

**Oracle[®] Retail Design
Operations Guide
Release 12.0
May 2006**

Copyright © 2006, Oracle. All rights reserved.

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

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

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

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

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

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

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

Contents

Preface	v
Who This Guide is Written For	v
Where You Can Find More Information	v
Customer Support	vi
1 Introduction	1
Functional and Technical Capabilities	1
Technical Architecture Overview	3
Oracle Retail Design Potential Integration Points	4
2 Minimum Requirements	5
Internet Connection Requirements Common to Both PC and Macintosh Systems	5
Mail Server Requirements	5
Requirements for PC Systems	6
Requirements for Macintosh systems	6
Browser Support	7
Recommended	7
3 Backend System Administration	9
Logging and Exception Handling	9
Location of Logs Diagram	10
Exception Handling and General Troubleshooting	11
A Word About a System Monitor	11
Starting and Stopping the System Order of Operation	12
Stopping the System	12
RetailServer Properties Documentation	12
db.properties	12
mailserver.properties	13
mailtestfile	14
main.properties	14
security.properties	15
warp.properties	15
rcapps.properties	15
Integrator.properties	16
license.bin	16
Troubleshooting	16
4 Integration Interface Dataflows	21
Overview	21
System-to-System Dataflow	21
From Oracle Retail Design to Oracle Retail WebTrack	22
From an External system to Oracle Retail Design	23
From Oracle Retail Design to an External System	23
5 How to Build a Test System	25
Refresh Process	25
6 Running Oracle Retail Integrator Batch Processes	27
Batch Process Architectural Overview	27
Import and Export File-Based Batch Processes	27
Import Processing Description and Diagram	28
Export Processing Description and Diagram	29

Running a Batch Process	31
Executable Korn Shell Script.....	31
Scheduler and the Command Line	31
Summary of Command Line Parameters	31
Functional Descriptions and Dependencies	33
Import Batch Processes.....	33
Export Batch Processes.....	35
General Batch Processes.....	38
Validation Processing	39
Summary of Batch Logging.....	39
Key GUI Field Descriptions Including Batch Return Values	48

A Appendix: DTD Files Used in Oracle Retail Integrator Processing51

Import Batch Processes.....	51
enterprisepartner.dtd used in PartnerImport.....	51
techspec.dtd used in plxImport	52
parameters.xsd	53
Export Batch Processes.....	59
style.dtd used in XMLItemExtract and in DesignXMLExtract	59
styleids.dtd used in DesignActiveStyleExport and in DesignActiveStyles	65

This operations guide serves as a Oracle Retail Design reference to explain ‘backend’ processes. The guide is designed so that you can view and understand key system administered functions, the flow of data into and out of the application, and the application’s behind-the-scenes processing of data.

Who This Guide is Written For

Anyone who has an interest in better understanding the inner workings of the Oracle Retail Design system can find valuable information in this guide. There are three audiences in general for whom this guide is written:

- System analysts and system operation personnel who:
 - Are looking for information about the Oracle Retail Design processes internally or in relation to the systems across the enterprise.
 - Operate Oracle Retail Design on a regular basis.
- Integrators and implementation staff who have the overall responsibility for implementing Oracle Retail Design into their enterprise.
- Business analysts who are looking for information about processes and interfaces to validate the support for business scenarios within Oracle Retail Design and other systems across the enterprise.

Where You Can Find More Information

This guide does not show you how to use the front-end of Oracle Retail Design. Rather, the focus here is on how data is managed, how it flows, and how it is processed.

If you wish to find further information, see the following applicable Oracle Retail documents:

- Oracle Retail Design Release Notes
- Oracle Retail Design User Guide
- Oracle Retail Design Online Help
- Oracle Retail Design Configuration Guide
- Oracle Retail WebTrack Configuration Guide
- Oracle Retail WebTrack Release Notes
- Oracle Retail WebTrack User Guide
- Oracle Retail WebTrack Online Help
- Oracle Retail Retail Server Installation Guide
- Oracle Retail Retail Server Data Model
- Oracle Retail Integrator User Guide
- Oracle Retail Integrator Online Help

Customer Support

- <https://metalink.oracle.com>

When contacting Customer Support, please provide:

- Product version and program/module name.
- Functional and technical description of the problem (include business impact).
- Detailed step-by-step instructions to recreate.
- Exact error message received.
- Screen shots of each step you take.

Introduction

Oracle Retail Design is a web-based, collaborative product development solution that enables clients to organize and manage their product development and sourcing process. As products are being developed, changes based on fashion trends, sample results, raw material costs, and other factors are an inevitable part of the process. As these changes occur, they must be communicated throughout the internal and external supply chain community. Because trading partners such as third-party agents, manufacturing suppliers, and raw materials vendors are located throughout the world, communication is challenging, and reaction times can be slow. For many clients, there is very little automation to support this process.

Oracle Retail Design brings the members of the supply chain to a single, secure place on the web for the most current product information. Product details (including initial product sketches and images, detailed specifications, and costing and supply data) are collected in one central database from season to season.

Functional and Technical Capabilities

The functional design and technical architecture of Oracle Retail Design results in the following capabilities:

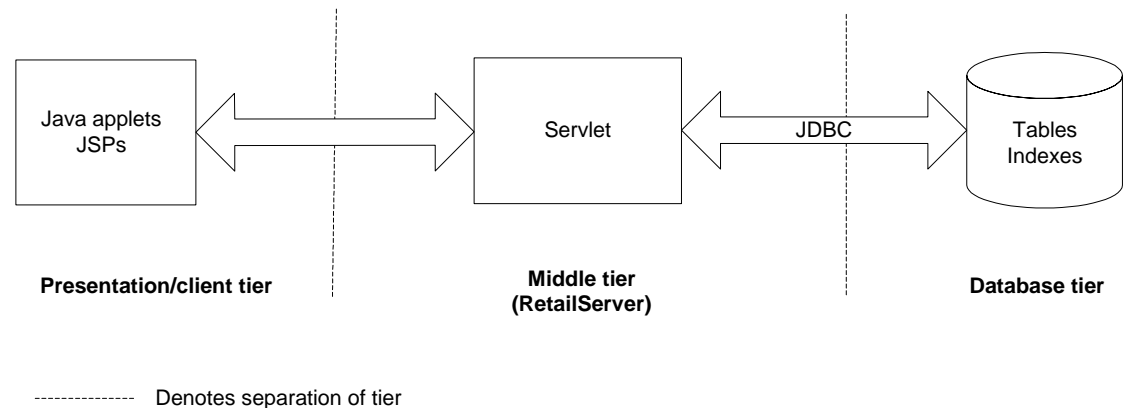
- The web-based collaborative architecture enables everyone involved in the product development and sourcing process to have secure access to the same information and to progressively develop the complete details of each new product.
- Oracle Retail Design recognizes that various types of products are developed differently. Configurable forms for various product types provide an efficient and flexible process to communicate and represent all product development details specific to the unique product type.
- Image management and annotation consolidate the visual representation of the product and the product notes in one place. This consolidation improves communication and leads to a greater overall understanding of the product. Improved communication surrounding the product concept reduces sample iterations and improves accuracy.
- The application's document upload capability allows retailers to attach or import multiple types of output from other sources, facilitating the collection and communication of product details in one place.
- Agent and vendor assignment capabilities support multiple sourcing scenarios and allow competitive, secure closed-bidding scenarios among multiple suppliers. This functionality also facilitates communication with key sourcing organizations early in the product development process.
- The application's flexible configuration enables product costs to be collected by component to calculate a total product cost. Early visibility to costs improves buying trip efficiency and automated visibility to overall landed cost. Sourcing decisions are improved, and the need for additional spreadsheets and/or manual, complex calculations is eliminated.

- Mass change functionality enables product changes to be made to multiple products during one edit process. Improved efficiency in the process makes product changes easier to update and manage, improving the overall accuracy and timeliness of the data.
- Oracle Retail Design provides the ability to automatically create and update projects within Oracle Retail WebTrack based on the product information being developed within Oracle Retail Design. In addition, the Oracle Retail Design user is able to navigate directly to the supporting product file's project or track within Oracle Retail WebTrack. This functionality allows the user to make updates to the event dates and completions while developing the product details.
- Data extract utilities allow for exports to a .csv file or a standard .xml extract. Data is therefore more accessible, and reporting can be performed on the product development process. If the system did not contain this functionality, and the information were stored only in email and spreadsheets, reporting on product development metrics would not be feasible.
- Because the system utilizes some features of the Java 2 Enterprise Edition (J2EE) architecture, clients have a choice in their selection of databases and application servers.
- Market-proven and industry-standard application programming interfaces (API) are utilized (that is, RMI, JDBC, and so on).
- Java applications such as Oracle Retail Design have enhanced portability which means the clients are not 'locked' into a single platform. Upgrades are easier to implement, and hardware is easier to change.

Technical Architecture Overview

The three-tiered architecture of Oracle Retail Design allows for the separation of presentation, business logic, and data, making the software cleaner and easier to maintain.

The diagram and brief explanation below offer a high-level conceptual view of the tiers. For a detailed description of this diagram, see Chapter 4, “Technical Architecture.”



Technical Architecture Overview

To facilitate the functional demands and complexity of a web-based application, the presentation tier facilitates robust client-side processing. Oracle Retail Design provides a ‘fat’ client with Java applets, which enable a portion of the application’s logic to be built into the client. JSPs serve as a place from which the application’s Java applets can launch. No logic resides within the system’s JSPs, except for that which facilitates utility processing (such as logging in, logging out, authentication, and so on). A browser is all that is need to access the application, and when a browser views a page that contains an applet, the applet’s code is transferred to the client and executed by the browser’s Java Virtual Machine (JVM).

The middle tier’s primary responsibility is to service the requests of the client (that is, to fetch data, to store data, and so on). The client, after all, has no direct connection to the database or its tables and must rely upon the middle tier to accommodate its requests. The middle tier (in which the server resides) is comprised primarily of a single Java servlet called retailserver. The Java applets in the client tier communicate with retailserver, which, in turn, communicates with the database on behalf of the client. The retailserver is comprised of a number of different modules. Once the modules are loaded by the servlet, each is responsible for performing specific functional tasks. Specific system functionality sometimes requires the use of only a subset of these modules. Sub-modules, called add-ons, lie within the modules to help facilitate the system’s integration functionality.

The database tier is the application’s storage platform, containing the physical data (user and system) used throughout the application. The database’s tables house data which is retrieved by the middle tier and then passed to the client. Oracle Retail Design supports two database products, Oracle and DB2.

Oracle Retail Design Potential Integration Points

The high-level diagram below shows the overall direction of the data among systems and products across the enterprise. For a detailed description of this diagram, see Chapter 5, “Integration Interface Dataflows.”

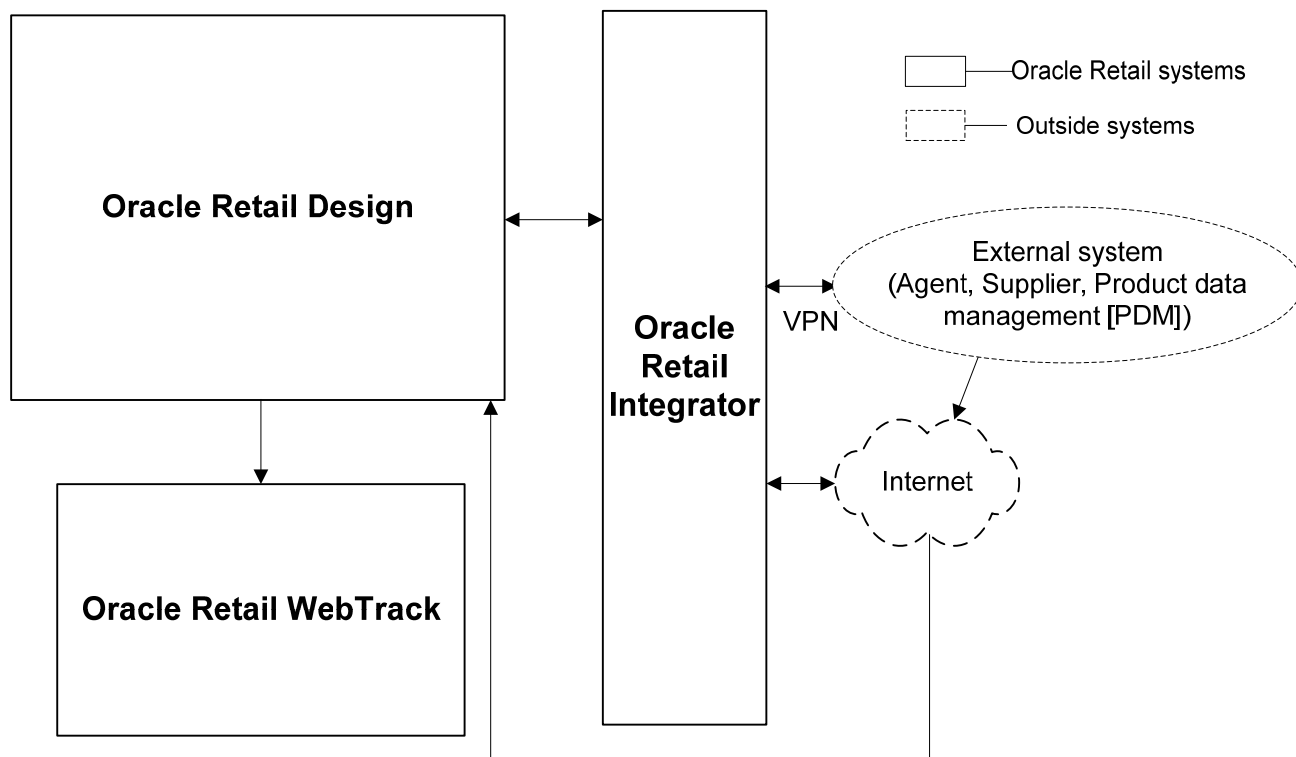
Oracle Retail Design does *not* have to be integrated in order to provide a client with functionality because all of its data can also be populated through the front-end. The diagram illustrates the potential ways an external system can access the Oracle Retail systems. In production, multiple trading partners could exist, and their interfaces to the system would depend upon their individual circumstances.

When a product enters a certain stage, a project within Oracle Retail WebTrack can automatically be created using approved product details data (such as color/short description data, department name data, and so on) collected in Oracle Retail Design.

An external system can send product images and technical specification documents to Oracle Retail Design and attach them to specific product files within Oracle Retail Design.

Oracle Retail Design can send technical specification data and product files (including season details data and active styles data) to an external system, ensuring that all external systems share synchronized information.

Note: Oracle Retail WebTrack, Oracle Retail Design, and Oracle Retail Integrator share the same schema.



Oracle Retail Design-related dataflow across the enterprise

Minimum Requirements

This chapter describes the following:

- The minimum requirements necessary to run Oracle Retail Design.
- What constitutes a supported browser.

Note: Products associated with mydomain.com run on PC or Macintosh systems using the minimum configuration requirements listed below. However, better response times are achieved using the recommended configurations. Note that running Web-based applications stresses connectivity more than Web browsing.

Internet Connection Requirements Common to Both PC and Macintosh Systems

System component	Minimum requirements	Recommended requirements
Internet connection	56K Internet connection	DSL/T1 or cable modem internet connection
Proxy servers	Oracle Retail advises its clients to ensure that their servers are sufficiently well configured to support the number of users (that is, CPU speed, memory, network connectivity).	

Mail Server Requirements

Oracle Retail Design must use a simple mail transfer protocol (SMTP) mail server configured to operate over Transmission Control Protocol (TCP) port 25. SMTP is the internet's standard host-to-host mail transport protocol. Most UNIX systems include sendmail, a widely-used SMTP server for email.

Requirements for PC Systems

System component	Minimum requirements	Recommended requirements
Hardware	Pentium 233Mhz processor 128Mb RAM	Pentium 1.1Ghz processor 512Mb RAM
Desktop resolution	1024 x 768 pixels	1024 x 768 pixels
Operating system	MS Windows 98SE or Windows NT 4.0 SP 3 plus Sun Java 1.5	MS Windows 2000 or Windows XP with the latest service pack plus Sun Java 1.5.0
Software	MS Internet Explorer 5.01 SP3 or above – downloadable from http://www.microsoft.com/ie Also, see the section, 'Browser Support,' later in this chapter.	MS Internet Explorer 6.0 with the latest service pack – downloadable from http://www.microsoft.com/ie Also, see the section, 'Browser Support,' later in this chapter.

Requirements for Macintosh systems

System component	Minimum requirements	Recommended requirements
Hardware	400 MHz processor 128Mb RAM	800 MHz processor 256Mb RAM
Desktop resolution	1024 x 768 pixels	1024 x 768 pixels
Operating system	Panther Mac OS X v10.3.1 plus Java 1.4.2	Tiger Mac OS X v10.4 plus Java 1.4.2
Software	Safari 2.0 downloadable from http://www.apple.com/safari/download/ Also, see the section, 'Browser Support,' later in this chapter.	Safari 2.0 downloadable from http://www.apple.com/safari/download/ Also, see the section, 'Browser Support,' later in this chapter.

Browser Support

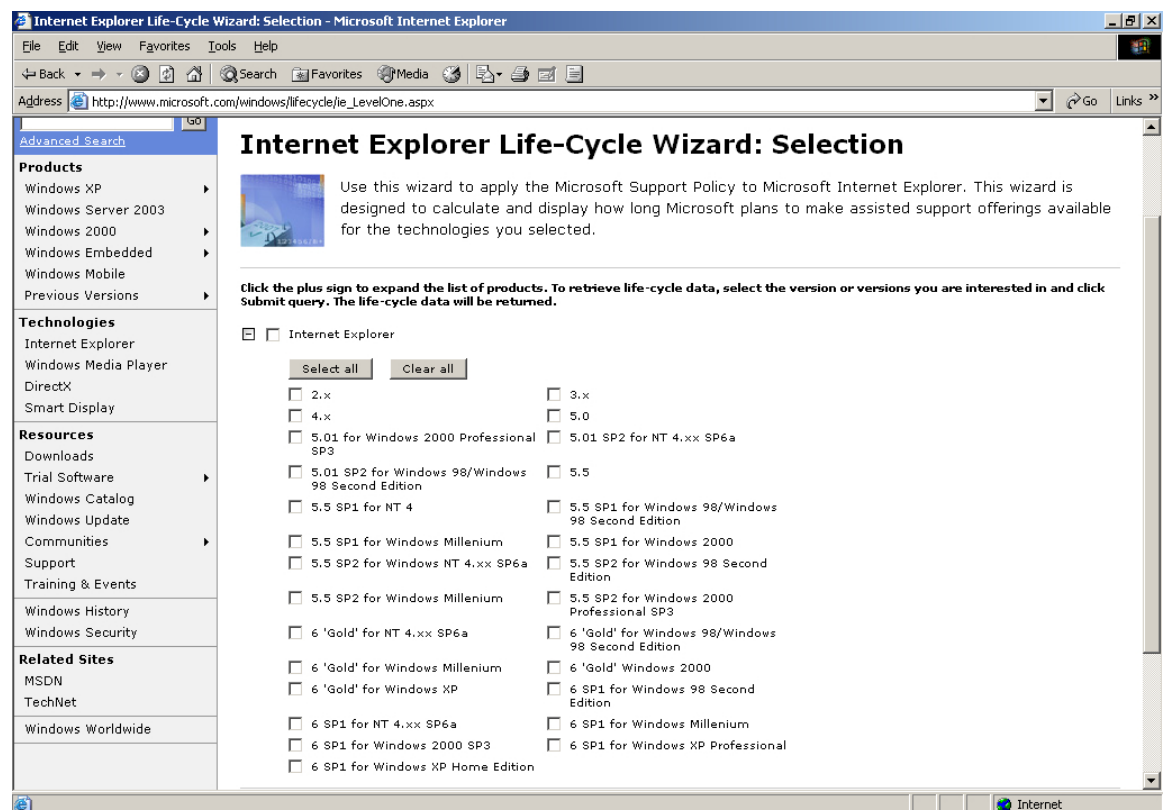
At a minimum, a Windows browser must support Java 1.1 to run the application. For Macintosh users, the browser must support Java 1.3.1 to run the application.

Recommended

Oracle Retail recommends browsers currently supported by Microsoft on the PC and by Apple on the Macintosh.

The Internet Explorer support lifecycle information can be found at the following internet location (shown also in the screen capture below to offer context in the event that the site changes):

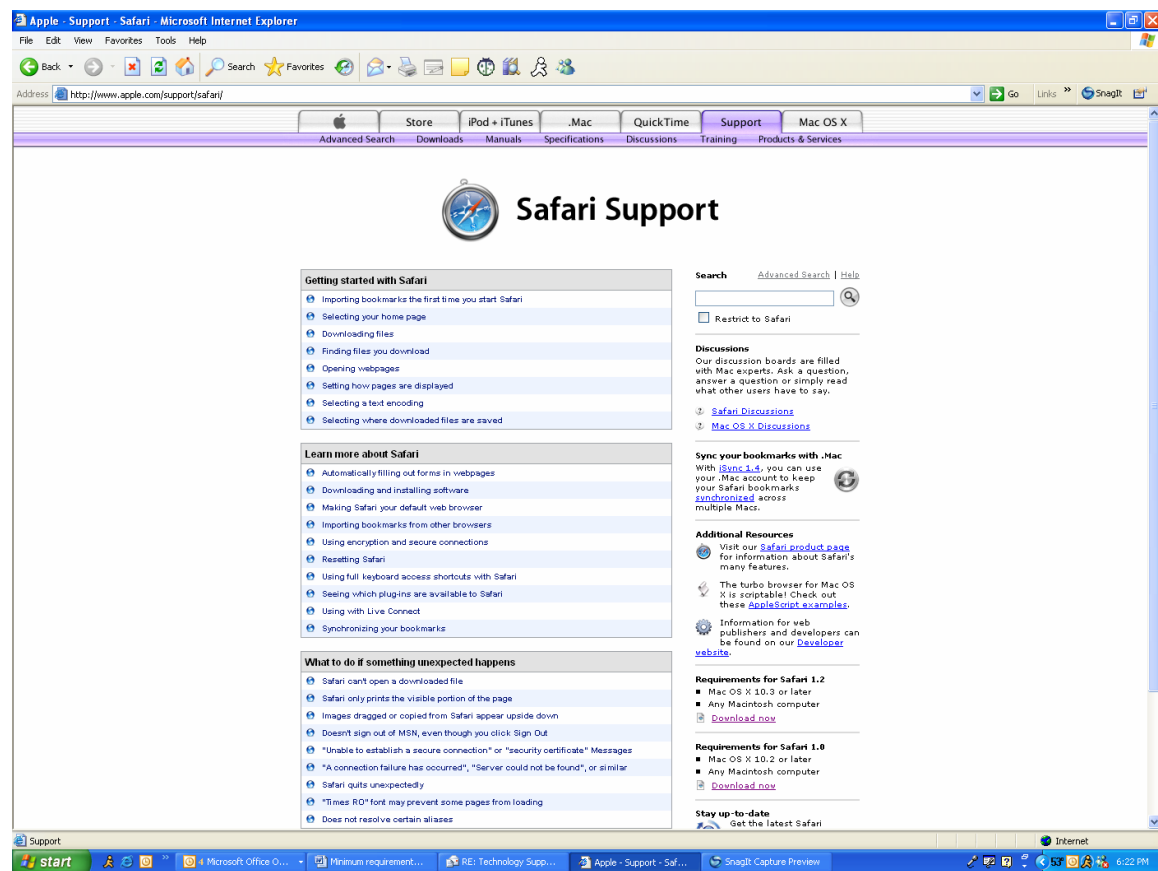
http://www.microsoft.com/windows/lifecycle/ie_LevelOne.aspx



Microsoft web page related to browser support data

The Safari support lifecycle information can be found at the following internet location (shown also in the screen capture below to offer context in the event that the site changes):

<http://www.apple.com/support/safari/>



Apple web page related to browser support data

Backend System Administration

This chapter of the operations guide is intended for database administrators who provide database support and monitor the running system.

The content in this chapter is not procedural but is meant to provide descriptive overviews of key system parameters. The troubleshooting section identifies situations that might arise and their remedies.

To find further system administration information including those values which can be defined through the graphical user interface (GUI), see the Oracle Retail Server Administration Guide.

Logging and Exception Handling

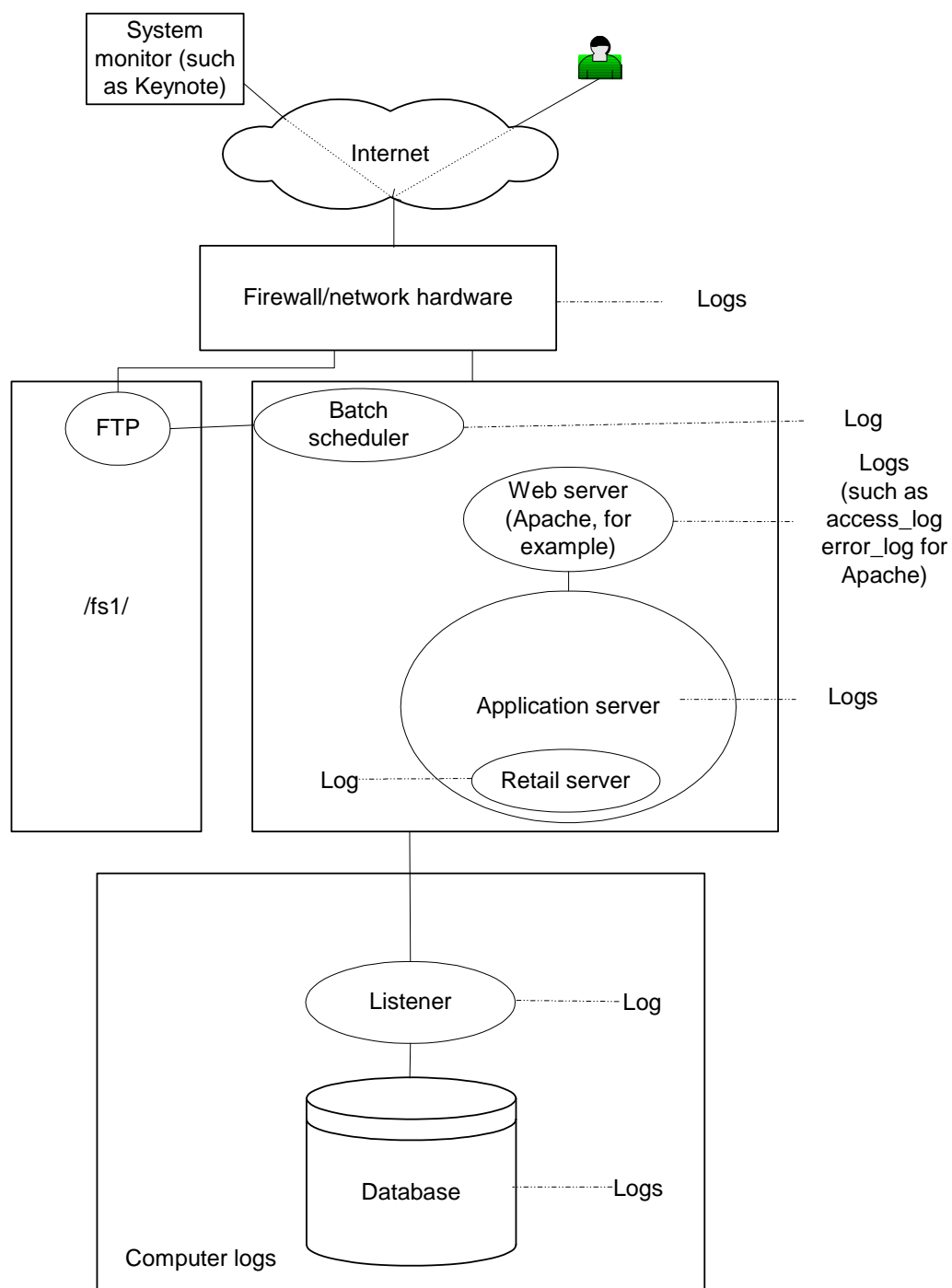
Log files to be monitored exist across a number of locations throughout the system.

Oracle Retail recommends that the client purchase a system monitoring tool (such as Patrol, Unicenter, and so on). Such a tool can usually be set up to send emails or pages when certain events occur in the system.

The diagram below illustrates the location of the system's log files (listed below from the bottom of the diagram up):

- Computer logs run by a system administrator
- Database logs (for Oracle or DB2)
- Listener-related log
- Retail server code log for the code running on the application server
- Application server logs (for WebSphere, BEA WebLogic, Oracle Application Server, and so on)
That is, when WebSphere is being used, these are located in
`/opt/WebSphere/AppServer/logs`
- Apache logs
That is, when Apache is bundled with WebSphere, these logs are located in the following directory:
`/opt/IBMHttpServer/logs`
- Batch processing-related logs
See Chapter 7, "Running Oracle Retail Integrator batch processes" for specific information and paths related to batch logging.
- Firewall and network hardware-related logs
- System as a whole logs (not shown on diagram)

Location of Logs Diagram



Logging locations throughout the system to be monitored

Exception Handling and General Troubleshooting

Note: For specific troubleshooting suggestions, see the section, 'Troubleshooting,' later in this chapter.

When an exception arises that must be investigated, the log files listed above and shown on the diagram should be searched for errors. Errors can indicate a variety of problems, such as the following:

- The database is full or the database is having other problems.
- A filesystem is full.
- The operating system log file indicates a hardware, network file system (NFS), or other operating system error.
- The database listener is down.
- The system is having domain name system (DNS) or name resolution problems.
- There is a missing 'complete' file to initiate a batchjob.
- There is invalid xml.
- A worm overloads the firewall.
- The firewall timeouts.

A Word About a System Monitor

Oracle Retail recommends the use of a third party service (such as Keynote) that inspects the site from many different locations (sixty, that is) around the world. Such third party system monitors usually have machines scattered in different places around the world, and these machines access websites and measure their response. When the third party system monitor encounters an issue, it contacts the administrator through a page or other method.

Starting and Stopping the System Order of Operation

The following order of operation should be followed when starting the system (after, that is, application maintenance, and so on):

1. Start the applicable database network listener.
2. Start the database (such as DB2, Oracle, and so on).
3. Start the application server (such as WebSphere, and so on). Note that Apache is always started in conjunction with the application server because the two are connected.

Stopping the System

Note: A graceful shutdown of the system is always preferred. However, in the event that this method does not succeed, an administrator can force an aborted shutdown.

The following order of operation should be followed when stopping the system (for application maintenance, that is):

1. Shut down the application server. Note that Apache is always stopped in conjunction with the application server because the two are connected.
2. Shut down the database.
3. Shut down the applicable database network listener.

RetailServer Properties Documentation

A system administrator defines configurations for Oracle Retail Design in a number of properties files. Most key configurable system parameters contained in these files are described in this chapter. That is, when clients build the code to their environment, they must update many of these values to their specific settings. Parameters within the system's properties files that are *not* changed (except perhaps under exceptional circumstances) are *not* described in this chapter.

Note that within a property file (and thus in some of the examples from that file below), a # sign that precedes a value in the properties file signifies that what follows is a comment and is not being utilized as a setting.

db.properties

This file contains configuration values related to the system's database.

- common.prop.db

This value defines the database the system is utilizing. This value is set to one of the following:

- oracle
- db2

Oracle Parameters

- `common.prop.oracle.sid`
This value defines the database name the system is utilizing. SID stands for system identifier.
- `common.prop.oracle.host`
This value refers to the database listener. This value defines the 'host:port' that the database listener is utilizing.
- `common.prop.user`
This value defines the 'username/password' of the database.

DB2 Parameters

- `common.prop.db2.db`
This value defines the database name the system is utilizing.
- `common.prop.db2.host`
This value refers to the database listener. This value defines the 'host:port' that the database is utilizing.
- `common.prop.user`
This value defines the 'username/password' of the database.

mailserver.properties

This file contains configuration values related to the system's use of email. The system uses these settings, that is, when sending emails related to batch process status (successful, failed, and so on).

- `common.prop.mailfrom`
This value defines the sender of email sent by Oracle Retail Design or Oracle Retail WebTrack. When a user views the email, this value appears in the 'From' field.
- `mailhost`
This value defines the server's name or IP address that Oracle Retail Design or Oracle Retail WebTrack uses as a SMTP mail server.
- `mailtest`
This value is used to redirect mail to a deadend account rather than to the intended recipient. This setting is designed as a precaution, so that users do not get confused by bogus email from a test system. The following table explains potential values and their results:

Value	Result
<code>#mailtest=deadend@mydomain.com</code>	Because the value is commented out, all mail works.
<code>mailtest=deadend@mydomain.com</code>	All mail is redirected to the deadend account. Nobody receives mail.

Value	Result
mailtest=deadend@mydomain.com,+@mydomain.com	All mail is sent to the deadend address <i>and</i> any mail going to within mydomain.com is sent to the correct addresses. For testing purposes, copies of all the mail being sent are thus available for inspection at the deadend account.
mailtest=deadend@mydomain.com,+john.doe@oracle.com	All mail is sent to the deadend address <i>and</i> any mail going to john.doe@oracle is sent to the correct address.

mailtestfile

By establishing the parameter, the client designates a custom alternate properties file that contains the mailtest value. This setting is used so that changes can be made ‘on the fly.’ If this parameter is used, changes to values do *not* require that the server be bounced. Note that a mailtest value in the yourfilename.properties file overrides any value established in the mail.properties file. If applicable, this value is set to the following:

mailtestfile=yourfilename.properties

main.properties

The system’s module definitions are provided in this file. Each module definition consists of a class name and any addon submodules that the module may need. Some modules may not be loaded, depending upon licensing. For an understanding of the role that modules and addons play within the application, see Chapter 4, “Technical Architecture”.

Other important settings within this file include the following:

- **base**
Because this value must correspond to the context root setup of where the application was installed, this setting should *not* be changed in this file once Oracle Retail Design has been implemented. That is, this value could be one of the following:
 - ‘/’ for a production system
 - ‘/test1’ for the first of several test systems on a single physical computer
- **jvmlogfile**
Path and logfile name in which to log information about the jvm and browser used for each connection to the system.
- **jvmlogformat**
Sets additional formatting for the entries in the jvmlogfile. The format was ‘inspired’ by the logging formats in Apache. “%a” is the client IP address and “%t” is time (date/time). The remaining values are standard Java system properties on the client.
 - The default is %a [%t] "%{java.vendor}" "%{java.version}" "%{os.name}" "%{os.version}" "%{os.arch}"
 - A common entry is: jvmlogformat+= "%{enterprisename}" "%{userfullname}"

- **securemode**
This value determines whether the system forces connections to switch from http (non-secure) to https (secure) upon logon. That is, this value can be one of the following:
- '1' forces connections to switch from http (non-secure) to https (secure) upon logon.
- '0' does *not* force connections to switch from http (non-secure) to https (secure) upon logon.

security.properties

- **trackeradmin.prop.adminhosts**
This file value contains the list of IP addresses that are allowed to log in to the exchange administration console. The list is a comma-separated list of address or address/mask values. If no values are entered, all hosts can connect to the administration console (*not recommended*).

warp.properties

Every night, the application sends email. The values in this file ensure that only one web server sends out email. Note that changes made to this setting *require* a bounce.

Note: Only one of the web servers should be instructed to process the mail run.

- **monitor**
A value of '1' instructs the web server to process the midnight mail run. A value of '0' instructs the web server *not* to process the mail run.

rcapps.properties

Values in this file change the GUI. This file thus allows you an easy way to override certain aspects of the GUI. That is, if you would like a testing system to have a different look than the production system, these settings could be used.

Colors at www.yourdomain.com are defined by the values below. Some values are in hexadecimal (Hex), which represent red, green, blue (RGB) values but in a form that browsers can interpret. Hex is a notation which mixes letters and numbers and allows three digit numbers to be expressed as two digit numbers. That is, a green background would be (0, 255, 0) in RGB, and 00FF00 in Hex. The # sign preceeding the value signifies that the value is in Hex.

- **appbanner.bg**
Main banner backgroup color (default is #E13128)
- **appmenu.bg**
Menu banner background (default is #F9F8C7)
- **apphdr.bg**
Header background (default is #E13128)
- **securemode.allow**
Set to "*" to enable securemode on all clients, or "*, !Mac" to enable securemode on all clients except those using a Macintosh. Secure mode must be disabled on Mac OS X systems if you are using a test SSL certificate because the 1.3.1 Java VM does not function without a valid host certificate.

- `text.fg`
Main test color (default is #E13128)
- `webmeter.allow`
The webmeter in the GUI provides an approximate visual indication of the quality of the network connection to the server. Set to “*” to display webmeter for all clients, or “* , !Mac” to display webmeter for all clients except those using a Macintosh. The webmeter is disabled for Macintosh systems because it can cause issues with JDK 1.3.1 on the Mac OS X system.

Integrator.properties

- `maxfilesize`
This value instructs the system to only accept import XML files below a given number in megabytes (that is, 128 megabytes).

license.bin

This file defines the applications that a client has licensed. Oracle Retail supplies this file when the client receives the system. Should a client choose to add a new application to the license, this file would have to be updated. Once updated, the new application would become available upon a system restart.

Troubleshooting

Issue

I am uploading a file through Oracle Retail Integrator, but system says the file is too big.

Response

Adjust the ‘maxfilesize’ parameter in `integrator.properties`. See the ‘Integrator.properties’ passage earlier in this chapter.

Issue

I am receiving a Remote Method Invocation (RMI) error in a batch process.

Response

Determine whether any properties refer to ‘localhost’. If so, ‘localhost’ must be resolvable by the system. For UNIX, add a record ‘127.0.0.1 localhost’ to `/etc/hosts`. Also, the web server (such as Apache) must listen on ‘localhost.’ This action is accomplished by adding the directives ‘Listen 80’ and ‘Listen 443’ to the conf file (that is, ‘httpd.conf’). The conf file is the main parameter file for Apache. See the Oracle Retail Server Installation Guide for more information about conf file settings for the application.

Issue

When I browse to `http://www.mydomain.com`, the login page is visible. However, when I browse to `https://www.mydomain.com`, the following error appears: ‘This page cannot be displayed.’ In addition, one or both of the following occurs:

- An error in the Apache log file says ‘handshake failed, invalid date’.
- When logging in, a certificate warning appears.

Response:

Your secure socket layer (SSL) protocol certificate may have expired. Renew it and install the new certificate from VeriSign. For more information about SSL, see your application server documentation and/or Apache documentation.

Issue

When I browse to `http://www.mydomain.com`, the login page is visible. However, when I browse to `https://www.mydomain.com`, the following error appears: ‘This page cannot be displayed.’ However, *neither* of the following occurs:

- An error in the Apache log file says ‘handshake failed, invalid date.’
- When logging in, a certificate warning appears.

Response:

The web server does not have the uniform resource locator (URL) and port added as a virtual host. That is, you may need to configure the virtual host to listen for ‘* : 443’ or ‘www.mydomain.com : 443’ or ‘a.b.c.d : 443’ depending upon which of these URLs you would like to use to access the system.

Issue

When I browse to `http://www.mydomain.com`, strange characters appear on the screen.

Response

An incorrect configuration in the Apache conf file (`httpd.conf`) may be causing Apache to use the SSL (https) protocol on port 80, which should *not* use SSL. Ensure that the `SSLEnable` directive only appears inside the ‘<VirtualHost :443>’ directive.

Issue

Note: This issue is addressed with the assumption that WebSphere is the application server that you are using.

When I perform a ‘`ps -ef | grep -i WebpShere`’, two processes are shown running rather than the usual one. Is something wrong?

Response

There is probably nothing wrong. You or someone else may have the admin console open. By closing the admin processes and then performing a ‘`ps -ef | grep -i WebpShere`’, the normal number of processes appears.

Issue

When I browse to `http://www.mydomain.com`, I see ‘Internal Server Error 500’ or ‘my custom error document.’ A similar error also appears on the Apache ‘CustomLog’ file.

Response

For reasons that cannot always be ascertained, the Apache front-end can sometimes lose its connectivity to the rest of the application server. The quickest way to restore the system is to bounce the application server. If this issue occurs frequently, examine all system log files and try to identify any errors that may coincide with the instability. Reevaluate the patch level of all software including the operating system. Also, consider a cold boot of all hardware to re-initialize the entire system. See the section, ‘Starting and Stopping the System Order of Operation,’ earlier in this chapter.

Issue

Errors occur related to the reading of data that is stored on an NFS filesystem shared by more than one application server.

Response

For reasons that cannot always be ascertained, NFS filesystems can sometimes become corrupt. This corruption is often accompanied by an NFS error in the operating system log files. In addition, sometimes the same directory, when viewed from different servers, displays inconsistent data. The quickest way to restore the system may be to unmount and remount the NFS filesystems. If this issue occurs frequently, examine all system log files and try to identify any errors that may coincide with the instability. Reevaluate the patch level of all software including the operating system. Also, consider a cold boot of all hardware to re-initialize the entire system. See the section, ‘Starting and Stopping the System Order of Operation,’ earlier in this chapter.

Issue

The following exception is displayed soon after I log in: ‘SQL exception...’.

Response

The wrong database and/or the incorrect machine name could be specified in the `db.properties` file. To resolve the issue, correct the entries in the `db.properties` file, and stop and restart the application server (such as WebSphere). See the sections, ‘Starting and Stopping the System Order of Operation’ and ‘`db.properties`’ earlier in this chapter.

Issue

The following message is displayed soon after I log in: ‘Virtual Host not defined ... unable to connect using port 443’.

Response

Note: This issue is addressed with the assumption that WebSphere is the application server that you are using.

In WebSphere’s admin GUI, select Environment->Virtual Hosts->default Host->New:

1. Enter Port = 443 and Host= *
2. Apply and save the changes you have made.
3. Stop and restart WebSphere.

Issue

First, I log into mydomain.com and click on the WebTrack. I then select the track option from the next window. My issue is that the tracks window takes a very long time to be displayed.

Response

*If the user does **not** need to create tracks, the user can be defined as 'maintenance only,' so that the tracks button loads more quickly. Thus, to reduce the access times, follow these steps:*

1. Log in as the admin for the user.
2. Click **webtrak** and select user.
3. Click the **Users** tab and edit the applicable user.
4. Click the **Permissions** tab and select **WebTrack**.
5. Under available types, select '**track maintenance only**' and click the right arrow. 'Track maintenance only' is now one of the selected types.
6. Log in as the user modified above and access the tracks. Access times should be significantly reduced.

Integration Interface Dataflows

This chapter provides an overview of how Oracle Retail Design can integrate on a functional level with other systems.

Overview

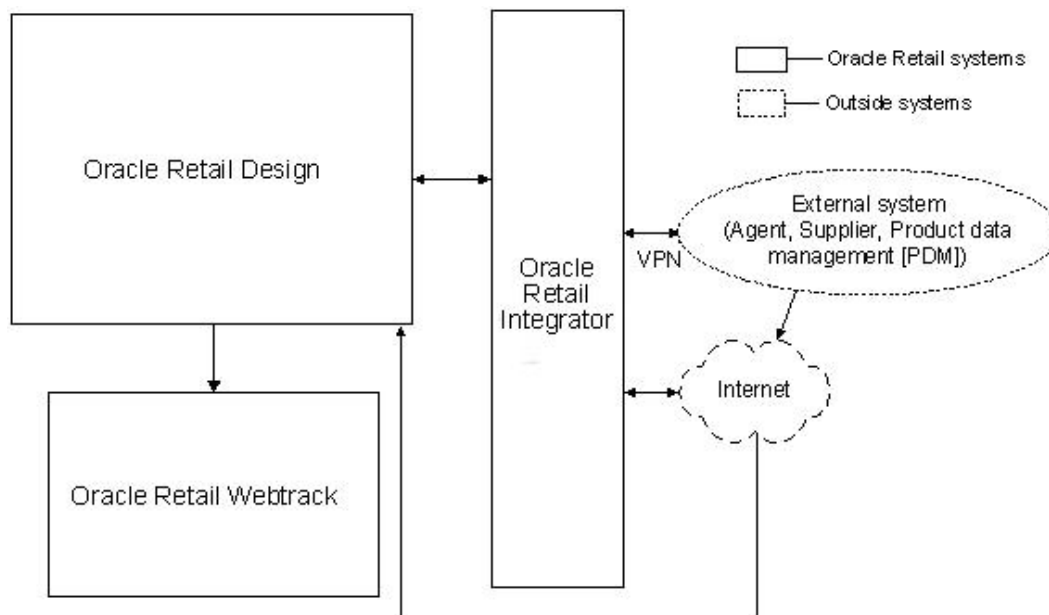
This chapter provides you with a diagram illustrating the systems that Oracle Retail Design interfaces with as well as the overall dataflow among the products. The chapter is intended for those clients who utilize the application's integration capabilities. Oracle Retail Design does *not* have to be integrated in order to provide a client with functionality because all of its data can be also populated through the front-end.

The accompanying explanations are written from a system-to-system perspective, illustrating the movement of data. Note that this discussion focuses on the functional use of data; the means of data movement (that is, batch processing) and other sites associated with those means (that is, the File Transfer Protocol [FTP] server) are not illustrated in this chapter.

Note: Oracle Retail WebTrack, Oracle Retail Design, and Oracle Retail Integrator share the same schema.

System-to-System Dataflow

Note: The diagram illustrates the potential ways an external system can access the Oracle Retail systems. In production, multiple trading partners could exist, and their interfaces to the system would depend upon their individual circumstances.



Oracle Retail Design dataflow across the enterprise

From Oracle Retail Design to Oracle Retail WebTrack

- **Approved product details for automatic project creation data**
When a product enters a certain stage, a project within Oracle Retail WebTrack can automatically be created using approved program details data collected in Oracle Retail Design. Once Oracle Retail WebTrack creates a project, the client can create a track from the project that includes the data originating in Oracle Retail Design. Approved product details can include the following:
- **Color/short description data**
Oracle Retail Design supports products (a shirt, that is). The color data represents the color of the product, and the short description data depicts the product (that is, silk, short-sleeved camp shirt). Oracle Retail WebTrack uses this data as its project name. This data is mandatory. The project integration between the two systems supports tracking by color or style.
- **Department name data**
Oracle Retail WebTrack uses this data as the department name. This data is mandatory.
- **Order required by date data**
This date is calculated within Oracle Retail Design based upon the initial availability date – transit time (freight on board [FOB] lead days). This data is mandatory.
- **Design identifier data**
Oracle Retail WebTrack uses this data as its project number. Note that because the same Oracle Retail Design ID may exist, the project number may not be unique.
- **Sell price multiplied by the quantity (per color) data**
Oracle Retail WebTrack uses this data element as the value.
- **Comments (created from the product file by the Oracle Retail Design user) data**
These creation comments identify the Oracle Retail Design user who triggered the creation of the project. Oracle Retail WebTrack uses this data as comments.
- **Unique style ID data**
Oracle Retail WebTrack uses this data as part of its attributes. Note that these fields are not visible to the user within Oracle Retail WebTrack until a track has been created. This data creates a unique relationship between data and allows the Oracle Retail WebTrack project to be linked to the specific product/color file in Oracle Retail Design. Once a project is created using data from Oracle Retail Design, the project key identifying the new Oracle Retail WebTrack project is referenced in the Oracle Retail Design database.
- **Supplier data**
Oracle Retail WebTrack uses this data as parts of its attributes. Note that these fields are not visible to the user within Oracle Retail WebTrack until a track has been created.

From an External system to Oracle Retail Design

Note: The DTD files associated with this dataflow are illustrated in the Appendix of this document, “DTD files used in Oracle Retail Integrator processing”.

- **Technical specification data**
Product images, technical specification documents, and a supporting XML file are sent to Oracle Retail Design. The detailed contents of the XML file determine how product images and technical specification documents are attached to specific product files within Oracle Retail Design.

From Oracle Retail Design to an External System

Note: The DTD files associated with this dataflow are illustrated in the Appendix of this document, “DTD files used in Oracle Retail Integrator processing”.

- **Technical specification and product file data**
Technical specification documents, a copy of the product file within Oracle Retail Design and a supporting XML file are sent from Oracle Retail Design to an external system.
- **Product files–season details data**
Oracle Retail Design generates an XML file for a trading partner that contains details about each product file. A season or set of seasons can be defined within the export’s parameters to determine which product files should be selected. The enterprise administrator can also configure the process to pick up all product files that have changed since a specific date or since the last time the export process has been run.
- **Product files–active styles data**
Oracle Retail Design generates an XML file containing details about all active product files that exist within Oracle Retail Design based on the parameter selection configuration. This extract uses the same season selection parameter as the season details extract described above and ensures that all external systems can be synchronized to include only active product files.

How to Build a Test System

This chapter provides a high-level illustration of how to create a test system using a production system as a source. That is, a production system called `yourdomain.com` could be reproduced as a test system called `test.yourdomain.com`. A client might wish to build a test system for a number of reasons, including the purchase of a new computer, the testing of new code, training, and so on.

Refresh Process

The following steps illustrate the refresh process: that is, how to copy a production system and build a test system. A system includes filesystem data, the schema within the database, and the J2EE application code (held within an `.ear` file).

Note: To ensure consistent data between the production system and the test system, copy the filesystem data near the time that the database is copied.

1. Set up the hardware for the test system.
Procure separate hardware, such as the disk, database, and application server, for the test system that is similar to or identical to the production system.
2. Create an empty database that is similar to the one in production.
The test system should have equivalent tablespace structure, but many of the values in the parameter file for the database (memory, sort space, and so on) should be scaled down because the load will most likely not be equivalent to production.
3. Install the application server in the test system (including the Apache front-end piece) in the same way that it was installed for production. See the Oracle Retail Server Installation Guide for more information. Note that after a fresh installation of the application server, adjustments to values may have to be made in the following places (though for the most part they should retain the values of the production system):
 - Ports
 - Mime-types
 - The Apache plug-in
 - The Apache conf file
 - The start script (that is, `startServer.sh` for WebSphere)
 - The JAVA libraries
4. Install the production J2EE ‘war’ file on the test system application server. This production J2EE ‘war’ file is available on the Oracle Retail-packaged CD-ROM or on the production system.
5. Browse to `test.mydomain.com` and utilize the Oracle Retail Private Exchange Setup utility. Type in applicable data (properties files settings, and so on) in the utility about the test system. Bounce the server code (by stopping and starting the server).

Note: You now have an empty system. You must now stop the application server and replace the database and filesystem with a copy of production.

6. Copy the production database schema using a database dump file. Oracle clients can note the following example command:

```
exp userid=system/manager owner=\\(retailserver\\) log=exp.log consistent=y  
file=design.dmp
```

See your database vendor's applicable documentation for more information.

7. Remove all objects from the database schema. Replace them with the exported contents of production. Oracle clients can note the following example command:

```
imp userid=system/manager fromuser=\\(retailserver\\) touser=\\(retailserver\\)  
log=imp.log file=retail.dmp
```

8. Remove the data directory (that is, /retailserver/data). Replace it with a copy of the system. One may already exist in your backup software. If not, a tar command may be used. That is:

```
tar -cvf retail_data.tar /retailserver/data
```

9. Extract the filesystem copy into the test system with a command. That is,

```
tar -xvfp retail_data.tar
```

10. Start up the application server.

11. Test a logon by accessing www.test.mydomain.com and entering your username and password.

12. Change the super user password so that it is different than it is for the production system if you wish enhanced additional security. To make this change, follow these steps:

- a. Browse to <http://www.yourdomain.com/phantasm>.
- b. Log on.
- c. Click **Users**.
- d. Select **Super Administrator**.
- e. Enter the new password in both boxes.
- f. Click **Save**.

13. Change all user passwords so that they are different than they are for the production system if you wish enhanced additional security. To make this change, follow these steps:

- a. Connect to the database.
- b. Enter the following SQL statement:

```
UPDATE ent_users SET password = 'test';  
commit;
```

Running Oracle Retail Integrator Batch Processes

This chapter provides the following:

- An overview of the batch architecture
- A description of how to run batch processes
- A list of the command line parameters used in batch processing
- A functional summary of each batch process, along with its dependencies
- A description of some of the features of the batch processes (batch logging, return values in the GUI, and so on)
- A list of other documentation related to configuring and running Oracle Retail Integrator batch processes

Batch Process Architectural Overview

Oracle Retail Integrator is the mechanism for sharing data between Oracle Retail Design and other systems through multiple channels (primarily FTP) using XML.

Note the following characteristics of the application's batch processes:

- They are run in Java.
- They are scheduled by the client.
- Although key parameters are established through the application's GUI, batch processes must be run through the command line, not the GUI.
- They are designed to process large volumes of data.
- Oracle Retail recommends that they be executed during 'off-hours' (that is, during a time when users are not in the system such as nights).

Import and Export File-Based Batch Processes

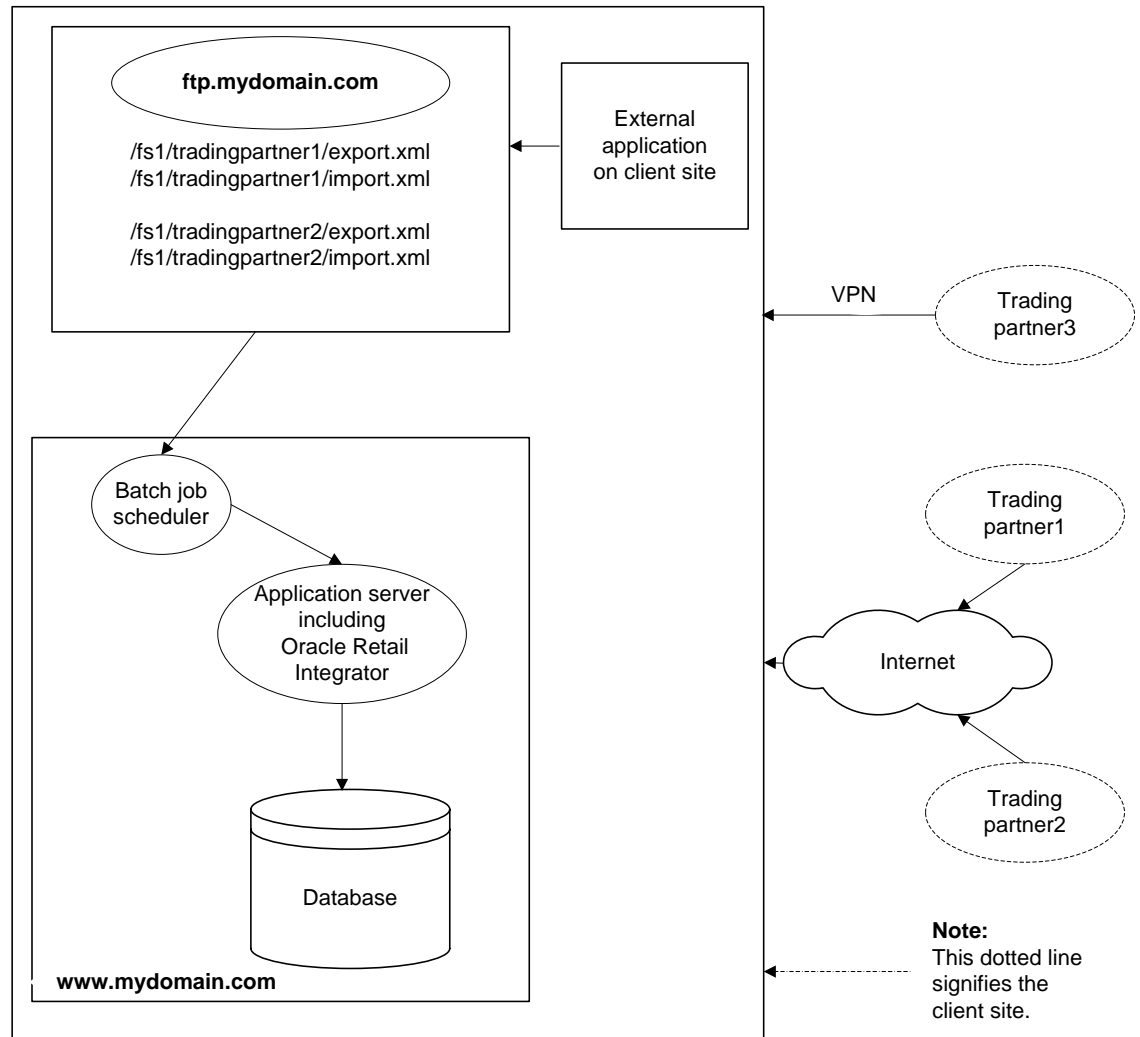
Although Oracle Retail Integrator supports a variety of functional processes, the system is responsible for two primary mechanisms, import and export. These import and export mechanisms process a variety of XML formats. The steps involved in each mechanism are consistently applied regardless of the functional process or application supported.

Import Processing Description and Diagram

This description and the diagram that follows it provide a brief overview to the import process.

FTP is used as a communication channel for sharing files between systems and Oracle Retail Integrator. The external system (that is, an external application on the client site, a trading partner, and so on) connects in as a user to access the FTP site. The trading partner then downloads the file for the client site to the FTP directory (that is, /fs1/trading partner1/import.xml, where fs1 stands for file system 1). Note that a trading partner *not* on the client's site uses the internet or a VPN encryption-based connection.

Once Oracle Retail Integrator receives the file, the file is validated and stored as a data set before it is processed. The XML file is then translated for internal processing between Oracle Retail Integrator and the applicable application (such as Oracle Retail Design). The XML file is parsed and individual records of data are shared with the applicable application to validate and perform defined updates. The applicable application processes the data and returns the status (including success, failure, or processed with errors). Once all processing is complete, Oracle Retail Integrator sends the applicable email notifications and stores the results within Oracle Retail Integrator for viewing through the GUI or the log file.



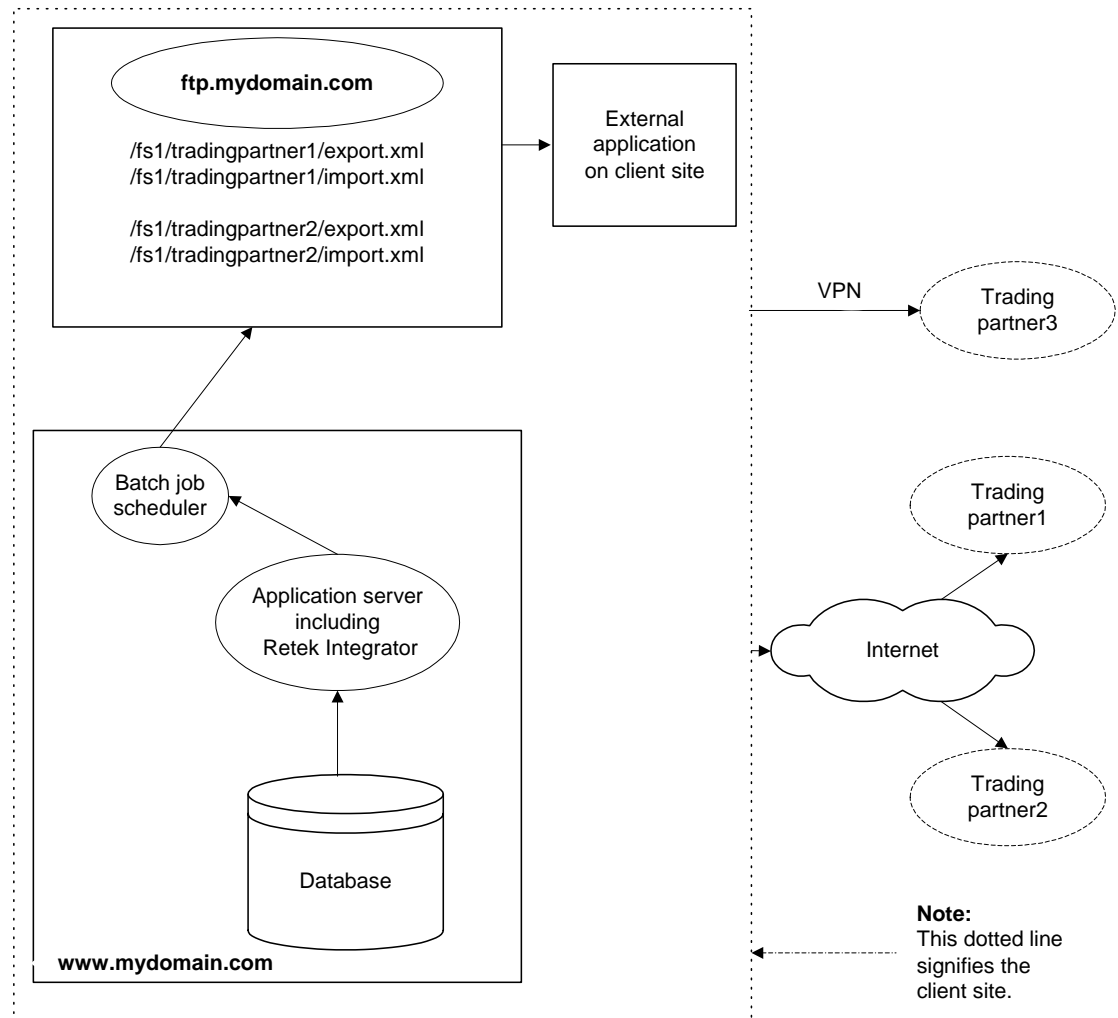
Batch import flow overview

Export Processing Description and Diagram

This description and the diagram that follows it provide a brief overview to the export process.

A scheduled batch process triggers the export process to begin processing. Based on a set of rules, application-specific processes determine the data to be included within the specific export process. When the applicable batch process runs, a file is extracted from the database. Oracle Retail Integrator is called for processing. Oracle Retail Integrator generates the supporting XML files, stores the entire data set contents in the form of an XML file, and generates the applicable success or failure responses.

A second 'companion' batch job runs placing the file into an FTP directory such as /fs1/trading partner1/export.xml, where fs1 stands for file system 1. The trading partner is the company with whom the client is collaborating. The trading partner has a computer that uses the internet or a VPN encryption-based connection to access the FTP site and connect in as a trading partner user. The trading partner can then perform whatever tasks it chooses (printing, uploading, and so on).



Batch export flow overview

Running a Batch Process

Executable Korn Shell Script

Java processes are scheduled through a hard-coded executable Korn shell script. This .ksh file is called `cronexe.ksh`. Oracle Retail provides this Korn shell script during installation (located in the 'configuration' directory you specified in the setup page). The script performs the following internally:

- Sets up the Java runtime environment variables before the Java process is run.
- Triggers the Java batch process in conjunction with the command line parameters that follow it.

Scheduler and the Command Line

If the client uses a scheduler, arguments are placed into the scheduler.

If the client does not use a scheduler, arguments must be passed in at the UNIX command line.

The installation creates `cronexe.ksh` *without* the execute permission. To execute `cronexe.ksh`, you must either grant the UNIX execute permission (that is, `chmod u+x cronexe.ksh`), or preface the call to `cronexe.ksh` with 'ksh' (that is, `ksh cronexe.ksh..`).

That is,

```
/yourpathtconfigdir/cronexe.ksh java com.retail.integrator.BatchStep0 -i 9999 -c
BatchReport -r DesignActiveStyles
1>/yourpathtodatadir/ftp/9999/export/ActiveStyleBatchStep0.out
```

Summary of Command Line Parameters

The following is a list of command line parameters for each of the application's batch processes. The commands 'run' a batch job.

In the command line parameters shown below, the top line specifies the class (that is, `BatchStep0`) and the arguments determining which batch process runs. The line beneath that beginning with `1>` specifies the directory location of the output file that accompanies the batch process. When a script is run and the batch process does *not* proceed to the point where Oracle Retail Integrator is involved, the system directs errors (which are probably significant) to the designated output file. The output file is a logical place to begin searching for an issue when you are troubleshooting batch process run problems. The location of the output file can be configured according to the client's needs.

The `-i` value (9999 in the example below) represents the client's enterprise database unique ID. Preexisting Oracle Retail clients cannot change this value.

Batch Process	Type	Command Line Parameters
Active Styles Export	Export	<pre>/yourpathtconfigdir/cronexe.ksh java com.retail.integrator.BatchStep0 -i 9999 -c BatchReport -r DesignActiveStyles 1>/yourpathtodatadir/ftp/9999/export/ActiveStyleBatchStep0 .out</pre>

Batch Process	Type	Command Line Parameters
Active Styles FTP	Export	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.FTPStep0 -i 9999 -d exportNoVal -r DesignActiveStyleExport 1>/yourpathtodatadir/ftp/9999/export/DesignActiveStyleExport.out
Season Details Export	Export	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.BatchStep0 -i 9999 -c BatchReport -r XMLItemExtract 1>/yourpathtodatadir/ftp/9999/export/XMLBatchStep0.out
Season Details FTP	Export	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.FTPStep0 -i 9999 -d exportNoVal -r DesignXMLExtract 1>/yourpathtodatadir/ftp/9999/export/DesignXMLExtract.out
Technical Specification Export	Export	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.RDOExportStep0 -e 9999 -f plxAutoExLog -rt plxBatchRDOExport 1>/yourpathtodatadir/ftp/9999/export/plxAutoExport.out
Technical Specification FTP	Export	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.FTPStep0 -i 9999 -d exportNoVal -r plxTechSpecExport 1>/yourpathtodatadir/ftp/9999/export/plxExport.out
Technical Specification Import	Import	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.FTPStep0 -i 9999 -d import -r plxImport 1>/yourpathtodatadir/ftp/9999/import/plxImport.out
Partner Import	General	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.FTPStep -I 9999 -d import -r PartnerImport 1>/yourpathtodatadir/ftp/9999/import/PartnerFTPStep0.out 2>&1
Purge	General	//yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.IntegratorPurgeStep0 -e all -pt all -ft all -ld 1 1>/yourpathtoconfigdir/purgeStep0_regular.out 2>&1
Shared File Purge	General	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.IntegratorPurgeStep0 -e 9999 -pt all -ft plxSharedTSpdf -ld 1 1>/yourpathtoconfigdir/purgeStep0_shared.out 2>&1
DesignParameterImport	Import	/yourpathtoconfigdir/cronexe.ksh java com.retail.integrator.IntegratorFTPStep0 -e 9999 -d import -r DesignParameterImport 1>/yourpathtoconfigdir/DesignParameterImport.out 2>&1

Functional Descriptions and Dependencies

The following tables summarize the application's batch processes and their dependencies. A functional description of each batch process is provided, along with the file, where applicable, that is involved in the batch processing.

Note: DTD stands for Document Type Definition. In the context of Oracle Retail Integrator, DTDs are schema specification methods for XML documents.

Import Batch Processes

Batch Process (and Name Used in Command Line)	Functional Description	File	Batch Dependencies (Run Before/After)
Partner import (PartnerImport)	Oracle Retail Design allows for a new trading partner to be imported. Once the client has been set up with a set number of user licenses (configured through Enterprise Administration), the client can submit an XML file containing the new trading partner's name, code, address and contact information. This trading partner is then automatically set up as an enterprise in yourdomain.com.	enterprisepartner.dtd	No dependencies. Can be run independent of any other batch process.

Batch Process (and Name Used in Command Line)	Functional Description	File	Batch Dependencies (Run Before/After)
Technical specification import (plxImport)	Product images, technical specification documents, and a supporting XML file are imported via a ZIP file to Oracle Retail Integrator. Detailed contents of the XML file determine how product images and technical specification documents are attached to specific product files within Oracle Retail Design.	techspec.dtd	No dependencies. Can be run independent of any other batch process.
DesignParameterImport	The parameter import allows parameter definitions and corresponding values to be automatically created within Oracle Retail Design-Parameters administration via Oracle Retail Integrator. Therefore, if there are a number of values to be uploaded, the parameter import allows the Oracle Retail Design administration to complete this more efficiently.	parameters.xsd	No dependencies. Can be run independent of any other batch process.

Export Batch Processes

Batch Process (and Name Used in Command Line)	Functional Description	File	Batch Dependencies (Run Before/After)
Active styles export (DesignActiveStyleExport)	This export process generates an XML file containing details about all active product files that exist within Oracle Retail Design based on the parameter selection configuration. This extract uses the same season selection parameter as the season details extract and ensures that all external systems can be synchronized to only include active product files. Once the XML file has been generated within Oracle Retail Integrator, a separate FTP process can be used to transfer data to specific directories on an FTP server.	styleids.dtd	Must be run prior to Active styles FTP.
Active styles FTP (DesignActiveStyles)	This batch process moves the files generated during the DesignActiveStyleExport script from the application directory to the FTP server.	styleids.dtd	Must be run after running Active styles export.

Batch Process (and Name Used in Command Line)	Functional Description	File	Batch Dependencies (Run Before/After)
Season details export (XMLItemExtract)	This export process generates an XML file containing details about each product file that, based on the parameter configuration, have been selected to appear within this extract. The administrator has the ability to select the season or set of seasons that should be used as a parameter for this extract and a timeframe for determining which product files should be selected. Specifically, the enterprise administrator can configure the process to pick up all product files that have changed since a specific date or since the last time the extract process has been run. Once the XML file has been generated within Oracle Retail Integrator, a separate FTP process can be used to transfer data to specific directories on an FTP server.	style.dtd	Must be run prior to Season details FTP.
Season details FTP (DesignXMLExtract)	This batch process moves the files generated from the XMLItemExtract script from the application directory to the FTP server.	style.dtd	Must be run after Season details export.

Batch Process (and Name Used in Command Line)	Functional Description	File	Batch Dependencies (Run Before/After)
Technical specification export (plxBatchRDOExport)	This export process supports the easy distribution of the Oracle Retail Design product file contents in a document format and any attached technical specification documents via Oracle Retail Integrator. Documents and the supporting XML file are exported from Oracle Retail Design and placed in a ZIP file within Oracle Retail Integrator. A separate FTP process can be used after the export process to transfer data to specific directories on an FTP server.	specsheel.dtd Note: See the Oracle Retail Design Configuration Guide for this file.	Must be run prior to Technical specification FTP.
Technical specification FTP (plxTechSpecExport)	This batch process moves the files generated from the plxBatchRDOExport script from the application directory to the FTP server.	specsheel.dtd Note: See the Oracle Retail Design Configuration Guide for this file.	Must be run after Technical specification export.

General Batch Processes

Batch Process (and Name Used in Command Line)	Functional Description	File	Batch Dependencies (Run Before/After)
Purge (IntegratorPurge)	The purge script purges data in the file system based on parameters entered in Enterprise Administration. The script includes the following parameters: enterprise ID, purge type, and file types. If a file processed through Oracle Retail Integrator is purged, the file no longer is accessible through the Oracle Retail Integrator GUI. It is deleted from the file system.	N/A	No dependencies. Can be run independent of any other batch process.
Shared file purge (IntegratorPurge)	The shared file purge script is in almost all respects the same as the purge script, but the shared file purge script includes the <code>-ft plxSharedTSpdf</code> parameter. This value tells the system to purge <i>only</i> the <code>plxSharedTSpdf</code> files. These files are <i>not</i> purged through the purge script. This separate script must be run to remove them. <code>PlxSharedTSpdf</code> files are PDF attachments that are no longer used by any style file within Oracle Retail Design. The PDF attachments are uploaded during the <code>plxImport</code> process.	N/A	No dependencies. Can be run independent of any other batch process.

Validation Processing

As part of some of its import processing, Oracle Retail Integrator performs validation. Once Oracle Retail Integrator converts the incoming XML to a class structure, Oracle Retail Integrator validates the XML to the schema, making sure that the data is formatted according to the schema. If the data is determined to be valid, it is passed along to the application (such as Oracle Retail Design or Oracle Retail WebTrack) which performs additional business rules validation (checking to determine whether this department exists in the database, that is) before submitting the data to the database. Data that Oracle Retail Integrator determines is not valid is returned to an XML format (that is still valid against the schema but tagged with an error). Note that Oracle Retail Integrator processes data piece by piece, so that only 'bad' data is tagged with an error (one division out of twenty in the division XML file, that is). The user can access this data, make corrections, and retry the import process.

Summary of Batch Logging

The following table associates batch processes with the type of log files they create. A description of the location of the log file is also included. Note that apart from errors located within the UNIX output file, most errors can be viewed through the applicable folder in the Oracle Retail Integrator GUI. Although folder paths are included in the notes within the table below, also see the Oracle Retail Integrator User Guide.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
Technical specification import (plxImport)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/import/plxImport.out (where 9999 is replaced by the enterprise_id) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.
	This log file contains the general processing steps for the FTP process. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.	/yourpathtodatadir/import/9999/FTPimportLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the FTPimportLog.txt file link seen in the /FTP/Import/Log folder in Oracle Retail Integrator.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
	This log file contains general processing steps of the ZIP file that was included in the plxImport.	/yourpathtodatadir/import/9999/STYLEID_DATASETID_Log.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the Log file link seen by clicking the radio button to the left of the filename in the /Design/Tech Spec Zip/AutoImport folder in Oracle Retail Integrator.
	This log file contains the errors the XML file generates when an import is 'Processed with Errors.'	/yourpathtodatadir/import/9999/error/STYLEID.xml_DATASETID.xml (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the Errors.xml file link seen by clicking the radio button to the left of the filename that was 'Processed with Errors' in the /Design/Tech Spec Zip/AutoImport folder or the /Design/Tech Spec/AutoImport folder in Oracle Retail Integrator.
Partner import (PartnerImport)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/import/PartnerFTPStep0.out (where 9999 is replaced by the enterprise_id) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
	This log file contains the general processing steps for the FTP process. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.	/yourpathtodatadir/import/9999/FTPimportLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the FTPimportLog.txt file link seen in the /FTP/Import/Log folder in Oracle Retail Integrator.
	This log file contains the Errors.xml file, which is generated when an import is 'Processed with Errors.'	/yourpathtodatadir/import/9999/error/FILENAME.xml_DATASETID.xml (where 9999 is replaced by the enterprise_id, FILENAME is the import filename and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the Errors.xml file link seen by clicking the radio button to the left of the filename that was 'Processed with Errors' in the /Partner Enterprise/AutoImport folder in Oracle Retail Integrator.
Active Styles Export (DesignActiveStyleExport)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/export/ActiveStyleBatchStep0.out (where 9999 is replaced by the enterprise_id) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
	<p>This log file contains the general processing steps produced when running the Active Styles Export script. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.</p>	<p>/yourpathtodatadir/import/9999/BatchLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file)</p> <p>Note: This file can also be viewed by clicking the BatchLog.txt file link seen when clicking the radio button to the left of the filename in the /Batch Jobs/Reports/Design/Active Styles folder in Oracle Retail Integrator.</p>
	<p>This log file contains the general processing steps for creating the Distributed Active Styles Export file. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.</p>	<p>/yourpathtodatadir/import/9999/ReportDistLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file)</p> <p>Note: This file can also be viewed by clicking the ReportDistLog.txt file seen when clicking the radio button to the left of the filename in the /Batch Jobs/Reports/Design/Distributed Active Styles folder in Oracle Retail Integrator.</p>
Active Styles FTP (DesignActiveStyles)	<p>This UNIX script output file contains any errors generated through the execution of the script.</p>	<p>/yourpathtodatadir/ftp/9999/export/DesignActiveStyleExport.out (where 9999 is replaced by the enterprise_id)</p> <p>Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.</p>

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
	This log file contains the general FTP processing steps.	/yourpathtodatadir/import/9999/FTPexportLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the FTPexportLog.txt file link seen in the /FTP/Export/Log folder in Oracle Retail Integrator.
Season Details Export (XMLItemExtract)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/export/XMLBatchStep0.out (where 9999 is replaced by the enterprise_id) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.
	This log file contains the general processing steps produced when running the season details export script. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.	/yourpathtodatadir/import/9999/BatchLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the BatchLog.txt file link seen when clicking the radio button to the left of the filename in the /Batch Jobs/Reports/Design/Season Details folder in Oracle Retail Integrator.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
	This log file contains the general processing steps for creating the Distributed Season Details Export file. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.	/yourpathtodatadir/import/9999/ReportDistLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking theReportDistLog.txt file seen when clicking the radio button to the left of the filename in the /Batch Jobs/Reports/Design/Distributed folder in Oracle Retail Integrator.
Season Details FTP (DesignXMLExtract)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/export/DesignXMLExtract.out (where 9999 is replaced by the enterprise_id) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.
	This log file contains the general FTP processing steps.	/yourpathtodatadir/import/9999/FTPexportLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the FTPexportLog.txt file link seen in the /FTP/Export/Log folder in Oracle Retail Integrator.
Technical specification export (plxBatchRDOExport)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/export/plxAutoExport.out (where 9999 is replaced by the enterprise_id) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
	<p>This log file contains the general processing steps for the client (retailer), including which files were exported when the Technical Specifications export script was run. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.</p>	<p>/yourpathtodatadir/import/9999/AutoExportLog.txt_DATASETID.txt (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file)</p> <p>Note: This file can also be viewed by clicking the AutoExportLog.txt file link seen in the /Batch Jobs/Design/AutoExportLog folder in Oracle Retail Integrator.</p>
	<p>This log file is an XML file generated for each file produced during the Technical Specification Export. This file contains the filename and corresponding enterprise_id.</p>	<p>/yourpathtodatadir/import/9999/AutoLog.xml_DATASETID.xml (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file)</p> <p>Note: This file can also be viewed by clicking the AutoLog.xml file link seen in the /Design/Tech Spec/AutoExportLog folder in Oracle Retail Integrator.</p>
	<p>This log file contains general processing steps for the AGENT, for each ZIP file produced when running the script.</p>	<p>/yourpathtodatadir/import/9999/plxAutoExTechSpecZipLog.txt_DATASETID.txt (where 9999 is replaced by the AGENT enterprise_id and DATASETID is a unique data set ID assigned to the file)</p> <p>Note: This file can also be viewed by clicking the plxAutoExTechSpecZipLog.txt file link seen when clicking the radio button to the left of the filename in the /Design/Tech Spec Zip/AutoExport folder in Oracle Retail Integrator for the AGENT.</p>

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
Technical Specification FTP (plxTechSpecExport)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/ftp/9999/export/DesignXMLExtract.out (where 9999 is replaced by the enterprise_id for the AGENT assigned to the style file) Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.
	This log file contains the general FTP processing steps.	/yourpathtodatadir/import/9999/FTPexportLog.txt_DATASETID.txt (where 9999 is replaced by the AGENT enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the FTPexportLog.txt file link seen in the /FTP/Export/Log folder in Oracle Retail Integrator.
Purge (IntegratorPurge)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/bin/purgeStep0_regular.out Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.
	This log file contains general purge processing steps. This data includes what was and was not purged for the enterprise_id. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.	/yourpathtodatadir/import/9999/purgeLog.txt_DATASETID (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the purgeLog.txt file link seen in the /Batch Jobs/Purge/Log folder in Oracle Retail Integrator.

Batch Process (and Name Used in Command Line)	Description of Log File	Log File Location
Shared File Purge (IntegratorPurge)	This UNIX script output file contains any errors generated through the execution of the script.	/yourpathtodatadir/purgeStep0_shared.out Note: This file <i>cannot</i> be viewed within the Oracle Retail Integrator application.
	This log file contains general purge processing steps. This data includes what was and was not purged for the enterprise_id. This content is identical to that in the email notification sent to users identified for this run type in Exchange Administration.	/yourpathtodatadir/import/9999/purgeLog.txt_DATASETID (where 9999 is replaced by the enterprise_id and DATASETID is a unique data set ID assigned to the file) Note: This file can also be viewed by clicking the purgeLog.txt file link seen in the /Batch Jobs/Purge/Log folder in Oracle Retail Integrator.

Key GUI Field Descriptions Including Batch Return Values

Because backend and front end batch-related data overlap in Oracle Retail Integrator, the following key front-end field descriptions are described here as a helpful reference. See the Oracle Retail Integrator User Guide for the latest information.

Field Name	Field Description
Date	Displays the time and date that the file was sent to retail.com Displayed in GMT format: MM/DD/YY HH:MM:SS
Status	<p>Displays one of five messages. The message describes the result of the file processing.</p> <p><i>Success.</i> The operation successfully completed and no errors were encountered.</p> <p><i>Error: NOT Processed.</i> Something occurred to block the file import. Try the import again or contact your site administrator.</p> <p><i>Processing File ...</i> File has been imported, but processing is not complete. When this message is displayed, click Refresh to update the status.</p> <p><i>Processed with Errors.</i> The import partially succeeded. An error file containing those records that were NOT imported should have been created and a link to it displayed in the Errors.xml field in the Detail frame. Save the error file to your desktop and fix any errors. Try the import again.</p> <p><i>Error: Processing Failed.</i> A major problem occurred while processing the file. This could be the result of invalid XML formatting, missing required seed data, or some retail.com resource that is not available. More information is available in the Detail frame's error message field. Check file formatting and/or contact your site administrator.</p>
Records Imported	Displays the number of records imported to the Oracle Retail WebTrack database tables from the Oracle Retail Integrator staging tables during the file import process. This number should not be zero and should be equal to the total number of records loaded. If this is not true then an error occurred during the process.
Records Loaded	Displays the number of records loaded directly from the XML file to the Oracle Retail Integrator staging tables during the file import process. This number should not be zero and should be equal to the total number of records imported. If not, an error occurred during the process.

Field Name	Field Description
Errors.xml	Contains a link to an Errors.xml file containing records that could not be imported. This is seen only if the file status in the List frame is 'Imported with errors.' When the link is clicked on, the current Detail frame is replaced with the contents of the error file. Each record or sub-record that could not be imported will contain a non-empty <error_text>...</error_text> element that describes the error that was encountered.
Error Message	Displays any error message or messages about the file import process.

Appendix: DTD Files Used in Oracle Retail Integrator Processing

Note: DTD stands for Document Type Definition. In the context of Oracle Retail Integrator, DTDs are schema specification methods for XML documents.

Import Batch Processes

enterprisepartner.dtd used in PartnerImport

```
<?xml version="1.0" encoding="UTF-8"?>
<!--DTD generated by XML Spy v4.3 U (http://www.xmlspy.com)-->
<!ELEMENT account_number (#PCDATA)>
<!ELEMENT address (address1?, address2?, city?, state_province?, postal_code?,
country_name?, phone?)>
<!ELEMENT address1 (#PCDATA)>
<!ELEMENT address2 (#PCDATA)>
<!ELEMENT administrator (first_name, last_name, user_name, password, email)>
<!ATTLIST administrator
    initial_admin CDATA #REQUIRED
>
<!ELEMENT administrators (administrator)>
<!ELEMENT city (#PCDATA)>
<!ELEMENT company_name (#PCDATA)>
<!ELEMENT company_type (#PCDATA)>
<!ELEMENT country_name (#PCDATA)>
<!ELEMENT default_contact (#PCDATA)>
<!ELEMENT department_number (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT enterprise (company_name, enterprise_code, industry?, company_type?,
address*, services+, administrators+, users*, partnership+, errors*)>
<!ATTLIST enterprise
    user CDATA #REQUIRED
    send_mail CDATA #REQUIRED
    send_password CDATA #REQUIRED
    action CDATA #REQUIRED
>
<!ELEMENT enterprise_code (#PCDATA)>
<!ELEMENT enterprises (enterprise)>
<!ATTLIST enterprises
    version CDATA #IMPLIED
    enterprise CDATA #REQUIRED
>
<!ELEMENT first_name (#PCDATA)>
<!ELEMENT industry (#PCDATA)>
<!ELEMENT last_name (#PCDATA)>
<!ELEMENT partner_code (#PCDATA)>
<!ELEMENT partner_info (#PCDATA)>
<!ELEMENT partner_roles (service_code, permission*)>
<!ELEMENT partnership (account_number?, partner_code, default_contact?,
partner_info?, scope+, permissions*)>
<!ELEMENT password (#PCDATA)>
<!ELEMENT permission (#PCDATA)>
<!ATTLIST permission
    all_users CDATA #REQUIRED
```

```
>
<!ELEMENT permissions (partner_roles+)>
<!ELEMENT phone (#PCDATA)>
<!ELEMENT postal_code (#PCDATA)>
<!ELEMENT scope (department_number+)>
<!ELEMENT service (service_code, service_limit?)>
<!ELEMENT service_code (#PCDATA)>
<!ELEMENT service_limit (#PCDATA)>
<!ELEMENT services (service+)>
<!ELEMENT state_province (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT user (first_name, last_name, user_name, password, email, type?,
services*)>
<!ATTLIST user
    account_manager CDATA #REQUIRED
>
<!ELEMENT user_name (#PCDATA)>
<!ELEMENT errorMessage (#PCDATA)>
<!ELEMENT tag_name (#PCDATA)>
<!ELEMENT users (user)>
<!ELEMENT errors (tag_name, errorMessage)>
```

techspec.dtd used in plxImport

```
<?xml version="1.0" encoding="UTF-8"?>
<!--DTD generated by XML Spy v4.3 U (http://www.xmlspy.com)-->
<!ELEMENT ContactEmail (#PCDATA)>
<!ELEMENT DepartmentCode (#PCDATA)>
<!ELEMENT Division (#PCDATA)>
<!ELEMENT FileName (#PCDATA)>
<!ELEMENT FolderName (#PCDATA)>
<!ELEMENT Height (#PCDATA)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT ObjectiveSheetName (#PCDATA)>
<!ELEMENT SeasonCode (#PCDATA)>
<!ELEMENT StyleID (#PCDATA)>
<!ELEMENT StyleImage (Name, Height, Width, FolderName?)>
<!ELEMENT TechSpec (DepartmentCode, FileName?, Division?, SeasonCode,
ObjectiveSheetName?, VendorName?, VendorCode?, StyleID, TechSpecName*,
StyleImage?, ContactEmail?)>
<!ELEMENT TechSpecName (#PCDATA)>
<!ELEMENT TechSpecs (TechSpec)>
<!ATTLIST TechSpecs
    EnterpriseID CDATA #IMPLIED
    UserName CDATA #IMPLIED
>
<!ELEMENT VendorCode (#PCDATA)>
<!ELEMENT VendorName (#PCDATA)>
<!ELEMENT Width (#PCDATA)>
```

parameters.xsd

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Schema for retailserver parameter name and value import -->

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:jaxb="http://java.sun.com/xml/ns/jaxb"
  xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
  jaxb:extensionBindingPrefixes="xjc"
  jaxb:version="1.0"
  elementFormDefault="qualified">

  <!-- JAXB global settings -->

  <xs:annotation>
    <xs:appinfo>
      <jaxb:globalBindings generateIsSetMethod="true">
        <xjc:serializable/>
      </jaxb:globalBindings>
    </xs:appinfo>
  </xs:annotation>

  <!-- Shared definitions -->

  <xs:include schemaLocation="common.xsd"/>

  <!-- Outermost element; contains one or more parameter or values elements -->

  <xs:element name="parameters">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="parameter" type="parameterType"/>
        <xs:element name="values" type="parameterValuesType"/>
      </xs:choice>
      <xs:attributeGroup ref="enterprise"/>
    </xs:complexType>
  </xs:element>

  <!-- Individual parameter

  Elements:

      name          descriptive name
      code          lookup code
      key_definition* key definitions

  Attributes:

      type          integer, decimal, string, sysdate or localdate
      numkeys
      decimalplaces
      multiplevalues
-->

  <xs:complexType name="parameterType">

    <xs:sequence>

      <!-- name and code are always present -->

      <xs:element name="name" type="nonempty"/>
```

```

<xs:element name="code" type="nonempty"/>

<!-- default_str and default_num are, use just one

For system dates, use a value in YYYY-MM-DD format; for local dates,
use
YYYY-MM-DD hh:mm:ss zzz

zzz specifies the time zone.
-->

<xs:choice minOccurs="0">
  <xs:element name="default_str" type="nonempty"/>
  <xs:element name="default_num" type="xs:double"/>
  <xs:element name="default_date" type="nonempty"/>
</xs:choice>

<!-- keys -->

<xs:element name="key_definition" type="parameterKeyDefinitionType"
minOccurs="0" maxOccurs="5"/>

<!-- import error repor -->

<xs:element name="error_text" type="xs:string" minOccurs="0"/>
</xs:sequence>

<!-- attributes -->

<xs:attribute name="action" type="action"/>

<!-- type is optional on updates, required on additions -->

<xs:attribute name="type">
  <xs:simpleType>
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="integer"/>
      <xs:enumeration value="decimal"/>
      <xs:enumeration value="string"/>
      <xs:enumeration value="sysdate"/>
      <xs:enumeration value="localdate"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>

<!-- number of keys, also optional on updates -->

<xs:attribute name="numkeys">
  <xs:simpleType>
    <xs:restriction base="xs:int">
      <xs:minInclusive value="1"/>
      <xs:maxInclusive value="5"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>

<!-- multiple values flag -->

<xs:attribute name="multiple_values" type="xs:boolean"/>

```

```

<!-- decimal places option -->

<xs:attribute name="decimal_places" default="2">
  <xs:simpleType>
    <xs:restriction base="xs:int">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>

<!-- global flag: this can be set for enterprise zero imports only -->

<xs:attribute name="global" type="xs:boolean"/>

</xs:complexType>

<!-- Key definition type -->

<xs:complexType name="parameterKeyDefinitionType">

  <xs:sequence>

    <!-- Label and detail are always optional, although the importer may
         place some semantic restrictions on the detail value.

    If the key type is 'location' or 'partnerlocation' then the
    detail value must be a system-wide 'location type'.

         If the key type is 'parameter', then the detail value must be
         the lookup code of an existing parameter.
    -->

    <xs:element name="label" type="nonempty" minOccurs="0"/>
    <xs:element name="detail" type="nonempty" minOccurs="0"/>

  </xs:sequence>

  <!-- Attributes -->

  <!-- key number: if omitted, one after the number of the previous key -->

  <xs:attribute name="number">
    <xs:simpleType>
      <xs:restriction base="xs:int">
        <xs:minInclusive value="1"/>
        <xs:maxInclusive value="5"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

```

```

<!-- key type: required on inserts -->

<xs:attribute name="type">
  <xs:simpleType>
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="partner"/>
      <xs:enumeration value="location"/>
      <xs:enumeration value="partnerlocation"/>
      <xs:enumeration value="producttype"/>
      <xs:enumeration value="elctype"/>
      <xs:enumeration value="season"/>
      <xs:enumeration value="division"/>
      <xs:enumeration value="department"/>
      <xs:enumeration value="class"/>
      <xs:enumeration value="subclass"/>
      <xs:enumeration value="colour"/>
      <xs:enumeration value="country"/>
      <xs:enumeration value="freeform"/>
      <xs:enumeration value="parameter"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>

</xs:complexType>

<!-- Set of values. This does not use the standard 'action' attribute;
      instead there is a mode attribute with values:

               replace      all values for the parameter are replaced
               append      new values are added
               update      existing values are updated or new values added
               delete      existing values are deleted

      For update and delete, existing values are identified by the
      'external value'; it is required that this is unique.

      The values element contains a set of individual value elements for
      the parameter.
-->

<xs:complexType name="parameterValuesType">
  <xs:sequence>
    <xs:element name="value" type="parameterValueType" minOccurs="0"
maxOccurs="unbounded"/>

    <!-- import error repor -->

    <xs:element name="error_text" type="xs:string" minOccurs="0"/>
  </xs:sequence>

  <!-- mode attribute -->

  <xs:attribute name="mode" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="replace"/>
        <xs:enumeration value="append"/>
        <xs:enumeration value="update"/>
        <xs:enumeration value="delete"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

```

```

        <!-- parameter attribute identifies the parameter by lookup code -->
        <xs:attribute name="parameter" type="nonempty" use="required"/>
    </xs:complexType>

    <!-- Single value -->

    <xs:complexType name="parameterValueType">
        <xs:sequence>

            <!-- value as string, double or date: see default_date above for
formats -->

            <xs:choice>
                <xs:element name="value_str" type="xs:string"/>
                <xs:element name="value_num" type="xs:double"/>
                <xs:element name="value_date" type="nonempty"/>
            </xs:choice>

            <!-- Optional start and end dates in YYYY-MM-DD format -->

            <xs:element name="start_date" type="xs:string" minOccurs="0"/>
            <xs:element name="end_date" type="xs:string" minOccurs="0"/>

            <!-- Optional external value -->

            <xs:element name="external_value" type="nonempty" minOccurs="0"/>

            <!-- keys -->

            <xs:element name="key_value" type="parameterKeyType" minOccurs="1"
maxOccurs="5"/>

        </xs:sequence>

        <!-- active attribute -->

        <xs:attribute name="active" default="true" type="xs:boolean"/>
    </xs:complexType>

    <!-- Value key type

```

The parent element defines the scope for keys which may not be unique in themselves, as follows:

partnerlocation	parent is partner name
department	parent is division number
class	parents are division number, department
number	
subclass	parents are division number, department
number, class number	

The key element is the key value itself, as follows:

partner	partner name
location	location name
partnerlocation	location name (in enterprise named by
parent)	
producttype	type name
elctype	ELC type name
season	season code

division	division number
department	department number
class	class number (extradata)
subclass	subclass number (extradata)
country	country name
colour	colour code
freeform	string
parameter	external value for parameter

-->

```
<xs:complexType name="parameterKeyType">  
  <xs:sequence>  
    <xs:element name="parent" type="xs:string" minOccurs="0" maxOccurs="3"/>  
    <xs:element name="key" type="xs:string"/>  
  </xs:sequence>  
</xs:complexType>  
</xs:schema>
```


Export Batch Processes

Note: See the Oracle Retail Design Configuration Guide for the specsheet.dtd used in plxBatchRDOExport.

style.dtd used in XMLItemExtract and in DesignXMLExtract

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- DTD for Design XML export -->

<!-- Style_files is the wrapper for a list of individual style items -->

<!ELEMENT Style_files (Style_file+)>

<!-- Style_file contains data on one style.

      The Style_file_id attribute is the internal unique integer ID for the
      item. The number of digits depends on the number of style records in the
      retail.com database. It will not be more than 10 digits.

      New for 10.3: Last_Edit_User and Last_Edit_Enterprise identify the last
      user to edit the style. The fields will not be present if no information
      is recorded with the style (for edits prior to 10.3 migration).
-->

<!ELEMENT Style_file (Creation_Date, Change_Date, Last_Edit_User?,
Last_Edit_Enterprise?, Season, Season_Code,
      Division_Name, Division_Number, Department_Name, Department_Number,
      Summary, Specification, Bids, ELC?, Miscellaneous, Labels)>
<ATTLIST Style_file Style_file_id CDATA #REQUIRED>

<!-- Creation_Date and Change_Date are timestamps on the style

      The format will be YYYY-MM-DD.
-->

<!ELEMENT Creation_Date (#PCDATA)>
<!ELEMENT Change_Date (#PCDATA)>

<!-- User and enterprise enames -->

<!ELEMENT Last_Edit_User (#PCDATA)>
<!ELEMENT Last_Edit_Enterprise (#PCDATA)>

<!-- Season and Season_Code are the name and code for the season -->

<!ELEMENT Season (#PCDATA)>
<!ELEMENT Season_Code (#PCDATA)>

<!-- Division_Name, Division_Number, Department_Name and Department_Number are the
names
      and numbers of the style division and department

      Note: Division_Name and Division_Number are new for Design 10.
-->
```

```
<!ELEMENT Division_Name (#PCDATA)>
<!ELEMENT Division_Number (#PCDATA)>

<!ELEMENT Department_Name (#PCDATA)>
<!ELEMENT Department_Number (#PCDATA)>

<!-- Summary contains data from the Design summary tab -->

<!ELEMENT Summary (Customer, Product, Supplier)>

<!-- Customer contains the customer data

      NOTE: FOB_Country and FOB_Location are new for Design 10.  FOB_Location_Code
is new for Design 10.1;
      it contains the new location code for the FOB location.
-->

<!ELEMENT Customer (Customer_Name, Buyer, Quality, Designer, Design_ID, Theme,
Bid_Deadline, Phase, Class, Subclass, Price_By,
      Qty_Reqd, Retail_Price, Num_Deliveries, Buying_Margin, IA_Date,
ELC_Target, Customer_Status, VAT,
      FOB_Country, FOB_Location, FOB_Location_Code)>

<!-- Customer name -->

<!ELEMENT Customer_Name (#PCDATA)>

<!-- Users within the customer enterprise -->

<!ELEMENT Buyer (#PCDATA)>
<!ELEMENT Quality (#PCDATA)>
<!ELEMENT Designer (#PCDATA)>

<!-- Customer Design ID or Style number -->

<!ELEMENT Design_ID (#PCDATA)>

<!-- Other customer data -->

<!ELEMENT Theme (#PCDATA)>
<!ELEMENT Bid_Deadline (#PCDATA)>
<!ELEMENT Phase (#PCDATA)>
<!ELEMENT Class (#PCDATA)>
<!ELEMENT Subclass (#PCDATA)>

<!-- Price_By is an integer:

      0 for [none]
      1 for Size
      2 for Colour & Size
-->

<!ELEMENT Price_By (#PCDATA)>

<!ELEMENT Qty_Reqd (#PCDATA)>
<!ELEMENT Retail_Price (#PCDATA)>
<!ELEMENT Num_Deliveries (#PCDATA)>
<!ELEMENT Buying_Margin (#PCDATA)>
```

```

<!-- IA_Date is a date in the YYYY-MM-DD format -->

<!ELEMENT IA_Date (#PCDATA)>

<!ELEMENT ELC_Target (#PCDATA)>

<!-- Customer_Status is the status name as displayed on the summary screen -->

<!ELEMENT Customer_Status (#PCDATA)>

<!-- VAT will be 0.00 if no value has been entered -->

<!ELEMENT VAT (#PCDATA)>

<!-- FOB_Country and FOB_Location are new for Design 10 -->

<!ELEMENT FOB_Country (#PCDATA)>
<!ELEMENT FOB_Location (#PCDATA)>
<!ELEMENT FOB_Location_Code (#PCDATA)>

<!-- Product contains the basic product data

      NOTE: Documents is new for Design 10.
      ELC_Type is new for Design 10.2. It is not included in 'restriced mode'.
-->

<!ELEMENT Product (Product_Name, Product_Type, ELC_Type?, Short_name, Text_Area_1,
Text_Area_2, Text_Area_3, Category,
      Dispatch_Type, Documents, Colors, Size_Charts)>

<!-- Product_Name is not used currently -->

<!ELEMENT Product_Name EMPTY>

<!-- Product_Type is the product type which drives the specification and bid sheet
      formats. The value is the type name seen on the summary screen.
-->

<!ELEMENT Product_Type (#PCDATA)>

<!-- ELC_Type is the separate type which drives the ELC sheet format. The
      value is the type name seen on the summary screen.

      ELC type is not included in 'restricted' XML.
-->

<!ELEMENT ELC_Type (#PCDATA)>

<!-- Style name from summary -->

<!ELEMENT Short_name (#PCDATA)>

<!-- Text_Area_1, Text_Area_2 and Text_Area_3 are the three text areas below the
short
      name on the summary screen.
-->

<!ELEMENT Text_Area_1 (#PCDATA)>
<!ELEMENT Text_Area_2 (#PCDATA)>
<!ELEMENT Text_Area_3 (#PCDATA)>

```

```
<!-- Category and dispatch type from bottom of product section -->

<!ELEMENT Category      (#PCDATA)>
<!ELEMENT Dispatch_Type (#PCDATA)>

<!-- Documents is new in Design 10.

      It lists the document names attached to the style. It contains zero or
      more Document elements, each containing a Name element defining the
      document name.
-->

<!ELEMENT Documents (Document*)>
<!ELEMENT Document (Name)>
<!ELEMENT Name (#PCDATA)>

<!-- Colors lists the colors attached to the style.

      It contains zero or more Color elements each containing Name, Code and
      Proj_Key elements. Proj_Key is the internal key assigned when the style
      is exported to a WebTrack project. If the style has not been exported
      Proj_Key will be empty. If the colour set is derived from an additional
      colour entry component, the Proj_Key element will be absent.
-->

<!ELEMENT Colors (Color*)>

<!ELEMENT Color (Name, Code, Proj_Key?)>

<!ELEMENT Code (#PCDATA)>
<!ELEMENT Proj_Key (#PCDATA)>

<!-- Size_Charts defines the size chart attached to the style. If no size
      chart has been assigned the Size_Charts element will be empty.

      Chart_Name and Chart_Code are the name and code value for the chart. Each
      Size
      has one or two Size_Name elements.
-->

<!ELEMENT Size_Charts (Chart_Name?, Chart_Code?, Size*)>

<!ELEMENT Chart_Name (#PCDATA)>
<!ELEMENT Chart_Code (#PCDATA)>

<!ELEMENT Size (Size_Name+)>
<!ELEMENT Size_Name (#PCDATA)>

<!-- Supplier contains supplier and agent information.

      Designer is the supplier designer. It is distinct from the customer
      designer although is shares the same element name.

      Design_ID is the supplier design ID. It is distinct from the customer
      design ID although is shares the same element name.

      Factory_Code is new in Design 10.1. It is the location code for the factory.
-->
```

```

<!ELEMENT Supplier (Supplier_Name, Account, Acc_Mgr, Designer, Team, Factory,
Factory_Code, Country, Lead_Days,
                Order_Reqd_By, Cost_By, Qty_Offered, Item_Cost, Effective_Until,
Net_Cost, Supplier_Status,
                Design_ID, Agent, Agent_Contact)>

<!ELEMENT Supplier_Name (#PCDATA)>

<!-- Account is the supplier's account number at the customer -->

<!ELEMENT Account (#PCDATA)>

<!-- Acc_Mgr and Designer are users in the supplier -->

<!ELEMENT Acc_Mgr (#PCDATA)>
<!ELEMENT Technologist (#PCDATA)>

<!-- Team is a department in the supplier -->

<!ELEMENT Team (#PCDATA)>

<!ELEMENT Factory (#PCDATA)>
<!ELEMENT Factory_Code (#PCDATA)>
<!ELEMENT Country (#PCDATA)>
<!ELEMENT Lead_Days (#PCDATA)>

<!-- Order_Reqd_By is a date in the YYYY-MM-DD format -->

<!ELEMENT Order_Reqd_By (#PCDATA)>

<!-- Cost_By is an integer:
        0 for [none]
        1 for Size
        2 for Colour & Size
-->

<!ELEMENT Cost_By (#PCDATA)>

<!ELEMENT Qty_Offered (#PCDATA)>
<!ELEMENT Item_Cost (#PCDATA)>

<!-- Effective_Until is a date in the YYYY-MM-DD format -->

<!ELEMENT Effective_Until (#PCDATA)>

<!ELEMENT Net_Cost (#PCDATA)>

<!-- Supplier_Status is the supplier status name as displayed on the summary
screen -->

<!ELEMENT Supplier_Status (#PCDATA)>

<!-- Agent is the name of the atgent enterprise
        Agent_Contact is the contact user within the agent.
-->

<!ELEMENT Agent (#PCDATA)>
<!ELEMENT Agent_Contact (#PCDATA)>

```

```
<!-- Specification contains the mapped fields for the specification tab.

It contains zero or more Spec_Item elements each of which has a Map_Name
and the associated value. A simple scalar mapping (an item in a form, a
cell in a non-dynamic matrix region) will have a Value element.

A multiple-value mapping (matrix, row, col or dynamic area) will be
represented as a one- or two-dimensional array. An array is introduced
by an Array element; subelements will either be further arrays or simple
Elem elements. Two-dimensional arrays are represented by row first.

A Custom form item may have a special output format. The colour selector
custom item is represented as a Colors element (see above).
-->

<!ELEMENT Specification (Spec_Item*)>
<!ELEMENT Spec_Item (Map_Name, (Value | Array | Colors))>

<!ELEMENT Map_Name (#PCDATA)>
<!ELEMENT Value (#PCDATA)>

<!ELEMENT Array (Array | Elem)*>
<!ATTLIST Array size CDATA #REQUIRED>
<!ELEMENT Elem (#PCDATA)>

<!-- Bids contains the mapped fields for the bid tab.

It contains zero or more Bid_Item elements each of which has a Map_Name
and a value.
-->

<!ELEMENT Bids (Bid_Item*)>
<!ELEMENT Bid_Item (Map_Name, (Value | Array | Colors))>

<!-- ELC contains the mapped fields for the ELC tab.

It contains zero or more ELC_Item elements each of which has a Map_Name
and a Value.
-->

<!ELEMENT ELC (ELC_Item*)>
<!ELEMENT ELC_Item (Map_Name, (Value | Array | Colors))>

<!-- Miscellaneous contains the miscellaneous fields from spec/bid which have a
value.

It contains zero or more Misc_Item elements each of which contain a
Number element (the misc field number) and a Value element.
-->

<!ELEMENT Miscellaneous (Misc_Item*)>
<!ELEMENT Misc_Item (Number, Value)>
<!ELEMENT Number (#PCDATA)>

<!-- Labels contains the style labels.

It contains zero or mail Label elements.
-->

<!ELEMENT Labels (Label*)>
```

```

<!-- Each label element contains:

    Label_Code and Label_Desc: template code and description
    Packaging_Type and Packaging_SubType: packaging info from template
    Label_Comment: Comment attached to style label
    Label_Date: YYYY-MM-DD format date
-->

<!ELEMENT Label (Label_Code, Label_Desc, Packaging_Type, Packaging_SubType,
Label_Comment, Label_Date)>

<!ELEMENT Label_Code (#PCDATA)>
<!ELEMENT Label_Desc (#PCDATA)>
<!ELEMENT Packaging_SubType (#PCDATA)>
<!ELEMENT Packaging_Type (#PCDATA)>
<!ELEMENT Label_Comment (#PCDATA)>
<!ELEMENT Label_Date (#PCDATA)>

```

styleids.dtd used in DesignActiveStyleExport and in DesignActiveStyles

```

<?xml version="1.0" encoding="UTF-8"?>

<!-- DTD for Design active style IDs XML export -->

<!ELEMENT Style_files (Seasons, Style_file*)>

<!-- Seasons contains the list of season codes associated with this export -->

<!ELEMENT Seasons (Season_Code*)>

<!-- Season_Code is the code for an individual season -->

<!ELEMENT Season_Code (#PCDATA)>

<!-- Style_file defines the ID for a single style -->

<!ELEMENT Style_file EMPTY>
<!ATTLIST Style_file Style_file_id CDATA #REQUIRED>

```