

Oracle Insurance for Health

Installation, Configuration and DBA Manual

version 3.11

Part number: E23093-01

March 2011

Copyright © 2011 Oracle and/or its affiliates. All rights reserved.

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

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

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. 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, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

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

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Where an Oracle offering includes third party content or software, we may be required to include related notices. For information on third party notices and the software and related documentation in connection with which they need to be included, please contact the attorney from the Development and Strategic Initiatives Legal Group that supports the development team for the Oracle offering. Contact information can be found on the Attorney Contact Chart.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your beta trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

CONTENTS

| | |
|--|-----------|
| Introduction | 10 |
| Purpose | 10 |
| Target Audience | 10 |
| Typographic Conventions | 10 |
| Command Syntax | 10 |
| Naming Products | 11 |
| Related Documentation | 11 |
| 1. Components and Requirements | 12 |
| 1.1. Architecture | 12 |
| 1.2. Client-Only Installation | 13 |
| 1.3. Related Publications | 13 |
| 1.3.1. RAC | 13 |
| 1.3.2. Single Sign On | 14 |
| 1.4. System Requirements | 14 |
| 1.4.1. Certification | 14 |
| 1.4.2. Hardware Requirements | 14 |
| 1.4.3. Configuration of Printers | 14 |
| 1.4.4. Software Requirements | 15 |
| 1.4.5. Operating System Requirements | 16 |
| 1.4.6. User Interface Requirements | 16 |
| 1.5. Disk Space and Memory Requirements | 16 |
| 1.5.1. Database Layer | 17 |
| 1.6. Restrictions | 17 |
| 1.6.1. National Language Support (NLS) | 17 |
| 1.7. Application Software and Templates | 17 |
| 1.7.1. Application Software | 17 |
| 1.7.2. Templates, Utilities & Sample Scripts | 18 |
| 2. Setting up the Environment | 19 |
| 2.1. Performing the Pre-Installation Tasks | 19 |
| 2.1.1. Creating an OS Account for the OHI Back Office Batchscheduler | 19 |
| 2.1.2. Assigning Default Shells for the Required OS Accounts | 19 |
| 2.1.3. Creating an OHI Back Office Database | 19 |
| 2.2. Setting up the Environment Variables | 22 |
| 2.2.1. Syntax of the Environment Variables | 22 |
| 2.2.2. Setting up the Environment Variables | 22 |
| 2.3. Setting up the OHI Directory Structure | 27 |

| | |
|---|-----------|
| 2.3.1. bin | 27 |
| 2.3.2. help | 27 |
| 2.3.3. install | 27 |
| 2.3.4. java | 28 |
| 2.3.5. report | 28 |
| 2.3.6. sh | 28 |
| 2.3.7. sql | 28 |
| 2.3.8. xml | 28 |
| 2.3.9. Custom | 28 |
| 2.3.10. log | 28 |
| 2.3.11. out | 28 |
| 2.3.12. admin | 29 |
| 2.3.13. patch | 30 |
| 2.4. Installation of OHI Back Office Application Logos | 30 |
| 2.5. Setting up Universal Oracle OS Scripts | 30 |
| 2.5.1. oratab | 30 |
| 2.5.2. oraenv | 31 |
| 2.6. Install additional Perl modules | 31 |
| 3. Configuration Oracle Fusion Middleware Software for OHI Back Office | 32 |
| 3.1. Installation and Initial Configuration Necessary Software | 32 |
| 3.1.1. Acquire the Software | 32 |
| 3.1.2. Install Oracle WebLogic Server | 32 |
| 3.1.3. Install Oracle Portal, Forms, Reports and Discoverer Product | 34 |
| 3.1.4. Configure Oracle Forms and Reports Domain and Instance | 35 |
| 3.2. Configure opmn.xml | 38 |
| 3.2.1. Forms | 38 |
| 3.2.2. Reports | 38 |
| 3.3. Configure Oracle HTTP Server | 39 |
| 3.3.1. Virtual Directories | 39 |
| 3.3.2. Viewing the Release Documentation | 39 |
| 3.4. Configure Forms Server | 39 |
| 3.4.1. Configuration formsweb.cfg | 39 |
| 3.4.2. Configure Environment Specific Settings | 43 |
| 3.4.3. Configure Registry.dat | 45 |
| 3.5. Configure Reports Server | 46 |
| 3.5.1. Change File System Access Rights | 46 |
| 3.5.2. Dynamic Environment Switching | 47 |
| 3.6. Configure Security and Manage Startup/Shutdown | 48 |
| 3.6.1. Node Manager - Additional User Definition | 49 |
| 3.6.2. Node Manager - Change Properties File | 49 |
| 3.6.3. Secure Storing of Credentials | 49 |
| 3.6.4. Server Lifecycles – Starting/Stopping Server Instances | 50 |
| 3.6.5. Securing Console Communication | 51 |

| | | |
|-----------|---|-----------|
| 3.7. | Limiting Run-time Footprint of WLS Servers | 51 |
| 3.8. | Adding Managed Servers to 'Cluster_forms' | 53 |
| 3.9. | Understanding the Batch Scheduler | 54 |
| 3.10. | Optimizing Oracle Reports | 56 |
| 3.10.1. | Reports Server System Parameter Settings in OHI Back Office | 58 |
| 3.10.2. | cacheSize | 58 |
| 3.10.3. | maxEngine | 59 |
| 3.10.4. | engLife | 59 |
| 3.10.5. | minEngine | 59 |
| 3.10.6. | keepConnection | 59 |
| 3.10.7. | maxConnect | 59 |
| 4. | Configuration Oracle Application Server (Self Service) | 60 |
| 5. | Installation OHI Back Office Application Software | 61 |
| 5.1. | Creating Accounts & Authorization | 61 |
| 5.1.1. | Security of the Batch Scheduler Account | 61 |
| 5.2. | Installation of the Application | 64 |
| 5.2.1. | An Initial Installation | 64 |
| 5.2.2. | A Copy of an Existing Installation | 64 |
| 5.3. | Compile and Check Application Software | 65 |
| 5.3.1. | Compilation | 65 |
| 5.3.2. | Check | 65 |
| 5.4. | Configuring Directories | 65 |
| 5.5. | Registration Reports Server | 66 |
| 5.5.1. | Registration of the Account in the Application | 66 |
| 5.5.2. | Registration Reports Server | 66 |
| 5.6. | Registration Batchscheduler Account | 66 |
| 6. | Completing the Installation | 67 |
| 6.1. | Synonyms for Database links | 67 |
| 6.2. | Starting OHI Back Office Batch Scheduler | 67 |
| 6.3. | Checking the Installation | 67 |
| 6.4. | Creating Backups | 67 |
| 6.5. | Activating Jobs | 68 |
| 6.6. | Installation Oracle Designer | 68 |
| 7. | Oracle Administration Related to OHI Back Office | 69 |
| 7.1. | Backup & Recovery | 69 |
| 7.1.1. | General | 69 |
| 7.1.2. | RAC | 70 |
| 7.2. | Startup and Shutdown | 70 |
| 7.2.1. | Order | 70 |
| 7.2.2. | Automation of Startup/Shutdown | 70 |
| 7.2.3. | Checks | 70 |
| 7.2.4. | RAC | 71 |

| | | |
|-----------|---|-----------|
| 7.3. | Access Control and Security Privileges | 71 |
| 7.4. | Performance Tuning and Monitoring | 71 |
| 7.4.1. | RAC | 72 |
| 7.4.2. | Rebuilding Indexes | 72 |
| 7.5. | Installation, Configuration, Upgrade, Migration and Version Control | 72 |
| 7.5.1. | RAC | 73 |
| 7.6. | Space and Storage Management | 73 |
| 7.7. | License Control | 73 |
| 7.8. | Networking | 73 |
| 7.8.1. | RAC | 74 |
| 8. | OHI Back Office Management | 75 |
| 8.1. | System Management/DBA | 75 |
| 8.1.1. | Oracle Environment | 75 |
| 8.1.2. | Clients | 75 |
| 8.1.3. | OHI Back Office Database | 76 |
| 8.1.4. | OHI Back Office Batch Scheduler | 76 |
| 8.1.5. | Changing Settings | 77 |
| 8.1.6. | Adding New OHI Back Office Application Environments | 78 |
| 8.2. | Creating New Application Users | 78 |
| 8.2.1. | Creating an Oracle Account | 78 |
| 8.2.2. | Creating Directories for the New User | 79 |
| 8.2.3. | Storing Application Authorization | 79 |
| 8.3. | Installation OHI (Patch) Releases | 80 |
| 9. | References for Active Management of OHI Back Office | 81 |
| 9.1. | Performance General | 81 |
| 9.1.1. | Memory and CPU | 81 |
| 9.1.2. | I/O | 83 |
| 9.2. | Performance Database | 84 |
| 9.2.1. | Database Settings | 84 |
| 9.2.2. | Statistics | 86 |
| 9.2.3. | Collecting Performance Data | 92 |
| 9.2.4. | Assess Settings during 'Regular' Performance | 100 |
| 9.2.5. | Preventive Application Management Actions | 100 |
| 9.2.6. | Examining Performance Problems | 100 |
| 9.2.7. | Notifying Performance Problems | 104 |
| 9.3. | Performance Application Server | 104 |
| 9.3.1. | Load Balancing Oracle Forms | 105 |
| 9.3.2. | Tuning Oracle Reports | 105 |
| 9.4. | Anticipate Preventive Resource Messages | 105 |
| 9.4.1. | Prevent Database Management Messages | 105 |
| 9.4.2. | Other Points of Special Interest | 106 |

| | |
|--|------------|
| 10. Uninstalling OHI Back Office | 107 |
| Appendix A – Configuration of Multiple batch Schedulers | 108 |
| Use of NFS Partition | 108 |
| Activities of NFS Server | 108 |
| Share Command under Sun Solaris | 108 |
| Share Command under IBM AIX | 108 |
| Security | 109 |
| Activities of NFS Client | 109 |
| Starting / Stopping the Batch Scheduler | 109 |
| Starting the Batch Scheduler | 109 |
| Stopping the Batch Scheduler | 109 |
| Dynamic Load Balancing Mechanism | 110 |
| Functionality | 110 |
| Example | 110 |
| Naming Reports Server | 111 |
| Appendix B - Installation & Configuration of OHI in a RAC Environment | 112 |
| Introduction | 112 |
| Related Publications | 112 |
| Architecture | 112 |
| Hardware and Software Requirements | 113 |
| Sizing and Performance | 113 |
| SYSAUX Tablespace | 113 |
| Instance Memory | 113 |
| Installation and Configuration | 113 |
| Migration | 114 |
| RAC Parameterization | 117 |
| Working with Services | 118 |
| Batch Processing | 119 |
| Failover | 119 |
| Management | 119 |
| Backup and Recovery | 119 |
| Startup and Shutdown | 119 |
| Networking | 119 |
| Performance Tuning and Monitoring | 119 |
| Appendix C - Installation & Configuration of SSO in OHI Back Office | 120 |
| Introduction | 120 |
| Related Publications | 120 |
| Architecture | 120 |
| Hardware and Software Requirements | 121 |

| | |
|---|------------|
| Some Important Remarks Regarding the Configuration of the OHI Back Office SSO Environment | 122 |
| Planning the (LDAP) Directory Tree | 122 |
| Migrating Database Users to Oracle Internet Directory | 122 |
| Oracle Internet Directory Single Point of Failure | 122 |
| Integration with other LDAP Directory Servers | 122 |
| Database Configuration | 123 |
| Creating the ldap.ora | 123 |
| Registering the Database in OID | 123 |
| Database Parameter ldap_directory_access | 123 |
| Configuring Database Authentication | 123 |
| Forms & Reports Configuration | 123 |
| Configuring the Forms Servlet | 124 |
| SSO System Administration | 124 |
| Managing Enterprise Users | 124 |
| Monitoring the Single Sign-On Server | 125 |
| Administering the Oracle Internet Directory | 125 |
| Appendix D - Installing Required Perl Modules | 126 |
| Introduction | 126 |
| Installation Procedure | 126 |
| Example | 127 |
| Specific Installation Instructions for DBD::Oracle | 128 |
| Prerequisites | 128 |
| SELinux on RHEL/OEL 5 | 128 |
| Specific Set up for Root Account before Installing DBD::Oracle | 129 |
| 32-/64-bit | 129 |
| Install DBD::Oracle | 130 |
| Term::ReadKey Issue | 130 |
| Appendix E - Installation Checklist | 131 |
| 1 – Prerequisites | 131 |
| 2 – Install Database | 131 |
| 3 – Install Application Server | 132 |
| 4 – Prepare OHI Back Office Installation | 132 |
| 5 – Install OHI Back Office | 133 |
| Appendix F - Migration from OAS10g to FMW WLS11g | 135 |
| What Has Changed? | 135 |
| Install and Configure Software | 135 |
| Post Install Steps | 136 |
| Recommendation | 136 |
| Appendix G - Configuration Oracle Application Server 10G (OHI Back Office) | 137 |
| Installation 10g AS Software | 137 |
| Install 10g AS Forms and Reports Services | 137 |
| Install 10g AS Patch Set | 137 |

| | |
|---|-----|
| Creating an Oracle*Reports Configuration File | 137 |
| Register in-process Reports Server | 138 |
| Configure web.xml | 138 |
| Configure opmn.xml | 138 |
| Forms | 138 |
| Reports | 139 |
| Configure Reports Server | 140 |
| Introduction | 140 |
| Reports Server Parameters | 143 |
| cacheSize | 144 |
| maxEngine | 144 |
| engLife | 144 |
| initEngine & minEngine | 145 |
| keepConnection | 145 |
| maxConnect | 145 |
| defaultEnvId | 145 |
| Dynamic Environment Switching | 145 |
| Configure Reports.sh File | 146 |
| Move in-process Reports Server to Standalone | 147 |
| Standalone Reports Server | 147 |
| Configure Oracle HTTP Server | 148 |
| Virtual Directories | 148 |
| Viewing the Release Documentation | 149 |
| Restarting 10g AS | 149 |
| Configuration ozg_web.cfg | 149 |
| Configuration Jinitiator | 149 |
| Check Port Conflicts | 150 |

INTRODUCTION

PURPOSE

This document describes the installation and configuration information for Oracle Health Insurance Back Office.

TARGET AUDIENCE

This document is intended for database managers and others responsible for installation of Oracle products. Even though various command samples are provided in this document, this document does not at all intend to offer a course in Oracle management.

TYPOGRAPHIC CONVENTIONS

Given that UNIX/Linux are case sensitive, conventions in this document may differ from the ones included in other Oracle product documentation.

COMMAND SYNTAX

Command syntax is represented in font `monospace`. The following conventions apply to command syntax:

`monospace`

Monospace type indicates OS commands, directory names, user names, path names, and file names.

brackets []

Words enclosed by brackets indicate keys (e.g., Key [Return]). Attention that brackets have a different meaning when used in command syntax.

italics

Italics indicates a variable, including variable parts of the file names. It is also used for emphasizing.

UPPERCASE

Uppercase letters indicate Structured Query Language (SQL) reserved words, initialization parameters and environment variables.

backslash \

Each backslash indicates a command that is too long to fit on one line:

```
dd if=/dev/rdsd/c0t1d0s6 of=/dev/rst0 bs=10b \  
count=10000
```

braces { }

Braces indicate mandatory items: `.DEFINE {macro1}`

brackets []

Brackets indicate optional items: `cvtrct termname [outfile]`

Attention that brackets have a different meaning when used in ordinary text.

ellipses ...

Ellipses indicate a random number of similar items:

CHKVAL fieldname value1 value2 ... valueN

italics

Italics indicates a variable. Replace the variable by value:

library_name

vertical bar |

The vertical bar allows choosing either braces or brackets:

SIZE filesize [K|M]

NAMING PRODUCTS

Whenever Oracle Health Insurance (OHI) is mentioned in this document, the *entire suite* is intended (containing product Oracle Health Insurance Back Office (OHI Back Office)).

Product Oracle Health Insurance Back Office is used when information applies to this product *only*.

RELATED DOCUMENTATION



Oracle Health Insurance Certification Matrix



Reading, Writing and Authorizing Oracle Health Insurance
Application Files

1. COMPONENTS AND REQUIREMENTS

In order to perform a quick and successful installation, a number of requirements have to be met regarding Oracle Software in the system. This chapter describes the requirements for the installation of OHI Back Office. Check to ensure that the system meets these requirements before starting the installation process.

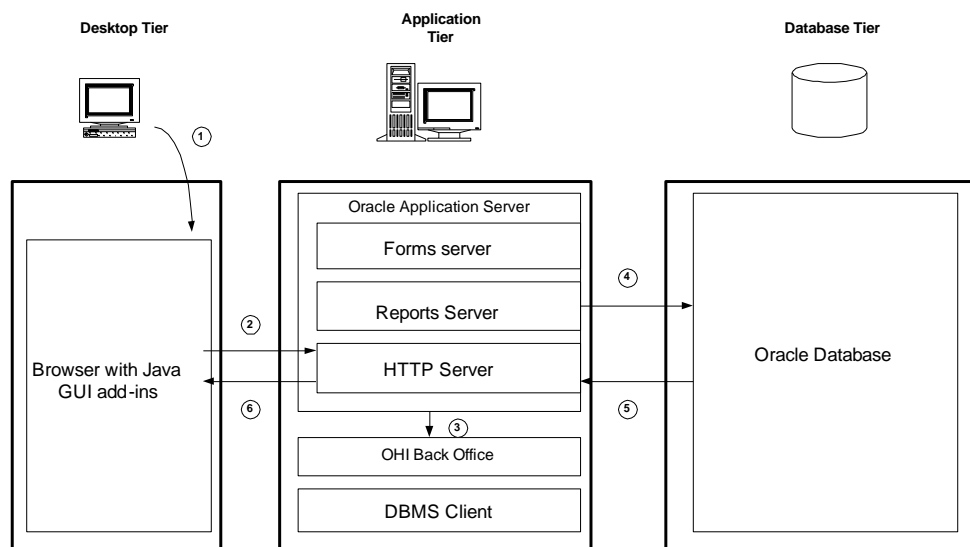
1.1. ARCHITECTURE

Application OHI Back Office is implemented in the so-called *web-architecture*. This architecture is based on 3 layers:

1. **Database Layer**
The Database Server (RDBMS), which contains the OHI Back Office data and business logic, runs on this layer.
2. **Application Layer**
The Application Server (AS), Database Client and the OHI Back Office application software run on this layer.
3. **Desktop Layer**
The browser software runs on this layer. The browser software is either installed on the PCs or runs via a *terminal server* architecture.



Attention: This refers to *logical* layers; the structure does not entail the *physical* implementation of the layers; the physical aspects and any other relevant details can be implemented on 1 or multiple servers.



This architecture results in the following software layers:

| Software layers |
|--------------------------------------|
| OHI Back Office application software |
| Oracle Application Server software |
| Oracle Database Client software |
| Oracle Database Server software |
| Operating System software |

The Application Server, Database Client and Database Server software are also referred to as *Oracle system software*.

Installation of this Oracle system software is required in order to run the OHI Back Office application software.

It is very convenient to install all related Oracle software such as Application Server, Database Client and Database Server under the same operating system user.



Attention: If OHI Back Office is used in a RAC environment, then a number of layers will be added. For a complete overview, see appendix B.

1.2. CLIENT-ONLY INSTALLATION

OHI Back Office has to be installed in an ORACLE_HOME directory, separated from the ORACLE_HOME directories which contain the Oracle Database and/or Oracle Application Server software.

Installation of OHI in the same home directory as Oracle system software is not supported.

1.3. RELATED PUBLICATIONS

Additional information regarding Oracle Database and Oracle Application Server software is available in the following documentation for the relevant platform:



Oracle Database Installation Guide



Oracle Fusion Middleware Documentation Library

1.3.1. RAC

Additional information regarding Oracle Real Application Clusters software is available in the following documentation for the relevant platform:








Oracle Clusterware and Oracle Real Application Clusters Installation and Configuration Guide



Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide

1.3.2. Single Sign On

Additional information regarding the installation and configuration of Oracle Identity Management (SSO) up to OHI Back Office release 2010.03 is available in the following documentation for the relevant platform:

-  Oracle® Database Enterprise User Administrator's Guide
-  Oracle® Internet Directory Administrator's Guide
-  Oracle® Identity Management Guide to Delegated Administration
-  Oracle® Application Server Single Sign-On Administrator's Guide
-  Oracle® Application Server Forms and Reports Services Installation Guide
(Platform specific)

Single Sign On is temporarily not supported starting with OHI Back Office release 2011.01.


1.4. SYSTEM REQUIREMENTS

1.4.1. Certification


OHI Back Office is certified for a number of platforms and specific Database, RAC, ASM, Application Server and browser versions.

Before starting the installation, check if the desired product combination is certified for OHI Back Office.

The certification information is available in the following publication:

-  Oracle Health Insurance Certification Matrix

For certification of standard Oracle products, see:

-  My Oracle Support

1.4.2. Hardware Requirements

Depending on e.g. the number of users, the usage type of OHI Back Office and the desired performance, a number of requirements regarding CPU, memory, disk space, networking, etc. have to be met.

A specific sizing check has to be performed in order to determine the numbers for a specific customer environment. Such a sizing check can be performed by Oracle.

1.4.3. Configuration of Printers

The printers that will be used for OHI Back Office have to be configured and tested under your OS.

This step has to be performed on the Application layer.

1.4.4. Software Requirements

OHI Back Office uses the following Oracle system software, which should be installed and configured before OHI Back Office is installed:

Database Layer

- Oracle Database Server
- Oracle Database Server patch set
- Oracle Database Server interim patch(es) (if required)

The installation of the Database software is not described in this manual.

Application Layer

- Oracle Fusion Middleware WebLogic Server
- Oracle Fusion Middleware Portal, Forms, Reports and Discoverer
- Oracle Fusion Middleware Portal, Forms, Reports and Discoverer patch set
- Oracle Fusion Middleware Portal, Forms, Reports and Discoverer interim patch(es) (if required)
- Oracle Database Client
- Oracle Database Examples (formerly Companion)
- Oracle Database Client patch set
- Oracle Database Client interim patch(es) (if required)
-

Installation and configuration of the Application Layer software is described in more detail in Chapter 3.



Attention 1: The version of Oracle Database Client software has to be the same as the Oracle Database Server software version.



Attention 2: Of the Oracle Database Client software the DBMS Utilities and the Precompiler software are required.
To this end, install the Administrator software.
After installation, executables proc, imp, exp, sqldr and tkprof have to be available in \$ORACLE_HOME/bin.



Attention 3: Of the Oracle Database Examples software, the Oracle Database Products software has to be installed (for the Pro*C precompiler demo software).
After installation, directory demo has to be available in \$ORACLE_HOME/precomp.



Attention 4: Please consider the fact that if the software is installed on an existing software tree, the required patch set will have to be reinstalled again (because otherwise the older, unpatched utility versions and/or precompiler software will be used).



Attention 5: In case of RAC, a description of the software requirements can be found in Appendix B.



Attention 6: In case of SSO a description of the software requirements can be found in Appendix C.

Furthermore, the following additional software is required on the OS:

1. ANSI C compiler
2. Perl
3. `zip` and `unzip` utilities for Unix/Linux
4. JDK
5. `java` utility

Desktop Layer

- Browser

For exact versions of the required software and patch sets, see:



Oracle Health Insurance Certification form

1.4.5. Operating System Requirements

For more information on the requirements, please refer to the following platform-specific documentation:



Oracle Database Installation Guide



In case of RAC, please refer to appendix B.

1.4.6. User Interface Requirements

For detailed requirements regarding OHI Back Office and different screen resolutions and Windows settings, see:



Oracle Health Insurance Certification form

1.5. DISK SPACE AND MEMORY REQUIREMENTS

The Installation Guide provides an elaborate description of the requirements pertaining to Disk Space and memory.

The following table displays the required disk space for OHI Back Office components. These are minimum estimations, not exact calculations.

1.5.1. Database Layer

| Product | Required Disk space (MB) |
|--------------------------------------|--------------------------|
| Initial OHI Back Office database (*) | 2000 |
| Total | 2000 |

Application Layer

The components marked with (*) are variables and will develop as the number of end users increases.

| Product | Required Disk space (MB) |
|---|--------------------------|
| Forms (Forms, Libraries & Menus) | 1000 |
| Reports (Reports & Libraries) | 200 |
| Other (Pro*C, OS shell scripts, tools, configuration files) | 100 |
| Batchscheduler output (*) | 150 |
| Batchscheduler log files (*) | 50 |
| Installation directory for patches | 1000 |
| Total | 2500 |

For OHI Back Office no additional memory requirements apply, other than the requirements for the Application and Database Server.

For more information on these requirements, please refer to the following (platform-specific) information:



Oracle Database Installation Guide



Oracle Fusion Middleware Documentation Library



In case of RAC, please refer to appendix B.

1.6. RESTRICTIONS

1.6.1. National Language Support (NLS)

OHI Back Office is compatible with Dutch or English/American settings. However, 1 restriction has to be considered:

1. The *territory* component of the NLS parameter has to have a final status *before* the application goes live and *may not* be changed. If a change is required, however, this can be requested from OHI Back Office; software will be provided which allows for this change to be implemented.

1.7. APPLICATION SOFTWARE AND TEMPLATES

1.7.1. Application Software

OHI application software should not be modified, conform SLAs; this would annul the Oracle guarantee.

Reference software is maintained by Oracle and *issued via (patch) releases* and can be

recognized by the uppercase prefix and lowercase extension.

Some samples of the reference software include e.g. `OZG_PROC.sh`,
`OZG_START_BATCH.sh`, `SYS1107S.sh`, etc.

1.7.2. Templates, Utilities & Sample Scripts

Templates, utilities and sample scripts, however, *can* be altered by customers.

Templates are *not* distributed by Oracle by means of (patch) releases, but they are *distributed via the iProjects website*.

They contain non-generic, often customer-specific settings (e.g. directory paths, host names, etc.) and therefore they cannot be part of the OHI application software.

Templates, utilities and sample scripts can be recognized by the lowercase naming.

Some examples include `ozg_init.env`, `ozg_fmw_main.sh` and `move.sql`.

iProjects

All tools, utilities, sample scripts, templates and technical documentation which are mentioned in this manual and have to or can be used for installation and implementation, are available on the Oracle iProjects website.

For further information regarding the Oracle iProjects website, see:



Manual iProjects Portlet

Change History

When personal changes are made in templates, utilities and scripts, it is advised to always keep a history of changes in a uniform way.

Indicate here when something is altered, who made the change and the reason for it.

Example

Change history

```
=====
Date       Author      Version: Change
-----
13-09-2007 M. Koel     1.2: Use of MS Cabinetfiles is deleted.
```

2. SETTING UP THE ENVIRONMENT

This chapter deals with the preparations of the environment before installing OHI Back Office, after all requirements listed in chapter 1 have been checked and met.

2.1. PERFORMING THE PRE-INSTALLATION TASKS

2.1.1. Creating an OS Account for the OHI Back Office Batchscheduler

The `batch` account is the OS account under which OHI Back Office Batchscheduler software runs.

Log on as `root` under OS and use the operating system user administration utility for creating the `batch` account.

This task has to be performed on the Application layer.

The name `batch` is recommended, but this is not mandatory. A different account name can be used as well.

2.1.2. Assigning Default Shells for the Required OS Accounts

For the OS accounts `batch` and `oracle` the default shell in Unix has to be set to Korn shell (`ksh`), in Linux it has to be set to `bash`.

2.1.3. Creating an OHI Back Office Database

Before OHI Back Office is installed, a database has to be created because the OHI Back Office installation has to create a number of database accounts, roles and objects in this database.

Unicode Character set

For non-western European customers the AL32UTF8 character set must be used. After creating the database, set `NLS_LENGTH_SEMANTICS` to `CHAR`, using the following command:

```
alter system set NLS_LENGTH_SEMANTICS=CHAR scope=both;
```

The database needs a restart before it takes effect. See also My Oracle Support note 144808.1 for more information about this setting.

Parameters

The following database parameters have to be set to the values specified; what is indicated is whether or not an alert or an error occurs if the value is not met:

Warning

```
DB_CACHE_SIZE          = 16M*n # multiple of 16Mb
RESOURCE_MANAGER_PLAN   = SYSTEM_PLAN
SHARED_POOL_SIZE        = 16M*n # multiple of 16Mb
```

Error/Fatal

```
AQ_TM_PROCESSES        = 1
```

```
DB_BLOCK_SIZE      = 8192
DML_LOCKS          >= 500
EVENT              not set
JOB_QUEUE_PROCESSES >= 10    # Needed for OZGISTAS
NLS_LANGUAGE        = DUTCH  # Choose the appropriate language
NLS_NUMERIC_CHARACTERS = ",." ** # personal choice
NLS_SORT            = BINARY
NLS_TERRITORY        = "THE NETHERLANDS"
OPEN_CURSORS        >= 500
OPTIMIZER_INDEX_COST_ADJ = 25
OPTIMIZER_MODE       = FIRST_ROWS_10
OS_AUTHENT_PREFIX    = ""    # empty string
PGA_AGGREGATE_TARGET > 0
PARALLEL_MAX_SERVERS = 0 ***
PROCESSES            >= 200
SESSION_CACHED_CURSORS >= 500
STATISTICS_LEVEL     = TYPICAL
UNDO_MANAGEMENT      = AUTO
SEC_CASE_SENSITIVE_LOGON = FALSE
OPTIMIZER_FEATURES_ENABLE = "11.1.0.6"
WORKAREA_SIZE_POLICY = AUTO/MANUAL ****

** = this setting can not be changed once the application is used
*** = during OHI Back Office installation > 1 is required
**** = during OHI Back Office installation MANUAL is required,
      in all other cases AUTO
```

Tablespaces

The following tablespaces have to be created for OHI Back Office data and indexes (OZG_DIM.. is for storing configuration data, OZG_FACT.. is for storing the operational changing data):

```
OZG_DIM_FIN_IND
OZG_DIM_FIN_TAB
OZG_DIM_REL_IND
OZG_DIM_REL_TAB
OZG_DIM_SYS_IND
OZG_DIM_SYS_TAB
OZG_DIM_ZRG_IND
OZG_DIM_ZRG_TAB
OZG_FACT_FIN_IND
OZG_FACT_FIN_TAB
OZG_FACT_REL_IND
OZG_FACT_REL_TAB
OZG_FACT_SYS_IND
OZG_FACT_SYS_TAB
OZG_FACT_ZRG_IND
OZG_FACT_ZRG_TAB
OZG_LOG_IND
OZG_LOG_TAB
```

The created tablespaces have to meet the following requirements:

1. Locally Managed

2. Automatic allocation
3. Automatic Segment Space Management
4. 8K block size



Attention 1: OHI Back Office requires the use of a *default temporary* table space for temporary segments.

Other Requirements

Other requirements that will be checked:

- NLS_CHARACTERSET either WE8ISO8859P15 or AL32UTF8
- PLAN_TABLE must exist
- System statistics must be available
- Direct grants from Secure Application Role `OZG_ROLE` are not allowed
- Table settings must be:

| | |
|------------|-----------------------------|
| PCTFREE = | 15% (5% for logging tables) |
| INITRANS = | 4 |



Attention 2: It is *not* permitted in the production environment to have activated events in the database, unless this has been requested explicitly by OHI Development or Oracle Support Services.



Attention 3: It is *not* permitted in the production environment to have activated Reports Server tracing, unless this has been requested explicitly by OHI Development or Oracle Support Services.

For Reports tracing, see [Troubleshooting Reports Server](#).



Attention 4: If in the Database or Application, Server settings have been changed which were not advised or prescribed, it is possible to ask the customer to have these settings reversed if problems occur that may be related to this.

The underlying reason is to prevent unnecessary instability risks. When using customized applications this has to be considered as well.



Advice 1: OHI advises the use of *Oracle Resource Management*.
Script `OZGI002S.sql` can be used to set up a standard configuration.
Set up the following database parameter:

```
RESOURCE_MANAGER_PLAN="SYSTEM_PLAN"
```



Advice 2: For the Western Europe customers OHI advises the use of character set `WE8ISO8859P15`, which supports the Euro symbol €. For non-western European customers OHI recommends `AL32UTF8`.

The character set can be selected when creating the database.
When changing the character set of an existing database, see



Migration of database character set



Advice 3: OHI advises the use of *Flash Recovery Area*.
The following parameters have to be set up for this:

```
DB_RECOVERY_FILE_DEST
DB_RECOVERY_FILE_DEST_SIZE
```



Advice 4: OHI advises the use of *Oracle Managed Files*.
The following parameters have to be set up for this:

```
DB_CREATE_FILE_DEST
DB_CREATE_ONLINE_LOG_DEST_n
```



Attention 1: If RAC is used, please refer to appendix B.

2.2. SETTING UP THE ENVIRONMENT VARIABLES

OHI applications require the necessary settings for environment variables (so these are *not* database parameters in the initialisation/server parameter file) to be included in the file `ozg_init.env`, to be called in the startup file, normally the file `.profile` (or `.bash_profile` in Linux bash shell) in the OS login home directory, of the account under which OHI is installed (normally speaking `oracle`).

Log on with the `oracle` account and set up the environment variables analogous to the instructions in this section.

An example of a call in `.profile`:

```
./u01/app/oracle/product/OHI/admin/ozg_init.env
```

Setting up the mentioned variables is *mandatory*.

A sample file with the required settings for the OS environment variables is again available on iProjects.



Hint: Copy file `ozg_init.env` to the directory `$OZG_ADMIN` (please read the next paragraph about the mandatory directory structure where this directory is introduced and the meaning of this variable later on in this paragraph) and apply the desired settings, using the instructions of this chapter.

2.2.1. Syntax of the Environment Variables

The syntax for setting up environment variables in Korn/Bash shell is as follows:

```
export variable_name=value
```

2.2.2. Setting up the Environment Variables

The environment variables required for OHI Back Office consist of 3 subsets:

1. Variables for Oracle Database Server

These variables have to be set up in the Database layer, analogous to the instructions in the *Oracle Database Installation Guide*.

2. Variables for Application Server

These variables have to be set up in the Application layer, analogous to the instructions in the *Fusion Middleware Installation Guides*.

3. Variables for OHI Back Office

These variables have to be set up in the Application layer and partly in the Database layer, if they are installed on separate servers.

The following sections provide a more detailed description of the configuration of the *specific* database/application server and OHI Back Office variables, for each variable.



Attention: *Other* Database and Application Server variables have to be set up analogous to the instructions in the relevant product installation manuals.



Attention 1: For specific environment variables for RAC, please refer to appendix B.

Setting up the NLS_LANG Variable

This variable has to be configured for the desired language, territory and character set for use by the client tools (such as Oracle Forms and Oracle Reports).

For the Dutch implementation this is normally:

```
"DUTCH_THE NETHERLANDS.WE8MSWIN1252".
```

For Czech implementations this is normally:

```
"CZECH_CZECH REPUBLIC.EE8MSWIN1250".
```

This variable has to be configured on the Application *and* Database layer.

Setting up the DBMS_LANG Variable

This variable has to be configured for the desired language, territory and character set for use by the database tools (such as import, export etc.).

For the Dutch implementation this is normally:

```
"DUTCH_THE NETHERLANDS.WE8ISO8859P15".
```

For Czech implementations this is normally:

```
"CZECH_CZECH REPUBLIC.AL32UTF8".
```

The indicated character set has to be the character set of the database, which can, if needed, be determined by means of:

```
select property_value
from   database_properties
where  property_name = 'NLS_CHARACTERSET'
/
```

This variable has to be configured on the Application *and* Database layer.

Setting up the NLS_DATE_FORMAT Variable

This variable has to be configured for the desired date format. For the Dutch implementation this is normally DD-MM-RRRR.

This variable has to be configured on the Application *and* Database layer. This setting can not be changed once OHI Back Office is used.

Setting up the NLS_NUMERIC_CHARACTERS Variable

This variable has to be configured for the desired combination for decimal and group separator. For the Dutch implementation this is normally: “, .”.

This variable has to be configured on the Application *and* Database layer. This setting can not be changed once OHI Back Office is used.

Setting up the NLS_SORT Variable

This variable has to be set on BINARY for performance reasons.

This variable has to be configured on the Application *and* Database layer.

Setting up the TNS_ADMIN Variable

This variable has to be configured to the directory which includes the Oracle Net files. On the application server (on which no Listener runs) this includes files `tnsnames.ora` and `sqlnet.ora`, on the database server (on which the Listener does run) the file `listener.ora` will be added.

This variable has to be configured on the Application *and* Database layer. It is recommendable (so as to keep the management easy) to set them in both layers referencing a location in the DBMS directory (given that this directory, within the Database Client software, is also available on the application server), this means in the Oracle Home of the database, the `$ORACLE_HOME/network/admin` directory.

Setting up the OZG_ORATAB_OZG Variable

This variable has to specify the entry in `$ORATAB` which refers to the home (root) directory of the OHI Back Office software. Typically this is an entry OZG.

This variable has to be configured on the Application *and* Database layer.

Setting up the OZG_ORATAB_DB11G Variable

This variable has to specify the entry in `$ORATAB` which refers to the home directory of Oracle 11g R1 Database software.

This variable has to be configured on the Application *and* Database layer.

Setting up the OZG_ORATAB_FRS10G2 Variable

This variable has to specify on the entry in `$ORATAB` which refers to the home directory of Oracle 10g R2 Application Server Forms & Reports Service (used upon release 2010.03 of OHI Back Office).

This variable has to be configured on the Application layer.

Setting up the OZG_ORATAB_FRS11G1 Variable

This variable has to specify the entry in `$ORATAB` which refers to the home directory of the Oracle 11g R1 Forms & Reports Service software (the Oracle Home directory within the Fusion Middleware home as used starting with release 2011.01 of OHI Back Office).

This variable has to be configured on the Application layer.

Setting up the OZG_ORATAB_IDM10G Variable

This variable has to specify the entry in `$ORATAB` which refers to the home directory of Oracle 10g Identity Management Infrastructure

This variable is *optional* and applies only if SSO is used.

Setting up the OZG_ORATAB_OMS10G Variable

This variable can be used to specify the entry in `$ORATAB` which refers to the home directory of Oracle Enterprise Manager Management Server.

This variable is *optional* and can be configured on the Application and/or Database layer (in which Oracle Enterprise Manager Management Server is set up).

Setting up the OZG_ORATAB_OMA10G Variable

This variable can be used to specify the entry in \$ORATAB which refers to the home directory of Oracle Enterprise Manager Management Agent.

This variable is *optional* and can be configured on the Application and/or Database layer (in which Oracle Enterprise Manager Management Agent is set up).

Setting up the OZG_ORATAB_CRS10G2 variable

This variable can be used to specify the entry in \$ORATAB which refers to the home directory of the Oracle Clusterware.

This variable is *optional* and applies only if RAC is used. If this parameter is set, then this has to be chosen for all nodes.

Setting up the OZG_ROOT Variable

This variable has to specify the home directory of OHI: \$ORACLE_BASE/OHI.

This variable has to be configured on the Application *and* Database layer.

Setting up the OZG_ADMIN Variable

This variable has to specify the directory in which OHI administration and configuration files are located: \$OZG_ROOT/admin.

This variable has to be configured on the Application *and* Database layer.

Setting up the PATH Variable

The directory in \$OZG_ADMIN has to be included in the PATH variable.

This variable has to be configured on the Application *and* Database layer.

Setting up the OZG_BASE Variable

This variable has to be configured to specify the home directory of OHI Back Office for the current environment (more than one OHI Back Office can exist next to each other within one OHI home directory; it is typically referring to a folder with the name of \$ORACLE_SID within \$OZG_ROOT): \$OZG_ROOT/\$ORACLE_SID.

This variable has to be configured on the Application layer.

Setting up the OZG_PATCH Variable

This variable has to be configured to specify the directory in which the OHI (patch) releases have to be downloaded and installed: \$OZG_ROOT/patch.

This variable has to be configured on the Application layer.

Setting up the FORMS_PATH Variable

This variable has to be configured for the directory in which the OHI Back Office runtime Forms modules are located for the current environment: \$OZG_BASE/bin.

This variable has to be configured on the Application layer.

Setting up the FORMS_USER_DATE_FORMAT Variable

This variable, which indicates the date format used in screens, has to be configured on DD-MM-RRRR.

This variable has to be configured on the Application layer.

Setting up the FORMS_SCROLL_ALL_BUT_ONE Variable

This variable, which is user-friendly when scrolling through records in a multi-record block in a screen, has to be set to `TRUE`.

This variable has to be configured on the Application layer.

Setting up the FORMS_TRACE_PATH Variable

This variable has to be set up to specify a directory in which the dump files due to crashes of the Forms runtime executables have to be located.

This variable has to be configured on the Application layer.

Setting up the FORMS_FLAG_DIFFERENT_SUBORD Variable

This variable has to be set to 1.

Setting up the REPORTS_PATH Variable

This variable has to be configured to specify the directory in which the OHI Back Office Reports library is located for the current environment: `$OZG_BASE/report`.

This variable has to be configured on the Application layer.

Setting up the REPORTS_NO_DUMMY_PRINTER Variable

This variable has to be set to `TRUE`.

This variable has to be configured on the Application layer.

Setting up the ORACLE_PRINTER Variable

Due to an Oracle Reports bug, this variable has to be set up for a printer, which will be used for printing from OHI Back Office.

This variable has to be configured on the Application layer.

Setting up the OCR_BACKUP Variable (Only in Case of RAC)

This variable is used for creating a backup of the Oracle Cluster Registry and has to be configured on a location of a shared file system.

This variable has to be configured on *each* RAC node.

Setting up the OVD_BACKUP Variable (Only in Case of RAC)

This variable is used for creating a backup of the Oracle Voting Disk and has to be configured on a location of a shared file system.

This variable has to be configured on *each* RAC node.

Setting up the OZG_OUT Variable

This variable has to be configured to specify the *physical* directory in which the OHI Back Office batchscheduler writes the output, e.g. `$OZG_BASE/out`.

This variable can be used in the system parameters for OHI Back Office subsystem SYS (“System”); screen `SYS1010F`.

This variable has to be configured on the Application layer.

Setting up the OZG_LOG Variable

This variable has to be configured to specify the *physical* directory in which the OHI Back Office Batchscheduler writes the log files, e.g. `$OZG_BASE/log`.

This variable can be used in the system parameters for OHI Back Office subsystem SYS (“System”); screen `SYS1010F`.

For a detailed description of the use of OS environment variables for use in the output and log path of the OHI Back Office Batchscheduler, please refer to the following document:



Reading, writing and authorising of OHI application files

This variable has to be configured on the Application layer.

2.3. SETTING UP THE OHI DIRECTORY STRUCTURE

Under operating system user `oracle`, the following *mandatory* directory structure (the “custom” directory being an exception) has to be created under the `$OZG_BASE` directory on the Application Server. The OHI application source files and executables for each OHI Back Office environment (so for each ‘environment’ directory) require a separate environment subdirectory structure:

```
$OZG_ROOT          #Oracle Health Insurance install directory
/admin            #Configuration files          = $OZG_ADMIN
/patch            #(Patch)releases              = $OZG_PATCH
/environment      #Directory for each environment = $OZG_BASE
                  #Can occur n times in 1 $OZG_ROOT
/bin              #Forms
/help             #Online help documentation
/install          #DDL scripts
/java             #Java files
/log              #Batchscheduler log files
/out              #Batchscheduler output
/report           #Reports & Reports printer definitions
/sh              #OS shell scripts
/sql              #SQL modules
/xml              #XML files
/custom           #Custom Development managed files
```

2.3.1. bin

This directory contains all OHI Back Office Forms (Forms, Menus, Forms Libraries, Reference Forms and Object Libraries), Pro*C software and application logos.

2.3.2. help

This directory contains all OHI Back Office online help documentation, which can be accessed from the application.

2.3.3. install

This directory contains all OHI Back Office DDL scripts e.g. tables, constraints, indexes and triggers.

These scripts are used to partition or compress large tables at any moment of your choice (this is possible only for tables characterized by OHI Back Office as *to be partitioned/compressed*).

2.3.4. java

This directory contains all OHI Back Office Java files.



Attention: Certification and/or keystores which are possibly required for secure connections on the Internet, have to be located in the `$OZG_BASE` directory of the relevant environment.

2.3.5. report

This directory contains all OHI Back Office Reports (Reports and Reports Libraries) software & Reports printer definition files.

2.3.6. sh

This directory contains all OHI Back Office OS shell scripts and SQL*Loader control files.

2.3.7. sql

This directory contains all OHI Back Office SQL modules.

2.3.8. xml

This directory contains all OHI Back Office XML files.

2.3.9. Custom

This directory contains custom development managed files. For instance files that are processed by custom developed, so non system-native, scripts should be placed here or in a subdirectory of this directory. The “custom” directory is not mandatory, however the naming is prescribed.

Additionally, the following 2 directories have to be created.

Normally, this takes place under `$OZG_BASE`; however, it is permitted to move these directories to a different physical location:

2.3.10. log

This directory contains all OHI Back Office Batchscheduler log files (see the environment variable `$OZG_LOG`).

The OS users `oracle` and `batch` have to have write rights for this directory.

2.3.11. out

This directory contains all OHI Back Office Batchscheduler output files (see the environment variable `$OZG_OUT`).

The OS users `oracle` and `batch` have to have write rights for this directory.

Under OS user `oracle`, the following *mandatory* directory structure has to be created under `$OZG_ROOT` on the Application layer for the OHI Back Office application tools & utilities and releases:

2.3.12. admin

This directory contains all OHI Back Office tools, management files and configuration files.



Attention: If the Database layer and the Application layer have been installed on different servers, the directory `$OZG_ROOT/admin` *also* has to be created on the Database layer.

In that case there is a `$OZG_ADMIN` directory on the application server(s) *and* a `$OZG_ADMIN` directory on the database server(s).

These directories can *not* be shared; the content of the directories happens to be different; also the content of the configuration files *in* the directories is different.

The directory `$OZG_ADMIN` has to contain the following files on the application server. The directory `$OZG_ADMIN` on the database server contains only the files that are marked with (**).

| Filename | Purpose |
|---------------------------|---|
| d10g*/d11g* | Utilities for generation/compilation of Oracle Forms and Reports 10g or 11g sourcefiles. |
| OHI.dat | Oracle Forms Registry file with specific OHI Back Office settings (upon 2010.03). |
| ozdefine.sql | Configuration file for OHI Back Office SQL*Plus variables. |
| ozdefine.sql.<ORACLE_SID> | Environment specific configuration file for OHI Back Office SQL*Plus variables. |
| ozg_init.env (**) | Environment file for OHI Back Office settings. |
| ozg_main.sh (**) | OS command file for starting/stopping the complete Oracle environment; Oracle system- and OHI Back Office software. |
| ozg_fmws_main.sh | OS command file for starting/stopping the Forms/Reports application server environment (starting with 2011.01). |
| OZG_GIF.jar | OHI Back Office jarfile for Oracle Application Server. |
| OZG_CMD.sh | OS command file to set correct environment when executing Oracle utilities and executables (upon 2010.03). |
| OZG_CMD.pl | Perl command file to set correct environment when executing Oracle utilities and executables (starting with 2011.01). |
| OZGPATCH.pl | OHI Back Office Perl installation menu for installing major releases, patch sets and interim patches. |
| OZGPLIB.pm | OHI Back Office Perl library module. |
| ozg_web.cfg | Configuration file for Oracle Application Server (upon 2010.03). |
| OZG_BASEJINI.htm | Configuration file for Oracle Jinitiator implementation (upon 2010.03). |
| OZG_BASEJPI.htm | Configuration file for Sun's Plugin implementation (upon 2010.03). |
| OZG_JPI.htm | Configuration file for forms Java Plugin implementation (starting with 2011.01). |
| ozgrf34w.res | OHI Back Office Forms Terminal Resource File (for configuring keyboard mappings). |
| ozg_batch_start.sh | OS command file to start OHI Back Office Batchschedulers for all environments. |
| ozg_batch_stop.sh | OS command file to stop OHI Back Office Batchschedulers for all environments. |
| ozg_oracle_start.sh (**) | OS command file to start all Oracle software on the system. |
| ozg_oracle_stop.sh (**) | OS command file to stop all Oracle software on the system. |

2.3.13. patch

This directory contains all OHI (patch) releases which have to be installed.
This directory only applies to the application server(s).

2.4. INSTALLATION OF OHI BACK OFFICE APPLICATION LOGOS

OHI Back Office uses 2 logos (which are loaded dynamically on runtime):

- **OZGIMG01.gif**
Is displayed on the OHI Back Office start screen OZGSTART.
Size: 390 x 160 pixels
- **OZGIMG02.gif**
Is displayed on the OHI Back Office Info screen OZGABOUT.
Size: 120 x 45 pixels

The logos have to be placed in directory `$OZG_BASE/bin` on the Application layer.

Initial logos are available via the OHI Helpdesk; the customer is free to replace these logos by private company logos; the only requirement is that the naming has to be maintained.

2.5. SETTING UP UNIVERSAL ORACLE OS SCRIPTS

2.5.1. oratab

In file `$ORATAB` (normally `/etc/oratab` or `/var/opt/oracle/oratab` for SunSolaris) entries *have to be* included (Oracle home directories for the relevant software) for the following products.

Be sure to use transparent naming of the entries put in `ORATAB`. The best way is to name the entry the same as the directory where the software is stored. There are exceptions for this and these are `DB<99>G` and `FRS11G1`, these names usually do not point to a directory with the same name, all other entries should be the same.

The codes that are used for the entries can be selected freely (but they have to be indicated in uppercase letters). By means of variables `$OZG_ORATAB_*` in `$OZG_ADMIN/ozg_init.env` the entries become available in the environment:

1. Oracle Fusion Middleware Forms & Reports Services

Sample code: `FRS11G1`

2. Oracle Database

If the Application layer and the Database layer are installed on 1 server, the entry refers to the Database Server software home, otherwise to the Database Client software home on the application server.

Sample code: `DB11G`

3. OHI

Sample code: `OHI` (previously `OZG`)

The following entries are optional:

4. Oracle Enterprise Manager Management Agent

Sample code: OMA10G

5. Oracle Enterprise Manager Management Server

Sample code: OMS10G

Sample entries in \$ORATAB

```
...
FRS10G2:/u01/app/oracle/product/frs10g2:N
FRS11G1:/u01/app/oracle/product/fmw11g1/pfrd_1:N
DB11G:/u01/app/oracle/product/11.1.0/db_1:N
OHI:/u01/app/oracle/product/OHI:N
...
```

Sample corresponding variables \$OZG_ADMIN/ozg_init.env

```
...
export OZG_ORATAB_FRS10G2=FRS10G2
export OZG_ORATAB_FRS11G1=FRS11G1
export OZG_ORATAB_DB11G=DB11G
export OZG_ORATAB_OZG=OHI
...
```

The aforementioned entries in `ozg_init.env` are used by means of the standard Oracle utilities `dbhome` and `oraenv`.

It is of importance that no other `oratab` is indicated in these utilities; ensure to check that `$ORATAB` in these files (available in `/usr/local/bin` and the `$ORACLE_HOME/bin` directories) does not refer to a non-existing `oratab` file.

When reference is made to a non-existing `oratab` file, this can be modified in the relevant utilities, or a symbolic link can be created for the non-existing `oratab` file which refers to the correct `oratab` file.

2.5.2. oraenv

For certain (older) versions of `oraenv` error message “unlimited: bad number” may be encountered, or the performance of `oraenv` may get stuck (on `exec $ORACLE_HOME/bin/osh`).

In that case the workaround as described in My Oracle Support note 1023496.6, has to be introduced.

2.6. INSTALL ADDITIONAL PERL MODULES

In order to use the OHI installation menu `OZGPATCH`, it is required to install additional Perl modules.

It is *mandatory* to install these modules *before* starting an initial installation of OHI software. The installation menu will not work until these modules are installed.

For installation instructions, see [Appendix D – Installing required Perl modules](#).

3. CONFIGURATION ORACLE FUSION MIDDLEWARE SOFTWARE FOR OHI BACK OFFICE

This chapter treats the configuration of the Oracle Forms and Reports components of the Oracle Fusion Middleware (OFM) product “Oracle Portal, Forms, Reports and Discover” (PFRD) as it is needed for the OHI Back Office application starting with release 2011.01.

3.1. INSTALLATION AND INITIAL CONFIGURATION NECESSARY SOFTWARE

This installation has been verified on a Linux platform. Installations on different certified platforms are supported. Specific instructions for these platforms will be documented when available from customer experiences.

3.1.1. Acquire the Software

Make sure you download the correct software for your platform. As only the 64 bit platform is supported download the following components:

1. Oracle WebLogic Server 11gR1 (10.3.3) + Coherence - Package Installer (the generic version without a JDK, make sure the .jar file is not renamed to a .zip file)
2. Oracle Portal, Forms, Reports and Discover (the base version typically existing of several zips and the update version which contains the latest version; we will refer to this product with PFRD)

It is assumed you have a 64bit JDK for your platform available. Please make sure you use a certified JDK version.

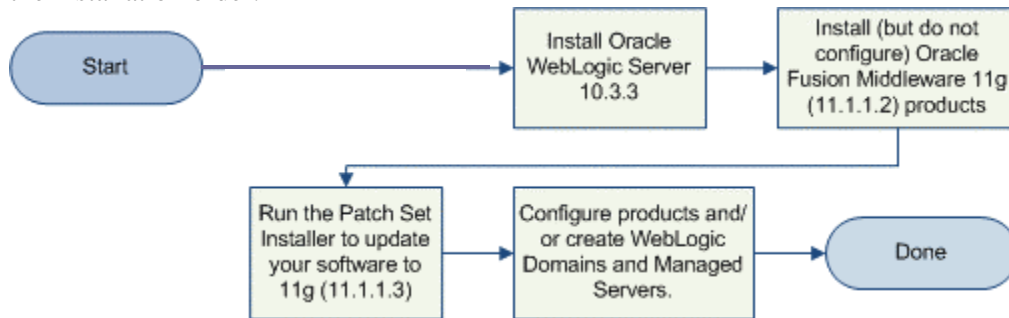
Currently no other Oracle software is required (the Installer will check on most OS requirements).

3.1.2. Install Oracle WebLogic Server

OHI Back Office uses only the Forms and Reports services of the Oracle FMW product PFRD.

For the 11g ‘application server’ it is required to install the Oracle WebLogic Server (WLS). The WLS installation creates a Middleware home directory structure.

It is not directly possible to install the required version. The schema below explains the installation order.



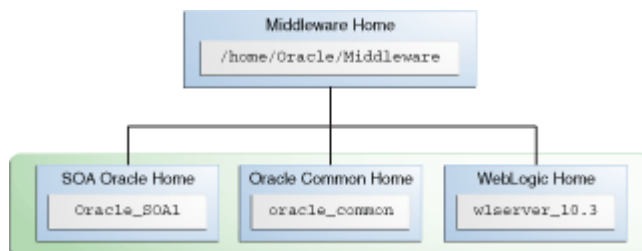
For all documentation describing the Oracle Fusion Middleware environment, concepts, installation requirements, installation process, etc. please see:



Oracle Fusion Middleware Documentation Library
(version 11.1.1.3, part nr E14571)

With the introduction of WLS new environment variables will be used, referencing the new directory structure needed for FMW products.

For the top level Middleware home we will use variable `$MW_HOME` to refer to it. As name we will use `fmw11g1` (**F**usion **M**iddleware **11g** **R1**) as for example in `/ozg/app/oracle/product/fmw11g1`. This is the top level folder in the directory structure below (we will refer to this as the FMW home, the middleware home directory).



The picture below contains a SOA Oracle Home within the middleware home directory, on the same level as the WebLogic Server (WLS) home directory. A FMW home will always contain one WLS home, which will be identified with variable `$WL_HOME`. Name this for example `wls11g1` (resulting in `/ozg/app/oracle/product/fmw11g1/wls11g1`).

Both environment variables are set automatically when the WLS environment scripts are run.

For starting the installation please run (be sure you use a certified JDK version; this JDK will be configured to run WLS with):

```
java -jar wls1033_generic.jar
```

**Attention:**

It is assumed the java bin directory is present in your PATH, otherwise prefix the executable name with the relevant path.

Because you are installing WebLogic Server on a 64-bit platform using a .jar installation program please:

- Include the -d64 flag in the installation command when using a 32/64-bit hybrid JDK (such as for the HP-PA, HPIA, and Solaris64 platforms). For example, if installing in graphical mode using the Package installer:

```
java -d64 -jar wls<version>_generic.jar
```

- Run the java -version command (or java -d64 -version command on platforms using a 32/64-bit hybrid JDK) to ensure that your JAVA_HOME refers to a 64-bit JDK.
- If you are using the Sun 64-bit JDK, use the following command to install WebLogic Server:

```
java -Xmx1024m -jar wls<version>_generic.jar
```

When you have started the installer please follow the instructions below:

- Specify a location for the Fusion Middleware home, i.e.
/ozg/app/oracle/product/fmw11g1
- At a certain moment you need to choose between a typical or custom installation: choose custom.
- Deselect Coherence on the highest level. And deselect within WLS the Evaluation Database. Also deselect the Server Examples (only these 2 need to be deselected within WLS, the rest should remain selected).
- Leave the selected JDK checked (it should be the JDK you run the installer with).
- Specify the name and location for the WebLogic home directory, i.e.
/ozg/app/oracle/product/wls11g1
- Do not (!) run the quickstart at the end of the installation.

3.1.3. Install Oracle Portal, Forms, Reports and Discoverer Product

After installation of the WebLogic Server software the PFRD product has to be installed. This cannot be installed at once given the current PFRD distribution: please follow the 2 step approach as shown in the figure in the previous paragraph (this is based on the current releases as available at the end of 2010; this will change of course in the future).

For exact versions of the required Oracle application software and patch sets, see:



Oracle Health Insurance Certification form

Within the FMW home the FMW product PFRD will be installed within its own Oracle Home directory (in the example figure in the previous paragraph the SOA product is installed in its own Oracle Home directory).

Use as \$ORACLE_HOME naming for example pfrd_1 for your first PFRD product install (resulting in for example /ozg/app/oracle/product/fmw11g1/pfrd_1).

This makes clear an Oracle Home folder is on the same level as the WLS home directory. Several Oracle Home folders next to each other can be installed within one Middleware Home folder. Only one WLS installation is possible within a Middleware Home.

Summary of the installation steps for PFRD:

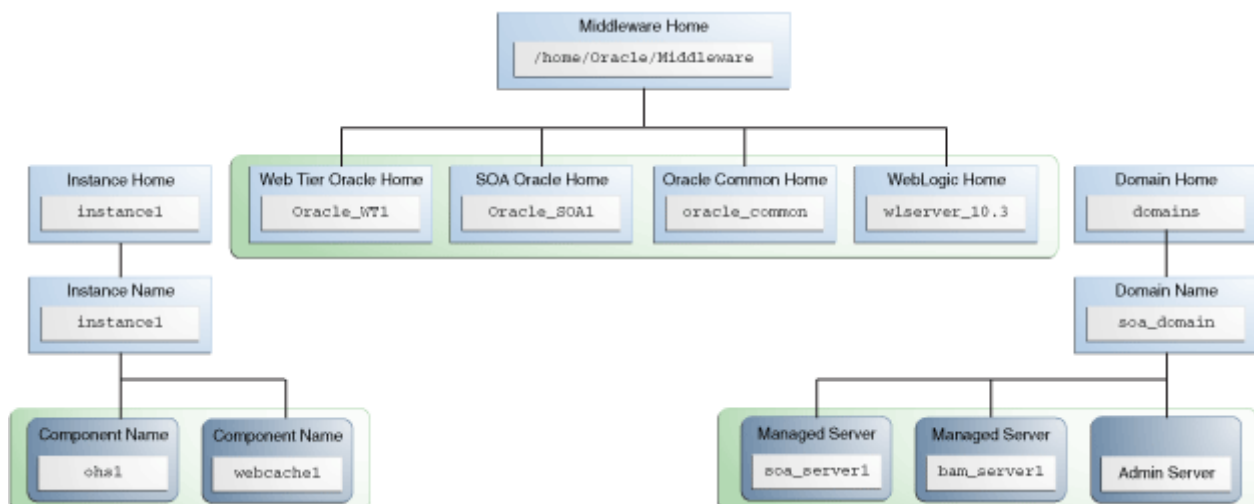
- Start the installer of the complete distribution: cd to the first Disk directory and execute `./RunInstaller` (it might be you have to run rootpre scripts on certain platforms).
- Choose the option 'Install Software – **Do Not Configure**' and proceed with the installation steps (and implement failing prerequisites if needed) until the installation is finished.
- Start the installer of the 'update' distribution: cd to the first Disk directory and execute `./RunInstaller`

After this you need to install at least the patches as indicated in the certification matrix. For this you need to install a recent version of Opatch in the PFRD Oracle Home. **There are required patches!**

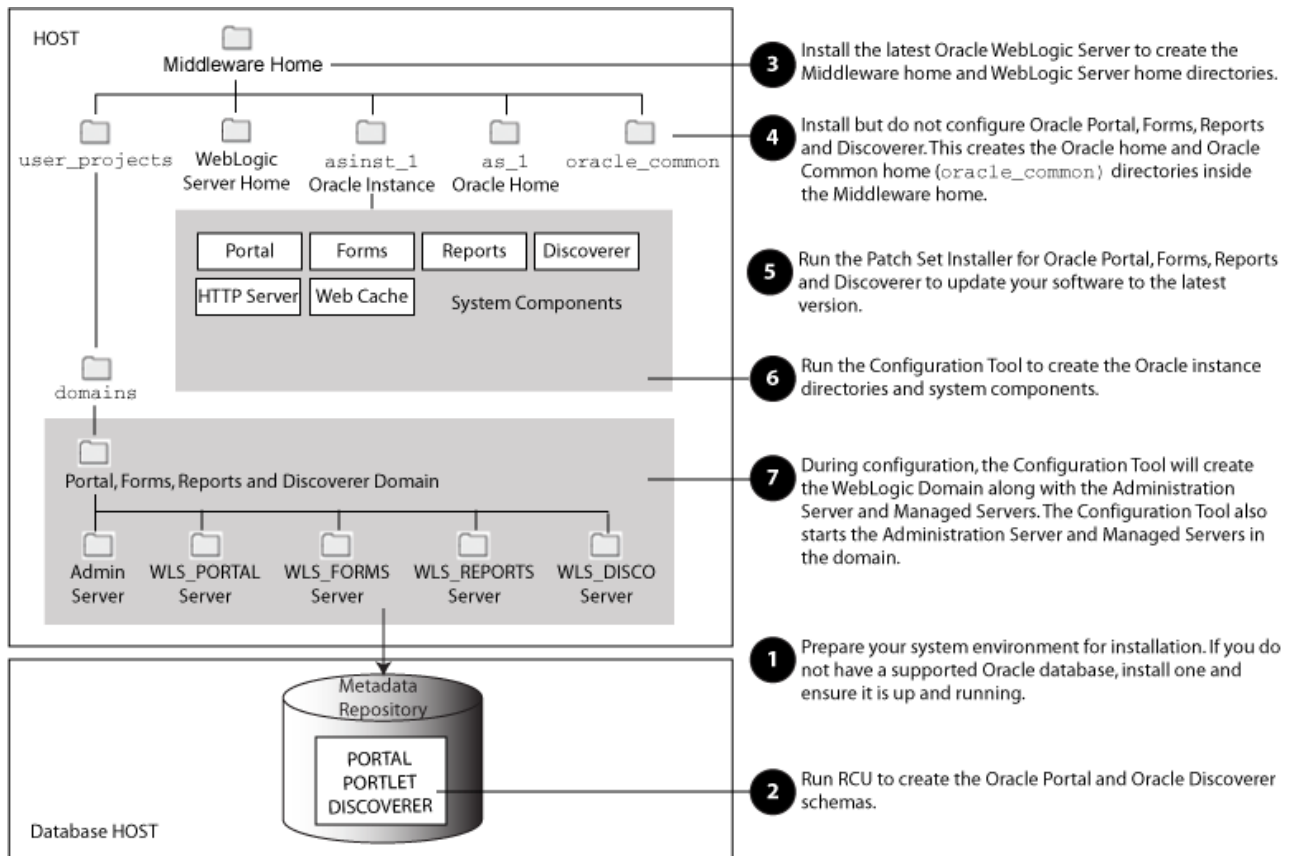
3.1.4. Configure Oracle Forms and Reports Domain and Instance

When the installation is finished the configuration can start. Only Forms and Reports services will be configured. Configuring these means both an *Oracle Instance* directory as well as a *WLS domain* will be created. The instance will contain the *system components* such as the Oracle HTTP server (OHS) and the reports server. The WebLogic domain will contain the *java components*, the Managed Servers (JVM's) for running the forms and reports services.

An example of a Middleware home with separate instance and domain structure is shown below.



A typical installation for the PFRD product could look like:



Before starting the forms and reports configuration make sure you implement the necessary settings in your environment:

- `ozg_init.env`: an entry for `OZG_ORATAB_FRS11G1` should be present which looks like the settings in the template file on iProjects. Most important is that the additional settings for this environment are set like `DOMAIN_NAME`, `DOMAIN_HOME` and `ORACLE_INSTANCE`.
- It is expected the file `oratab` (referred to by variable `$ORATAB`) contains an entry for the identifier identified by `OZG_ORATAB_FRS11G1`.

When these settings are present please switch to the PFRD environment settings through:

```
. ozg_init.env $OZG_ORATAB_FRS11G1.
```

```
Start $ORACLE_HOME/bin/config.sh.
```

It may be you run into the errors below:

Could not reserve enough space for object heap

Could not create the Java virtual machine.

In that case first:

```
export _JAVA_OPTIONS=-XX:MaxPermSize=512m
```

When you run the configuration tool please take notice of:

- Specify a username/password combination for the domain to be created to manage it later through the console.
- Where a domain name can be entered (default `ClassicDomain`) we will create and enter domain name `frs_d1` (`frs_d1` for Forms Reports Services domain 1) and refer to this name in later documentation steps. So the other options 'extend domain' and 'expand cluster' are not chosen.
- The instance directory to be created can be inside or outside the middleware directory, for example location `<$MW_HOME>/frs_i1` or `<$ORACLE_BASE>/product/frs_i1` (`frs_i1` for Forms Reports Services instance 1; please enter the complete path instead of using a variable) with instance name `frs_i1`. In the template script `ozg_init.env` it is assumed to be located within the Middleware home directory.
- When the screen is shown for choosing what to configure, deselect Oracle Portal, Oracle Discoverer, Oracle Web Cache (but leave Oracle HTTP server selected). As a result of this only Forms and Reports services are configured.

It is advised to select the Developer Builder tools, which can be helpful in case of investigating problems.

If you will run on a stand alone application server machine and the environment is only used by a small user community (typically a custom development or test environment) you can deselect the 'clustered' option at the bottom of the screen.

For production environments and environments used for testing the production situation always select the clustered option because this offers much more flexibility and ease of use for when more than one managed server is needed (the cluster can easily be extended when the forms and reports 'applications' are deployed on the cluster instead of a specific managed server).

- For port configuration there is no requirement, the option that fits best should be chosen.
- Deselect the option 'Use Application Level Identity Store'.
- When you apply the configuration it can take a considerable amount of time for executing all the steps.

Make sure you save the configuration results at the end the configuration report. Especially the URL of the Admin Server Console (`./console`) and the EM Console (`./em`) should be kept.

Beware that at the end of this step you have running WLS processes. In case you want to stop them at this moment you can follow these steps:

- Connect to the Admin Server console and shutdown the managed servers `WLS_FORMS` and `WLS_REPORTS`.
- `$DOMAIN_HOME/server/bin/stopWebLogic.sh`

When you need the Admin Server Console to proceed use the command below to start this server:

- `$DOMAIN_HOME/server/bin/startWebLogic.sh`

Starting and stopping will be discussed in more detail later.

3.2. CONFIGURE OPMN.XML

In file `$ORACLE_INSTANCE/config/OPMN/opmn/opmn.xml` the generic environment variables have to be specified.

This involves the generic variables, OHI environment independent, that are required for Oracle Forms and Oracle Reports, and a generic `OZG` variable required for `OZG_CMD.pl`.

3.2.1. Forms

For forms no specific entries are needed within `opmn.xml`. The environment variables as set within `ozg_init.env` are passed on.

3.2.2. Reports

Add the following to the entry

```
<process-type id="ReportsServer" module-id="ReportsServices">
```

under `<environment>` add/replace the two following variables with your environment specific values:

```
<variable id="TNS_ADMIN"
value="/u01/app/oracle/product/11.1.0/db_1/network/admin"/>
```

```
<variable id="OZG_ROOT" value="/u01/app/oracle/product/OHI"/>
```

And make sure you remove (!) the `NLS_LANG` variable setting within that location. It does not have a value and overrides in that way the existing environment variable. By removing this variable it will be derived from your environment when the OPMN processes are started. This prevents redundant settings which might be conflicting.

Example:

```
<ias-component id="rep_myhost" status="enabled" id-matching="false">
  <process-type id="ReportsServer" module-id="ReportsServices">
    <process-set id="rep_myhost" restart-on-death="true" numprocs="1">
      <environment>
        ...
        <variable id="TNS_ADMIN"
value="/u01/app/oracle/product/11.1.0/db_1/network/admin"/>
        <variable id="OZG_ROOT" value="/u01/app/oracle/product/OHI"/>
        ...
      </environment>
    </module-data>
    <category id="general-parameters">
      ...
    </category>
  </process-set>
</process-type>
</ias-component>
```



A reload and restart of your OPMN processes is necessary after implementing this changes.

3.3. CONFIGURE ORACLE HTTP SERVER

In a standard PFRD installation an Oracle HTTP Server (OHS) is installed and configured.

The following settings have to be modified in the configuration file for OHI Back Office to work correct. Afterwards the HTTP Server has to be restarted.

3.3.1. Virtual Directories

In configuration file `$ORACLE_INSTANCE/config/OHS/<OHS_INSTANCE_NAME>/moduleconf/forms.conf`, the Virtual Directory Mapping `/OHI/` has to be implemented to support the physical mapping to directory `$OZG_ROOT`.

This Virtual Directory Mapping can subsequently be used in the following setting:

- path for `OZG_GIF.jar` file

Sample setting:

Alias `/OHI/` `"/u01/app/oracle/product/OHI/"`

3.3.2. Viewing the Release Documentation

In order to allow for an interactive (via screen “Raadplegen release informatie” /”Consult release information”) view of the release documentation, the following settings (with of course the sample directory replaced by your actual directory) have to be added in `forms.conf`:

```
<Directory "/u01/app/oracle/product/OHI/patch/2???..???..???/doc/">
  Options Indexes MultiViews
  AllowOverride None
  Order deny,allow
  Allow from all
</Directory>
```

This code ensures that the `/doc` directory of OHI (patch) releases (always in format `2999.99.9.9999`) can be viewed via a browser.

Other directories are deactivated.

3.4. CONFIGURE FORMS SERVER

In this paragraph it is described how to configure your Forms Server environment.

3.4.1. Configuration `formsweb.cfg`

Configuration file

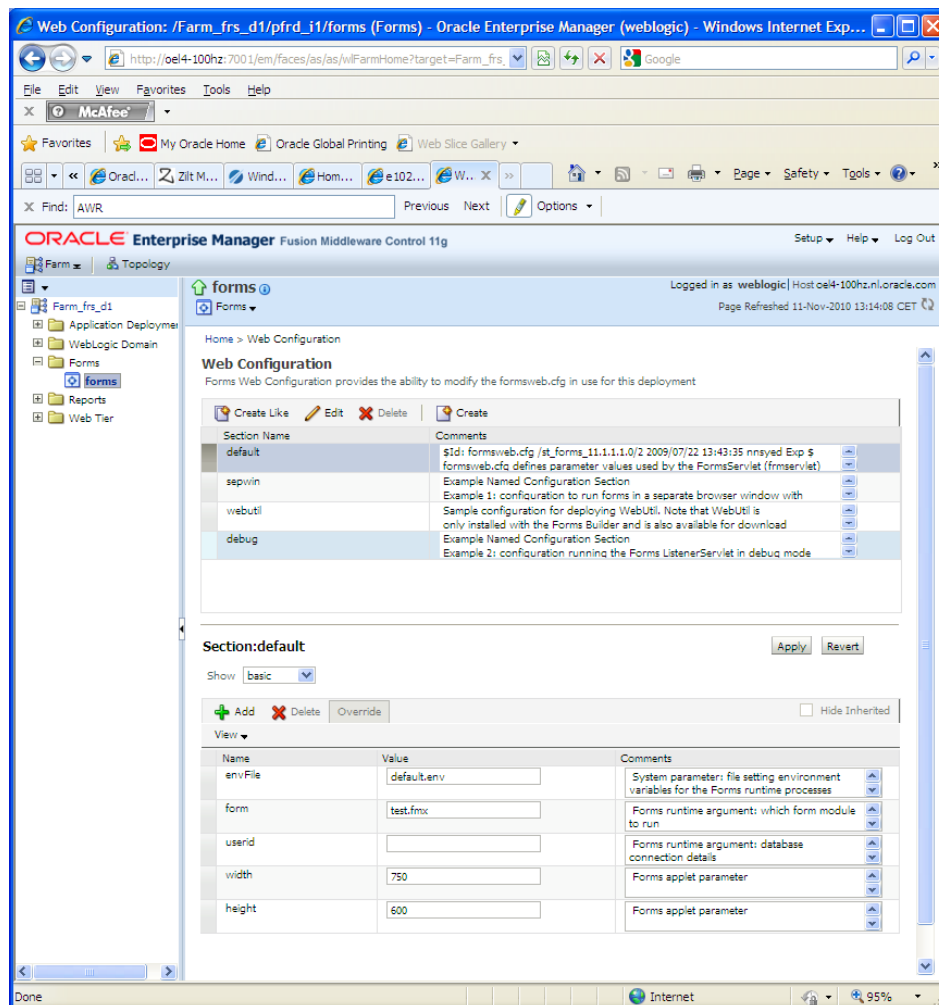
`$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.1/config/formsweb.cfg` now has to be configured for the environment-specific settings.

This file can be edited manually but it is also possible to change this using Oracle Enterprise Manager Fusion Middleware Control (the <hostname>:<consoleport>/em URL can be used to start this Oracle Enterprise Manager based environment).

An example of the window page where the configuration parameters can be changed is shown below. This can be reached by using the Web Configuration hyperlink for the Forms branch shown in the left hierarchy pane.

The <default> configuration will be inherited by all other configurations, so generic changes can be added overthere. Specific configurations with different settings can be created to support different environment settings for example.

The settings itself are grouped logically together. The group name used in the screen will be used to help in finding where to specify the setting.



The table below shows the parameters which are required to be set for the use of OHI Back Office. These can best be specified in the default configuration because they are usually set identical for each OHI BO environment. Be sure you adapt file paths and names to your actual server setup!

Beware: When editing the field values initially locating the cursor in the field and scrolling through the value might not work. Relocating the cursor position can help in this situation.

| Parameter name | Group | Value | Comment |
|----------------|----------|---|--|
| form | basic | OZGSTART.fmx | First window to open when forms application is started (fixed value). |
| archive | plugin | frmall.jar, /OHI/admin/OZG_GIF.jar | Forms applet archive setting for other clients (fixed value). |
| otherparams | advanced | term=/u01/app/oracle/product/OHI/admin/ozgrf34w.res | Replace the term=%term% with a line as shown overhere, referencing the correct folder for the keyboardmapping file, as it is defined in ozgrf34w.res. Beware: in older versions also useSDI=no was needed. Because this is now the default this is no longer required. |
| baseHTMLjpi | html | /u01/app/oracle/product/OHI/admin/OZG_JPI.htm | Location of OHI specific base HTML file for use with Sun's Java Plug-In: OZG_JPI.htm. Delivered in \$OZG_ADMIN. |

The following parameters apply to the Sun Java Plug In (Sun JPI). It is possible to use static or dynamic versioning. Static versioning mandates an exact version of Sun JPI to be installed on the workstation, whereas dynamic versioning just states a minimum version and every higher version will also do.

Dynamic versioning means value clsid:8AD9C840-044E-11D1-B3E9-00805F499D93 for JPI_CLASSID while static versioning uses clsid:CAFEEFAC-<major version>-<minor version>-<patch version>-ABCDEFEDCBA as format.

With static versioning JPI_CODEBASE should be a URL to the .CAB file to download this exact version.

JPI_MIMETYPE is a mime-type in a format like “application/x-java-applet;<version_type>=<implementation_version>”. For static versioning, <version_type> should be set to “jpi-version” (use “version” for Firefox to accept higher JPI versions).

Example values using static versioning are shown below (referencing Java SE 6 Update 20).

It seems wise to set these values also in the default environment.

| Parameter name | Group | Value | Comment |
|----------------|-------|-------|---------|
|----------------|-------|-------|---------|

| Parameter name | Group | Value | Comment |
|-------------------|--------|--|---|
| jpi_classid | Plugin | clsid:CAFEEFAC-0016-0000-0020-ABCDEFEDCBA | The ClassID of the Sun JVM to use (Internet Explorer specific). Static versioning is used. |
| jpi_codebase | Plugin | http://java.sun.com/products/plugin/autodl/jinstall-1_6_0-windowsi586.cab#Version=1,6,0,20 | Download location of the CAB file for the Sun JVM (IE specific). |
| jpi_mimetype | Plugin | application/x-java-applet;version=1.6 | Used primarily for Netscape/Firefox but also passed on to IE. Use ;version= instead of ; jpi_version= to get this working within Firefox. |
| jpi_download_page | Plugin | http://java.sun.com/products/archive/j2se/6u20/index.html | Download location of the JRE installer for Netscape/Firefox |



Attention: If a newer version of Sun's Plugin is to be used, then the relevant JPI tags in file `formsweb.cfg` have to be changed.

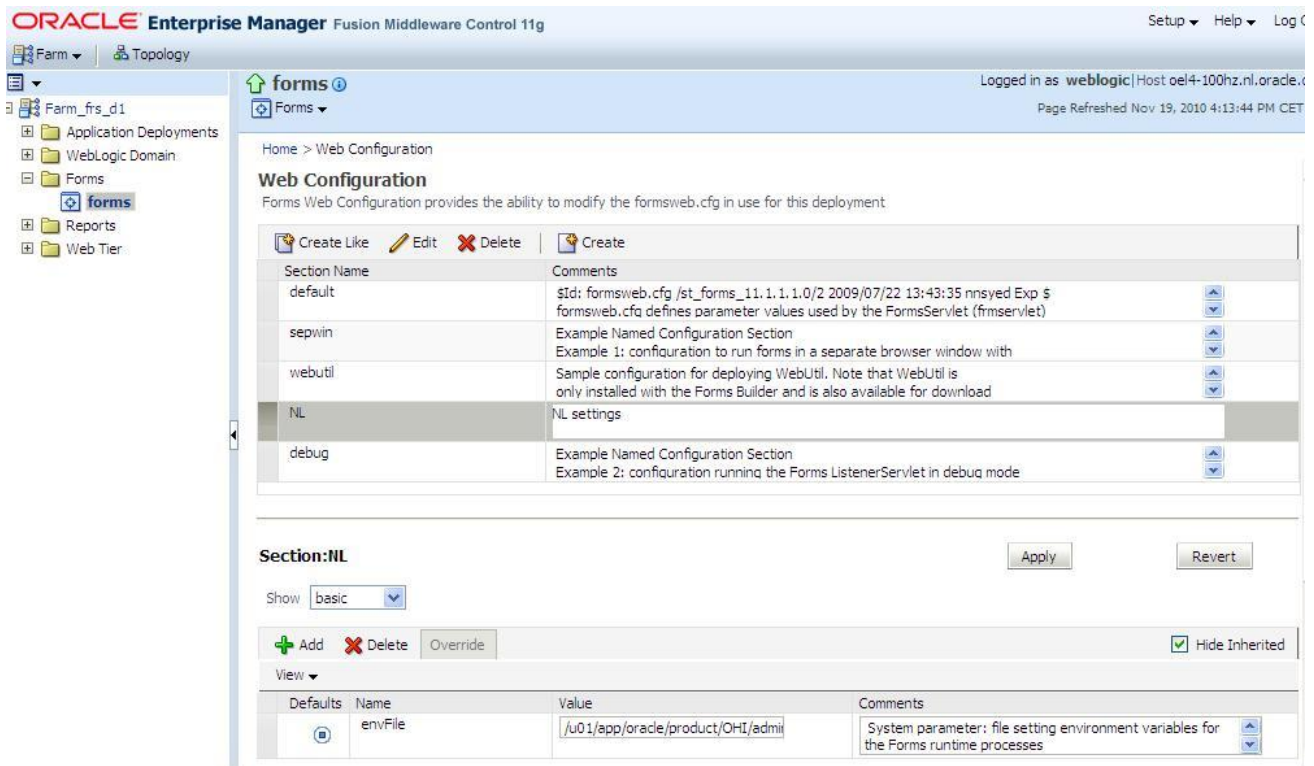
The following parameters are optional. Values are given but are not required. Normally these settings are applied to the default configuration but you can implement these different for each (OHI BO) environment. This is typically done by implementing different sections within the web configuration (`formsweb.cfg`), which is done in the next paragraph.

| Parameter name | Group | Value | Comment |
|----------------|--------|-------|---|
| splashScreen | Applet | no | Specifies the .GIF file that should appear before the applet appears. Set to NO for no splash. Leave empty to use the default splash image. |
| logo | Applet | no | Specifies the .GIF file that should appear at the Forms menu bar. Set to NO for no logo. Leave empty to use the default Oracle logo. |

| Parameter name | Group | Value | Comment |
|----------------|------------|-------|--|
| background | Applet | no | Specifies the .GIF file that should appear in the background. Set to NO for no background. Leave empty to use the default background. |
| separateFrame | Applet | false | If set to false the applet will appear within the browser window. If set to true the applet will appear as separate MDI window. |
| width | Applet | 100% | Default width of the plugin. A percentage value uses all available browser space. Values above 100% will activate a scrollbar. Omitting the percentage sign means a size in pixels. |
| height | Applet | 99% | Default height of the plugin. Values above 100% will activate a scrollbar. |
| clientDPI | <unlisted> | 100 | 100 enforces the standard size (compatible with earlier releases). <u>If not set the screens are a little smaller than in older releases which may result in prompts or characters partially invisible.</u> |

3.4.2. Configure Environment Specific Settings

Customer and/or OHI BO environment-specific entries in formsweb.cfg have to be set up depending on the actual situation.



It is advised to create a new section, which inherits the settings from the default section (use the **Create** button, do not use the **Create Like** button which copies the complete configuration), for each environment where at least the following variables should be overruled:

| Parameter name | Group | Value | Comment |
|----------------|------------|--|--|
| envFile | basic | /u01/app/oracle/product/OHI/admin/ozg_se rvlet_wls_NL.env | Specifies the environment specific file that contains environment specific environment settings. FORMS_PATH and NLS_LANG are examples. |
| formParams | <unlisted> | p_mdi_window_label=O HIVM | OHI specific variable which is used in the title of the MDI window (only visible if separeteFrame=true). |
| pageTitle | html | OHI Test Environment | The title for the browser page which is used for this environment. |

Be sure to apply each separate change to prevent losing them when switching to a new group of settings.

An environment file as indicated by the envFile setting defines at least the environment specific values for the following environment variables:

```

#-----
TWO_TASK=vozg
ORACLE_SID=vozg
#-----
#
#-----
OZG_LOG=/u01/app/oracle/product/OHI/vozg/log

```

```

OZG_OUT=/u01/app/oracle/product/OHI/vozg/out
#-----
#
#-----
FORMS_PATH=/u01/app/oracle/product/OHI/vozg/bin
#-----

```

Each environment file should be based on the default.env file which is delivered in folder

\$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.1/config. In the example setting above a file \$OZG_ADMIN/ozg_servlet_NL.env is referenced which contains settings for for example Dutch speaking production users.

From the settings in default.env you should normally comment out:

```

ORACLE_HOME
ORACLE_INSTANCE
TNS_ADMIN (should already be set in environment)
FORMS_PATH (set it at the bottom of the file)
FORMS_RESTRICT_ENTER_QUERY
PATH

```

Suppose the section is named NL, like in the screen print. In such a situation that environment settings are applied within forms by using a URL like:

<http://myhost.example.com:8890/forms/frmservlet?config=NL>

In the .env file also other environment settings can be specified like a different value for NLS_LANG in case a multi-language environment is configured.

3.4.3. Configure Registry.dat

Using Oracle Enterprise Manager Fusion Middleware Control it is possible to specify specific aspects of the look and feel of the application. Using the Fonts and Icons page from the drop down menu at the left top corner this is possible.

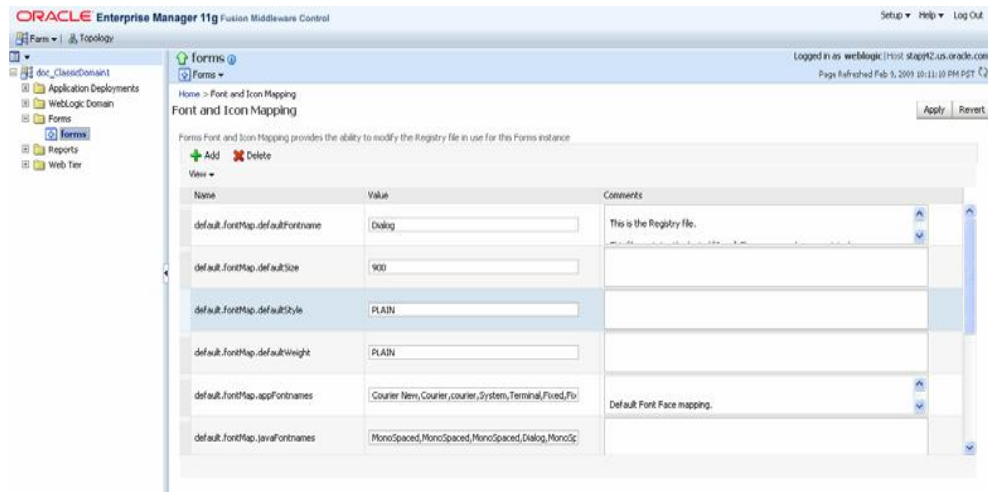
You can define whether an LOV button should appear at the right side of an item when a list of values functionality is defined.

See the picture below for such a button on the active second field:

The screenshot shows a form with five fields. The first field is 'Polisnummer' with an empty text box. The second field is 'Verzekeringnemer' with a green highlight and a button with three dots. The third field is 'Merk' with a dropdown menu showing '1 ZFW'. The fourth field is 'Collectief contract' with an empty text box. The fifth field is 'Tussenpersoon' with an empty text box.

This setting can be activated through the screen below, which shows the Font and Icon mapping page. This page shows the settings which are stored in the file

\$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.1/config/forms/registry/oracle/forms/registry/Registry.dat.



It is an option to specify `true` for the `app.ui lovButtons` setting.

Be aware that the users need to know about this setting because the look and feel slightly changes. Next to that the button may temporarily cover part of a directly following field when the cursor is in the field.

On the same page the required field visual attribute should be set to highlight **required fields**:

Please specify `true` for the `app.ui.requiredFielddVA` setting.

And adapt the color setting, for example to: 255,242,203

3.5. CONFIGURE REPORTS SERVER

In order to enable the Reports Server functionality for one or more environments a number of configuration steps are needed.

3.5.1. Change File System Access Rights

For the reports server configuration it is necessary to grant additional rights on the executables for the Oracle Instance and Oracle Home files.

It is assumed user `batch` is in the same group as the oracle software owner (typically user `oracle` and group `dba`). If not additional actions might be needed to get correct access rights.

The following changes are needed (where `world` access is not strictly necessary):

```
. ozg_init.env $OZG_ORATAB_FRS11G1
```

```
chmod 755 $ORACLE_HOME -R
```

```
chmod 755 $ORACLE_INSTANCE -R
```

```
chmod 775 $ORACLE_INSTANCE/diagnostics/logs -R
```

The `chmod` on `$ORACLE_HOME` will result in some messages about insufficient privileges. These can be ignored.

The additional privilege settings make sure the Reports Server processes can be started by the operating system user `batch` which is used for running the batch scheduler and batch processes. User `batch` is typically part of the `dba` group and receives the necessary file system access rights in this way.

When other accounts than batch need read access to the FMW software tree please additionally execute the commands below:

```
chmod 755 $MW_HOME
```

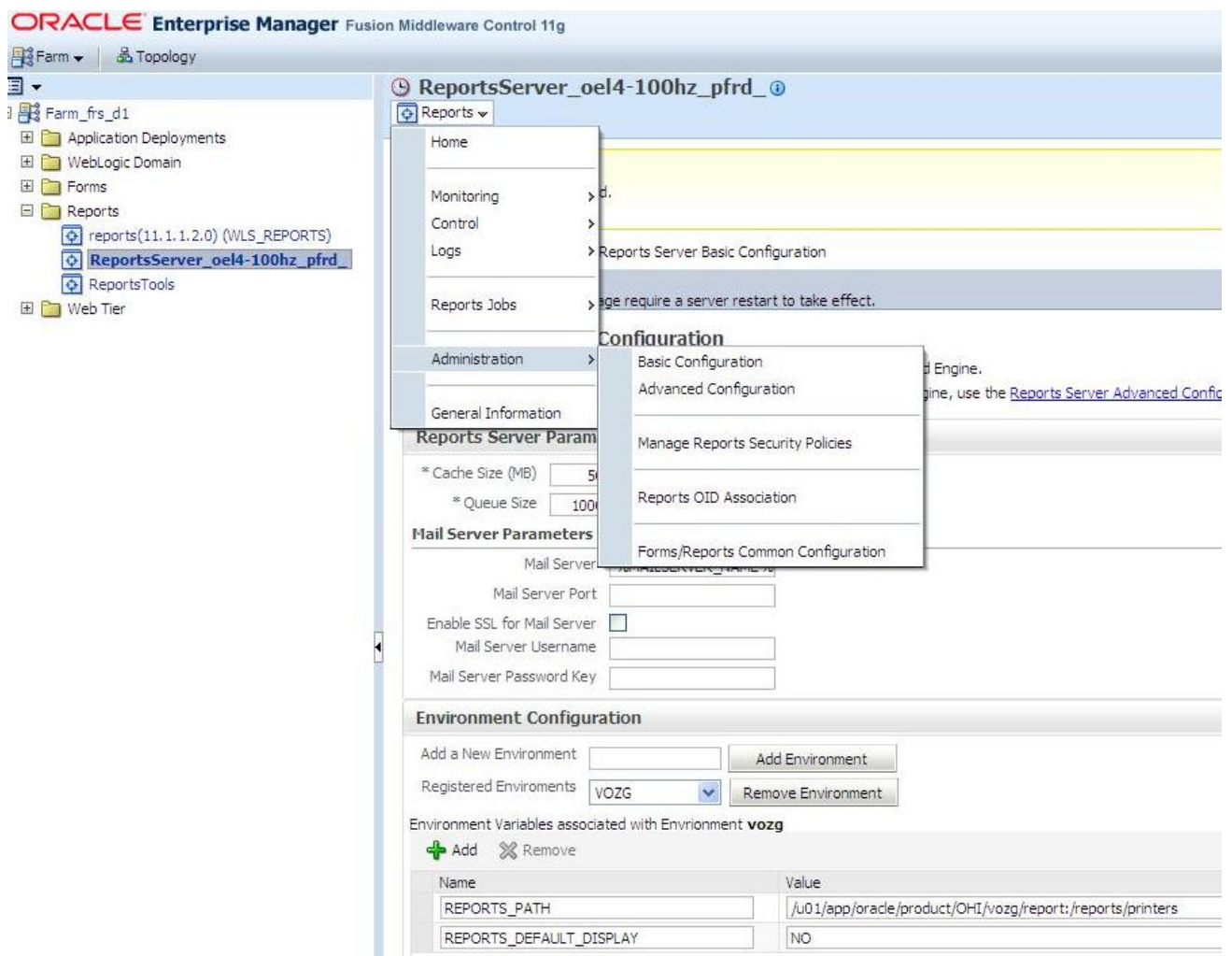
```
chmod 755 $WL_HOME -R
```

In this way also the WebLogic software becomes available for other accounts.

3.5.2. Dynamic Environment Switching

For each OHI Back Office environment specific settings should be applied.

This can be arranged again using Oracle Enterprise Manager Fusion Middleware Control 11g as shown below.



The picture shows in the left panel the navigation tree. Within the Reports branch the Reports Server definition is shown as created during the initial configuration of the PFRD environment. The actual Reports Server name is shown.

Within the the right panel the Reports drop down menu offers a 'Basic Configuration' option within the Administration option. When this Basic Configuration option is chosen an 'Environment Configuration' page appears which can be used to create an OHI Back Office environment specific setting section.

In the shown example an environment is added with name VOZG and for that environment the REPORTS_PATH and REPORT_DEFAULT_DISPLAY variable definitions are set.

In this way the file \$ORACLE_INSTANCE/config/ReportsServerComponent/<your reports server>/rwserver.conf is edited.



Attention: The environment name is case sensitive. So enter it in lowercase or uppercase depending on what you use. And beware that the environment name as shown in Fusion Middleware Control is always uppercase, even if you entered it in lowercase.

So when you get the error...

```
The specified environment ID <your environment> is invalid for
Job Type report in the command line.
```

... you should check in the specified rwserver.conf file whether the environment id is in matching case.

Beware that after changing these entries you execute a 'restartproc' of the reports server using opmnctl.

You need to define the following setting:

- the variable REPORTS_PATH (incl. path for printer definitions) ; specify typically \$OZG_ROOT/<env>/report

The variable REPORTS_DEFAULT_DISPLAY is default set to YES so does not need to be set.

The stand-alone reports server is available by default and will be used by OHI Back Office. No additional configuration action is needed.

The In-process reports server is not used, which would require the WLS_REPORTS managed server in WebLogic. This means the WLS_REPORTS managed server process is not required for running reports within OHI Back Office. It does not have to be started (we leave it defined and shutdown though) for runtime use. However, for configuring the environment as described in this paragraph you need to start the WLS_REPORTS process. You can typically do this using the WLS console. Please do not forget to stop this Managed Server after the configuration is ready (otherwise the NodeManager may bring it up again in specific situations if configured to do so).

3.6. CONFIGURE SECURITY AND MANAGE STARTUP/SHUTDOWN

For production environments some basic actions should be executed to ensure a minimum level of robustness.

At least the communication with the console ports should be encrypted and the startup and shutdown should be automated. Automatic restart of servers in case of crashes is a requirement.

For this purpose a template script is delivered to automate starting and stopping of WebLogic Server and the additional Forms and Reports services components.

Principle is to use the node manager process to start WebLogic server processes. This will be configured in such a way that in case of failures of processes these servers will be automatically restarted. The steps for accomplishing that are described below as a guidance but there are other ways to do this.

Connecting to the Admin Server and firing shutdown commands is used for stopping the processes. This because there are experiences and known problems when stopping by the Node Manager is used: this results in hangs of the stop actions.

3.6.1. Node Manager - Additional User Definition

It is advised to setup an additional user definition for the Node Manager process. This is especially useful for potential integration with Enterprise Manager. The user name is typically “nodemanager”.

Make sure you have a running Admin Server (use
`$DOMAIN_HOME/bin/startWebLogic.sh` if needed).

The steps to create the additional user:

```
. ozg_init.env $OZG_ORATAB_FRS11G1
$WL_HOME/common/bin/wlst.sh
connect('weblogic','<pw>','t3://<host>:<port>')

edit()
startEdit()
secConfig = cmo.getSecurityConfiguration()
secConfig.setNodeManagerUsername('nodemanager')
secConfig.setNodeManagerPassword('<nodemanager pwd>')
save()
activate()
disconnect()
exit()
```

Make a note of the password.

3.6.2. Node Manager - Change Properties File

In order to enable automatic restart of failed/failing Admin or Managed servers you should enable crash recovery for the Node Manager as well as execution of start and stop scripts.

This is done by setting the properties ‘CrashRecoveryEnabled’, ‘StartScriptEnabled’ and ‘StopScriptEnabled’ in
`$WL_HOME/common/nodemanager/nodemanager.properties` to true. The start script will implement the necessary domain environment settings.

3.6.3. Secure Storing of Credentials

In order to prevent having to specify an unencrypted username and password in scripts it is advised to store the `weblogic` and `nodemanager` credentials in a set of secure files.

The commands below save the credentials for both user definitions in 2 files per user (a credential file and an accompanying key file).

User weblogic:

Encrypted in file \$OZG_ADMIN/weblogic_frs_d1-WLSConfig.properties which can be decrypted by WLST using the key file \$OZG_ADMIN/weblogic_frs_d1-WLSKey.properties.

User nodemanager:

Encrypted in file \$OZG_ADMIN/nodemanager-WLSConfig.properties which can be decrypted by WLST using the key file \$OZG_ADMIN/nodemanager-WLSKey.properties.

Run the commands below while you have an Admin Server process running.

```
. ozg_init.env $OZG_ORATAB_FRS11G1
$WL_HOME/common/bin/wlst.sh
lvOZGAdmin=os.environ.get("OZG_ADMIN")
connect('weblogic','<pw>','t3://<host>:<port>')
storeUserConfig(userConfigFile=lvOZGAdmin+'/weblogic_frs_d1-
WLSConfig.properties',userKeyFile=lvOZGAdmin+'/weblogic_frs_d1-
WLSKey.properties')
```

Run the commands below while you have a Node Manager process running (use \$WL_HOME/server/bin/startNodeManager.sh for starting).

```
. ozg_init.env $OZG_ORATAB_FRS11G1
$WL_HOME/common/bin/wlst.sh
lvDomainHome=os.environ.get("DOMAIN_HOME")
lvDomainName=os.environ.get("DOMAIN_NAME")
lvOZGAdmin=os.environ.get("OZG_ADMIN")
nmConnect('nodemanager','<pw>','<host>','<port>',lvDomainName,lvDomainHome)
storeUserConfig(userConfigFile=lvOZGAdmin+'/nodemanager-
WLSConfig.properties',userKeyFile=lvOZGAdmin+'/nodemanager-
WLSKey.properties',nm='true')
```

In order to have the Node Manager be able to start the Admin Server you need to store the credentials of the weblogic user once more: store them in the file \$DOMAIN_HOME/servers/AdminServer/security/boot.properties. Please create this file and add two lines as shown:

```
username=weblogic
password=<Password>
```

The first time the Admin Server is started this files will be encrypted. As a result of this the credentials of the weblogic user are stored twice. Be aware of that when you change the password.

3.6.4. Server Lifecycles – Starting/Stopping Server Instances

It is advised to use the provided template scripts ozg_fmw_main.sh to start and stop the Fusion Middleware WebLogic Server Forms and Reports services.

You can start the FMW WLS environment using:

```
$OZG_ADMIN/ozg_fmw_main.sh start
```



Attention: When you get errors during startup of the WLS_FORMS server and the server log file reports authentication errors, please startup manually by connecting to the AdminServer through WLST and issue the `start('WLS_FORMS')` command. This will update the files in `$DOMAIN_HOME/servers/<SERVER>/data/nodemanager` directory. It might be such files are outdated.

And stopping is similar:

```
$OZG_ADMIN/ozg_fmw_main.sh stop
```

This script takes care of starting and stopping the following processes:

- WebLogic processes
 - NodeManager
 - AdminServer
 - WLS_FORMS (managed server)
- OPMN processes

When you start and stop the environment with the provided template script the Admin Server may already be started when issuing the start command after the stop command (indicated by message `Error Starting server AdminServer: weblogic.nodemanager.NMException: Server 'AdminServer' has already been started`) because of the way it has been stopped: during the stop the wlst session will be disconnected and the script continues with stopping the NodeManager process. When this is executed fast the NodeManager assumes the AdminServer was still running and restarts it during the NodeManager start.

After a server reboot without a graceful shutdown you will receive messages the Admin as well as the Managed server are already started. This is done again by the NodeManager, which takes considerably longer in such situations to start.

It is advised to incorporate the calls to the template script `ozg_fmw_main.sh` within the `ozg_main.sh` template script so your environment can be started and stopped automatically when `ozg_main.sh` is called within a service definition.

3.6.5. Securing Console Communication

In order to protect your console access it is advised to secure the communication to the console port.

This is described in the WebLogic Server documentation. In a later version of this document instructions will be included for a potential setup using ‘custom created’ unregistered certificates by setting up your own certificate authority.

3.7. LIMITING RUN-TIME FOOTPRINT OF WLS SERVERS

As described in the “Managing Server Startup and Shutdown for Oracle Weblogic Server” the server processes can be started with an option to limit their footprint. This will limit their resource usage.

The option “wlx” starts all services except:

- Enterprise JavaBeans (EJB)

- Java EE Connector Architecture (JCA)
- Java Message Service (JMS)

Because these are not needed when running plain forms and reports services you can activate this limited footprint by editing file

```
$DOMAIN_HOME/bin/setDomainEnv.sh.
```

Add the following lines behind the complete statement `EXTRA_JAVA_PROPERTIES=`"
`-Xms512m -Xmx1024m...` followed by the `export` (so after the `export` statement):

```
EXTRA_JAVA_PROPERTIES="-DserverType=wlx  
${EXTRA_JAVA_PROPERTIES}"  
export EXTRA_JAVA_PROPERTIES
```

Be sure the line is added before the line where `JAVA_PROPERTIES` is set using this setting `EXTRA_JAVA_PROPERTIES`.

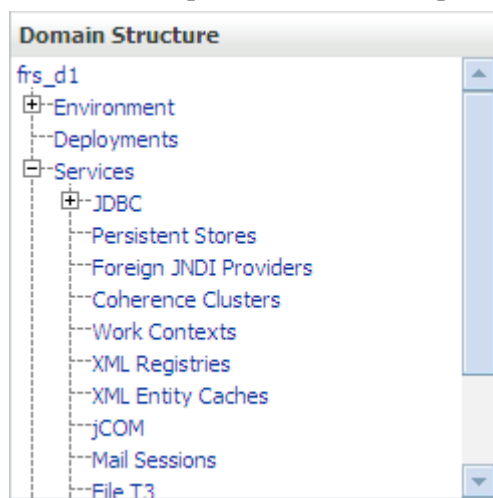
When you restart your server processes this setting is applied.

You can notice whether this setting is active by looking in the Administration console whether messaging services are available.

The picture below shows them:



And this is the picture when the footprint is limited:



The messages branch below Services is missing as you can see.

3.8. ADDING MANAGED SERVERS TO 'CLUSTER_FORMS'

As advised during configuration of the forms and reports services a cluster setup for the managed servers should be chosen. As a consequence of that a WebLogic cluster named 'cluster_forms' will have been created.

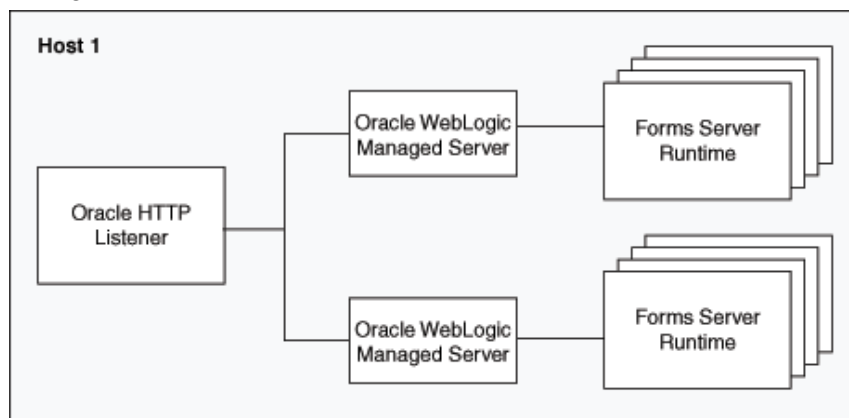
In EM control this will show up like:

| Name | Status | Host |
|-----------------------------|--------|-----------------------|
| WebLogic Domain | | |
| frs_d1 | | |
| AdminServer | ↑ | 192.168.10.6 |
| cluster_forms | ↑ | |
| WLS_FORMS | ↑ | nloz06.nl.oracle.c... |
| cluster_reports | ↓ | |
| WLS_REPORTS | ↓ | |
| Forms | | |
| forms | ↑ | nloz06.nl.oracle.c... |
| Reports | | |
| ReportsServer_nloz06_frs_i1 | ↑ | nloz06.nl.oracle.c... |
| ReportsTools | | nloz06.nl.oracle.c... |
| Web Tier | | |
| ohs1 | ↑ | nloz06.nl.oracle.c... |

It might be that given your user community and their usage at a certain moment one manager server is not able to handle the load the listener servlet processes impose.

The steps for tuning the Forms Listener servlet are similar to steps for tuning any high throughput servlet application. You have to take into account resource management and user needs for optimal tuning of your particular Forms Services configuration. For more information, see Oracle Fusion Middleware Performance Guide.

One of the options is setting up additional managed server processes and load balance the requests. This is described in the Forms Services Deployment Guide. A potential configuration is shown below:



An option to add an additional Managed Server to the cluster 'cluster_forms' is creating a new server manually and adding it to the cluster or by cloning it (the Clone option in the Control tab in the Administration Console):

Servers (Filtered - More Columns Exist)

New Clone Delete

| <input type="checkbox"/> | Name ↕ | Cluster |
|-------------------------------------|--------------------|-----------------|
| <input type="checkbox"/> | AdminServer(admin) | |
| <input checked="" type="checkbox"/> | WLS_FORMS | cluster_forms |
| <input type="checkbox"/> | WLS_REPORTS | cluster_reports |

New Clone Delete

You need to specify a new name and port for the new managed server.



Attention: In the WebLogic Server Basic license, the use of the WebLogic Server Administration Console for cloning a Managed Server instance is not permitted.

When the new Managed Server has been added to the cluster it can be started. It should be added to the `ozg_fmw_main.sh` script.

In order to make sure the requests are balanced over the available Managed Servers the Oracle HTTP Server settings have to be adapted in

```
$ORACLE_INSTANCE/config/OHS/<OHS_INSTANCE_NAME>/moduleconf/forms.conf.
```

Please change the WebLogicCluster line and add the additional servers like in the example below:

```
<Location /forms>
  SetHandler weblogic-handler
  WebLogicCluster <HostName>:9001, <HostName>:9010
  DynamicServerList OFF
</Location>
```

Of course you need to restart the OHS afterwards to enable these settings.

3.9. UNDERSTANDING THE BATCH SCHEDULER

This section (briefly) describes the functioning of the batch scheduler.

This description serves to clarify the Oracle Reports Server settings described elsewhere.

Batchscheduler

The OHI Back Office batch scheduler is a Pro*C daemon (a continuous process) which performs *iteratively* the following activities via the OHI Back Office application requested scripts:

1. Starting the new script requests
2. Checking the ongoing script requests

This process of starting and checking can be influenced by means of a number of settings in the application (screen *Systeem/Beheer/Algemeen/Systeemparemeter*, tab page *Batch scheduler*) (for a more elaborate description of the batch scheduler functionality, see function SY54 in the Oracle Designer Repository).

One of the settings is the so-called *Polling interval*; during this period the batch scheduler “sleeps” and will only be wakened by a newly created request; in this condition you have to wait for already planned script requests to be started. So only newly created requests will wake the batch scheduler immediately.

1 - Starting new script requests

When incoming script requests (newly created or already planned) have to be started, first there is a check to see if the maximum number of parallel script request has been reached or not.

In order to determine this number (to be set up in OHI Back Office by means of *Max. parallel script requests*) the number of script requests with status *Start* and *running* is summed.

If this number has not yet been reached, then the new script request can be started. If the number has been reached, then the script request to be started will still have status *waiting*.

The available “capacity”, this means the number of script requests that can still be started, can be retrieved in the log file of the batch scheduler (`$OZG_LOG/<batch account>_<sid>_<host>.log`) when it runs in *verbose* mode.

If no script requests are created in the *Polling interval* period, and if therefore no new script requests have to be started, the batch scheduler will proceed with activity 2; checking the ongoing script requests.

2 - Checking the ongoing script requests

Regarding the started processes (= “ongoing script requests”) 2 matters are checked:

1. Has the script request been started within the margin?
2. Is a script request with status “Running” (ongoing) indeed still active?

Re 1. Has the script request been started within the margin?

If the batch scheduler observes a script request in table `ALG_SCRIPT_AANVRAGEN` waiting to be started, then the status will have the value `W` (*waiting*).

If the request has to be started, then the batch scheduler will change the status to *Start* and the process will be started on the OS.

The data of the started process can be found in the log file of the batch scheduler if it runs in *verbose* mode.

An example of a script request process of the type Oracle Reports

```
11:07:10:nohup nice -10 OZG_CMD.pl rwclient.sh p_id=59384
recursive_load=no blankpages=no p_user=SCOTT batch=yes server=rep_myhost
report=$OZG_BASE/report/ZRG1234R desname=$OZG_OUT/scott/59384.out
mode=character nonblocksql=no destype=file envid=$ORACLE_SID
userid=OZG_BATCH/OZG_BATCH@$ORACLE_SID
>$OZG_LOG/scott/59384.log 2>&1 & echo $!
```

The script *itself* will (by means of the generic startup code) set the status of the relevant script request to *Running*.

If the status transition *Start*->*Running* has not occurred within the configured period (batch scheduler setting *Start delay*), the “Bijzonderheden” (Details) section of the script request will display the following message “*job niet gestart binnen marge*” (job not started within the margin). The status of the script request will then be set to “Failed”.

Normally, the script will be started though: the process has been started already, but it simply has not reached the startup code yet within the specified “*start delay*”.

Once the script is ready it will then set the status of the relevant script request to *Ready*

or `Error` (= functional error).

The possibly occurring **ERROR** message “*job not started within the margin*” will then be changed into an **INFO** message.

What are the possible causes of a script not starting in time?

1. **Start delay too low**

This setting has been configured so low that the server cannot activate the script fast enough.

2. **Server capacity insufficient**

The server is loaded so heavily that it takes long before a script is activated.

3. **Insufficient Oracle Reports Server resources**

There are insufficient available resources in Oracle Reports to start a script of the type Oracle Reports; the report will be on hold until these Oracle Reports resources become available.

Re 2. Is a script request with status “running” indeed still active?

If the script request has status `Running`, then there is a check on the OS or in the database (depending on the type of script) to ensure that a process is indeed still active for this script request.

If it turns out that this process is no longer active, then the script request is considered as `Failed`; the process has not been finalized correctly (given that the end code in which the status is set to `ready` or `error` has not been executed).

After performing the above checks (= activity 2 – *Checking the ongoing script requests*) the batchscheduler will proceed with activity 1 (*Starting new script requests*), etc.

Different types of batches

The scripts started by the batchscheduler can be of different types. These types include for example *SQL module*, *OS shell script* and *Oracle Reports*.

The behavior of scripts of the latter type can be influenced by the settings for *Oracle Reports*, a component of the *Application Server* (in case of OHI Back Office this refers to *Forms & Reports Services*). This is described in the next paragraph.

3.10. OPTIMIZING ORACLE REPORTS

Within the settings for Oracle Reports a distinction has to be made between the settings for the *Reports Server* and the settings for the *Reports Engines* (processes that actually handle the reports).

Functionality

What takes place in Oracle Reports when a script of this type (= a *report*) is requested?

By means of the batchscheduler, the previously mentioned (Oracle Reports) `rwclient.sh` process is started.

The Reports Server will now determine if a Reports Engine is active in order to handle the process.

If an Engine is active, this will be used to run the report.

If *no* Engine is active, one will be started.

An Engine is a java process (for which any relevant details can be set up by means of parameter `jvmOptions`), which can run *one report at the same time*, but with several reports after each other; the reports will then remain to use *the same* Engine (=

java process).

Given that as starting overhead of an Engine has a price, it could be profitable to have one Engine process multiple reports sequentially (*recycling* the process).

This and other aspects can be configured using the Enterprise Manager control. The file which is edited in this way is named:

\$ORACLE_INSTANCE/config/ReportsServerComponent/ReportsServer_<host>_frs_il/rwserver.conf

The number of reports that can be handled by an Engine is set up by means of parameter `engLife` which is specified in the Basic Administration page of the reports server with the 'Maximum Jobs Before Restart' setting.

See below:

| Reports Engine Parameters | |
|--------------------------------------|----------------------------------|
| * Maximum Engines | <input type="text" value="1"/> |
| * Minimum Engines | <input type="text" value="1"/> |
| * Maximum Jobs Before Restart | <input type="text" value="50"/> |
| * Maximum Idle Before Shutdown (min) | <input type="text" value="30"/> |
| Default Env ID | <input type="text" value="N/A"/> |
| JVM Options | <input type="text"/> |

After this value has been obtained, the Engine (and therefore the relevant java process) will be stopped. The following reports will then have to be handled by a different (existing or new) Engine.

The number of Engines that has to remain active at least is configured by means of parameter `minEngine`.

The maximum number of Engines is configured by means of parameter `maxEngine`. Both settings are shown above as the first two settings.

If an Engine has been idle longer than the configured parameter `maxIdle` allowed, it will stop automatically. This is the fourth setting shown above.

By means of parameter `keepConnection`, which must be set for OHI Back Office to "no" by means of the Advanced Configuration page with the 'Keep Database Connection' setting, it is indicated that for each report a new (database) connection has to be created; the various reports in 1 Engine therefore make use of different database sessions.

| Reports Server Parameters | Reports Engine Parameters |
|---|--|
| Maximum Connections <input type="text" value="50"/> | Engine Response Timeout (min) <input type="text" value="0"/> |
| Connection Idle Timeout (min) <input type="text" value="15"/> | Reports Source Directory <input type="text"/> |
| Maximum Cached Files <input type="text"/> | Reports Temp Directory <input type="text"/> |
| Job Retries <input type="text" value="0"/> | <input type="checkbox"/> Keep Database Connection |
| | <input type="checkbox"/> Enable Engine Diagnostics |

The maximum number of connections permitted for Engines and reports is configured by means of the parameter `maxConnect`, the 'Maximum Connections' setting above.

If this value is configured too low, this will lead to message "REP-56055: Exceed max connections allowed: ...". This can happen based on the other settings

(`maxEngines` & `keepConnection`) which may result in having more reports active than connections.

The documentation can provide additional information regarding this and other engine settings.

URLs

The following URLs display various data regarding Reports Server, Reports Engines and the Reports Server queue status:

<http://<host>:<port>/reports/rwservlet>

which contains an overview of other URL's like

<http://<host>:<port>/reports/rwservlet/help>

<http://<host>:<port>/reports/rwservlet/showjobs>

<http://<host>:<port>/reports/rwservlet/getserverinfo>

Sizing

Of importance for the configuration of `minEngine` and `maxEngine` is that Oracle Reports Server *strives* to meet these values; *in case insufficient resources (CPU & RAM) are available, then the values cannot be obtained.*

It is therefore of importance for the DBA to determine the real values for their system.

Also for the (`batchscheduler`) setting *Max. parallel script requests* the DBA has to determine what their system can (and is allowed) to handle.

The value for "*Max. parallel scriptaanvragen*" (*max. parallel script requests*) is strongly related to `maxEngine` & `maxConnect`.

If the parallel script requests are all of the type 'Oracle Reports' then this number of Engines & connections could be required to handle reports.

This number has to be greater than *Max. parallel script requests*; the reason for which is that a Reports process can still be active while writing the (very large) output file, while the status of the matching script request could be set to `Ready` (or `Error`) already.

The script request will no longer be considered as a "lopende scriptaanvraag" (running script request), while a process is still active though (during a short period of time); therefore it is (temporarily) required to have more connections than the allowed number of parallel script requests.

3.10.1. Reports Server System Parameter Settings in OHI Back Office

For a number of the settings mentioned earlier, the maximum number of parallel script requests is used, as can be configured in the system parameters of the OHI Back Office sub-system (*Max. parallel script requests*).

"*Max. parallel scriptaanvragen*" (*max. parallel script requests*) has to be configured based on the capacity of the relevant server; a guideline is $(2 \text{ to } 4) \times (\text{the number of available CPU cores on the server})$. Settings that are too *high* can lead to an overload of the server.

3.10.2. cacheSize

Property `cacheSize` has to be set to at least 3x the size of the largest *Reports* output file.

The value is indicated in MB.

The Basic Configuration offers the setting field for this.

| Reports Server Parameters | |
|---------------------------|-----------------------------------|
| * Cache Size (MB) | <input type="text" value="50"/> |
| * Queue Size | <input type="text" value="1000"/> |

Example:

The largest Reports output file is 100MB; `cacheSize` has to be 300MB

3.10.3. maxEngine

Parameter `maxEngine` has to be set to (the number of available CPU cores on the server) * (6 to 8 = recommended by Oracle for Unix/Linux platforms).

The value has to be at least identical to "Max. parallelle scriptaanvragen" (max. number of parallel script requests).

Example:

On a server with 8 CPU cores: $8 * 6 = 48$

3.10.4. engLife

For the same tag, parameter `engLife` has to be set to 4-30.

The default value of 50 (this means 1 Reports Engine, a java process, can process 50 reports sequentially before the process is started again) could lead to memory leaks.

3.10.5. minEngine

Parameter `minEngine` has to be configured based on the actual requirements; the setting depends on the load and the available capacity. It is recommended to choose *at least* 2-4. This setting will determine the minimum nr of engines which will be started.

See also [Tuning Oracle*Reports](#).

3.10.6. keepConnection

For this same tag, property `keepConnection` has to be set to `no` (to this end the comment symbols for property have to be deleted!).

3.10.7. maxConnect

Parameter `maxConnect` has to be configured to 10x "Max. parallelle scriptaanvragen" (max. parallel script requests).

Example:

```
ALG_SYSTEEM_PARAMETERS.BATCH_MAX_PROCESSEN = 24  
maxConnect=240
```

4. CONFIGURATION ORACLE APPLICATION SERVER (SELF SERVICE)

Please refer to



ORACLE HEALTH INSURANCE CERTIFICATION FORM

for specific instructions on installing the Application Server environment for OHI Self Service.

5. INSTALLATION OHI BACK OFFICE APPLICATION SOFTWARE

Installation of the OHI Back Office application software consists of the following steps:

5.1. CREATING ACCOUNTS & AUTHORIZATION

In order to create the required accounts and roles and to assign the correct authorizations, the OHI Back Office installation script `OZGI001S.sql` has to be run via `SQL*Plus` under account `SYS`.

This script creates an Oracle database scheme which contains OHI Back Office objects: `OZG_OWNER`.

Additionally, for object authorization a database role is created: `OZG_ROL`.

For the OHI Back Office Batchscheduler a separate account is created (formerly, up to release 2009.03 “identified externally”, currently a regular account in the database): `BATCH`. If necessary, it is possible to use a different account name; the name of the account to be used has to be registered in the System parameters.

In order to process script requests via the Batchscheduler of type Oracle Reports by means of the Reports Server, account `OZG_BATCH` is created. Also here a different name can be used; the account name to be used can again be configured in the System parameters. However, the password of the account to be used always has to be *identical* to the account name.

Finally, all required authorizations will be distributed.

5.1.1. Security of the Batch Scheduler Account

Access to the batch scheduler account is arranged by means of a Secure External Password Store (SEPS) from release 2010.01 onwards. This SEPS feature uses an Oracle Wallet. A SEPS can store one or more username/password combinations in an encrypted file.

The wallet will be used to pass the username/password combination for the batch account during batch processing.

Before the wallet can be used to pass credential information to the database for Oracle Net connections, the Oracle Net client must know where to look for the wallet. It is specified in the `sqlnet.ora` file as the `WALLET_LOCATION` parameter and should specify the directory location of the wallet created in the next chapter.

In this example we will create the wallet in the `$ORACLE_HOME/network/admin` directory, so the following entries will be added to the `sqlnet.ora` file:

```
WALLET_LOCATION =
(SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
        (DIRECTORY =
            /opt/oracle/product/11.1/db_1/network/admin)
        )
    )
SQLNET.WALLET_OVERRIDE = TRUE
SSL_CLIENT_AUTHENTICATION = FALSE
```

These settings cause all `sqlplus /@<db_connect_string>` statements to use the information in the wallet at the specified location to authenticate to the database. The `<db_connect_string>` identifies a username/password combination in the wallet, if present. So a connect string which is known in the wallet identifies exactly one username/password combination.

The wallet itself consists of two files, `ewallet.p12` and `cwallet.sso`. The last one contains the actual username/password combinations and must be protected with correct operating system access rights: any OS user who can read the wallet can use it to connect to the accounts stored in the wallet! Only the OS user `batch` and the oracle software owner need this file access.

Create the wallet using the syntax `'mkstore -wrl [wallet_location] -create'`. The example below creates it in the current directory (identified with '.'):

```
mkstore -wrl . -create
```

The two files are now created which implement the store (`ewallet.p12` and `cwallet.sso`). You will be asked to define a password to protect the contents of the wallet. This password will be asked for each later command you apply to the wallet.

Store a credential using:

```
mkstore -wrl [wallet_location] -createCredential [db_connect_string] [username] [password]
```

A credential consists of a combination of a database alias, username and password (the alias must be a known 'service name alias', a TNS entry, in `tnsnames.ora`; so add it to the `tnsnames.ora` if necessary).

The simplest way is creating an entry for the existing environment database alias, for example for `'acct'` or `'prod'`. However, this means that every OS user who uses for example the syntax `'sqlplus /@acct'` (and has operating system read access to the wallet files) will connect as user `batch` to the `acct` environment.

When you want to make it more clear that a specific alias identifies the username/password combination for the batch account it might be a good idea to use an alias like `'acct_batch'` and store the username/password for this connect string.

For a RAC environment this should be the way to go to have a specific service identifying the node(s) on which the batch scheduler (and the batches) should run.

When a connect string is used which is different from the standard connect string the starting and stopping command of the batch scheduler requires this special connect string as additional parameter (see later).

An example for when the wallet is in the current directory, a specific connect string is used and the password of the batch account has become `'ohibo'`:

```
mkstore -wrl . -createCredential prod_batch batch ohibo
```

Instead (or additional) you can add also an entry for the 'regular' Oracle Net alias, which is `prod` in the example below:

```
mkstore -wrl . -createCredential prod batch ohibo
```

This same change implies also for the BI environment. You should also create a credential for the alias that is used as database identification for the BI database in the BI related batches (`ZRG0E01S`, `ZRG0S01S` and `ZRG0D01S`).

IMPORTANT: If you do use a different connect string to identify the batch username/password combination for the BI batch user be aware that the 'application users' should use this new connect string to define the connection to the BI database. Inform them about this in that case!

Using the following syntax a list of currently defined credentials can be shown:

```
mkstore -wrl [wallet_location] -listCredential
```

Should it be necessary to change an existing credential, than use the following syntax:

```
mkstore -wrl [wallet_location] -modifyCredential [db_connect_string] [username]
[password]
```

An example for when the password of the batch account above has changed to ohibo2010, issue the following command:

```
mkstore -wrl . -modifyCredential prod_batch batch ohibo2010
```

Finally, a credential can be removed by using the syntax:

```
mkstore -wrl [wallet_location] -deleteCredential [db_connect_string]
```

So credential prod_batch can be removed by issuing:

```
mkstore -wrl . -deleteCredential prod_batch
```



Attention: When connecting as user oracle to the database using the syntax 'sqlplus / as sysdba' read access to the wallet file cwallet.sso is needed when the wallet is activated in the active sqlnet.ora file, even when TWO_TASK is unset. So make sure both the OS user which is owner of the oracle software as well as the batch user have read access but other users not.

In order to secure the batch scheduler account(s) additionally from unauthorized use, it is *possible* to create a logon database trigger under this account; starting with database version 11.1.0.7. this is no longer a *mandatory* activity.

Details

The Oracle user for batch processing is identified by means of the wallet file. So when someone gets hold of the wallet files he/she can log in with '/@<alias>' instead of by entering a username/password combination. When you know someone has 'stolen' the wallet you can of course change the password of the batch account but you can also additionally prevent unauthorized access.

Starting with database version 10.2.0.4, the batch user no longer acquires the secure application role OZG_ROL, which is a prerequisite for performing mutations. The privileges granted to this role are however also granted to the batch specific role OZG_ROL_BATCH, which in turn should be granted to the batch account.

The downside of a stolen wallet content is that PC users with sufficient permissions on their own computer may claim to be the batch user and they could then log in using the wallet on the OHI Back Office database. So it is clear the access to the wallet files must be limited.

This potential problem (which can easily be prevented) can be additionally safe guarded by checking, in a logon database trigger, for the Oracle batch user, by means of the IP address of the client session, to see if the client uses a reliable server and if he has the required permissions to log on. If the address is not included in the list of permitted IP addresses, then the session will be refused and the database connection will be aborted. In this way the administrator has an additional means to prevent improper use of the batch account.

Example

By way of an example, this is a logon trigger for a fictitious situation:

| | |
|-------------------|---|
| Batch user | batch (both under OS and the database) |
| Permitted servers | Local connection on the database server 144.21.160.66 (database server via Oracle*Net) 144.21.160.68 (application server) |

The matching trigger code is included here (and is also available on iProjects)

```
create or replace trigger ozg_on_logon_batch
after logon on batch.SCHEMA
begin
  /* Perform optional IP checks as mentioned in CTA13508.doc*/
  declare
    l_addr varchar2(100) := sys_context('userenv','ip_address');
  begin
    if nvl(l_addr,'local') not in ('local'
                                   , '144.21.160.66'
                                   , '144.21.160.68')
    then
      raise_application_error('-20001','Connection refused');
    end if;
  end;
end ozg_on_logon_batch;
/
```

Implementation of Logon Trigger

The implementation of the logon trigger is as follows:

- Determine the name of the batch account
- Determine the servers that have access to the batch account
- Modify the trigger code from the sample to the actual situation
- Create the trigger using a user with DBA privileges (e.g. SYS or SYSTEM).
If the trigger is not created under the batch user then it will *not* be activated when logging on!

5.2. INSTALLATION OF THE APPLICATION

There are a number of ways to install the application, of which the most important ones include:

5.2.1. An Initial Installation

This is involved if there is an empty scheme (created as described in the previous paragraph) in which all objects are installed from scratch.

In order to create the OHI Back Office objects for the owner account OZG_OWNER and to locate the application sources, an OHI Back Office initial release has to be installed.

A description can be found in the following document:



OHI Installation of releases

5.2.2. A Copy of an Existing Installation

This is involved when an existing environment is copied to the new environment;

1. on the one hand the database side of the application is transferred (see [database_reorganisatie](#) for a more elaborate description),
2. on the other hand the application side (by means of copying the \$OZG_BASE directory, and afterwards modifying the number of settings; to this end see the following paragraphs).

5.3. COMPILE AND CHECK APPLICATION SOFTWARE

5.3.1. Compilation

Now compile all relevant application objects (database and application server sid); execute this by means of the following activities in the installation menu:

1. Activity 120 for the compilation of database objects (e.g. packages and procedures in the database)
2. Activities 800 and 810 for the compilation of client objects (e.g. screens, menus and reports)

A description can be found in the following document:



OHI Installation of releases

5.3.2. Check

Now check the findings of OHI Back Office Object Check (via installation menu activity 900) or by starting a script request in the application (and process the findings).

5.4. CONFIGURING DIRECTORIES

On screen “System/Management/General/System parameter” directories have to be configured for reading, writing and authorizing OHI Back Office application files (such as output of script requests, online help information and release documentation).

For the virtual directories be sure to specify the correct port of the Oracle HTTP Server (OHS).



Attention: If RAC is used then the directories have to be created on a shared file system.

For a detailed description see:



Reading, Writing and Authorising OHI Back Office application files

5.5. REGISTRATION REPORTS SERVER

5.5.1. Registration of the Account in the Application

If after the creation of the Reports Server batch account name the name `OZG_BATCH` has not been selected, then the different name of this account has to be registered in the system parameters (screen "System/Management/General/System parameter", tab page "Batchscheduler", item "Reports batch account").

5.5.2. Registration Reports Server

Register the name of the Reports Server for OHI Back Office in the system parameters (screen "System/Management/General/System parameter", tab page "Batchscheduler", item "Reports server").

This name can be determined when listing the OPMN components likewise below:

```
. ozg_init.env <env>
$ORACLE_INSTANCE/bin/opmnctl status
```

This shows a table like below:

```
Processes in Instance: frs_i1
-----+-----+-----+-----+
ias-component      | process-type      |      pid | status
-----+-----+-----+-----+
emagent_frs_i1     | EMAGENT           |    15820 | Alive
ReportsServer_nloz05_frs_i1 | ReportsServerComp~ |    15818 | Alive
forms              | FormsRuntime      |     8558 | Alive
ohs1               | OHS               |    15819 | Alive
```

The process-type line starting with 'Reports' identifies the 'ias-component' name which should be used as Reports Server.

5.6. REGISTRATION BATCHSCHEDULER ACCOUNT

The name of the batchscheduler account (normally `batch`) has to be registered in the system parameters (screen "System/Management/General/System parameter", tab page "Batchscheduler", item "Batch account").

6. COMPLETING THE INSTALLATION

This chapter describes the completion of installation of OHI Back Office.

6.1. SYNONYMS FOR DATABASE LINKS

If synonyms have to be created for objects regarding database links (e.g. for Oracle Business Intelligence and/or GL objects), then the database link has to meet the following requirements:

1. The domain has to be included in the database link (e.g. `prod.world` instead of `prod`);
2. The database link has to have the same name as the global name of the database.

6.2. STARTING OHI BACK OFFICE BATCH SCHEDULER

The OHI Back Office Batchscheduler can now be started; see [Starting](#).

6.3. CHECKING THE INSTALLATION

The technical installation now has to be tested in order to determine whether or not it has been successful.

The minimal requirements to be tested include:

- Menu authorization
- OHI Back Office Batchscheduler
 - Functionality
 - Output
 - Messages in batchscheduler log file
 - Messages in log files of script requests
 - Availability of log and out directories for all users
 - Availability of database accounts for users of log and out directories
- Starting/stopping the complete OHI Back Office environment (including Database Server, Application Server, and OHI Back Office Batchscheduler)
- Checking the log files (see also [Checks](#))

6.4. CREATING BACKUPS

After testing the installation, a full offline backup has to be created for the relevant file systems.



Attention: In case of RAC: If ASM is used then the use of RMAN is mandatory when it comes to the database. RMAN in combination with ASM is further described in appendix B.

6.5. ACTIVATING JOBS

The regular jobs for administration now have to be activated.
See the following chapters for possible jobs in the area of Oracle and OHI Back Office.

6.6. INSTALLATION ORACLE DESIGNER

The OHI Back Office metadata can be accessed by means of an Oracle Designer Repository.

With each OHI Back Office major release an Oracle Designer dump file is provided which can be imported in an Oracle Designer Repository.

For additional information involving Oracle Designer and OHI Back Office, see:



OHI Form Certification

7. ORACLE ADMINISTRATION RELATED TO OHI BACK OFFICE

The OHI Back Office application runs on an Oracle database, and uses Application Server runtime software.

Administration of this environment is the responsibility of the DBAs of the customer.



Attention: As mentioned earlier on, this document does not intend to provide instructions in terms of the Oracle Database & Application Server administration.

The DBAs of the customer should have sufficient knowledge, skills and experience in order to perform this administration task in a correct way. If the administration is not performed correctly, Oracle can provide no guarantees as to the correct functioning of the OHI Back Office application.

In order to emphasize the importance, the most significant points of interest involving Oracle Database & Application Server administration *related to* OHI Back Office are mentioned here *explicitly*.

The following enumeration is therefore a *minimal set of mandatory tasks and activities, which are required for the correct implementation of* OHI Back Office, and they should *not* be considered as a "manual" for the DBAs. Evidently, each activity mentioned has to be tested and documented extensively.

Chapter 8 provides experience and advice as to how the administration could and should be completed. The administration activities described therein provide a good start to fill in the administration tasks actively (and therefore preventively).

7.1. BACKUP & RECOVERY

7.1.1. General

There are various options to create a database backup.



Oracle recommends the use of Recovery MANager as a database backup tool.



If ASM is used as a shared storage solution, the use of Oracle RMAN is mandatory.

Since the 10g release of the DBMS, Oracle provides the possibility to use a Flash Recovery Area (FRA) for storing backup related data.



Oracle recommends using FRA, particularly for a faster recovery of the Database.

Ensure proper backup and restore procedures for the software used (e.g. Database, Application Server, ASM, RAC, OHI Back Office, etc.) and the relevant file systems;

Testing and checking of the backup restorations is considered a standard procedure for system management, and has to be performed on a regular basis.

When it comes to using hot or cold backups, no specific requirements apply in combination with OHI Back Office.

7.1.2. RAC

In an Oracle RAC environment some additional requirements apply.

In addition to the database backup, a backup has to be created of the Oracle Cluster Registry (OCR) and the Oracle Voting Disk (OVD).

This is the case particularly if nodes have to be added or deleted.

To this end the environment has to be configured by means of `ozg_init.env CRS10G`.

Now two environment variables are configured; the link refers to the setting:

1. [OCR BACKUP](#)
2. [OVD BACKUP](#)

By means of command `ocrconfig -export $OCR_BACKUP` an export is created of the Oracle Cluster Registry.

Command `dd` allows for a backup to be created of the Voting Disk.

For example `dd if=<naam voting_disk> or=$OVD_BACKUP`.

Ensure that the backups are available on the shared disk.

7.2. STARTUP AND SHUTDOWN

7.2.1. Order

- After starting the database, the OHI Back Office Batchscheduler has to be started;
- Before the database shutdown, the OHI Back Office Batchscheduler has to be stopped;

7.2.2. Automation of Startup/Shutdown

- It is highly recommended to restart the Application Server software on a daily basis, in order to initialize the relevant processes and memory use (to clear it). If this is not an option, then it is recommended to only restart the Reports Server; to this end the *standalone* Reports Server has to be used (started through OPMN) instead of the (default) *in-process* Reports Server (running in a WLS Managed Server, which is not needed).
- Handling automatic Oracle system software startup/shutdown for startup/shutdown of the relevant servers;

7.2.3. Checks

- Check of the various Oracle system software files, e.g. trace files, alert files, audit files, AS log files `error_log*`, `access_log*`, `default-web-acces.log`, `em-web-acces.log`, OPMN log files (`o.a.opmn.log`, `ons.log` (see e.g. [Check port conflicts](#)), `ipm.log`, `application.log` (Forms & Reports!)) etc.
- Check the OHI Back Office Batchscheduler log files for possible errors;
- Check cause(s) for failed script requests (status M (failed)).

7.2.4. RAC

In an Oracle RAC environment some additional requirements apply.

In an Oracle RAC environment two or more instances may be active.

An important issue is that starting and stopping an Oracle RAC environment is performed in a slightly different manner than a single instance environment.



Use the `srvctl` command to start and stop RAC components *instead of* `SQL*Plus`.

When starting instances with `SQL*Plus`, services are not always started automatically as well. If afterwards these services are started manually, then CRS is not aware of these services.

7.3. ACCESS CONTROL AND SECURITY PRIVILEGES

- Access control of the relevant OS accounts (e.g. `root`, `oracle` and `batch`);
- Access control of the default Oracle database accounts (e.g. `SYS` and `SYSTEM`).
- Access control of the OHI Back Office Oracle database accounts (e.g. `OZG_OWNER` and `MANAGER`);
- Changing the passwords a.s.a.p. (after the initial installation);
- Configuring a password algorithm and validity duration, in other words setting up a policy in terms of password management;
- Access control when OHI Back Office is implemented in a web-based architecture on intranet or the Internet;
- Monitoring security alerts (<http://otn.oracle.com/deploy/security/alerts.htm>).
- etc.

7.4. PERFORMANCE TUNING AND MONITORING

- OHI Back Office uses e.g. the Oracle *cost-based optimizer*; to this end it is mandatory that regular jobs are run which e.g. perform the following tasks: Collecting *statistics* for *tables* and *indexes*, collecting *system statistics*, collecting statistics for the `SYS` schema.
For the OHI Back Office tables, *monitoring* is selected by default, which allows for *stale* statistics to be used.
- Regular tracing of activities in order to recognize performance bottlenecks.
- Deployment of the right disk settings for Oracle; e.g. keeping in consideration RAID levels, striping, controllers etc.;
- Performing regular checks of OS, Oracle and application performance by using the relevant diagnostics tools;
- Monitoring network performance, memory, disk I/O etc.
- Configuring and implementing Oracle Resource Management;
- etc.



Advice: Oracle recommends using Enterprise Manager Grid Control for the active day to day monitoring.

7.4.1. RAC



Oracle recommends using Enterprise Manager Grid Control for managing and monitoring an Oracle RAC environment.

To this end an Oracle Enterprise Manager *Management Pack* license is required as an extra option.

In an Oracle RAC environment one or more instances can be active.

For Database Management this entails more emphasis on performance monitoring.

In an Oracle RAC environment the most important performance issues to be recognized include the following:

- Performance of the Interconnect
As a guideline this is maximum 5ms.
- Performance of the Global Cache

If errors are encountered in terms of performance when using OHI Back Office in an Oracle RAC environment, the delivery of performance data will continue in the same way as described in section [Collecting Performance Data](#).

Additional information regarding RAC Performance Monitoring can be found in the following documentation:



Oracle Clusterware and Oracle Real Application Clusters Administration and Deployment Guide, chapter 13 ‘Monitoring Performance’.

7.4.2. Rebuilding Indexes

In actual fact, few indexes ever need rebuilding.

Conditions for rebuild would be

- Large free space (generally 50%+), which indexes rarely reach, *and*
- Large selectivity, which most index accesses never reach, *and*
- Response times are adversely affected, which rarely are.

Therefore the OHI general advice on rebuilding is:

- Do not rebuild indexes.
- Use `REBUILD` if the whole index structure is very poorly fragmented.
- Use `COALESCE` to reduce index fragmentation in a portion of the index.
- Use `SHRINK` to reduce index fragmentation in a portion of the index and the actual blocks associated to the index.

7.5. INSTALLATION, CONFIGURATION, UPGRADE,

MIGRATION AND VERSION CONTROL

- Installation of new releases and patches for Oracle Database and Application Server, possibly by means of Rolling Upgrades.
- Installation of new releases and patches for OHI;
- Planning upgrades and migrations;
- etc.

7.5.1. RAC

- If setup correctly, CRS will automatically start and stop the components required for the cluster (incl. database and listeners); check this configuration of dependencies by means of `crs_stat`;
- Use `crsctl check cluster` to check all nodes in CRS.
- Use `ocrcheck` to check the integrity of OCR (and mirror);
- Use before and after the different stages of the installation process the Cluster Verification Utility `cluvfy`.
- See [installation and configuration](#) for further information regarding the installation.

7.6. SPACE AND STORAGE MANAGEMENT

- Managing the available and required disk space;
- Managing storage settings for objects;
- Shrinking objects (e.g. based on Segment Shrink Advisor);
- Managing the development of various Oracle system software files, e.g. trace files, alert files, audit files, AS log files `error_log*`, `access_log*`, `default-web-acces.log`, `em-web-acces.log`, `OPMN log files` etc.
- Managing the development of OHI Back Office Batchscheduler log and output files in `$OZG_LOG` and `$OZG_OUT`.
- etc.

7.7. LICENSE CONTROL

- Checking license issues for the installation of new releases and/or patches or the use of certain options and/or tools;
- etc.

7.8. NETWORKING

- Making use of Oracle Net *dead connection detection* by means of configuring parameter `SQLNET.EXPIRE_TIME` in file `sqlnet.ora`;

- etc.

7.8.1. RAC

- For RAC environments it is crucial that the different nodes have the exact same system time; it is highly recommended to use NTP in this case (Network Time Protocol).
- If in an Oracle RAC environment an instance is positioned in restricted mode (= required for the installation of OHI (patch) releases), then the services for that instance will be stopped (by the Oracle Clusterware, only in case of *dynamic registration* with the listener).

Connection to the instance will still be possible via:

1. SID; this is to be used for the installation of OHI Back Office (patch) releases by means of the installation menu.
2. a Service Name; if this is used, then `tnsnames.ora` has to be modified in the connect string (see also My Oracle Support note 301099.1).

Between the `CONNECT_DATA` and `SERVICE_NAME` entries the following has to be added:
(UR=A)



Remark: Working with service names is preferred.

8. OHI BACK OFFICE MANAGEMENT

The management activities for OHI Back Office can be subdivided into tasks for System management/DBA and tasks for creating new application users.

8.1. SYSTEM MANAGEMENT/DBA

8.1.1. Oracle Environment

On the OS server *myhost* (a representative name for the servers on which OHI Back Office is run) the OS account `oracle` is the owner of the OHI Back Office files.

The OS account `batch` is the owner of the OHI Back Office Batchscheduler.

Script `$OZG_ADMIN/ozg_main.sh` can be used in the boot sequence and backup sequence of the OS server in order to stop and start the complete OHI Back Office environment (this means DBMS, AS, OHI Back Office Batchschedulers, OEM etc.).

Starting

Under OS account `root` enter the following command:

```
ozg_main.sh start
```

Stopping

Under OS account `root` enter the following command:

```
ozg_main.sh stop
```

This script uses the scripts `ozg_oracle_start.sh/ozg_batch_start.sh` and `ozg_oracle_stop.sh/ozg_batch_stop.sh` to stop/start all Oracle related processes and Batchschedulers.



If the Database layer and Application layer are installed on different servers, then script `ozg_main.sh` has to be available on both servers.



For using RAC other start and stop scripts are used. For more information see appendix B.

8.1.2. Clients

The user interface of the application OHI Back Office can be started on a client PC or Terminal client via a browser by means of the following URL:

<http://<host>:<port>/forms/frmservlet?config=Prod>

It is recommended to configure a DNS server that enables all relevant clients in the network to access the relevant host. If a DNS server is used then it is not required to manage the local host files on each client PC.

Additionally, it is recommended to position all relevant URLs for OHI Back Office on a company portal or homepage, in order to simplify the implementation and management.

The value of parameter `config` refers to the configuration as it can be found in file `$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.1/config/formsweb.cfg`.

8.1.3. OHI Back Office Database

The following provides an overview of the 4 general OHI Back Office database accounts and *initial* passwords:

1. `OZG_OWNER/OZG_OWNER`
Owner account for the OHI Back Office data structure.
2. `MANAGER/MANAGER`
OHI Back Office application manager with full menu authorization.
3. `BATCH/BATCH`
Owner account for the OHI Back Office Batchscheduler.
4. `OZG_BATCH/OZG_BATCH`
Reports Server batch account for the Reports modules via the OHI Back Office Batchscheduler.

8.1.4. OHI Back Office Batch Scheduler

The OHI Back Office Batchscheduler handles script requests which are submitted by the end users of the OHI Back Office application.

Starting

Service Name Versus Environment Name

Under OS account `batch` enter the following command for the environment in question; the Environment Name of the environment for which the batchscheduler has to be started is provided as a parameter.

```
. ozg_init.env environment_name  
$OZG_BASE/sh/OZG_START_BATCH.sh environment_name
```

The script will start for 1 environment, 1 batchscheduler on 1 server.

It is possible to use a separate service name next to an environment name. This is typically useful when a Secure External Password Store is used with a name which is different from the environment name and which identifies the batch username/password combination. This service name can be passed as 'connect' parameter:

```
. ozg_init.env environment_name  
$OZG_BASE/sh/OZG_START_BATCH.sh -connect service_name  
environment_name
```

Verbose Mode

By means of parameter `-verbose` (optional) it is possible to start the batchscheduler in *verbose* mode; in this mode extra information is written in the log file, which allows for any problems or questions to be examined:

```
. ozg_init.env environment_name
$OZG_BASE/sh/OZG_START_BATCH.sh -verbose -connect service_name
environment_name
```

Retention Period for Log Files

When starting the batchscheduler, the last (optional) parameter is the retention term (in days) of the log files of the batchscheduler.
By default this is 7 days.

```
. ozg_init.env environment_name
$OZG_BASE/sh/OZG_START_BATCH.sh environment_name 14
```

Stopping

Under OS account batch enter the following command for the relevant environment:

```
. ozg_init.env environment_name
$OZG_BASE/sh/OZG_STOP_BATCH.sh environment_name
```

Or when a separate service name is used to connect to the user batch:

```
. ozg_init.env environment_name
$OZG_BASE/sh/OZG_STOP_BATCH.sh -connect service_name
environment_name
```

The script will stop for 1 environment, all batchschedulers on all servers.

In the overall scripts `ozg_batch_start.sh/ozg_batch_stop.sh`, which are called by `ozg_main.sh`, the aforementioned calls are included to start/stop *all batchschedulers for all environments on 1 server*.



The OHI Back Office Batchscheduler runs on the Application layer.



See [Appendix A - configuration multiple batchschedulers](#) for the implementation of *multiple* batchschedulers in case multiple application servers are allocated.



For the configuration of the Batchscheduler in an Oracle RAC environment, please refer to appendix B.

OHI Back Office Batchscheduler Output and Log Files

OHI Back Office recommends to regularly clear the OHI Back Office Batchscheduler output files in the directory `$OZG_OUT` and the OHI Back Office Batchscheduler log files in the directory `$OZG_LOG`.

8.1.5. Changing Settings

Changing Software Trees

In a standard configuration, after the installation of OHI Back Office, the Oracle environment is setup in the Application Server software tree.

In order to configure the Oracle environment in the 11gR1 Database Server software tree (in a configuration for which the Database layer and Application layer are installed on the same server), the following script can be used:

```
. ozg_init.env $OZG_ORATAB_DB11G
```

In order to reset the environment for the 11g R1 Application Server software tree, the following calls can be used:

Example:

```
. ozg_init.env $OZG_ORATAB_FRS11G1
```

Changing Environments

In order to configure the OS environment for a specific environment for which the Application Service software tree is used, the following commands can be used, e.g.:

```
. ozg_init.env prod # configuring the prod environment settings
```

```
. ozg_init.env test # configuring the test environment settings
```

In order to configure the OS environment for a specific environment for which the Database software tree is used (e.g. for startup/shutdown of the instance), the following commands can be used, e.g.:

```
. ozg_init.env prod          # configuring the prod service_name
```

```
. ozg_init.env $OZG_ORATAB_DB11G # configuring the 11g database software
```

```
. ozg_init.env test          # configuring the test service_name
```

```
. ozg_init.env $OZG_ORATAB_DB11G # configuring the 11g database software
```

8.1.6. Adding New OHI Back Office Application Environments

When a new OHI Back Office application environment is required, the following steps have to be performed in the Application layer:

1. Add a Batchscheduler start command for the new environment in
\$OZG_ADMIN/ozg_batch_start.sh.
2. Add a Batchscheduler stop command for the new environment in
\$OZG_ADMIN/ozg_batch_stop.sh.
3. Configure a new environment *SID* in
\$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.1/config/formsweb.cfg by means of the Enterprise Manager control application for the Forms and Reports services.
4. Add an ozg_servlet_wls_<SID>.env file with appropriate settings in
\$OZG_ADMIN (if you referenced this location in the variable).
5. Add an additional entry in the Reports Server setup for this environment to specify the correct REPORTS_PATH variable.
6. Create a new application directory \$OZG_ROOT/environment_name and complete it with the relevant OHI Back Office application software.
7. Install the OHI Back Office application software for the relevant environment.

8.2. CREATING NEW APPLICATION USERS

The following steps have to be performed when configuring a new end user for the OHI Back Office application:

8.2.1. Creating an Oracle Account

First of all, an Oracle account has to be created in the database for the new end user. Subsequently, the required system privileges have to be assigned to the new account.

This step can be performed by means of the `CREATE USER` command in SQL*Plus as user `SYS` or `SYSTEM`

Example

```
create      user          scott
identified by            tiger
default      tablespace  users
temporary   tablespace  temp
quota        unlimited on users
/
grant create session to scott
/
grant alter session to scott
/
```



If OHI Back Office is configured with SSO the user must be created in the Oracle Internet Directory, see section [Configuring Database Authentication](#)

8.2.2. Creating Directories for the New User

For each new end user directories have to be created in the Application layer for OHI Back Office Batchscheduler output files and OHI Back Office Batchscheduler log files.

This has to be performed under OS account `batch`:

```
mkdir $OZG_OUT/<new account>
mkdir $OZG_LOG/<new account>
```

<new account> is the name of the Oracle account of the new end user, in *lower case*.

Example

```
mkdir $OZG_OUT/scott
mkdir $OZG_LOG/scott
```



If a RAC environment is used then the directories have to be created on a shared file system. For more information see appendix B.

8.2.3. Storing Application Authorization

Now the new end user can be authorized for application OHI Back Office. In order to establish this, start the application and log on as `MANAGER`.

1. Create a “Functionaris” (executive) for the new end user in screen “Systeem/Beheer/Autorisatie/Gebruiker” (system/management/authorization/user).
2. Now assign the required roles by means of menu authorization in the same screen. The standard menu role `MANAGER_ROL` has full menu authorization; other roles can be created and set up as desired.
3. Close the application, restart it and log on as the new end user in order to test the newly created account.
If desired, the user settings can be modified by means of screen “Bewerken/Instellingen” (Edit/Settings).
4. It is optional to configure the required functional application authorization (company access, authorization for administrative organizations, brands etc.)

8.3. INSTALLATION OHI (PATCH) RELEASES

The installation process of OHI (patch) releases is described in the OHI Release Installation Manual and is performed on the Application layer:



Oracle Health Insurance Installation of Releases

For new customers this and any other relevant details will be shown by Oracle in a knowledge transfer session.

9. REFERENCES FOR ACTIVE MANAGEMENT OF OHI BACK OFFICE

In this chapter a possible approach is proposed as to 'actively manage' the OHI Back Office. This has to be used at the time that OHI Back Office is successfully configured and operational.

Active management entails the undertaking of preventive actions so as to prevent performance problems, in particular as a consequence of passive management. It also strives to simplify the search for the cause of performance problems when they do occur. Additionally, a number of matters are tackled to apply active space management.

In case performance issues are encountered, a suggestion is made for handling the issue.

The target group for this chapter is the OHI Back Office database manager, the DBA for an OHI Back Office environment. For Unix managers, the chapter could be useful to get the hang of the database management section of this chapter in particular. For functional application managers with a highly technical interest, certain sections could function as an illustration with respect to the cause of performance problems (and probably also possible solutions).

This chapter does not focus on the background of the techniques used. Sufficient documentation regarding this matter can be found elsewhere. Main goal is to provide sufficient information to handle the fundamental issues, even if not much expertise is available regarding the subject. Experts in certain subareas would want, and can collect better or more detailed information at times.

The setup has been selected so that acquired practical experiences are shared. As expected, the chapter is an evolving section of this document which will be further elaborated in the future (based on numerous acquired experiences, both internally at Oracle and externally at OHI Back Office customers). In that sense it will never be finished and always be open for improvement (and therefore also for comments and possible suggestions).

The setup is based on an OHI Back Office database. The content cannot simply be applied to an OHI Business Intelligence (open policy) environment given that additional or different matters may be applicable.

ATTENTION: No rights can be derived from this chapter. It is just a matter of advice (unless specifically indicated differently). This is to be used at your own risk.

9.1. PERFORMANCE GENERAL

9.1.1. Memory and CPU

The following graphs provide minimum guidelines for the required CPU and memory capacity for OHI Back Office.

The guidelines are based on internal and particularly external key figures of used equipment and test results. No computed values are involved. The values, such as the ones of the equipment of various customers in the first months of 2009 can be found in a graph. Based on additional customer-specific information and an estimate of the trend to be recognized, subsequently, a certain trend line can be defined. When using

the customer key figures, it is important to understand that each customer runs more or less customer specific additional workload on the server in question, in addition to the OHI Back Office application. The graph figures should therefore not be followed blindly, but they could be used as an extra tool for sizing a server configuration.

Memory

In terms of memory, for each OHI Back Office user *at least* 20MB has to be available, but in reality users sometimes clearly require more when they use multiple sessions and screens.

RAC

In case of RAC Oracle recommends setting up the following extra allocations per node:

1. 10% extra buffer cache
2. 15% extra shared pool

CPU

With respect to the number of CPUs, historically a *maximum* of 100 users has to be considered for each CPU. However, considering the multi core and multi threading techniques nowadays available exceeding this number is possible.

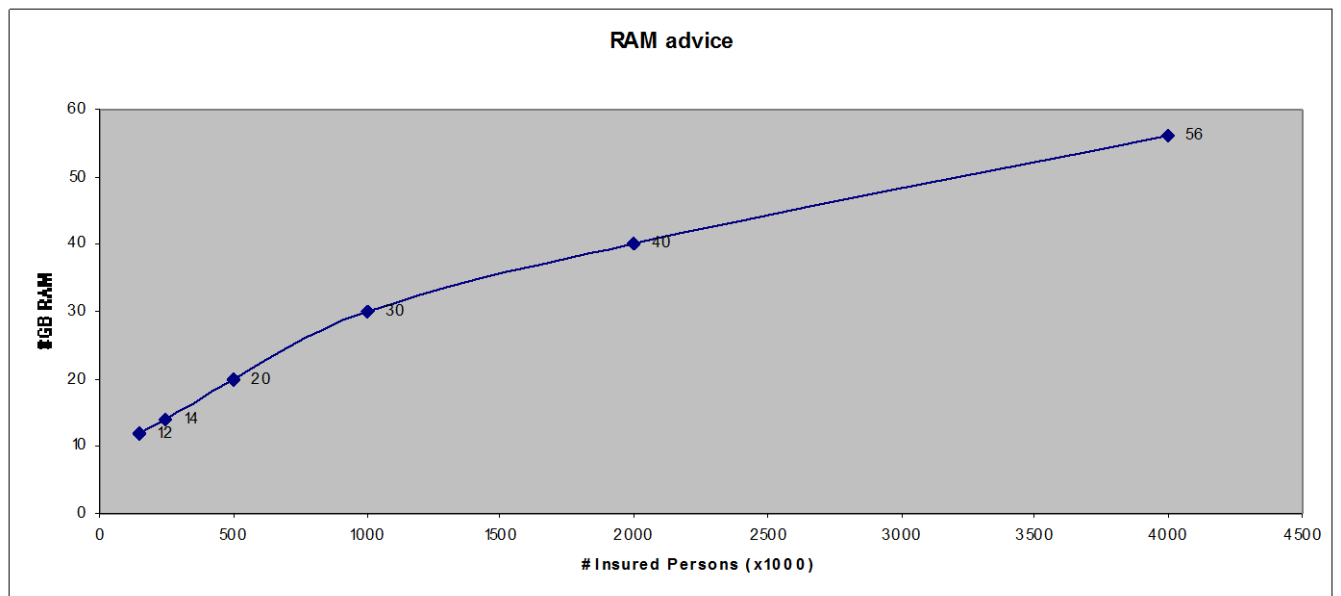
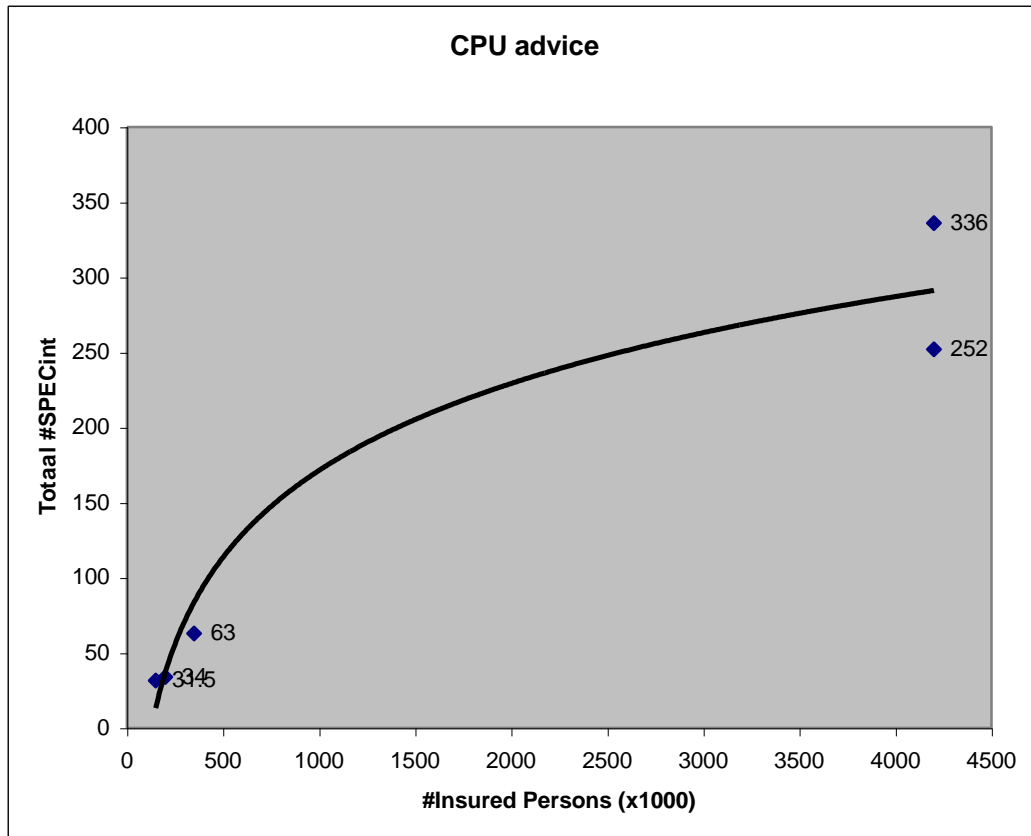
The CPU use is expressed in SPECint2006 benchmark values, which indicates the CPU capacity based on certain integer actions. This is not really a benchmark considered as representative for an application such as OHI Back Office, but it provides something to hold on to when comparing the CPU capacity of certain servers. Given that this value is available for a rather large diversity of equipment, this benchmark is applied. Transaction-oriented benchmark values would be preferable, but this is available for too few equipment in order to establish a thorough comparison of the equipment as is the case for the various customers.

On site <http://www.spec.org/cpu2006> information can be found regarding the benchmark applied.

By means of <http://www.spec.org/cgi-bin/osgresults?conf=cpu2006> it is possible to search for the benchmark values for certain types of servers. The figures used in the following CPU graph are the result of the value of field CINT2006 *rates*. For example Sun Fire E6900 with 16 dual core processors. The SPECint_rate2006 value for this server is 261.

The figures are based on combined database and application server allocation on a single server. However, for a separated database and application server, both servers will have to meet almost all of these criteria as the daytime work will primarily focus on the application server and the nighttime work primarily on the database server.

If RAC is used, Oracle recommends counting for each OHI Back Office RAC platform (i.e. all nodes together] with at least 10% extra CPU (SPECint).



9.1.2. I/O

With respect to I/O throughput it is possible to establish that a random I/O on a data file for OHI applications may take on average at the most about 5ms. More than 5ms can be considered as a potential bottle neck and cause for unwanted delay of the application response times.

In AWR output this is the column `Av Rd(ms)` in the `Tablespace IO Stats` overview

9.2. PERFORMANCE DATABASE

This section elaborates on the various aspects of performance of OHI Back Office. Important settings are assigned but utilities are also offered that may be of assistance when zooming in on possible performance issues.

The objective is to provide an overview as to how it is possible to monitor and possibly improve the performance of OHI Back Office.

By getting familiar with a normal performance and by collecting data about it, in case of actual performance problems it is easier to zoom in on the cause and nature of the problem. Especially if reference measures of a representative period with a normal/acceptable performance are available.

9.2.1. Database Settings

A smooth performance of OHI Back Office starts with optimally setting up and configuring a number of database settings.

Initialization of the Parameters

It is of importance to configure some crucial parameters in a correct manner. We list the parameters that are particularly of importance and for some cases we recommend possible values. Also many other parameters can be configured, but at certain times the following are of importance for the performance of OHI Back Office.



With respect to the installation of OHI (patch) releases, different settings apply. See the following document for more details:



Oracle Health Insurance Installation of releases

The Most Crucial Parameters

There are 2 parameters that are very significant as they determine the greater part of the SGA:

- `DB_CACHE_SIZE` (=Ym)
Use this instead of old parameter `DB_BLOCK_BUFFERS`. Specify the size of the buffer cache in multiples of 16MB. As a guideline use a value of 100MB per 50,000 insured persons in OHI Back Office. For environments with smaller numbers of insured persons, a clearly higher value has to be applied. For very large numbers - millions - this number can be adjusted.
- `SHARED_POOL_SIZE` (=Ym)
Therefore it is best to also specify multiples of 16MB. A nice guideline to start with is 1 MB per user in case of 100 users, however, with a minimum of 128 MB. For several hundreds of users it is possible to take a somewhat lower value.

Use the 'Advice' button in *Oracle Enterprise Manager (OEM)/Grid Control* to get a picture of the correctness of the configuration after the database has been used for a couple of hours (!) representatively. A good time is e.g. by the end of the morning of a typical weekday.

Currently Automatic (Shared) Memory Management is not advised. If activated make sure sufficient minimum values are set for `DB_CACHE_SIZE`, `SHARED_POOL_SIZE` as

well as `SGA_TARGET` (and also for `PGA_AGGREGATE_TARGET`, but more about that parameter follows).

Parameters for SQL Tuning

The parameters in `v$sys_optimizer_env` are of importance in order to activate the *Cost Based Optimizer* properly for OHI Back Office.

Parameters for Session Memory Allocation

There are various parameters to activate the allocation of memory for sessions, the allocation of the working memory. In the past it was possible to reserve a certain amount of memory per type of operation. However, it is possible to specify a target for all sessions together depending on the size of the total volume of working memory. The latter is normally preferred and is performed by means of the following PGA parameter.

- `PGA_AGGREGATE_TARGET` (=Ym)
It is recommended to use this setting to keep the total volume of working memory within certain limits. An automatic conversion takes place to the working memory allocation per session. The total target volume can become larger temporarily, but the memory is kept within certain limits. This has as the advantage that an objective can be set for all sessions together. A possible drawback is that specific optimization for a certain objective is not possible. Which ordinarily is not required anyway. As a guideline, a reserve of 5 MB per user can be kept for e.g. about 200 users. A multiple of 16 is not required.

For these parameters it is also possible to obtain some advice for optimization of this configuration by clicking the 'Advice' button. Advice can only be used after a representative load of the database has taken place. It is also possible to simply check if the PGA 'cache hit percentage' lies above 90% and if the maximum PGA volume and current volume do not exceed the objective by too much. Otherwise the value has to be revised upward.

Instead of automatically assigning working memory, it is possible to assign specific memory for certain operations by means of the so-called `....._AREA_SIZE` parameters. These parameters are used only if the parameter `WORKAREA_SIZE_POLICY` is set to `MANUAL`. If the last parameter has not been set then this will be set to `AUTO` by default when `PGA_AGGREGATE_TARGET` has a value that is greater than zero. It is recommended to use the following parameters only for certain purposes and to activate them by temporarily setting `WORKAREA_SIZE_POLICY` to `MANUAL`.

- `WORKAREA_SIZE_POLICY` (=manual)
Set to `manual` in order to have the following parameter values be used. Otherwise, set to `auto` or do not configure it at all.
- `SORT_AREA_SIZE` (=Ym)
Configure this parameter explicitly in case of very heavy sorting operations that have to be optimized extra. During the installation of an OHI Back Office major release, sometimes e.g. indexes are created on very large tables that benefit in case of a large value for this parameter. This could lead to a difference in throughput time when creating index factors (e.g. 2 hours becomes 30 minutes). It is recommended to set the value to e.g. 50MB or 250MB in case very heavy sorting operations have to be expedited temporarily. The relevant quantity of memory is allocated per session in case this is required, so a value of 50 MB is not desirable when

users log on. Otherwise, for certain sorting operations an unexpected high volume of memory could be allocated.

Parameters for Rollback (undo) Space

Management of the rollback space has to be automated by Oracle. The quantity of rollback space rendered available determines the throughput time for some batches as well. It is recommended to allocate some GBs of rollback space (or to allow for having this much space available by means of e.g. auto-extendable data files for the relevant table space).

The following parameter values are recommended:

- `UNDO_RETENTION (=10800)`
This value indicates the time to Oracle as to how long the rollback space should not be overwritten to provide a long-term statement - read 'consistent' - image. This value is provided in seconds. 3 hours would be an appropriate value. If sufficient rollback space is available (to be assessed via Oracle Enterprise Manager/Grid Control) and based on one hour settings (or more) 'snapshot too old' messages pop up, then the software has to be adapted to it (an incident message has to be reported).

9.2.2. Statistics

The statistics definitions have been changed since 10g. The 11g standard job is recommended for collecting statistics. This job handles collecting statistics of all schemes and for the dictionary statistics. The job will collect the statistics in the *maintenance window* for all objects in the database. The job determines the order of the database objects for which the statistics require altering. In that way, the most important statistics are defined at the moment that a *maintenance window* is closed.

System Statistics

For determining the optimum way to perform an SQL statement, the database uses a so-called execution plan. Depending on the postulated optimization objective for performing SQL statements - providing the first part of the result as soon as possible (`FIRST_ROWS_10 optimizer_mode`) or minimizing the time to determine all results (`ALL_ROWS optimizer_mode`), a different plan could be preferred.

In order to determine these 'execution plans', the 'Cost Based Optimizer' (CBO) may keep in consideration the speed of I/O of the system and the speed of the processors (the CPUs) of the server. To this end so-called 'system statistics' are required.

ATTENTION: The Cost Based Optimizer (CBO) is used by default to optimize the SQL statements as performed by the database. This is arranged via the aforementioned instance parameter.

What are System Statistics?

Up until database release Oracle 8i the Cost Based Optimizer considered the most appropriate execution plan for determining the I/O costs.

As of the launch of Oracle 9i, not only does the CBO have to keep account of the I/O costs, but also the CPU costs and the estimated required time table space (see 3 new columns in table `PLAN_TABLE`: `CPU_COST`, `IO_COST` and `TEMP_SPACE`).

The CBO since 9i can deal with CPU costs that are based on actual system characteristics, that allows for cost estimation for comparison of the complexity of planning which can take place that is close to reality.

A statement that requires relatively little (physical) I/O, but that uses a lot of CPU because many database blocks have to be accessed in cache, could clearly benefit from system statistics because the outcome of the required time for a certain plan is closely related to reality (in the past CPU and I/O used to be as expensive in a calculation, while in reality I/O is much more expensive, or more slow than CPU, although this might differ per machine used).

Therefore, the criteria for making the choice for most likely best execution plan can be established more reliably by the CBO.

Collecting System Statistics

Introduction

In order to optimize the performance of the CBO, the system statistics have to be collected again if the configuration or the use of the system undergoes clear changes (e.g. because the data is increasing and/or the database files are moved to a different location).

Collecting system statistics refers to an analysis of the database activity by Oracle during a certain period of time that is representative for the average database load (and the system as a whole).

Not having system statistics could result in choices for incorrect (or low performance) execution plans (if there are no records in table `AUX_STATS$` under user `SYS` then there are no system statistics).

Standard Values:

It is therefore possible to immediately create direct system statistics if these are not there yet, even though no representative load is involved. A number of standard settings are used (which are better than no statistics at all). To this end it is possible to indicate that statistics have to be determined without there being a representative 'workload'.

Since DBMS 10g this is by default performed by instance startup.

It is possible to perform this manually by means of the following (SQL*Plus) command, under `SYS` (just like all other commands that follow afterwards):

```
exec dbms_stats.gather_system_stats('NOWORKLOAD')
```



Attention: For OHI Back Office environments the `NOWORKLOAD` option (method *Standard values*) is the recommended setting for collecting system statistics, until further notice. This relates to negative experiences with respect to the actual collecting of system statistics.

Actually Measured Values

Subsequently, at the moment a representative load is involved 'verzamelen' (collect) can be activated. Collecting can be activated by entering the following command e.g. at 10 a.m.:

```
exec dbms_stats.gather_system_stats('START')
```

When closing the collecting around 3 p.m., the command will run:

```
exec dbms_stats.gather_system_stats('STOP')
```

ATTENTION: In the meantime the instance should not be stopped. Otherwise the user has to restart collecting the statistics all over.



Attention: Until further notice, the method of actually measured values should not be used, unless this is requested explicitly by OHI Development or Oracle Support Services.

Status Definition

By means of the following SQL*Plus commands it is possible to request the status of the collection system statistics

```
col sname format a20
col pname format a15
col pval2 format a30

select * from aux_stats$;
```

If the statistics are defined without a representative work load, this results in:

| SNAME | PNAME | PVAL1 | PVAL2 |
|---------------|--------|-------|------------------|
| SYSSTATS_INFO | STATUS | | NOWORKLOAD |
| SYSSTATS_INFO | DSTART | | 10-26-2004 13:59 |
| SYSSTATS_INFO | DSTOP | | 10-26-2004 13:59 |
| SYSSTATS_INFO | FLAGS | 1 | |

If afterwards collecting is activated, this provides the following output:

| SNAME | PNAME | PVAL1 | PVAL2 |
|---------------|-----------|---------|------------------|
| SYSSTATS_INFO | STATUS | | MANUALGATHERING |
| SYSSTATS_INFO | DSTART | | 10-26-2004 14:01 |
| SYSSTATS_INFO | DSTOP | | 10-26-2004 14:01 |
| SYSSTATS_INFO | FLAGS | 1 | |
| SYSSTATS_TEMP | SBLKRDS | 125167 | |
| SYSSTATS_TEMP | SBLKRDTIM | 391710 | |
| SYSSTATS_TEMP | MBLKRDS | 97238 | |
| SYSSTATS_TEMP | MBLKRDTIM | 73860 | |
| SYSSTATS_TEMP | CPUCYCLES | 467124 | |
| SYSSTATS_TEMP | CPUTIM | 1434067 | |
| SYSSTATS_TEMP | JOB | 0 | |
| SYSSTATS_TEMP | MBRTOTAL | 759624 | |

In that case initial values have been entered for certain values that are defined. If afterwards collecting is stopped then this could result in the following output:

| SNAME | PNAME | PVAL1 | PVAL2 |
|---------------|----------|-------|------------------|
| SYSSTATS_INFO | STATUS | | COMPLETED |
| SYSSTATS_INFO | DSTART | | 10-26-2004 14:01 |
| SYSSTATS_INFO | DSTOP | | 10-26-2004 14:04 |
| SYSSTATS_INFO | FLAGS | 1 | |
| SYSSTATS_MAIN | SREADTIM | 8.381 | |
| SYSSTATS_MAIN | MREADTIM | .626 | |
| SYSSTATS_MAIN | CPUSPEED | 36 | |
| SYSSTATS_MAIN | MBRC | 4 | |
| SYSSTATS_MAIN | MAXTHR | -1 | |
| SYSSTATS_MAIN | SLAVETHR | -1 | |

What can be derived from this is that collecting statistics is based on a period of 3 minutes. In case the statistics collection is not successful the line with value STATUS for column PNAME could have value BADSTATS for column PVAL2, as represented below:

| SNAME | PNAME | PVAL1 | PVAL2 |
|---------------|----------|-------|------------------|
| SYSSTATS_INFO | STATUS | | BADSTATS |
| SYSSTATS_INFO | DSTART | | 10-25-2004 22:41 |
| SYSSTATS_INFO | DSTOP | | 10-25-2004 22:41 |
| SYSSTATS_INFO | FLAGS | 1 | |
| SYSSTATS_MAIN | SREADTIM | -1 | |
| SYSSTATS_MAIN | MREADTIM | -1 | |
| SYSSTATS_MAIN | CPUSPEED | 221 | |
| SYSSTATS_MAIN | MBRC | -1 | |
| SYSSTATS_MAIN | MAXTHR | -1 | |
| SYSSTATS_MAIN | SLAVETHR | -1 | |

In such cases, the collection of statistics has to be restarted.

Deleting System Statistics

For disabling/deleting system statistics

```
exec dbms_stats.delete_system_stats
```

is sufficient (in case this is desirable for whatever reason).

Committing is not required for the above actions.

System Statistics and Execution Plans

Collecting system statistics does *not* nullify the execution plans in the cache, as opposed to *editing the table statistics*. For existing SQL statements for which an execution plan was available, after the creation of system statistics no new execution plans will be defined again right away. These have to be deleted from the 'cache' first (for testing purposes it is possible to add an extra space to a statement to stress that a plan is defined again for that statement after the system statistics have been changed).

Table Statistics

In addition to having system statistics available, it is more important to have current table (and index) statistics collected. The Cost Based Optimizer uses the statistics of tables and indexes (e.g. how many records are included in a table, what is the division of values in a column, how many records are there in an index) to identify the execution plan that best meets the optimizer requirements. To this end, these statistics, in case they are estimates, have to lie near the actual content of the table (and index). The statistics could be 'estimated' based on a part of the content of a table or 'calculated' when going through the table content entirely.

In order to establish representative statistics, they have to be updated on a regular basis if the table content changes. Tables that undergo changes more and more often when considering their total content have to be modified more regularly in terms of statistics than tables of which the content remains rather static. When a new OHI release is installed, for the adapted tables even the definition of statistics has to be performed again.

In order to abstract from this issue, a 'table monitoring' functionality has to be used in the database. Oracle keeps track of how many changes (inserts, updates, deletes) are performed on the data in a table. This data is used in certain utilities to determine statistics 'automatically' for the tables that require this. 'Temporary' tables (tables that are not permanent but that can store data only during a session) are dealt with in a different way.

Considering that there are various complicating factors for adequately collecting table statistics, script `OZGISTAS.sql` is included in the reference software. This script is

called during the installation of a new release. The script is modified if changed understandings encourage doing so.

When determining the statistics, Oracle decides about which part of the table (the number of records of the total number of records) the statistics are defined. Whether this takes place based on an estimate or on a small percentage of the data of the table or based on an actual calculation of 100% of the data, the choice is up to the standard software.

The script implements required OHI settings (such as preventing and/or dropping column histograms, locking the statistics for certain tables, etc.) and uses spare runtime to collect statistics.

The script can only be run by the user who requires his objects to be analyzed. It contains two input parameters. Both of them are mandatory.

Parameter 1 involves the number of hours during which the script can run; providing a value is mandatory.

Parameter 2 involves the environment in which the script runs. Possible values include `OBD` (=OHI BUSINESS INTELLIGENCE) and `OZG` (=OHI Back Office).

When called by the installation menu (`OZGPATCH`) these are passed automatically.

Data Dictionary Table Statistics

In a 10g database the creation of dictionary statistics is *mandatory*; to this end `dbms_stats.gather_dictionary_stats` procedure can be used under `SYS` account.

For a database with only OHI Back Office objects this takes about fifteen minutes. Should the data dictionary have experienced a lot of changes since the previous time the statistics were defined, then of course it is advisable to determine the statistics again in the aforementioned way.

Fixed Objects Statistics

Since the 10g database the creation of statistics is mandatory for so-called “fixed objects”; this allows to use `dbms_stats.gather_fixed_objects_stats` procedure under `SYS` account.

At the moment, conditions for collecting these statistics include:

1. The SGA size and the `init.ora` database parameters have to be stable.
2. The system has to have a representative load.

Dividing I/O - if Possible

In the past it was common to divide the I/O over physical disks by dividing the often used table spaces and data files as well as across the available disks.

At present logical volumes and certain RAID levels are used mostly, for which the division of the I/O is assigned to the controller(s) (of which all sorts of variants are possible). In general this should guarantee a suitable I/O division, certainly considering the availability of fast read and write caches with intelligent algorithms to activate the content.

One of the few options that remain is the division across multiple controllers if these are available. For an I/O bottleneck problem this could offer a solution.

If there is a possibility for that, it is in any case desirable to locate the redo log files on a separate and fast I/O device. It would be even better to put each redo log group on a separate device or in any case successive redo log groups (e.g. even groups on disk 1 and odd groups on disk 2).

The archived redo logs (if these are used) have to be written to a separate device as well, so that the other I/O do not form an obstruction.

This two last recommendations are not strictly necessary, but first of all they have to be considered if the transaction speed slows down and if statistics show that the redo log actions tend to reduce the speed. After all, a database on which read actions take place does not easily experience problems with slow write actions with the redo logs.

Resource Management – Sessions Mutual Prioritizing

A tricky phenomenon with regard to the performance of a database refers to the performance of long-term heavy (in terms of CPU use) database processes (long-term in the sense of a few seconds up to possibly many hours). If more of this type of processes are active, then processors have to be available that can notably influence this performance of short-term (online!) database actions.

Within OHI Back Office such long-term processes are, generally speaking, batch processes for which the Oracle session consumes a lot of CPU during longer periods of time. Such processes can sometimes occupy a processor for about 100%.

In such cases it is desirable to prioritize the short-term database actions that usually take place in screens over the long-term database actions. It is, however, not recommended (or better, the opposite) to do this on the level of the operating system: if an Oracle session has certain latches (short-term locks to protect memory structures) and the process would be given lower priority and therefore 'not attended to' for longer periods of time, it is possible that other database processes will be put on hold for the relevant latches.

For this and other reasons, Oracle has so-called resource management options at its disposal to prioritize the use of resources such as CPU. The documentation contains a lot of descriptions about this, including options to 'check' resources other than CPU. For now the recommendation to use this for prioritizing short-term database sessions over long-term database sessions so that on-line users have priority over batch processes is sufficient.

A simple application has been provided for a number of years in script `OZGI002S.sql`. This script makes use of the standard resource plan `SYSTEM_PLAN` in which there is a division between three groups of users (Resource Consumer Groups):

- Sessions of SYS/SYSTEM (`SYS_GROUP`)
- Low priority sessions (`LOW_GROUP`)
- Other users (`OTHER_GROUPS`)

Resource plan `SYSTEM_PLAN` has been configured in such a way that when users wish to use the CPU, they get priority depending on the group they belong to: `SYS_GROUP` has priority to `OTHER_GROUPS` and this has priority to `LOW_GROUP`.

By default user `SYS` and `SYSTEM` are located in group `SYS_GROUP` and all other users in `OTHER_GROUPS`. By locating the batch user now explicitly in `LOW_GROUP` by means of `OZGI002S.sql` online processes have priority over batch processes by default.

There has been a lot of work on the possibility to set up a finer division of resources in which also short-term online actions get priority over long-term online actions (such as an expensive query or online processing of a declaration). These long-term online actions have priority over short-term batches while long-term batches have the lowest priority. By having a session switch dynamically from one to another group, this function can be realized. Given that this mechanism is still insufficiently reliable and as further test experiences are required, only the simplified form of prioritizing is recommended.

ATTENTION: An important limitation of the resource management is that prioritizing database processes takes place within a single database. If another database runs on the same server then it is not possible to prioritize across multiple databases because of the resource management mechanism. This restriction has to be taken in consideration. Prioritizing on operating system level is no solution either. In that sense it is important that a production database runs as a bulk consumer alone on the server to benefit from resource management.

9.2.3. Collecting Performance Data

This section discusses the resources which can be activated to collect performance data.

Operating System Utilities

Simple use of some general operating system utilities (Unix based) will be discussed in some sections. These can be used to acquire insight into the load of the server(s). There are many more options, sometimes these are better and more advanced, but these tools allow for an initial impression of where the bottleneck is located in most cases.

vmstat

By means of e.g. `'vmstat 5'` it is possible to acquire insight into memory-related statistics every 5 seconds, amongst which page swapping (a value of 300 seconds, or 5 minutes, could also be useful to assess the average load in a screen during a longer period; an hour consists of 12 lines).

Whether or not page swapping takes place can be derived from the value in front of the column 'sr' (scan rate) in the group page. If this has a constant value in hundreds or higher, swapping is clearly involved (the value has to be compared with the number of available pages). Additionally, for the CPU load the percentage of idle time is represented. If idle is nearly continuously close to 0, then the CPU is heavily loaded.

With regard to group 'procs', the number under r(un queue), b(locked) and (can be run but s)w(apped) is very interesting for an initial impression. Values not equal to zero indicate that processes are not serviced right away.

For more information, see the manual pages.

iostat

By means of for example `'iostat 5'` the default output is given. In order to zoom in on I/O problems, specific options have to be used, but the four columns under CPU provide insight as to whether or not waiting for I/O is required. Column 'w' under CPU provides the percentage of the time required for I/O,

By means of `'iostat -xc 5 10'` it is possible to have overviews provided of the average service time 10 times during 5 seconds, the busy percentage etc. of all devices including how the CPU time is used. A busy device could possibly be relieved by distributing the database files better across the devices (if the database files are an explanation for the load of the devices).

sar

By means of the `sar` utility it is possible to register statistical data with a certain interval in a simple way. The data can be converted into readable reports by means of the same utility.

In order to optimize the use of this a set up is required with an interval of e.g. 10 minutes of 'recording' (24 *7). This allows reasonably well for zooming in on longer

peak periods and in this way data involving half an hour can be compared with statistical data on a database level (this will be discussed at a later stage).

By means of the following 2 lines in the `crontab` for `sar` (user `sys` on a Sun Solaris system) every 10 minutes recording can be made that are converted into a readable report at the end of the day (11:55 p.m.)

```
0,10,20,30,40,50 * * * * /usr/lib/sa/sa1
```

```
55 23 * * * /usr/lib/sa/sa2 -s 0:00 -e 23:50 -i 600 -A
```

CPU load for previous period (all ‘snapshots’ of the day):

```
sar
```

Reading memory use for previous period:

```
sar -r
```

Reading swapping of previous period:

```
sar -w
```

Reading the actual load five times every 10 seconds:

```
sar 10 5
```

Of course there are advanced tools to implement monitoring of the system. However, because of a comparison with other customers and installations and simply rendering available data to Oracle in case of performance problems, it is desirable to always have `sar` monitoring activated. The overhead is very limited and the advantage is that this allows for tracing the cause of performance problems faster because sometimes certain causes can be excluded.

top/topas/prstat

By means of a tool such as `prstat`, `top` or `topas` (or other system utilities) it is possible to retrieve which processes are used by most CPUs rather fast. Especially if certain processes are slow or if the CPU use on the server level is high, then it is possible to zoom in on the processes that require most CPU.

In case of frequent performance problems, it could also be desirable for Oracle to use such a utility. Access to one of these utilities (available via the ‘remote connect’ account) could speed up the process.

swap

By means of the `swap` command, certain data can be retrieved involving the swap memory. This could be desirable in case swapping seems to be involved.

Listing out the swap memory:

```
swap -l
```

Summary of the swap use:

```
swap -s
```

OS Watcher

Oracle’s OS Watcher (OSW) is a collection of UNIX shell scripts intended to collect and archive operating system and network metrics to aid support in diagnosing performance issues.

Information on how to setup and use OS Watcher can be found in the following My Oracle Support notes:



My Oracle Support note 301137.1, OS Watcher users guide



My Oracle Support note 461053.1, OS Watcher graph users guide

Database Monitoring

Oracle Enterprise Manager

For monitoring the database for recent activities there are various options. Particularly, by using the Oracle Enterprise Manager (OEM) in combination with an OEM repository and configured agent processes, a lot of useful information can be retrieved.

It is highly recommended in that case to use OEM and to have its (web) user interface available for OHI Development to inventory the performance problems rather fast.

Automatic Work load Repository

Therefore we choose to collect data periodically in a relatively easy way, so that comparisons can be made between data of various customers. In case of more general performance problems, this data allows for the possibility to obtain a clear view as to the bottlenecks in the database (presuming that there are any). Therefore we use AWR, the Automatic Workload Repository which is installed by default and configured since DBMS 10g. AWR creates periodic 'snapshots' of the database statistics automatically and stores them in the SYSAUX table space. Reports can be released on these snapshots in tables.

By default one snapshot is created every hour. Every night a job runs which automatically removes snapshots taken more than 7 days ago (the default retention setting).

OHI advises to create snapshots each 20 minutes and use a retention period of at least 35 days. This consumes more SYSAUX space but can be very helpful to investigate problems (and look for example whether they also occurred the previous month in the same way).

RAC

In a 10g R2 RAC environment, each AWR snapshot gathers information regarding all instances in the cluster. This data will not be aggregated.

AWR – Reporting Statistics

When statistics are collected, they can be used to run reports about certain periods. These periods are characterized by a 'start' snapshot and an 'end' snapshot (identified by numbers; a list of snapshot periods with identifying numbers is displayed by each AWR utility that requires the snapshot numbers).

By means of report script `awrrpt.sql` it is possible to create a general report about a certain period. If a problem is encountered during a certain period, then it is useful to limit the period to be reported to a time span that keeps the problematic period to a minimum. If a reduced speed experienced from 9:45 a.m. to 11:55 a.m. then it would be recommended to run the report for the period 9 a.m. to 12 p.m.

If a name for the file requested in which the report is 'spooled' then it would be useful to apply the standard proposed naming.

All sorts of statistic information is displayed in the relevant report. Additionally, various lists are provided of the heaviest SQL and PL/SQL statements.

For the statements for which there are uncertainties regarding the legitimacy of their size, it is possible to create a more detailed report by means of a separate script `awrsqrpt.sql`. In this case the identifying numeric hash value (SQL ID) has to be provided which is reported for each statement if the SQL statement report is performed correctly with `awrrpt.sql`. Identifying 'wrongful heavy' statements will ordinarily be performed by the development department with OHI Back Office (the development team) after an AWR report is delivered.

However, it is not necessary to examine problem statements that may be present in the OHI Back Office application as customer of OHI Back Office. The OHI Back Office

development team should perform this task. But by means of the AWR reports and any specific reports pertaining to certain statements, the development team can focus on pertinent solutions for any problems.

ADDM Report

When AWR reports are delivered to the development department for OHI Back Office, then for the relevant snapshot range also the ADDM (Automatic Database Diagnostics Monitor) report has to be delivered, which provides advice regarding the AWR collected data; to this end, script `addmrpt.sql` has to be run.

This script performs an ADDM analysis for the relevant range and creates a report about it.

In order to display the report of the *last* ADDM analysis, script `get_latest_addm.sql` can be used; this script does *not* first perform an analysis.

RAC

In a 10g R2 RAC environment, an ADDM report node is specific; for the different nodes therefore different ADDM reports have to be collected.

Tracing Sessions

Whereas AWR is used to collect all of the database data, in certain cases it may be desirable to zoom in on a specific process or a certain database session. Certainly when a specific session causes problems and no general problems are detected.

For such a session, creating so-called 'trace files' is recommended. Even though this is generally a known process, some indications are provided as to how to deal with tracing.

Setting a Session to Trace Mode

Activating tracing in a session is relatively easy by entering the following command:

```
alter session set sql_trace = true;
```

However, sometimes it may be required to set an ongoing session to trace mode. This is possible by entering command:

```
execute dbms_system.set_sql_trace_in_session(&sid,&serialnr, true)
```

This has to be performed with an account that has relevant rights.

While doing so, it is necessary to determine the values for '`sid`' and '`serial#`'. These can be retrieved from view `V$SESSION` and by means of column `USERNAME` it is usually possible to create a selection for the user sessions that are slow.

However, users who have opened multiple screens from the menu in OHI Back Office will have multiple sessions in the database. When a slow action in a screen has to be traced, it is possible to do as follows:

- Start the screen from the menu (or the calling screen once it appears in the start-up screen).
- As a first action, before performing the slow action, determine the 'sessionID': Start the Information window by means of the Help menu and the Info menu option. Determine the value of 'sessionID'. This value identifies a session by means of column `AUDSID` in `V$SESSION`. As long as the 'problem screen' does not close, this value will remain constant and by means of selecting `SID` and `SERIAL` on `V$SESSION` the `SERIAL#` values can be determined:

```
select sid
,      serial#
,      (select spid
        from v$process p
        where p.addr = s.paddr
       )
      oracle_process
from v$session s
where audsid = &sessionid
```

- Use the `sid` and `serial#` values in order to perform the above command and to set the session to trace mode.
- Perform the slow action.
- Check the trace directory to see if a trace file has been created with the name `oracle_process` number as provided by the above query.
- Close the screen (the trace file will be closed which would render the information complete and therefore easier to use).

In case a screen is already blocked by a long-term action and if it is still desirable to trace the session, try to identify it in the following way:

- Determine which oracle process uses a lot of CPU in the relevant database (by means of a peak-like tool). If there is no such process then the problem is supposedly not located in the database but in the forms. It is possible that the forms process consumes a lot of CPU. The database session does not have to be traced in that case.
- By means of utility script `form.sql` as available on iProjects, it is possible to determine which sessions the relevant user has in the database. The oracle process that consumes a lot of CPU should have the ID available in the list of process numbers in the 'SQL*Net' column as provided in this script.
- Use all `sid` and `serial#` combinations that match the 'heavy' oracle process in order to set the relevant sessions to trace mode.
- Check if a trace file appears (see above). If this is not the case, then it is possible that the session requires a lot of time with a single SQL statement. Only after the statement provides results or if the session starts with other statements then a trace file will appear (for heavy statements in batches this may be the cause if no trace file appears, for screens this is the case less frequently).

For running batch processes, searching the sessions based on a username is not a solution if multiple batches run at the same time. The batches all run under the same Oracle user, often referred to as `BATCH`. Also the `batchscheduler` runs under this Oracle user and has to be recognized. This is possible because the relevant session in column `MODULE` of `V$SESSION` has a value that starts with `SYSS004S`. This session should therefore not be traced.

If multiple sessions are active in the user batch, it is possible to identify the correct session by selecting the ID process the session has under Unix. If everything goes right, the session will return as a heavy Oracle process in a tool that displays the top sessions (see above). The process ID that matches the Oracle process can then be used to select the correct values from the `V$SESSION` and `V$PROCESS` view by means of the following statement:


```
select sid
,      serial#
,      process
,      username
from    v$session s
where   s.paddr = (select addr
                   from    v$process
                   where    spid = &oracle_proces_id
                   )
```

Should several heavy Oracle processes run, request the start time of the heavy processes by means of ps command (`ps -eaf | grep <procesid>`) and compare this with the start time of the script requests as displayed in the screen. This will have to be about the same as the start time of the running script request. In this way it can often be determined which Oracle process matches which script request.

Trace File Points of Attention

Some points of attention

- If a session is set to trace mode then normally a trace file will be created in the location provided by means of the initialization parameter `DIAGNOSTIC_DEST` (within a subdirectory 'trace' deep within the directory structure designated by this location setting).
- The names of the trace files normally have the form `<$ORACLE_SID>_ora_<process_id>.trc`. In this case the `process_id` can be determined by means of column `SPID` from view `V$PROCESS` (see previous section). For MTS sessions the process id of the 'server' process will be used (a so-called `Sxxx` process, `S001` for server process 1, etc.). Such a server process will also write statements of other sessions to the trace file. An MTS configuration is not as usable for tracing sessions.
- By means of the initialization parameter `MAX_DUMP_FILE_SIZE` a limit can be set for the size of trace files (in system block processes or in KB or MB). If this limit is set rather low (some MBs e.g.), then it is possible that trace information can be registered only for the first few minutes of an action, because afterwards the maximum volume is reached for the trace file. In order to trace a slow action in a screen this is generally not a problem, for tracing a slow batch sometimes hundreds of MBs of data per hour could be written already. Therefore it is of importance that these parameters are set to e.g. unlimited or to a high value when tracing long-term batch processes (and of course sufficient space has to be available on the volume on which the trace file will be written; if necessary use a symbolic link for the relevant directory for writing the trace files in a different location which has sufficient space available).
- If a trace file is filled up then it is no longer possible to get the writing of the trace information started by setting the parameter value higher dynamically. Only once a new session is started trace information will be written again.
- However, tracing may also be activated while no trace information is written (see the previous remark about tracing a screen which has been blocked for a longer period). Often the cause is that the session is occupied for a longer period with one and the same SQL statement and does not retrieve records (yet in case of a select statement). No trace information will be written in that case. This could be the case for batches if an SQL

statement is included which would have a result only after a couple of hours. Whether or not this is concerned can usually be verified by assessing if the session has been occupied with the same SQL statement for a long time in Oracle Enterprise Manager.

- If possible, it is desirable to have the trace file as complete as possible. Particularly for the slow screen actions it is possible to close the screen after 'reproducing' a slow action after which even the trace file is closed. This will also contain certain counting information that could be useful for assessing the trace file. Also for the batch it is recommended to have a trace file about the entire process that is as complete as possible, but usually this is not possible due to the size. Or it is not possible because the batch has been running for too long and does not end.

Formatting a Trace File

When a session is set to trace mode and the trace file containing the trace information has been identified, then it is desirable to format it in a format what is easily readable and useful. It is possible to already format a trace file while it is still being created. If possible, it would be better to wait until it is entirely ready.

Formatting has to take place via utility `'tkprof'`. The version that is located in the database bin directory of `ORACLE_HOME` has to be used. Switch therefore in advance the environment settings to the correct `ORACLE_HOME`.

In order to get the heaviest statements of the trace file on top in the formatted output, it is of importance to apply good sorting techniques. This optimizes the output because in that case it is not necessary to look for the slowest statements very long and therefore slow statements are missed less easily.

Useful sorting can be done on the sum of the elapsed times of parse, executing and fetching. In that way most long-term statements will for sure end up on top in the output. For this, parameter value `'sort=prsela,exeela,fchela'` is very useful. An example of call:

```
tkprof ontw_ora_7222.trc trace.txt explain=ozg_owner/ozg_owner  
sort=prsela,exeela,fchela
```

Some points of attention:

- Run `tkprof` under the account that has relevant rights for the objects used, for example the owner account of OHI Back Office. Or use an account that has `SELECT ANY TABLE` rights.
- Given that `tkprof` determines execution plans for the statements in the trace file at the moment on which `tkprof` is running, then it is important that nothing in the database is changed between the creation of the trace file and running `tkprof` (for example, the table statistics do not have to be updated in the meantime).

Statements – Determining Execution Plan

In case certain SQL statements are slow then it is desirable to examine the execution plan by consulting the explain plan output in which the execution plan is represented. By means of `tkprof` the execution plan is represented for the statements by default in the trace file. But it could also be desirable to assess a change in a statement in terms of execution plan.

In order to determine an execution plan it is important to have access to the right `PLAN_TABLE`. For this, the global temporary `PLAN_TABLE` has to be used. To facilitate

this it is *mandatory* to remove any available `PLAN_TABLE` tables under all schemes. The following action list can be used to obtain this:

- Select the owners of the existing `PLAN_TABLE`s:

```
select owner from dba_tables where table_name='PLAN_TABLE'
```

- Remove the `PLAN_TABLE`s for the various owners.
- Remove any public and private synonyms for `PLAN_TABLE` if they refer to a `PLAN_TABLE`.

Assessing the execution plans of slow statements is something that has to be performed by the development team of the OHI Back Office application and will not be discussed further in this document.

Check Index Use

It could be desirable to check if certain indexes are used. For this, it is required to activate the relevant monitoring for an index:

```
alter index &index_name monitoring usage
```

Via `view v$object_usage` it is possible to trace if an index is used.

By default this is not required and the development team of OHI Back Office will be asked to activate this if required for studying a certain performance problem.

Use of SQL Access Advisor

After release 2007.02 script `OZGTUNES.sql` (available in `$OZG_BASE/sql`) is available, a report can be created for a (problematic) SQL statement. This can be run if it turns out that e.g. based on an AWR report an SQL statement leads to many problems.

The script starts the *SQL Access Advisor* and hereby generates information that contains advice regarding the steps to be taken in order to optimize the SQL statement performance. The output can be sent along with the rest of the mandatory information for incident notifications of performance problems. The OHI Development can then probably solve the problem faster.

Parameters OZGTUNES.sql

1. The ID of the SQL statement about which information/advice has to be printed.
2. Optional: the logical name of the directory (= directory object in the database). If nothing is indicated then `OZG_TMP` is used.
3. Optional: product with permissible values `OZG` (=OHI Back Office) and `OBD` (OHI Business Intelligence). Parameter influences advice: either OLTP or datawarehouse. If nothing is indicated then `OZG` is used.

Examples of Possible Calls

```
@OZGTUNES ax0ufbmack2cg " "
```

This command writes file `ax0ufbmack2cg.out` to `OZG_TMP` with product `OZG` (OLTP).

```
@ OZGTUNES ax0ufbmack2cg " OBD
```

Writes file `ax0ufbmack2cg.out` to `OZG_TMP` with product `OBD` (datawarehouse).

```
@ OZGTUNES ax0ufbmack2cg OZG_DIR "
```

Writes file `ax0ufbmack2cg.out` to `OZG_DIR` with product `OZG` (OLTP).

@ OZGTUNES ax0ufbmack2cg OZG_DIR OBD

Writes file ax0ufbmack2cg.out to OZG_DIR with product OBD (datawarehouse).

SQL Profiles

Using the *SQL Tuning Advisor* it is possible to store SQL Profiles for specific statements. It is allowed to store SQL Profiles.

9.2.4. Assess Settings during 'Regular' Performance

This component will be developed in greater detail at a later stage.

Database

Components to be checked:

- Hit rate of the buffer cache. At least 90%, but preferably more than 95%.
- The same for shared pool.
- Checking the PGA settings

9.2.5. Preventive Application Management Actions

This component will be developed in greater detail at a later stage.

Issues to be considered:

- Max number of batches
- Max number of processes for splitters
- Using splitters

9.2.6. Examining Performance Problems

In case there actually are complaints about performance it is recommended to zoom in on the problem before drawing conclusions as to a possible cause. In this section a description can be found about a possible approach that can be applied.

Finding out the Scope

When performance problems occur it is important to first find out the extent of the problems at the moment they are encountered (the problem perhaps only occurs during certain periods, also these periods are of importance):

- Does everyone experience the problems on all locations regardless of the operation?
- Is it possible for the problem to be encountered in various workstations but does it only occur when logging in on the system with a certain name?
- Is the slowness related to the location (only certain workstations or PCs)?
- Does the slowness occur only in certain sections of the application?
- Can the slowness be detected only if a certain action is performed on the data?

- At the moment the problem occurs, is it just OHI Back Office that has a slow performance on the PC or are all of the actions in the relevant workstation(s) experiencing this problem?

If the problem is time-driven it is of importance to note down the exact time (up to the minute with a note specifying where the time is read; the clock of the system could differ by a couple of minutes).

With the answers to these questions the scope can be determined. This would probably lead to one of the following delineations:

- A PC-specific problem is involved.

In these cases the problem can be caused by the hardware, the network connection, the operating system, certain applications that run on the PC, alternative settings on the PC or, more tricky, a combination of these problems.

It is typical that the problem is encountered on one PC regardless of the user that logs in or the application and that the same user experiences no problem on other PCs.

- The problem is related to the network.

Only certain workstations in a specific section of the building or a specific building experience this problem.

Typically, the problem is not encountered when the same user performs the same operations on workstations in a different section of the building.

- The problem is related to the server (database and/or application server).

All applications on the server(s) experience the problem. In that case, via monitoring utilities it is possible to check that the server is overloaded at a certain point (too much swapping/paging is involved; the server is occupying nearly 100% CPU all the time or is waiting for I/O regularly; or a combination of the above).

Typically, all users experience problems with the slowness in the applications that run on the relevant server and all server applications are rather slow.

- It is either a database or a general application problem.

All actions on the relevant database are performed relatively slowly (generally speaking this will result in a server problem but in that case particularly the database actions will cause problems).

Typically, one particular application runs very slowly instead of other applications that run on the same server (if available).

On the outside it is often hard to make a distinction between the server and the application-specific database problems.

- The problem is caused by a problem in the application.

The problem is only encountered during certain actions. Depending on the gravity of the problem, only the relevant action experiences problems in performance or other components of the application are burdened by it as

well because the database and/or server is overloaded.

Typically, some of the actions in the application are performed at a normal speed, but certain actions are much slower than before (if there is a frame of reference).

The problem is caused by wrong application use.

Typically, new users are involved or users who perform certain actions for the first time. Or a function is involved that is used for the first time since its introduction.

Chances for incorrect use of certain functions or wrong expectation are the highest.

The problem is caused by a certain bottleneck in the infrastructure of the application.

Typically, for these problems no clear general overloading can be observed. A possible cause could be that many processes change the same file (outside of the database). An example, which actually did occur, involved many user sessions that wrote many messages at the same time in the Apache log files.

Zooming in on Problems

In order to figure out the cause of the performance problems experienced, the following suggestions are made:

- Checking the workstation: What is the CPU load of the PC like (e.g. use a 'task manager' to monitor the CPU load; if it is near 100% and this is not caused by the application process then usually this could be a reason).
- Does the relevant action (execute it as similarly as possible) run comparably slowly in a different workstation? If not, does it make a difference if the same account is used in a different workstation?
- Is a peak or an overload encountered on the server?
 - Is there hardly or no idle CPU time?
 - Is 'waiting for I/O' involved?
 - Is there a high paging or swapping rate and do all active processes insufficiently fit in the work memory of the server?

In case of one or more of these phenomena, the cause has to be ascertained. In case of a heavy CPU load the processes that cause the problems have to be defined as well as the application/database they belong to.

In case of a high I/O rate, the processes that cause the problem have to be checked (as well as the application they belong to).

If the memory use is too high, then what has to be checked is if there are certain processes that require large amount of memory or if the number of processes is higher than normally. The cause for that problem will then have to be ascertained.

- If no peak load is experienced, is there a clearly heavy process available at the moment the performance problems are experienced? Apparently the relevant operation is then very heavy and requires to be zoomed into.
- In case certain processes are heavy (a lot of I/O, CPU or memory usage) the kind of process has to be ascertained: For OHI Back Office only a few types of processes can be recognized:
 - an Oracle client process that handles the SQL and PL/SQL work;
 - a forms server process that runs on the server and that matches the applet

screen processes that run on the workstations;

- a batch 'client' process (reports, SQL*Plus, SQL*Loader, import/export or shell process).

In almost all cases the Oracle client process, which matches a certain database, will be the culprit. At times a web forms server process may crash or spin (users should never have such processes run for more than few CPU percentages; if this happens, it has to be killed).

If the Oracle client process is identified then by means of the process ID it is possible to determine which database session matches it, after which it is possible to examine the relevant database session further.

- If a certain process is identified as the culprit it can be examined further. For Oracle client processes additional tools are available without problems, for other processes even the source will have to be examined. Oracle client processes can be set to trace mode so that the produced trace file can be assessed afterwards by formatting it with `tkprof`. Additionally, it is possible to zoom in to the SQL statements that perform the process and to have a look at the statistics in Oracle Enterprise Manager or by means of a different tool.

In the future zooming in on problems will be focused on what appear to be encountered in the database. If research shows that the problem is to be found elsewhere, then this section will probably not be of any further use.

Before continuing, the changes made since the problem has occurred will also have to be discussed further:

- Has the frequency increased?
- Is there a new version of a component of the total 'stack' technology (hardware and software) in use?
- Have multiple changes been implemented (this makes it more tricky to have a clear view as to the cause, which would call for making as many changes as possible successively rather than simultaneously).
- Is it for certain that the problem has not occurred before?
- Is the problem related to time periods? In other words, is the problem encountered during the beginning or the end of the week, the month or the year?

The more information available, the more plausible certain causes may be or the more easily others can be excluded.

Search the Oracle Database for a General Problem

At a certain moment a general database problem can be questionable. In that case it has to be clear that no other causes are involved:

- No additional users are involved which would stretch the limits of the machine.
- There is sufficient physical memory available or, not too much swapping is taking place.
- There are no processes that are more prominent in terms of consumption.

If this impression of having problems with the entire database performance persists, then the general database matters should be focused on. Particularly, the AWR output which reports on the slow period could be useful.

Now examine the following:

- Is the problem an I/O problem:
 - do the service times in AWR no longer correspond with the 'good' periods?
 - has the buffer hit percentage clearly decreased?
- Are there any unusual events in the top 5 of the wait events with a substantially higher waiting period than normally?

In case e.g. a latch problem is involved, then this is to be zoomed into. There could be numerous other problems like this. An initial approach is to search on My Oracle Support for similar problems.

Generally, general database problems - not caused by some specific processes - do not tend to cause application problems that easily. A parameter setting value that is too low would more likely be a cause to a problem.

Search Oracle Sessions for Problems

If it is clear that certain sessions are perceived as top processes, they require to be examined. At a later stage, more information will be provided about a possible approach, using the previously described possibilities.

9.2.7. Notifying Performance Problems

If performance problems occur which are caused by the OHI Back Office function, then when notifying a performance incident the following information is mandatory:

1. ADDM report
2. AWR report in HTML-format
3. Output of Object check (see *Installation manual, Oracle Health Insurance (patch) releases* on iProjects; start the Object check via the *application menu* so that the *runtime* checks are performed!)
4. In case of ORA-00600/07445 error messages; the number of the Service Request logged with Oracle Support Services, and the corresponding alert- & tracefiles

Based on this information it is possible to ask for more information with respect to specific SQL statements; subsequently, in that case a delivery of the following will be requested:

1. AWR SQL report in HTML-format
2. Output of OZGTUNES.sql

For more information, see [Collecting Performance Data](#)

9.3. PERFORMANCE APPLICATION SERVER

This section briefly discusses the performance of Oracle Forms and Reports

9.3.1. Load Balancing Oracle Forms

In order to implement Load balancing for Oracle Forms (the screen interface component of OHI Back Office) the following My Oracle Support notes can be used:



294749.1 : Troubleshooting WebForms Tuning / Performance /Time out Problems

9.3.2. Tuning Oracle Reports

In order to set the use of Oracle Reports (one of the batch components of OHI Back Office) the following My Oracle Support Notes can be used (although they refer to previous releases they can still help):



282170.1 : How and Which Options to Set in 'jvmOptions' for a Reports Server / Engine



282406.1 : Engine Parameters, their significance and recommendations in 9i/10g Architecture



258225.1 : Oracle Reports 9i Server Does Not Appear to Equally Load Requests Across Engines ?



296533.1 : Reports Server is using one engine only, even when maxengine is set to a higher value

Troubleshooting Reports Server

Please read the paragraph 'Tracing Report Execution' (24.3.7 in 'Publishing Reports to the Web with Oracle Reports Services' which is incorporated in the FMW Documentation Library) to enable logging. The logging information might help in finding out what your problem is.

9.4. ANTICIPATE PREVENTIVE RESOURCE MESSAGES

If a certain type of resource is insufficiently available (memory, free disk space, free table space, reserved memory space for extra database sessions, etc.) error messages may be encountered in the application.

The objective of this section is to discuss a number of matters as to how preventive action can be undertaken to prevent certain resource-related messages as much as possible.

In a subsequent phase, this section will be worked out further based on practical experiences. The following indicates the type of matters involved.

9.4.1. Prevent Database Management Messages

By use of active management certain database messages can be prevented. Therefore attention will be paid to the following subjects:

- Active space management for data and index storage
- Auto space management
- Redo and temporary space management
- Monitor the file system space (particularly swap space)

- Processes parameter
- Undo parameter settings
- Session cached cursors

9.4.2. Other Points of Special Interest

Additionally, attention will be paid to the following remaining related matters:

- Number of parallel processes
- Monitoring batch jobs
- Advice to application managers with regard to batches
- Error messages in batchscheduler log
- Compiling sources as a preventive measure
- Locks

10. UNINSTALLING OHI BACK OFFICE

The uninstallation of OHI Back Office consists of the following activities:

- Remove the OHI Back Office database(s);
- Uninstalling the Oracle Database and Application Server software on the database, application and desktop layers (if this Oracle system software is not used by other software any longer);
- Deleting directory `$OZG_ROOT`;
- Deleting the OS account `batch`;
- Deleting the OS account `oracle` (if this is no longer used by other software);
- Cleaning the file system.

APPENDIX A – CONFIGURATION OF MULTIPLE BATCH SCHEDULERS

For the same database multiple batchschedulers can be launched. However, for the database provided no more than one batchscheduler can be active per server. If for example four servers are available, then a maximum of 4 batchscheduler processes can be started for the same database.

The work between the different batchschedulers is divided based on a *dynamic load balancing mechanism*. This entails that if a batchscheduler process is added or deleted, the load will afterwards be divided equally across the active batchschedulers.

In case of multiple batchschedulers it is important that the output and log information is written to a central location, so that when consulting the log/output from the OHI Back Office application it does not matter on which server the batch process has run. For this purpose a SAN can be used. An alternative is the use of NFS, for which one of the servers renders a directory available for creating log and output files.

USE OF NFS PARTITION

This section discusses how batchscheduler processes can store their log and output files in the same directory structure on different servers. In this process one server renders a directory available, which is the NFS *server*. Other servers may 'share' this directory they are the NFS *clients*.

Activities of NFS Server

Enter the required command as `root` to render available the desired directory as NFS share.

Share Command under Sun Solaris

Under Sun Solaris `share` is used to have a directory used as NFS partition. The easiest form of this command is:

```
share <directory>
```

If directory `$OZG_BASE` is rendered available, then the Solaris command is:

```
share $OZG_BASE
```

Share Command under IBM AIX

Under IBM AIX `exportfs` is used to render available a directory as NFS partition. The easiest form of this command is:

```
exportfs <directory>
```

If directory `$OZG_BASE` is rendered available, then the AIX command is as follows:

```
exportfs $OZG_BASE
```

Security

The above commands provide read-write access to all hosts and for all users. The manuals of the Operating System used by you contain the information required to improve the security.

Activities of NFS Client

On servers that function as NFS client, the NFS share has to be uploaded.

The first step entails defining the mount point:

```
mkdir -p <mountpoint>
```

Afterwards the remote NFS partition has to be uploaded

```
mount -F nfs <host>:<directory> <mountpoint>
```

If for example directory `/u01/app/oracle/product/OHI/prod` has to be uploaded on server `myhost` as `/u01/app/oracle/product/OHI/prod` then the command will look as follows:

```
myhost:/u01/app/oracle/product/OHI/prod \
mount -F nfs \
/u01/app/oracle/product/OHI/prod
```

STARTING / STOPPING THE BATCH SCHEDULER

Starting the Batch Scheduler

For each server only one batchscheduler can be active for a given database. If the batchscheduler process is activated, the batchscheduler reserves a user lock for the given server. If a batchscheduler process is active on the given server, the user lock is already occupied and the batchscheduler will stop.

If successful, the batchscheduler registers as an active process. This means that in table `ALG_BATCHSCHEDULERS` flag `IND_ACTIEF` of the row matching the server is set to J.

In the following example the batchschedulers are active on servers `myhost1` and `myhost2`. Query

```
select server
,      ind_actief
,      pid
from   alg_batchschedulers
/
```

provides in that case the following result:

| SERVER | I | PID |
|---------|---|-------|
| myhost1 | J | 17015 |
| myhost2 | J | 11098 |

Stopping the Batch Scheduler

The batchscheduler checks the row matching the process in table `ALG_BATCHSCHEDULERS` periodically. If column `IND_ACTIEF` is set to N, then the batchscheduler will stop.

The batchscheduler is stopped on server `myhost` by means of the following SQL statement:

```
update alg_batchschedulers
set    ind_actief = 'N'
```

```
where server      = 'myhost'
/
```

Script `OZG_STOP_BATCH.sh` notifies *all* batchscheduler processes of a certain environment to stop, by performing the following statement:

```
update alg_batchschedulers
set     ind_actief = 'N'
/
```

DYNAMIC LOAD BALANCING MECHANISM

Functionality

Each batchscheduler process uses the dynamic load balancing mechanism in order to determine how many new script requests can be processed. As no central 'key player' has been appointed, each batchscheduler process can use the mechanism autonomously.

For this reason the following information is required:

1. The number of batchschedulers (# batchschedulers)
2. The number of active processes started by the batchschedulers (#active_processes)
3. The number of active processes started by the proper batchscheduler process (#child_processes)
4. The maximum number of batch processes for the given database (#batch_max_processes).

This value is determined by the system parameter `BATCH_MAX_PROCESSEN`

The mechanism returns the smallest number of

1. `#batch_max_processes – #active_processes`
2. $(\#batch_max_processes / \#batchschedulers) - \#child_processes$

Example

Imagine the following situation:

- The value of `BATCH_MAX_PROCESSEN` is 100
- 4 batchschedulers online with 17, 22, 24 and 18 active processes respectively.
- 40 requests on hold

then we can see the following division:

| Scheduler | 1 | 2 | 3 | 4 | Total |
|------------------------------------|----|----|----|----|-------|
| Max. #processes per batchscheduler | 25 | 25 | 25 | 25 | 100 |
| Active | 17 | 22 | 24 | 18 | 81 |
| To be started per batchscheduler | 8 | 3 | 1 | 7 | 19 |

NAMING REPORTS SERVER

CAUTION: The instruction below still needs to be adapted for the situation where the new application server (Fusion Middleware WebLogic) is used. This will be updated soon (March 2011).

In case multiple application servers are active, the Reports Server to be used has to have the same name for each application server.

For this, a standalone Reports Server can be started by means of command `rwserver.sh`; for more details see the Oracle*Reports documentation.

APPENDIX B - INSTALLATION & CONFIGURATION OF OHI IN A RAC ENVIRONMENT

INTRODUCTION

This chapter describes the installation and configuration of OHI Back Office in an Oracle RAC environment.

This chapter only applies for customers who would like to migrate their current OHI Back Office 'single instance' environment to an Oracle RAC environment.



Via the Oracle website, under the denominator *Oracle By Example (OBE) Series*, it is possible to listen to free Mini Lessons in order to learn how to use the different features of the Oracle Database.

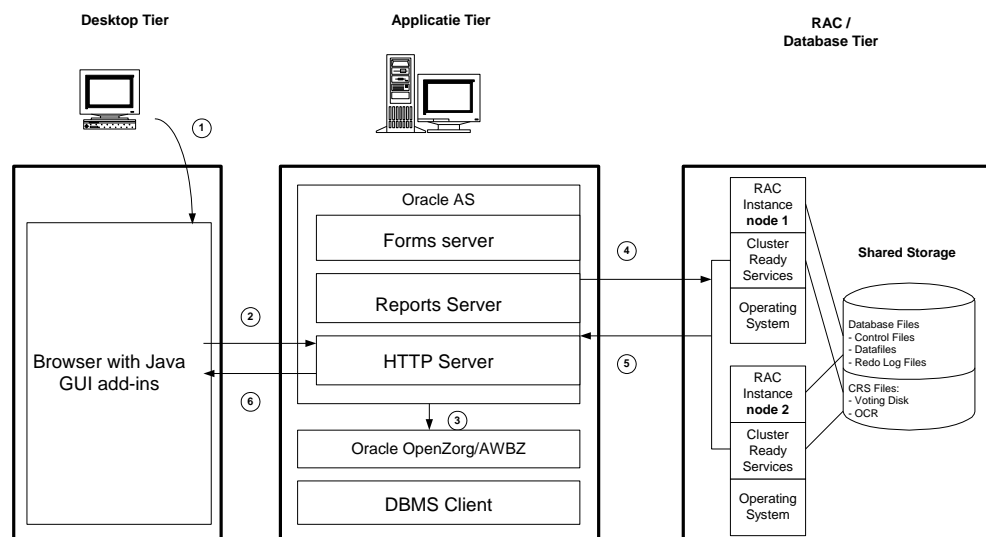
There are e.g. training sessions for RAC, ASM, Clusterware and Manageability.

RELATED PUBLICATIONS

See [Related publications/RAC](#).

ARCHITECTURE

The following provides a general overview of the architecture.



Some remarks about the architecture:

1. Deploying RAC alone is **not** a complete High Availability solution. If e.g. a SAN (Storage Area Network) breaks down, RAC does **not** offer protection and having a

second SAN (stretched cluster situation) could provide a solution.

For an overview of the Oracle Maximum Availability Architecture see:

<http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm>

2. For additional information on shared storage solutions, see the following My Oracle Support note (which is one of the reasons why ASM is the only certified choice):



183408.1 : Raw Devices and Cluster File systems With Real Application Clusters



Oracle recommends using ASM as a Shared Storage solution for the Database Files and a Shared RAW device for the Oracle Cluster Registry and the Voting Disk.

3. In terms of availability and performance Oracle recommends using at least two Host Bus Adaptors and at least two Network Interface Cards per server/node.
4. In case of RAC, Oracle recommends to separate the Application tier from the RAC / Database tier. This supports the Oracle Grid Strategy.

HARDWARE AND SOFTWARE REQUIREMENTS

In order to run OHI Back Office in an Oracle RAC environment, the following software has to be installed completely:

Cluster Layer:

- Oracle Clusterware (consult [Oracle Health Insurance Certification Form](#) for required version and interim patches)

Database Layer:

- Oracle Database Server (consult [Oracle Health Insurance Certification Form](#) for required version and interim patches)

SIZING AND PERFORMANCE

SYSAUX Tablespace

In case of RAC more space is required in the SYSAUX tablespace; in order to determine its size, use script `$ORACLE_HOME/rdbms/admin/utlsyxsz.sql`.

Instance Memory

In case of RAC Oracle recommends to set up extra allocations per node; see [memory advice RAC](#).

INSTALLATION AND CONFIGURATION

This section describes the installation steps that are required for configuring an Oracle RAC environment for OHI Back Office that has to be performed as a preparation for migration.

The steps have to be performed by the customer and go as follows:

1. Installation of the OS
Check if the kernel settings correspond with the values prescribed in the “Oracle® Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide”.

2. Configuration of the network

Keep in mind that RAC require at least three network interfaces:

- Public Interface – used for regular network traffic
- Virtual (Public) Interface – used for failover and RAC management
- Private Interface – used for the Interconnect



Oracle recommends using the interconnect network cards of at least 1 Gigabyte Ethernet and these have to be connected to a switched network, preferable on a separate switch.



Attention: As of Release **2007.02** the use of *Net Service* names is mandatory. For this release, these have to be the same as \$ORACLE_SID and \$OZG_BASE.

3. Configuration of the shared storage.
ASM and OCFS are supported. See the specific documentation for additional information.
4. Ensure that the Oracle user can log in automatically via ssh.
5. Move the OHI Back Office RAC installation and configuration scripts to the \$OZG_ADMIN directory.
6. Install the Oracle Cluster Ready Service and any patch sets. See also the Certification Form **2007.02.0000 (RAC)**.
7. Install the Oracle DBMS and any patch sets. See also the Certification Form **2007.02.0000 (RAC)**.

MIGRATION

The following describes the steps that have to be taken to migrate a single instance database to an Oracle RAC environment.

This section is an addition to chapter 3.

In order to migrate an OHI Back Office Single Instance Database to an Oracle RAC environment, the following conditions have to be met:

1. OHI Back Office Release 2007.01.0000 has been installed.
2. ASM has to be used.
3. For ASM, Recovery MANager has to be used as a backup restore tool.
4. The steps described in section 14.5 have been completed successfully.



This section presumes ASM as a Shared Storage solution. Other migration options as well as choices for shared storage will not be described in this manual.

For additional information, see the relevant installation and configuration manuals.

Creation of an ASM instance

Create an ASM instance with the following parameters:

```
INSTANCE_TYPE      =  ASM
ASM_DISKSTRING      =  ‘ ‘
```



Parameter `ASM_DISKSTRING` is customer-dependent and has to contain a reference to the shared storage.

This parameter is additionally intended to exclude disks that are not to be used.



In case of Linux `ASM Lib` can be used for managing an ASM environment.

The RPMs are downloaded from Oracle Technet via the following URL:

<http://www.oracle.com/technology/tech/linux/asmlib/index.html>

Creation of ASM disk groups

For OHI Back Office the following disk groups have to be created:

```
OZG_DG_DATA
OZG_DG_FRA
```



Oracle recommends ASM mirroring (normal redundancy), also external mirroring is supported.



The `OZG_DG_FRA` disk group (for Flash Recovery Area) is optional.

Migrating the database to ASM

For migrating the current OHI Back Office database to ASM the following whitepaper can be used:

Oracle Database 10g Migration to Automatic Storage Management

This is available on OTN via the following link:

www.oracle.com/technology/deploy/availability/pdf/Technical_WP_ASM_Migration.pdf

Check

Check if the files that are migrated to ASM have been moved to the correct disk group.

Migrating single instance to RAC

This section provides a description of how a single instance can be migrated to an Oracle RAC environment.

This calls for using the Database Configuration Assistant, abbreviated as `dbca`.



Oracle recommends performing the RAC migration by means of the *Database Configuration Assistant* (dbca).

1. For database type select the Oracle Single Instance Database. Subsequently select Manage Templates
2. Create a template while retaining data of the 'single instance' database.
3. Option

```
create a database template / from existing database (structure as
well data)
```



For ASM databases a different structure will have to be selected.

4. Choose the desired Database Instance, specify a name and possibly a description.
5. Subsequently, in this case, select the option: Maintain the file locations.
The template will be stored in the
6. Afterwards restart dbca but now with the database type Oracle Real Application Cluster Database.
7. From the list of nodes select the nodes that will be part of the cluster.
8. Subsequently select the template created in step 1.
9. For option Storage Mechanism choose the Cluster File System.
10. For option Specify File Locations for Database Files to be created select

```
Use Database File Locations from Template.
```

11. Dbcas will subsequently convert the single instance database to an Oracle RAC environment

- - For each instance a separate redo log and undo log will be created.
- - For each instance a new `init.ora` will be created.
- - For each instance a password file will be created.

During the conversion the following report may appear:

```
ORA-28010 cannot expire external or global accounts
This report can be ignored. dbca lets all accounts expire.
```

12. Check if all instances are active by means of the following query:

```
SELECT      *
FROM        v$active_instances
```

13. After migration, check which accounts have expired with the following query and also restore any other relevant details:

```
SELECT      username,
            lock_date,
            expiry_date
FROM        dba_users;
```

Migrate database to a different platform.

If the current OHI Back Office environment runs on a different platform than the RAC environment, migration can be performed by means of Transportable Tablespaces.

To recapitulate briefly, the following steps have to be performed:

1. check if both the source and destination are available in the list of supporting platforms, as well as the `endian_format`.

```
SELECT *
FROM v$tablespace
```

2. Check if the tablespaces are 'self contained' by means of the procedure

```
dbms_tts.transport_set_check.
```

3. Set the tablespace to read only
4. Export the tablespace metadata
5. Perform the tablespace / data file conversion so that the 'endianness' is the same.

This conversion has to be performed through RMAN by means of commands `convert tablespace`, or `convert datafile`.

For an extended description of these commands, see:



Oracle® Database Backup and Recovery Advanced User's Guide 10g
Release 2

6. Copy the data files to the RAC platform
7. Import the tablespace metadata

A description of steps 2, 3, 4, and 7 can be found in section [database reorganisation](#).

RAC PARAMETERIZATION

If OHI Back Office is used in an Oracle RAC environment then extra parameters will be configured for RAC in the parameter file.



Oracle recommends the use of an spfile in an Oracle RAC environment.

The advantage of an spfile is the joint use by multiple instances as well as the dynamic modification of the parameters.

If ASM is used, then the spfile will also have to be moved to ASM.

Spfiles of ASM instances are moved to the local file system.

The following parameters have to be configured to the indicated values.

When using an spfile a cluster consist of 2 nodes is presumed.

Mandatory RAC instance parameters:

`CLUSTER_DATABASE_INSTANCES=`

`REMOTE_LISTENER =`

```

LOCAL_LISTENER      =
<INSTANCE1>.INSTANCE_NUMBER  =
<INSTANCE2>.INSTANCE_NUMBER  =
<INSTANCE1>.THREAD      =
<INSTANCE2>.THREAD      =
<INSTANCE1>.UNDO_TABLESPACE =
<INSTANCE2>.UNDO_TABLESPACE =

```



The aforementioned mandatory RAC instance parameters are completed during the `dbca` RAC migration.



If ASM is used as shared storage, then parameter `CONTROL_FILES` has to refer to a valid ASM Disk group.

Mandatory ASM instance parameters:

```

ASM_DISKGROUPS      =  ozg_dg_data
DB_CREATE_FILE_DEST  =  '+ozg_dg_data'
DB_CREATE_ONLINE_LOG_DEST =  '+ozg_dg_data'

```

Optional ASM instance parameters, if an FRA is used:

```

DB_RECOVERY_FILE_DEST  =  '+ozg_dg_fra'
DB_RECOVERY_FILE_DEST_SIZE =

```



For the value of parameter `DB_RECOVERY_FILE_DEST_SIZE` take the size of the database including the generated archive log files on the busiest day and multiply this by 2 for an extra margin.



If ASM is *not* used then FRA has to be located on a shared file system.

WORKING WITH SERVICES

OHI Back Office uses services in an Oracle RAC environment. These have to be configured after the RAC installation.

For OHI Back Office two services have to be identified which are listed in the following table.

| Service name ^e | Preferred Instance | TAF Policy |
|---------------------------|---|------------|
| OLTP | All available nodes | none |
| BATCH | Preferred on node 1 Available on the other nodes | none |



Oracle recommends using `dbca` for defining the services.



Ensure that for OHI Back Office the *Connection Load Balancing Goal* is set to “Long” for both services.

BATCH PROCESSING

In an Oracle RAC environment it is possible to run the batch processing across multiple nodes. OHI Back Office *requires*, however, running the batch processes (= the `BATCH` service) on *one node*. This has to do with unnecessary 'block pings' between multiple instances. The nodes have to run on a server with the most memory and the biggest CPU capacity.



For future releases of OHI Back Office, there is a possibility to assign batches dynamically to services that could run across one or multiple nodes.

FAILOVER

If all conditions for RAC Installation and Configuration are met, the failover test is the last step that has to be performed.

Attention that OHI Back Office does not support a full Transparent Application Failover.

This has to do with the fact that the PL/SQL variables and package states used by Oracle Forms are lost in case of instance failure.

Therefore OHI Back Office will provide the following messages in case of an instance failure:

FRM-21011 PL/SQL Unhandled Exception Program Error

ORA-03113: End of File on Communication Channel

FRM-40655 SQL error forced rollback : clear form and re-enter transaction.

This is predictable behavior and the user will have to start a new connection with an active instance.



OHI Back Office recommends *not* configuring TAF considering that TAF offers no added value in a Forms environment.

MANAGEMENT

Backup and Recovery

See [Backup & Recovery/RAC](#).

Startup and Shutdown

See [Startup & Shutdown/RAC](#).

Networking

See [Networking/RAC](#).

Performance Tuning and Monitoring

See [Performance Tuning & Monitoring/RAC](#).

APPENDIX C - INSTALLATION & CONFIGURATION OF SSO IN OHI BACK OFFICE

INTRODUCTION

This chapter describes the installation and configuration of Single Sign on for OHI Back Office and applies only to customers who would like to migrate their OHI Back Office database users to the Oracle Internet Directory for use with SSO.

As of release 2008.01 OHI Back Office is certified to work with SSO. This certification is valid upon release 2010.03. With release 2011.01 it is deprecated until further notice.



Attention: Release 2008.01 SSO will only support database connections to exclusive schemas. In later releases, support will be added for shared schemas too.



Attention: In release 2008.01 only the Oracle Internet Directory is supported for SSO.



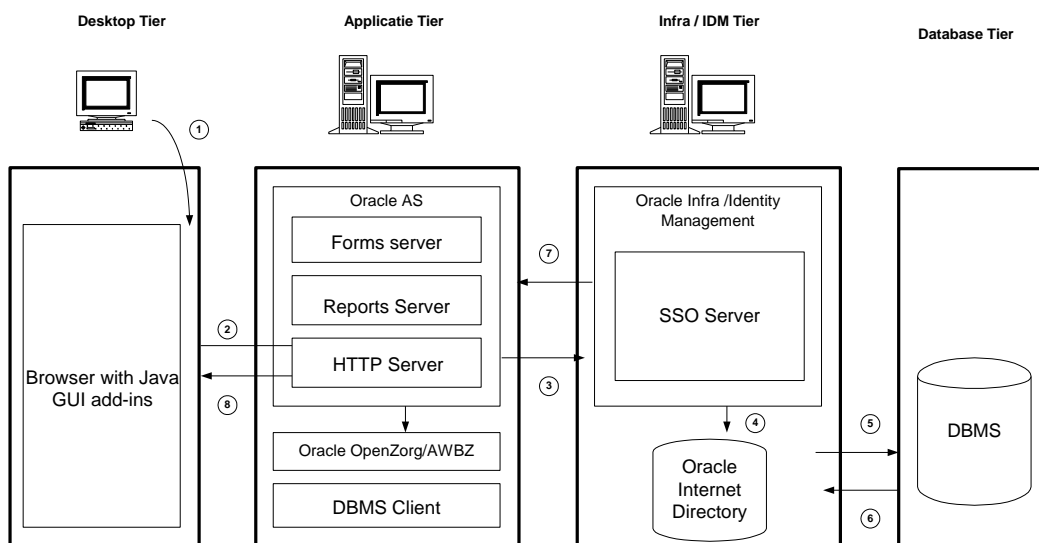
Attention: OHI Back Office release 2008.01 is only certified with password based authentication.

RELATED PUBLICATIONS

See [#Related_Publications_SSO](#)

ARCHITECTURE

The following picture provides a general overview of the architecture.



Global explanation of the architecture:

- User starts OHI Back Office in a Browser Session;
If user is not validated, redirection takes place to the SSO Server with the use of the HTTP Server Module `mod_osso`.
- The User is authenticated in the Oracle Internet Directory.
- If authentication has succeeded, the SSO Server sets the cookie in the user's browser.
- User is authorized in OHI Back Office with the use of the secure application role.

HARDWARE AND SOFTWARE REQUIREMENTS

OHI Back Office with SSO has been certified with the following software configuration:

- Forms & Reports Services 10.1.2.2.0;
- Oracle Application Server Identity Management 10.1.4.0.1 (Enterprise Edition).



In the Oracle Application Server Documentation Library (Oracle Identity Management) see the OS specific Installation Guides for Hardware and Software requirements.



SSO integration must be manually added with the existing Forms & Reports midtier, see Oracle® Application Server Forms and Reports Services Installation Guide, Chapter 6.



In the user's browser, the security settings must be configured in a way that it can accept cookies.

SOME IMPORTANT REMARKS REGARDING THE CONFIGURATION OF THE OHI BACK OFFICE SSO ENVIRONMENT

Planning the (LDAP) Directory Tree

Before one can migrate the database users to the Oracle Internet Directory, one must make some decisions about the Directory Tree.



For more information about implementing the Directory Tree, consult the person, or department responsible for the Directory Server Administration,

Migrating Database Users to Oracle Internet Directory

For migrating the current database users to Enterprise (LDAP) users, one can use the Oracle supplied User Migration Utility (umu), which is available in the Oracle Database Server installation.

The umu tool will generate random passwords for the database and the directory.

For more information see the manual: Oracle® Database Enterprise User Administrator's Guide, Appendix A.

Another way to migrate local database users is by executing the following command:

```
ALTER USER <username> IDENTIFIED GLOBALLY AS '<user DN in OID>';
```

One condition to this method is that every user must be manually added to the Oracle Internet Directory by using either the oidadmin tool or the Delegated Administration Service (DAS).



Both methods require new passwords for every user migrated to the Oracle Internet Directory

Oracle Internet Directory Single Point of Failure

Important to know is that if all the database users are located in the Oracle Internet Directory, OID becomes a single point of failure. Therefore it is advised to have a second OID environment available in case the first one becomes unavailable.



Oracle recommends implementing some form of High Availability for the Oracle Internet Directory.

Integration with other LDAP Directory Servers

Integration with other LDAP Directory Servers like Microsoft Active Directory is possible by using the Oracle Directory Integration and Provisioning Platform (DIP).

This is explained in the document Oracle® Identity Management Integration Guide.



At this moment the Oracle Internet Directory is the only Directory Server that is certified for OHI Back Office, therefore OID must be leading.

DATABASE CONFIGURATION

For using SSO with OHI Back Office some configurations must be done to the Database and Oracle Net Services.

Creating the ldap.ora

First Oracle Net Services must be configured so that the database can find the right Directory Server.

One can configure the `ldap.ora` with the use of Net Configuration Assistant (`netca`).

In `netca` choose Directory Usage Configuration and for Directory type choose Oracle Internet Directory, and after that choose the right OracleContext.

Registering the Database in OID

With the use of `dbca`, the database must be registered in the Oracle Internet Directory so that the database can login to it.

The following values must be filled:

```
User DN          directory superuser (e.g. orcladmin)
Password
Wallet password
```

DBCA creates a wallet and stores it in the `$ORACLE_BASE/admin/Oracle_SID/wallet`

This wallet is used to secure the connection between the database and the directory.

Database Parameter ldap_directory_access

Change the database parameter `ldap_directory_access` in the `pfile`, or `spfile`.

For OHI Back Office, `password` is the only supported value at this moment.

Configuring Database Authentication

For Database authentication, OHI Back Office SSO uses the Oracle Internet Directory to store User Identities.

A new user must be created the following way:

```
CREATE USER <username> IDENTIFIED GLOBALLY AS '<user DN in OID>';
```

And needs only the create and alter session privilege

```
GRANT CREATE SESSION TO <username>;
GRANT ALTER SESSION TO <username>;
```

FORMS & REPORTS CONFIGURATION

Because OHI Back Office is only certified with Forms & Reports the integration between the Midtier and the Identity Management Infrastructure must be manually configured.

This configuration is described in Oracle® Application Server Forms and Reports Services Installation Guide, Chapter 6.

After configuration, copy the following entries from the `formsweb.cfg` (located in `de FRS10G2/forms/server`) to the desired configuration section in the `ozg_formsweb.cfg`:

```
oid_formsid          = the identifier
formsid_group_dn     = the Distinguished Name in the LDAP
ssoMode              = true
ssoDynamicResourceCreate = true
```

And for the reports services copy the following entry from the `rwserver.template` to the report server conf file.

```
<property name="oidEntity" value = "value for %REPORTS_OID_ENTITY%"
```

Configuring the Forms Servlet

If the user's password contains numeric and alphanumeric characters, then the following parameter must be added to the `ozg_servlet` file.

```
FORMS_USERNAME_CASESENSITIVE = 1
```

SSO SYSTEM ADMINISTRATION

Managing Enterprise Users

When the database users are migrated to the Oracle Internet Directory the best way to administer those users is by means of the Delegated Administration Service. This has a web-based interface where the IAS Administrator, and/or end-users can manage their accounts.

The Delegated Administration Service is part of the infrastructure / identity management and it's automatically configured after installation.

With DAS, the following privileges can be delegated:

- Creation, editing and deletion of users and groups;
- Assignment of privileges to users and groups;
- Management of services and accounts;
- DAS configuration;
- Resource Management of Oracle Forms & Reports Services.

An important issue for OHI Back Office working with SSO is the configuration of Resource Access Descriptors, RAD.

OHI Back Office as a forms application needs a RAD for making connection to the database.

The Forms Servlet reads the RAD from a named entry in the Oracle Internet Directory.

For creating a RAD the following parameter must be specified:

```
Resource name  name of OHI Back Office configuration in
                ozg_web.cfg
Username       username as specified in OID
Password       the SSO password
Database       the database to where OHI Back Office must make a
                connection
```



If no RAD exists and `ssoDynamicResourceCreate = true`, the user is redirected to the 'resource creation' page where he or she can create a new RAD.



At this moment, when the user changes his or her SSO password, the password in the RAD must also be changed.

For more information about configuring DAS see the manual Oracle® Identity Management Guide to Delegated Administration.

Monitoring the Single Sign-On Server

One can monitor following items of the SSO Server from the IAS Console (Infrastructure):

- Status of the SSO server;
- Number of logins, successful logins and failed logins.

For more information about monitoring the SSO server see the manual Oracle® Application Server Single Sign-On Administrator's Guide.

Administering the Oracle Internet Directory

One can administer the Oracle Internet Directory with the Oracle Directory Manager that is located in the `IDM10G/bin` directory.

There are some important issues regarding the administration of the Oracle Internet Directory:

1. Password policy management

The default password expiration is 120 days, and that includes the `orcladmin` account.

In case a certain password policy is enabled for the superuser account (`orcladmin`) and the account becomes locked, it can be unlocked by the ODS user with the use of the `oidpasswd` tool with the following parameters:

```
oidpasswd [connect=<OID schema database connect string>]  
unlock_su_acct=true
```

My Oracle Support Note: 370247.1 gives a good explanation how to unlock the OID superuser account.

2. Statistics gathering

After an initial load, or significant changes in the OID Directory Data, one must update the Oracle Directory Server (ODS) statistics. This can be achieved by running the `oidstats.sql` script that is located in the `IDM_HOME/ldap/admin/` directory with the following parameters:

```
sqlplus ods/ods_password@connect_string@oidstats.sql
```

For more information about administering OID see the manual Oracle® Internet Directory Administrator's Guide.

APPENDIX D - INSTALLING REQUIRED PERL MODULES

INTRODUCTION

The following additional Perl modules need to be installed for the OHI installation menu `OZGPATCH` to work.

Some modules are also used by OHI application software.

All modules (from www.cpan.org) are available on Oracle's iProjects website; please find the names of the zip files below:

| Set | Installation order | Name of CPAN Perl module | Version | File to download |
|-----|--------------------|--------------------------|---------|------------------|
| | | File::NCopy | 0.34 | file.zip |
| | | File::Remove | 0.34 | remove.zip |
| | | Shell::Source | 0.01 | shell.zip |
| | | Term::ReadKey | 2.30 | term.zip |
| | | | | |
| 1 | 1 | Test::Simple | 0.86 | testsimple.zip |
| | 2 | DBI | 1.607 | dbi.zip |
| | 3 | DBD::Oracle | 1.22 | dbdoracle.zip |



Note 1: The `gcc` compiler is required for installation of the `Term::ReadKey` module on SunSolaris.



Note 2: All modules *in a set* should be installed in a *specific order*, as indicated in the table above.



Note 3: Installing module `Term::ReadKey` on Oracle Enterprise Linux can give errors during generating the module.
See [Term::ReadKey issue](#).

INSTALLATION PROCEDURE

The following steps have to be followed to install each module. The steps must be executed by `root`.

1. Download the zip file concerned to the application server which will run the installation menu and place it in a temporary directory, e.g. `$TMP`.
2. Unzip the zip file into a temporary directory with `unzip`.
3. Navigate to the directory created for the module.
4. Generate instructions for the installation by running the make file:

```
perl Makefile.PL
```

5. Run `make` to create the blib installation directory:

```
make
```

6. Test the functionality of the module with

```
make test
```

7. Install the module with

```
make install
```

8. Delete the temporarily created directory.

Example

The following is an example installation of the `Term::ReadKey` module on IBM AIX:

```
# unzip term.zip # *** step 2 ***
Archive: term.zip
  creating: TermReadKey-2.30/
  inflating: TermReadKey-2.30/genchars.pl
  inflating: TermReadKey-2.30/Makefile.PL
  inflating: TermReadKey-2.30/Configure.pm
  inflating: TermReadKey-2.30/test.pl
  inflating: TermReadKey-2.30/ReadKey.pm
  inflating: TermReadKey-2.30/META.yml
  inflating: TermReadKey-2.30/ReadKey.xs
  inflating: TermReadKey-2.30/ppport.h
  inflating: TermReadKey-2.30/MANIFEST
  inflating: TermReadKey-2.30/README
  extracting: TermReadKey-2.30/blib/lib/Term/.exists
  inflating: TermReadKey-2.30/blib/lib/Term/ReadKey.pm
  extracting: TermReadKey-2.30/blib/lib/auto/Term/ReadKey/.exists
  inflating: TermReadKey-2.30/blib/lib/auto/Term/ReadKey/autosplit.ix
  extracting: TermReadKey-2.30/blib/arch/auto/Term/ReadKey/.exists
  inflating: TermReadKey-2.30/blib/arch/auto/Term/ReadKey/ReadKey.so
  extracting: TermReadKey-2.30/blib/arch/auto/Term/ReadKey/ReadKey.bs
  extracting: TermReadKey-2.30/blib/man3/.exists
  inflating: TermReadKey-2.30/blib/man3/Term::ReadKey.3pm
  inflating: TermReadKey-2.30/Makefile
  extracting: TermReadKey-2.30/pm_to_blib
  inflating: TermReadKey-2.30/cchars.h
  inflating: TermReadKey-2.30/ReadKey.o
  inflating: TermReadKey-2.30/ReadKey.c
  extracting: TermReadKey-2.30/ReadKey.bs
# cd TermReadKey-2.30 # *** step 3 ***
# ls
Configure.pm  Makefile.PL  ReadKey.o    cchars.h     test.pl
MANIFEST      README       ReadKey.pm   genchars.pl
META.yml      ReadKey.bs   ReadKey.xs   pm_to_blib
Makefile      ReadKey.c    blib         ppport.h
# perl Makefile.PL # *** step 4 ***
Writing Makefile for Term::ReadKey
# make # *** step 5 ***
cc -c -D_ALL_SOURCE -D_ANSI_C_SOURCE -D_POSIX_SOURCE -qmaxmem=16384
1506-507 (W) No licenses available. Contact your program supplier to
Running Mkbootstrap for Term::ReadKey ()
chmod 644 ReadKey.bs
rm -f blib/arch/auto/Term/ReadKey/ReadKey.so
LD_RUN_PATH="" ld -bhalt:4 -bM:SRE -
I:/usr/local/lib/perl5/5.8.0/aix/CORE/per
chmod 755 blib/arch/auto/Term/ReadKey/ReadKey.so
cp ReadKey.bs blib/arch/auto/Term/ReadKey/ReadKey.bs
chmod 644 blib/arch/auto/Term/ReadKey/ReadKey.bs
Manifying blib/man3/Term::ReadKey.3
```

```
# make test # *** step 6 ***
PERL_DL_NONLAZY=1 /usr/local/bin/perl "-Iblib/lib" "-Iblib/arch" -w
test.pl
1 .. 8
ok 1
ok 2
ok 3
ok 4
ok 5
ok 6
ok 7
ok 8
# make install # *** step 7 ***
Installing
usr/local/lib/perl5/site_perl/5.8.0/aix/auto/Term/ReadKey/ReadKey.so
Installing
usr/local/lib/perl5/site_perl/5.8.0/aix/auto/Term/ReadKey/ReadKey.bs
Files found in blib/arch: installing files in blib/lib into architec
Installing
/usr/local/lib/perl5/site_perl/5.8.0/aix/auto/Term/ReadKey/autos
Installing /usr/local/lib/perl5/site_perl/5.8.0/aix/Term/ReadKey.pm
Installing /usr/local/man/man3/Term::ReadKey.3pm
Installing /usr/local/man/man3/Term::ReadKey.3
Writing
/usr/local/lib/perl5/site_perl/5.8.0/aix/auto/Term/ReadKey/.packlis
Appending installation info to
/usr/local/lib/perl5/5.8.0/aix/perllocal.pod
# cd ..
# rm -Rf TermReadKey-2.30 # *** step 8 ***
```

SPECIFIC INSTALLATION INSTRUCTIONS FOR DBD::ORACLE

To connect to the Oracle database, the Perl `DBI` module is used (independent database interface/access module), together with the Perl `DBD::Oracle` module (driver for Oracle databases).

This last module performs a *connection test* on the database.

To be able to successfully perform this test, as part of installing `DBD::Oracle`, perform following steps:

Prerequisites

First install the Perl modules (if not already installed!):

1. `Test::Simple`
2. `DBI`

SELinux on RHEL/OEL 5

When running on Red Hat/Oracle Enterprise Linux 5, *and* SELinux (Security-enhanced Linux) is enabled and running in Enforcing mode (that is; as `root` the `getenforce` command returns `Enforcing`), you might run into Oracle Security Service bug 6140224.

This bug will cause following error when installing `DBD::Oracle`:

```
Can't load '/tmp/DBD-Oracle-1.22/blib/arch/auto/DBD/Oracle/Oracle.so'
for module DBD::Oracle:
/u01/app/oracle/product/11.1.0/db_1/lib/libnnz11.so: cannot restore
segment prot after reloc: Permission denied at
```


/usr/lib64/perl5/5.8.8/x86_64-linux-thread-multi/DynaLoader.pm line 230.

To work around this bug (the workaround changes the file context (for 2 files), as SELinux is disallowing text-segment relocation in the shared object `libnnz10.so`) perform following commands as oracle user (on the application server):

```
. ozg_init.env $OZG_ORATAB_DB11G
cd $ORACLE_HOME/lib
ls -lZ libnnz11.so
chcon -t textrel_shlib_t libnnz11.so
ls -lZ libnnz11.so
ls -lZ libclntsh.so.11.1
chcon -t textrel_shlib_t libclntsh.so.11.1
ls -lZ libclntsh.so.11.1
```

For more information see



My Oracle Support note 454196.1 - ./sqlplus: error on libnnz11.so: cannot restore segment prot after reloc

Specific Set up for Root Account before Installing DBD::Oracle

As root user, now perform following steps to correctly indicate the Oracle OCI (Oracle Call Interface) environment for the connection test.

This is done by setting `$ORACLE_SID` to a relevant database, and `$ORACLE_HOME` to the database (client) software:

```
. <path to $OZG_ADMIN>/ozg_init.env
. ozg_init.env $OZG_ORATAB_DB11G
. ozg_init.env <SID>
export ORACLE_USERID=<OHI account>/<password OHI account>
export LD_RUN_PATH=$ORACLE_HOME/lib
```

Example

(For this example, it is presumed `ozg_owner` is the OHI application owner account set up in the PROD database; the Oracle environment will therefore have to be set to PROD)

```
. /u01/app/oracle/product/OHI/admin/ozg_init.env
. ozg_init.env $OZG_ORATAB_DB11G
. ozg_init.env PROD
export ORACLE_USERID=ozg_owner/ozg_owner
export LD_RUN_PATH=$ORACLE_HOME/lib
```

32-/64-bit

Be sure to use the right Perl version; when using 64-bit Oracle Database (Client) software, you should use the 64-bit version of Perl.

When switching from 32-bit to 64-bit, all of the Perl modules mentioned in the table in the Introduction paragraph, need to be reinstalled.

The Perl version can be checked with `file /usr/bin/perl` or `perl -v` (or `-V`).

AIX

On AIX both versions will be installed on your system; by switching Perl symbolic links it is possible to switch 32/64-bit. This is done as root, and is documented on your system (usually `/usr/opt/perl5/README`); ask system administrators for more info on your local system.

SunSolaris

When there's no 64-bit Perl on your local SunSolaris system, you can install the modules using 32-bit Perl (see the file `README.64bit.txt` in your local Perl installation).

For that, when installing `DBD::Oracle`, you should set following environment variables

```
export LD_LIBRARY_PATH=$ORACLE_HOME/lib32
export LD_RUN_PATH=$ORACLE_HOME/lib32
```

and set up the make file using

```
perl Makefile.PL -m $ORACLE_HOME/rdbms/demo/demo_rdbms32.mk
```

Install DBD::Oracle

Now install `DBD::Oracle` by following the standard installation procedure for Perl modules; see paragraph [Installation procedure](#).

AIX

On AIX, following error during the `make test` phase may be ignored:

```
t/32xmltype.....ok 3/4Unsupported named object type for bind
parameter at t/32xmltype.t line 71.
```

Linux

On Linux, following error during the `make test` phase may be ignored:

```
t/56embbeded.....ok 2/5DBD::Oracle::db do failed: ORA-00600:
Interne foutcode, argumenten: [kothc_uc_md5:lxerr], [], [], [], [],
[], [], []. (DBD ERROR: OCISStmtExecute) [for Statement "CREATE or
replace TYPE table_embeda_type as varray(10) of varchar(30) "] at
t/56embbeded.t line 48.
```

TERM::READKEY ISSUE

When installing the mandatory Perl modules on a Linux server a problem can occur with installing module `Term::ReadKey`. A workaround for this problem is to download and install the precompiled package.

The package for RedHat5 or OEL5

`perl-TermReadKey-2.30-3.el5.rf.x86_64.rpm` can be downloaded from:

<http://rpmfind.net/linux/rpm2html/search.php?query=perl-TermReadKey>

APPENDIX E - INSTALLATION CHECKLIST

Following is a *quick reference checklist* for installing an OHI Back Office environment. The hyperlinks refer to the relevant paragraphs in this document.

1 – Prerequisites

- [Check OHI Back Office certification](#)
- [Check hardware & software requirements](#)
- [Check OS requirements](#) (for both Database & Application Server)
- [Download all required software](#) (OS, Database, Application Server, OHI, Perl, Java, etc.)
- [Install & configure OS](#)
- [Install additional required Perl modules](#)

Validation

Validate OS requirements by starting Oracle Universal Installer; all requirements should have been met.

2 – Install Database

- Install Oracle Database Server
- *Install All Product Languages*
- Install Oracle Database Examples (formerly Companion)
- Install Oracle Database Server patch set
- Install Oracle Database Server interim patches
- Apply Oracle Database Server workarounds for open bugs
- Configure Oracle Database Server
- [Create an OHI Back Office database](#)

Validation

Use OEM and/or OPatch to see if all interim patches are installed.
Are all workarounds mentioned in Certification Matrix installed?

Check the Database languages.

```
. ozg_init.env <Net Service Name>
. ozg_init.env DB11G
sqlplus sys as sysdba
```

Enter a command that generates a message, for example:

```
SQL> exec dbms_output.put_line(1)
```

This command returns the message

```
PL/SQL procedure successfully completed.
```

Check if this message is in the correct language.

Check if the NLS_NUMERIC_CHARACTERS is set correctly.

3 – Install Application Server

- Install WebLogic Server
- Install Portal, Forms, Reports and Discoverer
- Install Portal, Forms, Reports and Discoverer patchset
- Install Portal, Forms, Reports and Discoverer interim patches
- [Initial Configuration Forms and Reports services](#)
- Detailed Configuration of Forms and Reports services

Validation

Check the connection to the database and the version of the database client.

```
. ozg_init.env <Net Service Name>
. ozg_init.env $OZG_ORATAB_DB11G
sqlplus sys as sysdba
sqlplus sys as sysdba@<env>
```

Check the version of import and export utility and the Pro*C compiler.

```
. ozg_init.env <Net Service Name>
. ozg_init.env $OZG_ORATAB_DB11G
imp help=y      => check version number
exp help=y      => check version number
proc -version => check version number
```

Check the version of the Developer software.

Use the following command to ensure the Oracle*Forms software is valid and check the version.

```
$OZG_ADMIN/OZG_CMD.pl frmcmp_batch.sh -help
```

Use the following command to ensure the Oracle*Reports software is valid and check the version.

```
$OZG_ADMIN/OZG_CMD.pl rwconverter.sh -help
```

Check if the NLS_NUMERIC_CHARACTERS is set correctly.

4 – Prepare OHI Back Office Installation

- Set up configuration templates (ozg_init.env)
- Set up directory structure

Validation

Check if Operating System specific settings are correct.

For Linux

```
. ozg_init.env $OZG_ORATAB_FRS11G1
```

LD_LIBRARY_PATH must be set

For AIX

```
. ozg_init.env
```

MALLOCTYPE must be set

OBJECT_MODE must be set

```
. ozg_init.env $OZG_ORATAB_DB11G
```

LIBPATH must be set

```
. ozg_init.env $OZG_ORATAB_FRS11G1
```

LIBPATH must be set

LDR_CNTRL must be set

```
. ozg_init.env $OZG_ORATAB_OWB10G
```

LIBPATH must be set

```
. ozg_init.env $OZG_ORATAB_OMA10G
```

LIBPATH must be set

```
. ozg_init.env $OZG_ORATAB_OMS10G
```

LIBPATH must be set

Check if the Perl installation software compiles without errors, and check the Perl version.

```
. ozg_init.env <Net Service Name>
. ozg_init.env $OZG_ORATAB_DB11G
perl -version
perl -c $OZG_ADMIN/OZGPLIB.pm
perl -c $OZG_ADMIN/OZGPATCH.pl
```

Check if the installation menu can be started.

```
OZGPATCH.pl <Net Service Name>
```

Fill the parameters and a release.

If the menu options appear, the following is correct:

- Perl libraries
- Developer version
- database connection

5 – Install OHI Back Office

- Perform installation
- Configure the application
- Set up logo's
- Set up Forms Terminal Resource file

Validation

Test following URLs to check the valid configuration of the HTTP server.

```
http://<server>:<port>/forms/frmservlet?config=<env>
http://<server>:<port>/OHI/<env>/bin/OZGIMG01.gif
http://<server>:<port>/OHI/<env>/bin/OZGIMG02.gif
http://<server>:<port>/OHI/admin/logo.gif
```

Start the application; use the URL:

http://<server>:<port>/forms/frmservlet?config=<env>

Now check:

- The images on the startform and Info screen;
- Icons on iconic toolbar;
- Availability release documentation => Double click on release in the *Releases* screen;
- Availability of Online Help (F1 key);
- Correct language of Forms messages (for example in login screen);
- Running Batch scheduler;
- Run script of type Reports => *Overzicht medium definitie*;
- Run script <> type Reports => *Valideren Business Rules*;
- Check logfiles of scripts;
- Run *Object Validation* => Parameter = B (Both database and filesystem)
Ensure no errors are reported.

APPENDIX F - MIGRATION FROM OAS10G TO FMW WLS11G

Starting with release 2011.01 the OHI Back Office application requires Fusion Middleware Oracle WebLogic Server (FMW WLS) including Forms and Reports services instead of Oracle Application Server (OAS). This Appendix describes the activities to migrate an existing OHI Back Office environment from OAS to FMW WLS.

WHAT HAS CHANGED?

When you are familiar with managing an existing OHI Back Office environment which is running the Oracle Application Server it is good to know what the major changes are in the configuration.

An overview:

- The Jinitiator plugin, which could be used in the browser as Oracle specific Java plugin instead of the standard (formerly Sun and now Oracle) Java plugin is deprecated: only the standard Java plugin is allowed and supported.
- Management and configuration previously was done by editing configuration files and using command line commands. Now an Administration Console for managing the WebLogic components and an Enterprise Manager Control web application are available for implementing an important part of this task. Where previously OHI specific files were used now standardized configuration files are used which can be maintained by these tools. The relevant configuration files are listed separately.
- \$OZG_ADMIN/ozg_web.cfg is replaced by the standard formsweb.cfg configuration file (see the relevant chapter for the location).
- \$OZG_ADMIN/OHI.dat is replaced by the standard Registry.dat file (see the relevant chapter for the location).
- \$OZG_ADMIN/OZG_JPI.htm replaces both OZG_BASEJINI.htm and OZG_BASEJPI.htm.

INSTALL AND CONFIGURE SOFTWARE

This part is described in the regular [Chapter](#) because it is not different from when installing a new environment. The installation and initial configuration can be executed without affecting the existing application server environment when this is executed carefully. Additional software will be set up which can run aside of the existing software. Of course sufficient disk space and system resources should be available for this.

When the configuration is ready and prepared you can migrate the environment from the required 2010.03 patch set level to 2011.01 in the usual way, using the OZGPATCH.pl menu.

POST INSTALL STEPS

When the 2011.01 installation is ready there are a few additional steps necessary to complete the migration to the new application server environment.

The names of the directories in the system parameter screen need to be adapted because normally the port nr of the Oracle HTTP server has changed. Please follow the instructions as mentioned in the paragraph [Configuring directories](#).

Next to that the reports server name has to be adapted so the new Reports server will be used. Please follow the instructions as described in [Registration Reports Server](#).

After having implemented these settings be sure you restart the batch scheduler when it was already running.

RECOMMENDATION

When you have an existing production environment running with Oracle Application Server make a list of all modifications you made to adapt the environment to your needs.

This can be tuning parameters or specific management settings to be informed, special logging configuration settings, etc.

During the acceptance test phase determine what to do with these settings:

- Are they still needed?
- What is their equivalent?
- Can similar settings be used or might it be they need to be adapted?

When you pay attention to the changes you made in the past they can maybe help prevent problems from occurring when you migrate to the new environment.

And when things are not clear it is better to have time available for finding out what might be the reason for a certain setting before going into production. This compared to looking for old settings which helped with 'a similar problem in the past' when you run into a production problem after you have migrated.

APPENDIX G - CONFIGURATION ORACLE APPLICATION SERVER 10G (OHI BACK OFFICE)

This chapter treats the configuration of Oracle AS, which is used up to release 2010.03. So when you are on release 2011.01 or later this Appendix can be ignored.



OHI Back Office can be configured for using Single Sign On. This configuration is described in Appendix C.

INSTALLATION 10G AS SOFTWARE

Install 10g AS Forms and Reports Services

OHI Back Office uses only the Forms and Reports Services of the Oracle Application Server.

With 10g AS it is possible to install *only these* products (required supporting components such as OC4J engine and HTTP server are implicitly installed). Of the Oracle Application Server 10g Enterprise Edition CD-pack therefore *only* the CD with *Forms and Reports Services* has to be installed.

To this end use `$ORACLE_HOME` naming for example `frs10g` (forms and reports services **10g** R1) or `frs10g2` (forms and reports services **10g** R2)

Installation of the complete AS suite is *not* supported for OHI Back Office.

If this is desirable (for other purposes) then the installation has to be performed in a separate `$ORACLE_HOME`.

Install 10g AS Patch Set

After installation of 10g AS Forms and Reports Services the certified patch set has to be installed.

For exact versions of the required Oracle AS software and patch sets, see:



Oracle Health Insurance Certification Matrix



Attention: If AIX is used as OS then for installation of an Oracle Application Server patch set command `slibclean` has to be executed.

Creating an Oracle*Reports Configuration File

Due to a bug in Oracle*Reports, a dummy configuration file has to be entered in the home directories of the OS accounts `oracle` and `batch`.

This is possible by entering the following OS command in these directories:

```
touch prefs.ora
```

This task has to be performed on the Application layer.

REGISTER IN-PROCESS REPORTS SERVER

Subsequently, render the in-process Reports Server (= the default Reports Server, created after installation) accessible to Oracle Enterprise Manager (based on a bug in the default configuration) by means of the following command:

```
addNewServerTarget.sh <name of the in-process Reports Server>
```

For example:

```
addNewServerTarget.sh rep_myhost
```

CONFIGURE WEB.XML

In file `$ORACLE_HOME/j2ee/OC4J_BI_Forms/applications/formsapp/formsweb/WEB-INF/web.xml` a reference has to be made to `ozg_web.cfg`.

To implement this change the following line:

```
<!--
  <init-param>
    <param-name>configFileName</param-name>
    <param-value><your configuration file name goes here></param-
value>
  </init-param>
-->
```

with:

```
<init-param>
  <param-name>configFileName</param-name>
  <param-
value>/u01/app/oracle/product/OHI/admin/ozg_web.cfg</param-value>
</init-param>
```

Do not forget to remove the comment markers `<!--` and `-->`



Attention: During the installation of a 10g AS *patch set* this file will be *overwritten*; it is therefore important to retain the original.

CONFIGURE OPMN.XML

In file `$ORACLE_HOME/opmn/conf/opmn.xml` the generic environment variables have to be specified.

This involves the generic variables that are required for Oracle Forms and Oracle Reports, and a generic OZG variable required for `OZG_CMD.sh`.

With variable `LD_LIBRARY_PATH` on Oracle Enterprise Linux 5 location from library `libjvm.so` is changed. The new location must be added to the value of `LD_LIBRARY_PATH`.

Forms

Add the following to the entry

```
<process-type id="OC4J_BI_Forms" module-id="OC4J">
```

add the following variables (for which `LD_LIBRARY_PATH` naturally has platform-specific entries, and for which the value in `OZG_ORATAB_DB11G` can be selected analogous to `ozg_init.env`).

```
<environment>
  ...
  <variable id="ORACLE_HOME" value="/u01/app/oracle/product/frs10g2"/>
  <variable id="TNS_ADMIN"
value="/u01/app/oracle/product/11.1.0/db_1/network/admin"/>
```

```

<variable id="DISPLAY" value="myhost:0.0"/>
<variable id="FORMS_USER_DATE_FORMAT" value="DD-MM-RRRR"/>
<variable id="FORMS_CATCHTERM" value="0"/>
<variable id="FORMS_TRACE_PATH" value="/tmp"/>
<variable id="FORMS_SCROLL_ALL_BUT_ONE" value="TRUE"/>
<variable id="FORMS_MESSAGE_ENCRYPTION" value="FALSE"/>
<variable id="FORMS_LOV_INITIAL" value="5000"/>
<variable id="FORMS_LOV_MINIMUM" value="1000"/>
<variable id="FORMS_LOV_WEIGHT" value="16"/>
<variable id="FORMS_RESTRICT_ENTER_QUERY" value="FALSE"/>
<variable id="OZG_ORATAB_DB11G" value="DB11G"/>
<variable id="NLS_LANG" value="DUTCH THE NETHERLANDS.WE8MSWIN1252"/>
<variable id="NLS_DATE_FORMAT" value="DD-MM-RRRR"/>
<variable id="NLS_NUMERIC_CHARACTERS" value=",""/>
<variable id="NLS_SORT" value="BINARY"/>
<variable id="LD_LIBRARY_PATH"
value="$ORACLE_HOME/lib:$ORACLE_HOME/network/lib:$ORACLE_HOME/jdk/jre/
lib/sparc"/>
<variable id="PATH"
value="/bin:/usr/bin:/usr/ccs/bin:/usr/local/bin:/usr/ucb:/usr/vac/bin
:$ORACLE_HOME/bin:/u01/app/oracle/product/OHI/admin" append="true"/>
...
</environment>

```



Attention: If the following message “/forms/lervlet:-1” is encountered on the SunSolaris platform when starting the Forms client via the browser, then for OC4J_BI_Forms *in front* in LD_LIBRARY_PATH the entry \$ORACLE_HOME/lib32 has to be added. For further information see Oracle Forms bug 4143510.



Attention: The value of FORMS_USER_DATE_FORMAT and NLS_DATE_FORMAT must match the setting of the environment variable NLS_DATE_FORMAT set in [ozg_init.env](#). This setting can not be changed once OHI Back Office is in use.



Attention: The value of NLS_NUMERIC_CHARACTERS can not be changed once OHI Back Office is in use.



Attention: Set NLS_LANG to the correct language. For Czech implementations this setting should be:
“CZECH_CZECH REPUBLIC.EE8MSWIN1250”



Attention: The values of FORMS_LOV_WEIGHT, FORMS_LOV_INITIAL and FORMS_LOV_MINIMUM are derived from My Oracle Support note 138159.1. These values must match the setting of the equally named environment variables set in [ozg_init.env](#)

Reports

Add the following to the entry

```
<process-type id="ReportsServer" module-id="ReportsServices">
```

under <environment> add the following variables:

```

<variable id="TNS_ADMIN"
value="/u01/app/oracle/product/11.1.0/db_1/network/admin"/>
<variable id="OZG_ROOT" value="/u01/app/oracle/product/OHI"/>
<variable id="NLS_LANG" value="DUTCH THE NETHERLANDS.WE8MSWIN1252"/>
<variable id="NLS_DATE_FORMAT" value="DD-MM-RRRR"/>
<variable id="NLS_NUMERIC_CHARACTERS" value=",""/>
<variable id="NLS_SORT" value="BINARY"/>

```

Example:

```

<ias-component id="rep_myhost" status="enabled" id-matching="false">
  <process-type id="ReportsServer" module-id="ReportsServices">
    <process-set id="rep_myhost" restart-on-death="true"
numprocs="1">
      <environment>
        ...
        <variable id="TNS_ADMIN"
value="/u01/app/oracle/product/11.1.0/db_1/network/admin"/>
        <variable id="OZG_ROOT"
value="/u01/app/oracle/product/OHI"/>
        <variable id="NLS_LANG" value="DUTCH_THE
NETHERLANDS.WE8MSWIN1252"/>
        <variable id="NLS_DATE_FORMAT" value="DD-MM-RRRR"/>
        <variable id="NLS_NUMERIC_CHARACTERS" value=","/>
        <variable id="NLS_SORT" value="BINARY"/>
        ...
      </environment>
      <module-data>
        <category id="general-parameters">
          ...

```



Attention: The value of NLS_NUMERIC_CHARACTERS can not be changed once OHI Back Office is in use.



Attention: The value of NLS_DATE_FORMAT must match the setting of the environment variable NLS_DATE_FORMAT set in [ozg_init.env](#). This setting can not be changed when OHI Back Office is in use.



Attention: Set NLS_LANG to the correct language. For Czech implementations this setting should be:

“CZECH_CZECH REPUBLIC.EE8MSWIN1250”

CONFIGURE REPORTS SERVER

Introduction

This section (briefly) describes the functioning of the batchscheduler in combination with the batch process of type Oracle Reports.

This description serves to clarify the Oracle Reports Server settings described afterwards.

Batch Scheduler

The OHI Back Office batchscheduler is a Pro*C daemon (a continuous process) which performs *iteratively* the following activities via the OHI Back Office application requested scripts:

1. Starting the new script requests
2. Checking the ongoing script requests

This process of starting and checking can be influenced by means of a number of settings in the application (screen *Systeem/Beheer/Algemeen/Systeemparameter*, tab page *Batchscheduler*) (for a more elaborate description of the batchscheduler functionality, see function SY54 in the Oracle Designer Repository).

One of the settings is the so-called *Polling interval*; during this period the batchscheduler “sleeps” and will only be awakened by a newly created request; in this

condition you have to wait for already planned script requests to be started. So only newly created requests will wake the batch scheduler immediately.

1 - Starting new script requests

When incoming script requests (newly created or already planned) have to be started, first there is a check to see if the maximum number of parallel script request has been reached or not.

In order to determine this number (to be set up in OHI Back Office by means of *Max. parallel script requests*) the number of script requests with status `Start` and `running` is summed.

If this number has not yet been reached, then the new script request can be started. If the number has been reached, then the script request to be started will still have status `waiting`.

The available “capacity”, this means the number of script requests that can still be started, can be retrieved in the log file of the batchscheduler (`$OZG_LOG/<batch account>_<sid>_<host>.log`) when it runs in *verbose* mode.

If no script requests are created in the *Polling interval* period, and if therefore no new script requests have to be started, the batchscheduler will proceed with activity 2; checking the ongoing script requests.

2 - Checking the ongoing script requests

Regarding the started processes (= "ongoing script requests") 2 matters are checked:

1. Has the script request been started within the margin?
2. Is a script request with status "Running" (ongoing) indeed still active?

Re 1. Has the script request been started within the margin?

If the batchscheduler observes a script request in table `ALG_SCRIPT_AANVRAGEN` waiting to be started, then the status will have the value `W` (`waiting`).

If the request has to be started, then the batchscheduler will change the status to `Start` and the process will be started on the OS.

The data of the started process can be found in the log file of the batchscheduler if it runs in *verbose* mode.

An example of a script request process of the type Oracle Reports

```
11:07:10: nohup nice -10 OZG_CMD.sh rwoclient p_id=59384
recursive_load=no blankpages=no p_user=SCOTT batch=yes
server=rep_myhost report=$OZG_BASE/report/ZRG1234R
desname=$OZG_OUT/scott/59384.out mode=character nonblocksql=no
destype=file envid=$ORACLE_SID
userid=OZG_BATCH/OZG_BATCH@$ORACLE_SID >$OZG_LOG/scott/59384.log
2>&1 & echo $!
```

The script *itself* will (by means of the generic startup code) set the status of the relevant script request to `Running`.

If the status transition `Start->Running` has not occurred within the configured period (batchscheduler setting *Start delay*), the "Bijzonderheden" (Details) section of the script request will display the following message "*job niet gestart binnen marge*" (job not started within the margin). The status of the script request will then be set to "Failed".

Normally, the script will be started though: the process has been started already, but it simply has not reached the startup code yet within the specified "*start delay*".

Once the script is ready it will then set the status of the relevant script request to `Ready` or `Error` (= functional error).

The possibly occurring **ERROR** message “*job not started within the margin*” will then be changed into an **INFO** message.

What are the possible causes of a script not starting in time?

1. **Start delay too low**

This setting has been configured so low that the server cannot activate the script fast enough.

2. **Server capacity insufficient**

The server is loaded so heavily that it takes long before a script is activated.

3. **Insufficient Oracle Reports Server resources**

There are insufficient available resources in Oracle Reports to start a script of the type Oracle Reports; the report will be on hold until these Oracle Reports resources become available.

Re 2. Is a script request with status "running" indeed still active?

If the script request has status `Running`, then there is a check on the OS or in the database (depending on the type of script) to ensure that a process is indeed still active for this script request.

If it turns out that this process is no longer active, then the script request is considered as `Failed`; the process has not been finalized correctly (given that the end code in which the status is set to `ready` or `error` has not been executed).

After performing the above checks (= activity 2 – *Checking the ongoing script requests*) the batchscheduler will proceed with activity 1 (*Starting new script requests*), etc.

Oracle Reports

The scripts started by the batchscheduler can be of different types. These types include for example *SQL module*, *OS shell script* and *Oracle Reports*.

The behavior of scripts of the latter type can be influenced by the settings for *Oracle Reports*, a component of *Oracle Application Server* (in case of OHI Back Office this refers to *Forms & Reports Services Standalone version*).

Within the settings for Oracle Reports a distinction has to be made between the settings for the *Reports Server* and the settings for the *Reports Engines* (processes that actually handle the reports).

Functionality

What takes place in Oracle Reports when a script of this type (= a *report*) is requested?

By means of the batchscheduler, the previously mentioned (Oracle Reports) `rwclient` process is started.

The Reports Server will now determine if a Reports Engine is active in order to handle the process.

If an Engine is active, this will be used to run the report.

If *no* Engine is active, 1 will be started.

An Engine is a java process (for which any relevant details can be set up by means of parameter `jvmOptions`), which can run *1 report at the same time*, but with several reports after each other; the reports will then remain to use *the same* Engine (= java process).

Given that as starting an Engine overhead has a price, it could be profitable to have 1 Engine process multiple reports sequentially (*recycling* the java process). The number of reports that can be handled by an Engine is set up by means of parameter `engLife`. After this value has been obtained, the Engine (and therefore the relevant java process)

will be stopped. The following reports will then have to be handled by a different (existing or new) Engine.

By means of parameter `keepConnection`, which must be set for OHI Back Office to "no", it is indicated that for each report a new (database) connection has to be created; the various reports in 1 Engine therefore make use of different database sessions.

The maximum number of connections permitted for Engines and reports is configured by means of the parameter `maxConnect`.

If this value is configured too low, this will lead to message "REP-56055: Exceed max connections allowed: ...". This can happen based on the other settings (`maxEngines` & `keepConnection`) which may result in having more reports active than connections.

The number of Engines started when starting the Oracle Reports Server (*prespawning*) is configured by means of parameter `initEngine`.

The number of Engines that has to remain active at least is configured by means of parameter `minEngine`.

If an Engine has been idle longer than the configured parameter `maxIdle` allowed, it will stop automatically.

The maximum number of Engines is configured by means of parameter `maxEngine`.

URLs

The following URLs display various data regarding Reports Server, Reports Engines and the Reports Server queue status:

<http://<host>:<port>/reports/rwservlet/help>

<http://<host>:<port>/reports/rwservlet/help>

<http://<host>:<port>/reports/rwservlet/showjobs>

<http://<host>:<port>/reports/rwservlet/getserverinfo>

Sizing

Of importance for the configuration of `minEngine` and `maxEngine` is that Oracle Reports Server *strives* to meet these values; *in case insufficient resources (CPU & RAM) are available, then the values cannot be obtained*.

It is therefore of importance for the DBA to determine the real values for their system.

Also for the (`batchscheduler`) setting *Max. parallel script requests* the DBA has to determine what their system can (and is allowed) to handle.

The value for "*Max. parallel scriptaanvragen*" (*max. parallel script requests*) is strongly related to `maxEngine` & `maxConnect`.

If the parallel script requests are all of the type 'Oracle Reports' then this number of Engines & connections could be required to handle reports.

This number has to be greater than *Max. parallel script requests*; the reason for which is that a Reports process can still be active while writing the (very large) output file, while the status of the matching script request could be set to `Ready` (or `Error`) already.

The script request will no longer be considered as a "lopende scriptaanvraag" (running script request), while a process is still active though (during a short period of time); therefore it is (temporarily) required to have more connections than the allowed number of parallel script requests.

Reports Server Parameters

In file `$ORACLE_HOME/reports/conf/<Reports Server>.conf` a number of settings have to be adjusted for this.

For a number of these settings, the maximum number of parallel script requests is used, as can be configured in the system parameters of the OHI Back Office sub-

system (*Max. parallel script requests*).

"*Max. parallelle scriptaanvragen*" (max. parallel script requests) has to be configured based on the capacity of the relevant server; a guideline is (2 to 4)*(the number of available CPU cores on the server). Settings that are too *high* can lead to an overload of the server.

Validation of the configuration file

Once the following modifications have been made in the configuration file, without *Dynamic Environment Switching* (this concept is explained later in this chapter), it is possible to use the following commands based on the corresponding `.dtd` file to validate whether or not the syntax of the configuration file is (still) correct:

```
$ . ozg_init.env $OZG_ORATAB_FRS10G2
$ cp $ORACLE_HOME/reports/conf/<naam reports server>.conf $TMP/r.xml
$ export CLASSPATH=$ORACLE_HOME/lib/xmlparserv2.jar:$CLASSPATH
$ oraxml -dtd $TMP/r.xml
```

=> this should not lead to any errors.

If modifications are *also* made with respect to *Dynamic Environment Switching* then it is possible to use the following additional command to validate if the XML syntax of the configuration file is (still) correct:

```
$ oraxml -novalidate $TMP/r.xml
```

=> this should not lead to any errors.

cacheSize

Property `cacheSize` has to be set to at least 3x the size of the largest *Reports* output file.

The value is indicated in MB.

Example:

The largest Reports output file is 100MB; `cacheSize` has to be 300MB

```
<property name="cacheSize" value="300"/>
```

maxEngine

For the tag `engine id="rwEng"` parameter `maxEngine` has to be set to (the number of available CPU cores on the server) * (6 to 8 = recommended by Oracle for Unix/Linux platforms).

The value has to be at least identical to "*Max. parallelle scriptaanvragen*" (max. number of parallel script requests).

Example:

On a server with 8 CPU cores: $8 * 6 = 48$

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl"
initEngine="1" maxEngine="48" minEngine="0" engLife="50"
```

engLife

For the same tag, parameter `engLife` has to be set to 4-30.

The default value of 50 (this means 1 Reports Engine, a java process, can process 50 reports sequentially before the process is started again) could lead to memory leaks.

initEngine & minEngine

For the same tag, parameter `initEngine` and `minEngine` have to be configured based on the actual requirements; the settings depend on the load and the available capacity. It is recommended to choose *at least* 2-4.

Attention `initEngine` has to be greater than `minEngine`.

See also [Tuning Oracle*Reports](#).

Example:

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl"
initEngine="2" maxEngine="48" minEngine="1" engLife="50"
```

keepConnection

For this same tag, property `keepConnection` has to be set to `no` (to this end the comment symbols for property have to be deleted!).

Example:

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl"
initEngine="1" maxEngine="12" minEngine="0" engLife="50" maxIdle="30"
callbackTimeOut="60000" defaultEnvId="default">
<!--property name="sourceDir" value="your reports source directory"/-->
<!--property name="tempDir" value="your reports temp directory"/-->
<property name="keepConnection" value="no"/>
```

maxConnect

For tag `connection` parameter `maxConnect` has to be configured to *10x "Max. parallele scriptaanvragen" (max. parallel script requests)*.

Example:

```
ALG_SYSTEEM_PARAMETERS.BATCH_MAX_PROCESSEN = 24
<connection maxConnect="240" idleTimeOut="15">
```

defaultEnvId

For tag `engine id="rwEng"` parameter `defaultEnvId` has to be set to `"default"` for *Dynamic Environment Switching* (see the next paragraph).

Example:

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl"
initEngine="1" maxIdle="30" callbackTimeOut="60000"
defaultEnvId="default">
```

Dynamic Environment Switching

For each environment that has to be serviced by this Reports Server, an `environment` tag has to be included which specifies the `REPORTS_PATH` (incl. path for printer definitions) and which configures `REPORTS_DEFAULT_DISPLAY` (YES for OHI Back Office, NO for the default installation for the demo reports).

Additionally, a default `environment` tag has to be included in order to run the standard settings of 10g AS (e.g. to be able to run the demo reports).

```
<environment id="default">
<envVariable name="REPORTS_PATH"
value="$ORACLE_HOME/reports/templates:$ORACLE_HOME/reports/samples/dem
```

```
o:$ORACLE_HOME/reports/integ:$ORACLE_HOME/reports/printers"/>
<envVariable name="REPORTS_DEFAULT_DISPLAY" value="NO"/>
</environment>
<environment id="<ORACLE_SID>">
<envVariable name="REPORTS_PATH"
value="<OZG_ROOT/ORACLE_SID/report:$ORACLE_HOME/reports/printers"/>
<envVariable name="REPORTS_DEFAULT_DISPLAY" value="YES"/>
</environment>
```

Example:

```
<environment id="default">
<envVariable name="REPORTS_PATH"
value="$ORACLE_HOME/reports/templates:$ORACLE_HOME/reports/samples/dem
o:$ORACLE_HOME/reports/integ:$ORACLE_HOME/reports/printers"/>
<envVariable name="REPORTS_DEFAULT_DISPLAY" value="NO"/>
</environment>
<environment id="prod">
<envVariable name="REPORTS_PATH"
value="$OZG_ROOT/prod/report:$ORACLE_HOME/reports/printers"/>
<envVariable name="REPORTS_DEFAULT_DISPLAY" value="YES"/>
</environment>

<environment id="test">
<envVariable name="REPORTS_PATH"
value="$OZG_ROOT/test/report:$ORACLE_HOME/reports/printers"/>
<envVariable name="REPORTS_DEFAULT_DISPLAY" value="YES"/>
</environment>
```

Configure Reports.sh File

In file \$ORACLE_HOME/bin/reports.sh the lines containing settings of the following variables have to be commented:

1. TNS_ADMIN
2. REPORTS_PATH
3. NLS_LANG

Additionally (i.e. setting these variables is a **mandatory** step when Application Server version 10.1.2.3. or higher is used), set following variables:

4. ORACLE_HOME
5. REPORTS_DEFAULT_DISPLAY (to remove the requirement of having to set the DISPLAY environment variable)

Example:

...

```
ORACLE_HOME=/u01/app/oracle/product/frs10g2; export ORACLE_HOME

#TNS_ADMIN=$ORACLE_HOME/network/admin; export TNS_ADMIN
## TWO_TASK=< two task name >; export TWO_TASK
## ORACLE_SID=< ORACLE SID >; export ORACLE_SID

## setting for Reports Runtime
RW=$ORACLE_HOME/reports; export RW
#REPORTS_PATH=$ORACLE_HOME/reports/templates:$ORACLE_HOME/reports/samp
les/demo:$ORACLE_HOME/reports/integ:$ORACLE_HOME/reports/printers;
export REPORTS_PATH
...
#NLS_LANG=AMERICAN_AMERICA.WE8ISO8859P15; export NLS_LANG
```

```
...  
REPORTS_DEFAULT_DISPLAY=YES; export REPORTS_DEFAULT_DISPLAY
```



Attention: On AIX 5.3 it is possible that the Reports Server cannot be started due to the settings of variable `MALLOCTYPE` in file `reports.sh`. See My Oracle Support Note 298175.1. The workaround consists of commenting the variable.

MOVE IN-PROCESS REPORTS SERVER TO STANDALONE

Now change the default in-process Reports Server to a standalone server with the same name. This report server is already registered with OPMN.

To do this adjust the `$ORACLE_HOME/reports/conf/rwservlet.properties` file and change `SERVER_IN_PROCESS=YES` to `SERVER_IN_PROCESS=NO`.

Restart all the services (using `opmnctl`) to activate the changes in the configuration. If something is not correct, please see [Standalone Reports Server](#).



Attention: For more details on How to move the in-process report server to standalone See My Oracle Support note [361498.1](#).

STANDALONE REPORTS SERVER

If for some reason the method described above does not give the expected result (occasionally this can happen), the alternative below describes how you can change the in-process Reports Server to a standalone Reports Server.

Steps are:

1. Make sure your environment variables point to forms & reports server `ORACLE_HOME`.
2. Make sure all components from Forms & Reports Server are stopped, including the ias console.
3. Copy `$ORACLE_HOME/sysman/emd/targets.xml` to `targets.xml.org`.
4. Modify `$ORACLE_HOME/sysman/emd/targets.xml`, remove all directives from the in-process reports server.
5. Modify `$ORACLE_HOME/reports/conf/rwservlet.properties`; change `SERVER_IN_PROCESS=YES` to `SERVER_IN_PROCESS=NO` and put `SERVER=<name_in-process_rep_server>` on comment.
6. Move `$ORACLE_HOME/reports/conf/<name_in-process_rep_server>.conf` to `/<name_in-process_rep_server>.conf.org`
7. Start all components with `opmnctl`.
8. Start ias console
9. Log on to the ias console, the reports server component must not be seen.
10. Stop ias console and all components with `opmnctl`.

11. Create a new reports server from the command prompt by use of;
 1. Make sure \$ORACLE_HOME points to Forms & Reports Server environment.
 2. Change your current directory to \$ORACLE_HOME/bin
 3. Execute `./rwserver.sh server=<name_reports_server> batch=yes`, a new reports server configuration file is build and reports server is started.
 4. Stop the reports server. Sometimes it happens that in the previous step the command prompt is not give back and it seems the session hangs. Just press ctrl-c and search for rwserver processes. If there are, stop them.
12. Registrare the created reports server at OPMN;
 1. Your environment and current directory should be the same as in step 10.
 2. Execute `./addNewServerTarget.sh <name_reports_server>`, an entry should be created in opmn.xml
 3. An entry is also added to targets.xml, so the new component should be seen on the ias console.
 4. Start all components with opmnctl.
 5. Start ias console
 6. Check if the newly registered reports server is started through opmnctl and it should be seen by use of the ias console.
 7. Try to stop and start the reports server by use of the ias console and/or opmnctl. This should work.
13. Configure the reports server as described [earlier](#) in this document.
14. Restart all the services (using `opmnctl`) to activate the changes in the configuration.



Attention: For more details on How to move the in-process report server to standalone
See My Oracle Support note 276810.1 and 260367.1

CONFIGURE ORACLE HTTP SERVER

In a standard Oracle Application Server installation the Oracle HTTP Server is installed and configured.

The following settings have to be modified in the configuration file for OHI Back Office. Afterwards the HTTP Server has to be restarted.

Virtual Directories

In configuration file `$ORACLE_HOME/Apache/Apache/conf/httpd.conf`, the Virtual Directory Mapping `/OHI/` has to be implemented to support the physical mapping to directory `$OZG_ROOT`.

This Virtual Directory Mapping can subsequently be used in the following settings:

- `java_progressimage` in `ozg_web.cfg`

- path for `ozg_gif.jar` file

Sample settings:

```
Alias /OHI/ "/u01/app/oracle/product/OHI/"
```

Viewing the Release Documentation

In order to allow for an interactive (via screen “Raadplegen release informatie” (consult release information)) view of the release documentation, the following code has to be added in `httpd.conf`:

```
<Directory "/u01/app/oracle/product/OHI/patch/2???..???..???/doc/">
    Options Indexes MultiViews
    AllowOverride None
    Order deny,allow
    Allow from all
</Directory>
```

This code ensures that the `/doc` directory of OHI (patch) releases (always in format 2999.99.9.9999) can be viewed via a browser.

Other directories are deactivated.

RESTARTING 10G AS

After implementation of the changes, the 10g AS software has to be restarted in order to activate these changes.

CONFIGURATION OZG_WEB.CFG

Configuration file `$OZG_ADMIN/ozg_web.cfg` now has to be configured for the environment-specific settings.

The standard settings have to be copied from sample file `ozg_web.cfg` in Oracle iProjects.

Customer and/or environment-specific entries have to be set up depending on the actual situation.



Attention: If a newer version of Oracle Jinitiator or Sun’s Plugin is to be used, then the relevant Jinitiator and/or JPI tags in file `ozg_web.cfg` have to be changed.

The current version of Oracle Jinitiator can be found in file `jinit-version.ini` (in the Application Server software tree).

CONFIGURATION JINITIATOR

Along with 10g AS R2 Oracle Jinitiator 1.3.1.28 is provided; OHI recommends, however, to use version 1.3.1.22oldfonts, given that *multiline* items are not displayed in an adequately readable form.

1.3.1.22oldfonts can be downloaded from iProjects, and has to be moved to directory `$ORACLE_HOME/jinit`.

Afterwards, configure the relevant `jinit*` variables in `ozg_web.cfg` to 1.3.1.22oldfonts (see template in iProjects/Beehive).

CHECK PORT CONFLICTS

If 10g R2 DBMS and AS are installed on the same machine, it is possible that Oracle Notification Server of 10g AS R2 and 10g DBMS R2 will use the same port; this leads to a conflict, which can be observed by messages such as “Local connection ...

missing/invalid form factor” in 10g AS R2

`$ORACLE_HOME/opmn/logs/ons.log` file.

In that case the following steps have to be performed:

1. Stop the 10g DBMS R2 Notification Server by means of 10g DBMS R2
`$ORACLE_HOME/bin/onsctl stop.`
2. Now modify the 10g DBMS R2 file `$ORACLE_HOME/opmn/conf/ons.config` and change the localport and remoteport for port numbers that are not used yet.
3. Start the 10g DBMS R2 Notification Server by means of 10g DBMS R2
`$ORACLE_HOME/bin/onsctl start.`
4. Restart the database and the 10g DBMS R2 database listener.

See also the following My Oracle Support note:



297411.1 : OPMN Uses a High Amount of CPU