Low Memory Sample Size	10	Low Memory Time Interval	600
JMS	JMS Thread Pool Size	0		
Bridge	Messaging Bridge Thread Pool Size	1		
MSLVDomain Logging	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10

	Parameter	Value	Parameter	Value
AppServer Logging	Log to Stdout	Check box cleared	Debug to Stdout	Check box cleared
	Rotation Type	By Size	File Min Size	10000
	Number Of Files Limited	Check box selected	File Count	10
	Log to Domain Logfile	Check box cleared	HTTP Enable Logging	Check box cleared
	Rotation Type	Size	Enable JDBC Logging	Check box cleared
Web Application	Files Reload Period (for all)	-1		
Cluster	Name	ClusterV880	MULTICAST IP	237.5.5.7
	Cluster Address	Server11:6063	MULTICAST PORT	6062
		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		
	WebLogic Plug-In Enabled	Check box cleared		

Index

Numerics

999 DBMS 105

A

account number 90, 95
 Active Ticket Queue check point 19
 addresses 123
 Admin Server 148
 administration console 40
 Runtime page link 41
 starting 42
 administrative tasks 1
 aliases 81
 Allow users to customize My Desktop 76
 API server
 logon validation 171
 parameters 164
 startup 30
 AppendToLog parameter 173
 application server
 administration console 42
 applications 26
 architecture 2
 BEA documentation v
 client access 3
 configuration files 13, 23
 customizing startup 154
 database connection 152
 environment changes 33
 environment variables 154
 Gateway.ini 35
 Integration server 32
 Jmaster application 102
 log files 45
 performance 29
 reconnecting 153
 restarting 83, 152
 restartingrestarting
 custom portlets 81
 Runtime page 43
 servername parameter 105
 starting 39
 startup problems 148
 stopping 40

UNIX background process 153
 version 153
 VM memory 155
 WebLogic Server tasks 42
 ASR API 164
 Assign Permissions check point 19
 AutoCommit parameter 105

B

Background Processor
 administrator responsibilities 110
 automatic job submissions 113
 errors 150
 file locations 102
 incomplete job 149
 Jmaster.ini 103
 Job Manager 98
 Job Master 100
 Job Worker 101
 Notifications 106
 performance 109
 preferences 106
 restarting 113
 setup 102
 tuning 109
 backup and recovery methodology 72
 BEA
 administration console 41, 42
 directory 150
 logging 147
 security 42
 startup 148
 benchmarksdatabase
 benchmarks 64

C

CA parameters 183
 Cache parameters 184
 caching recommendation 183
 Circuit ID 90
 circuit ID 89
 coding exits 89
 CommitOnDisconnect parameter 105
 configuration files

- application server26
 - application server performanceapplication server
 - performance 29
 - changing 32, 35
 - client35
 - database 26, 27
 - descriptions24
 - editing32
 - enabling API securityAPI security14
 - environment changes33
 - file changes35
 - functions24
 - Init.ora68
 - relationships26
 - Sharedini= parameter28
 - tracking logons14
 - ZAC37
 - configuration settings
 - medium appserver deployment226
 - medium deployment clustered 226, 228
 - small appserver deployment222
 - connection failure149
 - connection pools3, 41, 45, 58, 148
 - ConnectionManager parameter175
 - custom attributes
 - Custom Attributes tool4
 - Custom Attributes Utility141
 - load183
 - parameters183
 - custom reports85
 - custom validation 89, 91
- D**
- database149
 - adding DBAs61
 - archive log files71
 - asynchronous I/O70
 - baseline reference64
 - blocks 64, 69
 - character set62
 - connections 42, 149
 - copy61
 - data buffer69
 - data cache hit ratio63
 - data model differencesData Model68
 - db_block_buffers 63, 69
 - db_block_size, init.ora64
 - db_file_multiblock_read_count69
 - db_file_multiblock_read_count parameterperformance
 - db_file_multiblock_read_count 69
 - DBA responsibilitiesDBA responsibilitiesresponsibilities
 - DBAAdministrator tasks
 - DBA 60
 - enqueue_waits70
 - error repair139
 - exporting and importing61
 - extended characters62
 - health138
 - import and export61
 - Init.ora68
 - international data62
 - locally vs. dictionary managed Tablespaces61
 - Log_checkpoint_interval parameter70
 - log_checkpoint_timeout parameter70
 - mdl_cur.txt report72
 - MTS-related parameters63
 - optimizer_features_enable, init.ora64
 - optimizer_mode, init.ora64
 - parallel_max_servers, init.ora64
 - performance 64, 66, 68
 - permissions66
 - problem and performance scripts64
 - rebuilding66
 - recovery time70
 - scalability63
 - scalabilityscalability63
 - scriptsscripts64
 - sort_area_retained_size parameter70
 - sort_area_size parameter69
 - sort_write_buffer_size parameter70
 - STATSPACK reports64
 - storage parametersstorage parameters65
 - structure problems64
 - tablespace tablespaces61
 - database parameter105
 - DatabaseName parameter170
 - DatabasePassword parameter105
 - DB Health140
 - DB Status report140
 - db_block_buffers69
 - db_file_multiblock_read_count69
 - DB_Health Utility123
 - db_writer parameterdatabase
 - db_writer parameter70
 - DBMS
 - DBMS profiles108
 - DBMS_STATS60
 - multiple job masters108
 - DBMS parameter105
 - DbParm, parameter105

DBPollingInterval	29, 30
DBPollingInterval parameter	176, 177, 178
ddlcmp.txt	67
debug	
Debug level	52
debugging and monitoring	42
DebugLevel parameter	173
DebugOn parameter	173
Default Background Server	107
default desktop	77
default fields	89
DefaultPortalId parameter	186
DelimitIdentifier parameter	105
dependencies	89
deployments	42
desktop	75
Dial Up Info tab	94
DLR	
issues	136
issues preferences	125
logging category	52
Mass DLR Reconcile check pointMass DLR Reconcile check point	19
Purging DLR issues	136
DLRSERVER parameter	164
dml_locks parameterdatabase dml_locks parameter	63

E

Edit Tasks checkpoint	19
element types	123
Enabled	18
enqueue_resources parameterdatabase enqueue_resources parameter	70
Environment parmparameter	172
event statuses	30
Event2 parameters	177
EVENTPROC parameter	167
Events parameters	176
Exception Queue Access check point	19
Exe files	
application serverapplication server EXE files 26	
Background Processor	97
configuration files	24
Jmanager.exeJmanager.exe	24
Jmaster.exe	103, 113
Jworker.exe	103
Lerg.exeLerg.exe	24
MSSINI	38

Npasplit.exeNpasplit.exe	24
Oracless.exeOracless.exe	69
Tbs.exeTbs.exe	24
Tbs_util.exe	21, 25
TBSGraphicLoad.exe	150
executing queues	42
exits	
coding	89
definition	88
pkg_generate_freeformat_ecckt.sp_generate_ecckt	90
pkg_psr_custom_keys.sp_custom_dialup_password	92
pkg_psr_custom_keys.sp_custom_dialup_userid	92
pkg_psr_custom_keys.sp_custom_email_password	93
pkg_psr_custom_keys.sp_custom_url	94, 95
pkg_psr_custom_keys.sp_custom_webhost_password	93
pkg_psr_custom_keys.sp_custom_webhost_userid	93
sp_NL_custom_validation	91
sp_PSR_cust_acct_number	90, 95
sp_PSR_custom_validation	91
sp_wm_custom_validation	92
expired reservations	136, 137

F

Factory parameter	181
files	
changing	32
collocations	102
configuration	1, 24
database communication	26
database log files	71
INI file relationships	26
Init.ora	68
Jmaster.ini	105
location	35, 99
log files	48
logons	14
ZAC	37

G

Gateway Event server suspension parameter	176
Gateway Events transition	31
Gateway parameters	174
Gateway.ini	
description	24
Gateway.ini file	157
Gateway.ini file sample	159
groups and permissions, definition	9
groups and usersusers and groups	10

I

I/O performance	71
i_rpt.pbl	86
Index Compare report	74
indexes	67
Infrastructure	164
INI files	
after installation	103
installation	99
Job Worker	103
relationships	26
Tbs_db.ini, See Tbs_db.ini file	110
init.ora	24, 68
INTEGRATIONSERVER parameter	166
Internet Dial Up user ID	92
INTERNET_SIGNAL_SERVER parameter	168
Inventory and Capacity Management API	164
IORPATH parameter	173

J

Java Naming and Directory Interface	180
JDBC driver (Session Parameter)	170
Jmanager.exe	102
Jmaster.ini	24, 105
JNDI parameters	180
JNDI parameters parameter	180
Job Manager	
definition	98
INI file	103
Job Manager window refresh rate	106
preferences	106
security	102
users	102
Job Master	
job problems	111
Master.log	112
multiple Job Masters	108
performance	109
preferences	106
starting	100
Job Parameters window	112
Job Queue	
clearing	102, 150
clearing jobs with reports	114
definition	111
management	111
speed	111
tasks	110
Job Status	

Complete	101
Delete	150
Job Type table	101
Job Viewer	111
Job Worker	
definition	101
INI file	103
Jmaster.ini file	103
processing	101

L

LERG	
application setup fields	117
definition	114
flow	115
functions	114
importing data	120
restarting	120
Local parameter	182
Location and Routing Gateway	
configuration	116
loading process	115
Lock parameter	105
log file	
performance	39
log files	45, 112
Appsrverlog.xml	45
Appsrverlog.xmlAppsrverlog.xml log file	48
archiving	49
Background Processor	110
Background Processor Job Log	111
Background Processor types	112
BEA	147
location	49
Loggingconfig.xml	24
performance	39
viewing	46
WebLogic	42
Log Viewer	46, 149
Log Viewer problems	149
LogFileDir parameter	173
logging file specifications	51
Loggingconfig.xml	
file	148
filelog files	
Loggingconfig.xml parameterscon-	
figuration files	
Loggingconfig.xml 50	
severity levels	52

specifications	51
loggingconfig.xml file	187
LoggingOn parameter	173
LogId parameter	105
logon problems	147
LogPassword parameter	105
LogToFile parameter	173
LogToSystemOut parameter	173
LSOG columns	123
LSR API	164
LSRSERVER parameter	164

M

management tasks	1
MANAGER.LOG	112
MASTER.LOG	112
MaxThreads parameter	173, 178
mdl_cmp.sql script	66, 67, 148
mdl_cmp.txt	74
mdl_cmp.txt report	67
mdl_cur.sql	67
mdl_cur.sql script	67
mdl_cur.txt	67, 68
mdl_cur.txt report	68
mdl_upd.cmp	66
mdl_upd.com.txt	67
mdl_upd_cmp.txt	74
mdl_upd_cur.sql	67
mdl_upd_cur.txt	68, 72
mdl_upg.cmp	66
memory	69, 109, 148
memory/disk swapping	71
Migration	28
MTS related parametersdatabase	
MTS related parameters	63
My Navbar	84
my-portlets.xreg	77, 79

N

Network Locations	91
network systems	123
network templates	123
Next Generation Migration utility	141
NISERVER parameter	164
No Access	18
Not Set	18
Notification purge lag days	106

O

o_analyz.sql	66
o_analyz.sql	66
o_cfggen.sql	66
o_cfgprt.txt	66
o_cfgprt.txt report	66
o_config.sq	65
o_config.sql	65, 66
o_config.sql script	66
o_config.sql scriptscripts	
o_config.sql	65
o_granta	66
o_granta, o_grante, o_grantb, o_grantj.sql script	66
o_grantb	66
o_grante	66
o_grantj.sql	66
open_cursors parameterdatabase	
open_cursors parameter	63
Oracle Client	123
Oracle errors	64
OracleThinTNSNamesLocation parameter	171
ORBPort parameter	169
OrbProperties parameter	169

P

paging	71
Password parameter	170
Password paramter	181
passwords	
checkpoint	19
expiration date	15
Internet Dial Up password	92
management tasks	15
preference	15
PSR email password	93
user default	15
Web Hosting password	93
Path of shared directory where reports are stored	106
PBDatabaseTrace check point	19
Performance	106
performance9, 52, 64, 66, 71, 109, 147, 148, 155, 183	
Background Processor	107
db_block_buffers	69
parameters to adjust	70
permissions	19
check points	19
controls	18
definition	17
pop-up menus	18

windows	18	redo logsdatabase	
physical memory	71	redo logs	70
Pmat.ini	29	refresh rate, CAs	147
portal	75	reinstallation	149
Portal parameters	186	reports	
portlet logons	81	ddlcmp.txt	67
portlets	75	mdl_cmp.txt	67
portlets, adding	81	mdl_cmp.txt (sample)	74
preferences		mdl_cur.txt	67, 68
Background Processor	102, 106	mdl_cur.txt (sample)	72
check point	19	o_cfgprt.txt	66
Jmanager.exe	97	reports, customizing	85
password preference	15	restarting	
permission	13	application server	157
system preferences	34	Job Master	111
Pre-Migration Analysis Tool	28	jobs	113
primary keys	67	server or database	152
procedures		Restarting a canceled import	120
connecting to a database	108	Runtime page	43
installing multiple Job Masters	108	Runtime pageLinks, Runtime page	45
managing log filesLogfiles	112		
system administrator	110, 111		
processes parameterdatabase		S	
processes parameter	63	scanning controls	20
.....	105	scheduled jobs	113
Prompt parameter	105	scripts	
PSR Ancillary API	164	mdl_cmp.sql	67
PSR API	165	mdl_cur.sql	67
PSR End User Billing API	164	o_analyz.sql	66
PSR End User Billing server	177	o_cfggen.sql	66
PSRANCILLIARYSERVER parameter	164	o_config.sql	65, 66
PSRSERVER parameter	165	Sec_Admin group	
Purge		permissions	19
preferences	124	security	
Purge Completion Lag Days	106	Background Processor	99
Purge completion lag days for reports	106	prerequisites	11
recurring scheduling	113	utilities	123
reports	114	Security Reports check point	19
service requests	125	sequences	67
purge	114	server processes	32
purge jobs	114	server settings	
PURGEUTIL	124	large deployment	231
		large deployment appserver configuration settings	231
		large deployment clustered appserver	234
		large deployment clustered appserver settings	233
		large, clustered configuration	234
		large, clustered setup	233
		medium	226
		medium cluster setup	228
		medium deployment clustered	228
		medium, clustered	228
Q			
Queue, See Job Queue	111		
R			
Read Only	18		

T

Threads	43
timeout settings	154
TMSSERVER parameter	165
trace levels	52
TraceLevel parameter	173
Trouble Notifications Background Server	106
tuning the server	42

upgrade presparation	88
customized reports	88
user exits	89
URL parameter	181
URLNamingServicePort parameter	172
user exits	88
User parameter	170, 181
UserID parameter	105
users	
accessing accessory productsZAC	
enabling user access 36	
adding	12
administrative privileges	13
distributing updates	38
identifying	9
IDs	111
Job Manager	102
Job Queue access	111
notification	106, 111
passwords	15
preferences	106
problems	111
supporting	1
tracking logons	14
Users and Groups check point	19

validateUserAndPassword parameter	171
variables	154

w_row_in_use checkpoint	19
warning icon	152
Warnings	
Gateway.ini Server names	163
Job Type table changes	101
Job Type table data	62

LERG reconciliation	116
new structured formats	143
NPA NXX Split	143
Purge Utility	124
service request tasks	125
SharedIni parameter	105
ThreadProcs	166
trace levels	52
Web Hosting user ID	93
WebLogic logs	42
WMSERVER parameter	165
Work Management API	165
Work Queue completion date	19
Work Queue Manger check point	19
worker log	112
WorkManagement parameters	178

Z

ZAC

browser settings	148
configuration files	25, 36
user updates	35
Zac.log	149
Zac_config.xml	35