



Implementation Guide for Oracle Billing Insight

Version 7.1

July 2016

ORACLE®

Copyright © 2005, 2016 Oracle and/or its affiliates. All rights reserved.

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

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

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

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

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

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

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

Documentation Accessibility

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

Access to Oracle Support

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

Contents

Chapter 1: What's New in This Release

Chapter 2: Overview of Oracle Billing Insight

About Oracle Billing Insight	13
About the Self-Service Application	14
About the Assisted Service Application	15
About the Command Center Application	15
About the RESTful Web Services	16
About Oracle Billing Insight Architecture	17
About the Oracle Billing Insight Database	17
About Oracle Billing Insight Installation Directories	18
About Application EAR Files	19
Debugging Oracle Billing Insight	23
Accessing Oracle Billing Insight Data Dictionary and Javadoc	25
Identifying the Installed Version of Oracle Billing Insight	26

Chapter 3: About Customizing Oracle Billing Insight

Best Practices for Customizing Oracle Billing Insight	27
Non-Permitted Database Changes	28
About User Interface Files	28
About Repackaging EAR Files	29
Customizing the Default UI CSS	29
About Custom JSP Pages and Action Classes	31
About Velocity Templates	31
About Customizing Reports	31
Customizing the User Interface for Multiple Branding	31
Localizing the User Interface	32
Changing the Default Language	33
Localizing the User Interface for Additional Languages	33

Adding Your Own Custom Language Resource Bundle Files 38

Chapter 4: Customizing User Management

Deactivating and Reactivating the Bootstrap User	41
Customizing the Assisted Service User Password Update Requirements	43
Customizing the Administrator User Password Validation Rule	43
Customizing End User and Agent Password Validation Rules	44
Customizing User Enrollment	46
Customizing the Validation Code Generator	47
Customizing Account Lockout	48
Customizing User Authentication and Authorization With Spring Security	49
About Authentication Request Processing	49
Customizing Authentication URLs	50
About Resource Authorization	51
Adding a New User Role	55
Adding a New Permission Set	56
About Deleted Users	58
Input File Format for the DefUsrAcctRelationship Job	59
Input File Format for Migrating Consumer Users in Batch	59
Input File Format for Deleting Consumer Users in Batch	61

Chapter 5: Customizing Billing Statements

About Split Billing Rules Management	63
Updating Disputes in an External Customer Relationship Management System	64
Defining Billing Periods	65

Chapter 6: Using and Customizing Email and SMS Notifications

About Batch Email Notification Processing	69
Notification Methods: Instant and Batch	71
Email Notification Template Content for Business Users	74
Email Notification Template Content for Consumer Users	101
SMS Notification Template Content	130

Customizing the Content of Notifications	134
Customizing the Default Display Patterns Used in Notifications	135
Configuring Messaging Properties	137
Adding a Custom Message Provider	140
Configuring the Secure Bill-Ready Notification Password	142
Disabling SMS Notifications	142

Chapter 7: Using the Reporting Engine

Reporting Engine Features	143
Reporting Engine Architecture	145
Reporting Engine Object Model	147
Components Used by the Reporting Engine	148
Using the Report List Properties File	149
Configuring Batch Reporting	149
Report XML	149
Creating a PDF Template for Reporting	178
Predefined Context Variables	179
Integration with Struts and Tiles	181
Reporting API	183
Core Reporting Features	184
Sorting Feature of the Reporting Engine	184
Paging Feature of the Reporting Engine	185
Dynamic SQL	186
Internationalization and Localization of Reporting	187
Object Data Source	190
DSV Data Source	191
Drilldown and Breadcrumb Link	191
Customizing the Reporting Engine	192
Write Your Own Report XML	192
Customize the Report Template	192
Write Your Own Action Classes and ReportBean	193
Packaging	193
Hiding Report Columns and Manipulating IReport	194
Unlimited Paging	194
Reloading Report XML and Templates without Restarting the Server	195
Customizing Threshold Values for Batch Reporting	195
Customizing Charts	197
Configurable Chart Properties	198

Customizing the Statement Summary Chart	206
Changing the Local Storage Location for the HTML Report Formatting Files	207

Chapter 8: Payment Processing

About Check Processing	209
Example of Adding a Check Account at Enrollment	209
Check Account Enrollment Status Workflow	210
Check Payment Transaction Workflow	211
Automated Clearing House (ACH) Return Codes	213
About Credit Card Processing	216
Paymentech Authorization Transactions	216
Credit Card Payment Status	217
Credit Card Payment Transactions	218
Instant Credit Card Payments	219
Scheduled Credit Card Payments	219
User Options	220
Using PayPal Payflow Pro as a Payment Gateway	220
Using Chase Paymentech Orbital Payment Gateway	221
Using CyberSource Payment Gateway	221
Address Verification Service	221
About Recurring Payment Processing	222
Tables Updated by Recurring Payments	224
Recurring Payment Transaction Cycle	224
User Interface Updates to the Recurring Payments Table	226
Recurring Payment Job	227
Example of Scheduling the Amount Due Before the Due Date	232
Example of Scheduling the Amount Due on a Fixed Pay Date	234
Example of Scheduling a Fixed Amount Before the Due Date	236
Example of Scheduling Fixed Amount and Fixed Pay Date	239
About Payment Auditing	240
Using Payment Debug	244
Input File Format for Canceling Payments for Consumer Users in Batch	244

Chapter 9: Customizing Payment

About Payment Plug-Ins	245
Creating an ACH Plug-In	252
Creating a Credit Card Plug-In	252
Creating a Payment Reminder Plug-In	253
Creating a Recurring Payment Plug-In	254

Customizing Oracle Billing Insight Payment Template Files	254
Generating Accounts Receivables (A/R Files)	265
Customizing the Payment Amount Format	271
Packaging Oracle Billing Insight Payment Custom Code	271
About the APIs for Implementing Custom Payment Cassette Cartridges	272
Implementing Custom Payment Cassette Cartridges	274
Configuring Convenience Fee Payments	277
Handling Multiple Payee ACH Accounts	279
Using Payment APIs	280
Configuring International Bank Routing	281
Customizing Anonymous Payment Validation	282
Updating the Back-End Billing System With a Top-Up Payment	283

Chapter 10: Customizing the Payment Consolidator Integration Module

Overview of Processing with a Payment Consolidator	285
Payment Consolidation Jobs	287
Payment Consolidation Input and Output Files	289
The Payment Consolidation Template Engine	290
Process of Implementing a Payment Consolidator	290
Creating Customized Payment Consolidation Templates	290
Sample Payment Consolidation Template Files	291
The File Structure XML Template	293
The Record XML Templates	295
Template Attributes	299
Template Variables	300
Logic Operators	303
Accessing an Object's Public Fields and Methods	304
Adding a New Java Object Instance or Static Class	305
Adding Cartridge Property Keys	306
Setting Payment Consolidation Configuration Parameters	308
Customizing the Bill Detail URL	309
Implementing Payment Consolidator Plug-Ins	310
Payment Consolidator Java APIs	313
Default File Formats for Payment Consolidation	314

Account Enrollment Input File Format	314
Enrollment Acknowledgement Output File Format	320
Bill Summary Output File Format	323
Bill Summary Acknowledgement Input File Format	330
Data Element Mappings - Example A	331
Data Element Mappings - Example B	342
Payment Consolidation Database Tables	351

Chapter 11: Customizing the Assisted Service Application

About the Assisted Service Application Functionality	353
Assisted Service Application Files	353
About Impersonation Auditing	354

Chapter 12: Reporting on Audit Data

Reporting on User Audit Data	357
Reporting on System Administrator Audit Data	362

Chapter 13: Load Input File Specifications and Data Mapping

Preprocessor Tasks	365
Data File Loading Tasks	366
About Data Load Processing	366
File Record and Dimension Table Mapping	367
File Record and Fact Table Mapping	367
File Formats for Dimension-Level Information	368
File Formats for Statement-Level Information	375
File Formats for Account-Level Information	380
File Formats for Service-Level Information	383
File Formats for Service Detail-Level Information	387
File Formats for Unbilled Detail-Level Information	391
File Formats for Prepaid Information	393
File Formats for Batch Provisioning Data	397
Internationalization Support Settings	403

Index

1

What's New in This Release

What's New in Implementation Guide for Oracle Billing Insight, Version 7.1

Table 1 lists changes described in this version of the documentation to support this release of the software.

Table 1. New Product Features in Implementation Guide for Oracle Billing Insight, Version 7.1

Topic	Description
"Non-Permitted Database Changes" on page 28	New topic. Lists the non-permitted changes to the Oracle Billing Insight database.
"Localizing the User Interface" on page 32	Modified topic. New language resource property file sets have been added for Brazilian Portuguese, Dutch, German, Italian, Spanish (Worldwide), and Turkish. Language (property) file names and content have been updated for global display formats for numbers and dates. These changes facilitate localization. Steps have also been added for implementing a custom resource properties file.
"Customizing the Content of Notifications" on page 134 "Customizing the Default Display Patterns Used in Notifications" on page 135	Modified topics. Updated the email and SMS notification property file names. In addition, global display formats used for numbers and dates in the UI have been updated.
"File Formats for Dimension-Level Information" on page 368	Modified topic. Updated file formats for dimension-level data.
"File Formats for Service Detail-Level Information" on page 387	Modified topic. Updated file formats for service call usage detail data.
Multiple topics.	Modified topics. Removed references to debit cards, which are no longer supported.
Implementing the Assisted Service Application with CRM or Other Back-Office Systems	Moved topic. This topic has been moved to <i>Integration Guide for Oracle Billing Insight</i> .

2

Overview of Oracle Billing Insight

This chapter presents an overview of Oracle Billing Insight. It includes the following topics:

- [About Oracle Billing Insight on page 13](#)
- [About Oracle Billing Insight Architecture on page 17](#)
- [About the Oracle Billing Insight Database on page 17](#)
- [About Oracle Billing Insight Installation Directories on page 18](#)
- [About Application EAR Files on page 19](#)
- [Debugging Oracle Billing Insight on page 23](#)
- [Accessing Oracle Billing Insight Data Dictionary and Javadoc on page 25](#)
- [Identifying the Installed Version of Oracle Billing Insight on page 26](#)

About Oracle Billing Insight

Oracle Billing Insight is a Java J2EE multi-tiered Web application suite for viewing, analyzing, managing, and paying bills. Business and consumer users interact with the self-service interface and agent-assisted services. Oracle Billing Insight can be deployed on the Oracle Cloud Platform or On Premise.

Oracle Billing Insight is built on Struts 2, the Spring Framework, Spring Security, and Hibernate, and runs on Oracle WebLogic and Oracle Database. The Struts actions talk with the Service APIs which then access different functionality, such as hierarchy or reporting.

Oracle Billing Insight includes four applications, each packaged as one Enterprise Archive (EAR) file:

- **Self-Service.** End users interact with the self-service online interface to view billing information, manage payments, manage their business hierarchies, and so on. Use the information in this guide to help you customize the preconfigured functionality for your company's implementation. In general, you deploy the Self-Service application EAR file in a cluster environment for the purpose of failover and load balance.

For information about preconfigured Self-Service application use cases provided with Oracle Billing Insight, see *Self-Service Application Guide for Oracle Billing Insight*.

- **Assisted Service.** Agents, such as customer service representatives, use the Assisted Service application to view billing analytics, statements, and payment information for end users. Agents can also make payments and perform other billing- and account-related tasks on behalf of a customer. You can also configure your Assisted Service application for access by a CRM or other back-office system.

For information about preconfigured Assisted Service application use cases provided with Oracle Billing Insight, see *Assisted Service Application Guide for Oracle Billing Insight*.

- **Command Center (System Administration).** Your system administrator uses the Command Center to manage the live Oracle Billing Insight production environment, which comes with a set of predefined batch jobs for payment processing and reporting. A job consists of one or more tasks, and each task performs a specific piece of the processing. Each task is implemented as an EJB and has its own configuration parameters which you also configure using the Command Center UI. The Command Center UI is based on Servlet-JSP technology, not struts and tiles. You deploy the Command Center on a separate application server. You do not customize this application, although you can create custom jobs if necessary.

For more information about configuring and running jobs in the Oracle Billing Insight Command Center, see *Administration Guide for Oracle Billing Insight*.

- **Web Services (Machine-to-Machine).** Oracle Billing Insight Web services provide a programmatic interface that system integrators can use to integrate back-office systems with the Oracle Billing Insight applications and database.

For information about using Web services with Oracle Billing Insight, see *Web Services Reference for Oracle Billing Insight*.

This guide assumes you have installed Oracle Billing Insight and deployed these applications, and can run and view them successfully. For information about installing Oracle Billing Insight, see *Installation Guide for Oracle Billing Insight*.

Related Topics

[About the Self-Service Application on page 14](#)

[About the Assisted Service Application on page 15](#)

[About the Command Center Application on page 15](#)

[About the RESTful Web Services on page 16](#)

About the Self-Service Application

The Oracle Billing Insight Self-Service is a Web-based application that empowers customers with electronic bill presentment and payment capabilities, and extensive analytics that highlight the value of products and services.

The Oracle Billing Insight Self-Service application:

- Provides convenient, efficient, and secure payment collection.
- Lets users view statement summaries and drill-down to transaction detail.
- Provides bill or transaction-level dispute functionality.
- Lets users annotate and categorize transactions.
- Provides dynamic, interactive views of spending and usage based on the customer's own business hierarchy and cost centers.
- Provides cross-channel access by the Assisted Service application, enabling agents to view customer data and perform tasks on behalf of a customer.
- Is an engaging, personalized, and graphical cross-platform application.

- Can aggregate invoice data from multiple billing sources.
- Provides SMS alerts and email notifications with PDF attachments to attract customers to service their accounts online.
- Provides notifications and alerts on overages.

For information about the preconfigured use cases provided in the Self-Service application, see *Self-Service Application Guide for Oracle Billing Insight*.

About the Assisted Service Application

The Oracle Billing Insight Assisted Service application is a Web-based application that empowers agents, such as customer service representatives, to assist users with all the functions they can perform in the Self-Service application, plus additional functionality to manage agents.

The Oracle Billing Insight Assisted Service application:

- Lets agents view statements and run analytics for any customer account.
- Provides the agent with the exact data views that a particular user sees in the Self-Service application, based on user's role and privileges.
- Gives the agent the ability to make payments on behalf of a user perform all of the functions available to the user they are assisting, based on user's role and privileges.
- Lets an agent search for organizations and modify organization information.
- Lets an agent administrator create agent users.
- Gives agents the ability to impersonate users.

For information about preconfigured Assisted Service application use cases provided with Oracle Billing Insight, see *Assisted Service Application Guide for Oracle Billing Insight*.

About the Command Center Application

The Oracle Billing Insight Command Center provides a Web-based production application for system administration.

Your system administrator uses the Oracle Billing Insight Command Center application to:

- Manage the live Oracle Billing Insight production environment, including scheduling and running batch jobs for loading and managing billing data, processing payments, generating user notifications, managing hierarchies, and running reports.
- Configures each job and task for your organization and environment.
- Configure payment gateways, accounts, and related functionality.
- Perform other ongoing production and maintenance tasks, such as reviewing job reports and log files, and purging old data.

For more information about configuring and running jobs in the Oracle Billing Insight Command Center, see *Administration Guide for Oracle Billing Insight*.

About the RESTful Web Services

The Oracle Billing Insight provides a RESTful Web Services programmatic interface that system integrators can use to integrate back-office systems with the Oracle Billing Insight applications and database.

The RESTful Web Services:

- Provides hundreds of APIs to cover majority of functionality provided in the online Self-Service and Assisted Service applications.
- Can be extended.
- Provides outbound Web services.
- Can trigger plug-ins to update external systems.
- Performs token-based authentication with the same authorization rules used online.
- Is locale-aware.
- Based on JAX-RS specification with Jersey Implementation (version 1.14).

For information about using Web services with Oracle Billing Insight, see *Web Services Reference for Oracle Billing Insight*.

About Oracle Billing Insight Architecture

Figure 1 shows an overview of the Oracle Billing Insight architecture.

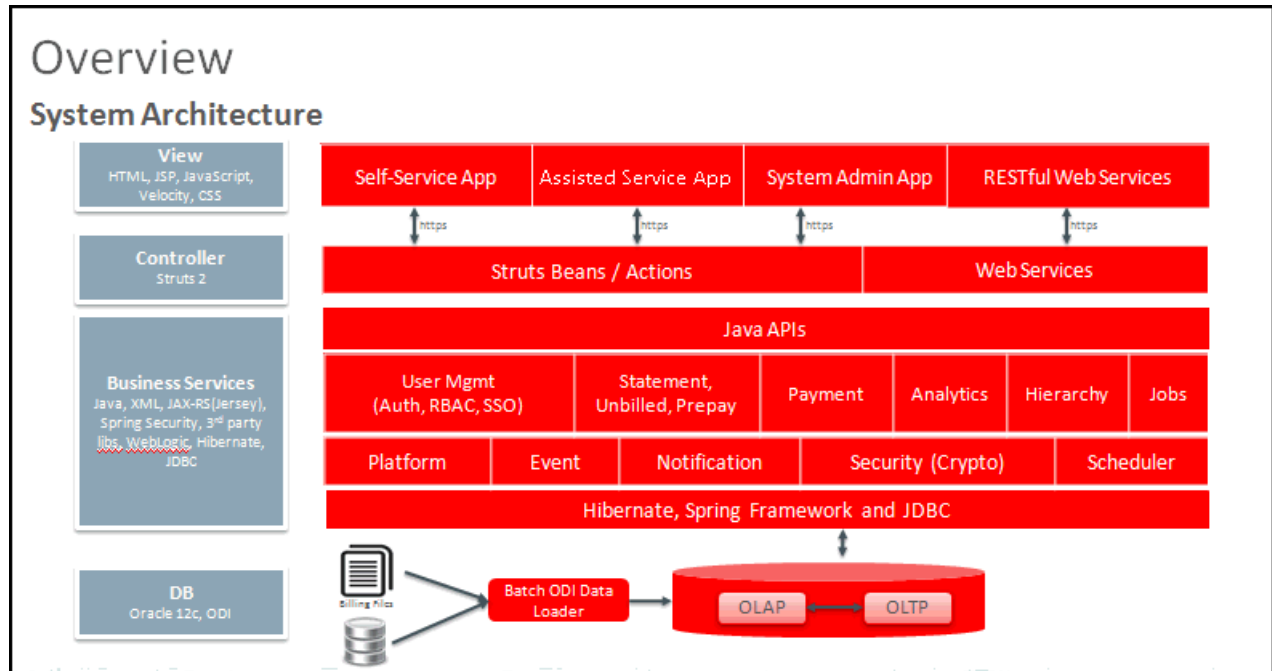


Figure 1. Overview of Oracle Billing Insight Architecture

About the Oracle Billing Insight Database

OLTP is the Oracle Billing Insight transactional data schema. Oracle Billing Insight expects access to product tables to go through the Oracle Billing Insight APIs. Do not change the existing product schema. However, you can add your own customization tables.

OLAP is the Oracle Billing Insight data warehouse. These tables are accessed directly through report XML files. For information on how to use report XML files to retrieve data from the OLAP schema, check the report XML files used to generate various billing reports. These files are defined in the *EDX_HOME*/config/rpt directory (or the *EDX_HOME*\config\rpt directory on Windows). In these directories, *EDX_HOME* is the directory where you installed Oracle Billing Insight.

Auditing Database Administration Activity

Oracle Billing Insight does not audit database administrator activity. However, to remain compliant with the Payment Card Industry Data Security Standard (PCI DSS), you must implement auditing functionality that documents each time an administrator logs in, creates new tables or attributes, deletes information including tables, attributes, or transaction details, or runs an external script against the Oracle Billing Insight database.

About Oracle Billing Insight Installation Directories

The default installation directory for Oracle Billing Insight is:

- **UNIX.** /opt/Oracle/BillingInsight
- **Windows.** Oracle\BillingInsight

It is possible to change the default directory during installation to fit your deployment environment. This guide refers to the directory where you have installed Oracle Billing Insight as *EDX_HOME*.

Table 2 describes the types of files stored in each main product directory defined under *EDX_HOME*.

Table 2. Oracle Billing Insight Directories

Directory	Description
bin	Contains scripts for Scheduler, notification template generation, and purging data.
config	Contains configuration files for Oracle Billing Insight.
db	Contains scripts for creating and configuring the Oracle Billing Insight database.
docs	Contains the Javadoc and Data Dictionary files for Oracle Billing Insight.
Input	Provides a place for you to place input files that you intend to process using batch Command Center jobs.
integration	Contains components used when integrating Oracle Billing Insight with Oracle Communications Billing and Revenue Management (BRM).
J2EEApps	Contains the Oracle Billing Insight application J2EE EAR files.
keystore	Contains the necessary files and directories for creating the Java KeyStore.
lib	Provides a location for placing downloaded third-party libraries.
logs	Provides a (recommended) location for directing and storing log output files.
META-INF	A J2EE metadata directory.
Output	Provides a place for you to direct output files generated by batch Command Center jobs.
payment	Contains the configuration files for the Oracle Billing Insight payment functionality.
repackage	Provides a location for placing downloaded LGPL libraries.
samples	Contains sample files you can use for implementing and customizing Oracle Billing Insight, and for implementing a single sign-on system.

Table 2. Oracle Billing Insight Directories

Directory	Description
temp	Used by Oracle Billing Insight for the placement of temporary files created during processing.
template	Contains the Velocity templates used for displaying UI reports.
Uninstall	Contains the executable for uninstalling Oracle Billing Insight.
xma	Contains XMA configuration files for Oracle Billing Insight.

About Application EAR Files

Each Oracle Billing Insight application is a separate EAR file, which you can deploy on the appropriate server for the application.

Self-Service Application EAR File Structure

This Self-Service application EAR file is located in the following directory, where *EDX_HOME* is the directory where you installed Oracle Billing Insight:

- **UNIX.** *EDX_HOME*/sel fservi ce/J2EEApps/sel fservi ce/webi ogi c/sel fservi ce-webi ogi c-7.1.ear
- **Windows.** *EDX_HOME*\sel fservi ce\J2EEApps\selservi ce\webi ogi c\selservi ce-webi ogi c-7.1.ear

Table 3 describes the components packaged under the directories inside the Self-Service application EAR file, `selfservice-weblogic-7.1.ear`.

Table 3. Self-Service EAR File Directories

EAR File Directory	Contents
/	This is the root directory and it contains the EJB JavaARchive (JAR) and Web ARchive (WAR) files.
/lib	Contains the list of third-party library files used by the Self-Service application.
/META-INF	Contains the J2EE META-INF directory. Oracle Billing Insight uses <code>BillingInsight</code> as its URL prefix. The prefix is defined in the <code>application.xml</code> file in this directory. Oracle Billing Insight uses relative URLs to ensure that after you change the URL prefix, the application still works. You can change this to fit your deployment environment.
/xma	Contains internal library files used by the Self-Service application. In this directory there is one JAR file called <code>api-version_number.jar</code> , which has all the public Oracle Billing Insight APIs defined. In the file name, <code>version_number</code> is the Oracle Billing Insight version.

NOTE: Some of the WAR file subdirectories are inherited from another legacy Oracle Billing Insight application and are not used directly by Oracle Billing Insight.

Table 4 describes the components packaged under the directories inside the Self-Service application WAR file, `selfservice-web-1.0-SNAPSHOT.war`:

Table 4. Self-Service WAR File Subdirectories

WAR File Directory	Contents
/	This is the root directory and contains common JSP pages.
/includes /_templates	Contains JSP page fragments used by the Self-Service application. Many of these fragments are tiles.
/_assets	Contains images, JavaScripts, and CSS files used by the Self-Service application.
/hierarchy	Contains hierarchy-related JSP pages.
/usermanagement	Contains user-management-related JSP pages.
/reporting	Contains reporting-related JSP pages.
/payment	Contains payment-related JSP pages.
/contacts	Contains contacts-related JSP pages.
/dispute	Contains dispute-related JSP pages.

Table 4. Self-Service WAR File Subdirectories

WAR File Directory	Contents
/unbilled	Contains unbilled-related JSP pages.
/WEB-INF	Contains the J2EE WAR file WEB-INF directory.
/WEB-INF/classes/azcfg/ policy	Contains the Oracle Billing Insight RBAC policy file.
/WEB-INF/classes/lib	Contains the libraries used by the WAR file.

About the Assisted Service Application EAR File

The Assisted Service application EAR file is located in the following directory, where *EDX_HOME* is the directory where you installed Oracle Billing Insight:

- **UNIX.** *EDX_HOME*/J2EEApps/csr/weblogic/csr-app-7.1.ear
- **Windows.** *EDX_HOME*\J2EEApps\csr\weblogic\csr-app-7.1.ear

Table 5 describes the components packaged under the directories inside the Assisted Service application EAR file, csr-app-7.1.ear.

Table 5. Assisted Service Application EAR File Directories

EAR File Directory	Contents
/	This is the root directory and it contains the EJB JavaARchive (JAR) and Web ARchive (WAR) files.
/lib	Contains the list of third-party library files used by the Assisted Service application.
/APP-INF	Contains common jar files.
/META-INF	Contains the J2EE EAR file META-INF directory for meta information.
/xma	Contains internal library files used by the Assisted Service application. In this directory there is one JAR file called <i>api-version_number.jar</i> , which has all the public Oracle Billing Insight APIs defined. In the file name, <i>version_number</i> is the Oracle Billing Insight version.

NOTE: Some of the WAR file subdirectories are inherited from another legacy Oracle Billing Insight application and are not used directly by Oracle Billing Insight.

Table 6 describes the components packaged under the Assisted Service application WAR file, csr-web-1.0-SNAPSHOT.war.

Table 6. Assisted Service Application WAR File Subdirectories

WAR File Directory	Contents
/	This is the root directory and contains common JSP pages.
/manage-report	Contains report-related JSP pages.
/manage-profile	Contains profile-related JSP pages.
/manage-org	Contains organization-related JSP pages.
/manage-csruuser	Contains agent user-related JSP pages.
/integration	Contains integration-related JSP pages.
/html	Contains report-related JSP pages.
/access-cust	Contains customer-access related JSP pages.
/_templates /_includes	Contains JSP page fragments used by the Assisted Service application. Many of these fragments are tiles.
/hierarchy	Contains hierarchy-related JSP pages.
/contacts	Contains contacts-related JSP pages.
/WEB-INF	The J2EE WAR file WEB-INF directory.
/WEB-INF/classes/azcfg/policy	Contains the Oracle Billing Insight RBAC policy file.
/WEB-INF/classes/lib	Contains libraries used by the WAR file.
/usermanagement	Contains user-management-related JSP pages.
/reporting	Contains reporting-related JSP pages.
/payment	Contains payment-related JSP pages.
/manage-search	Contains search-related JSP pages.
/manage-cust	Contains customer-related JSP pages.
/manage-account	Contains account-related JSP pages.
/dispute	Contains dispute-related JSP pages.
/_assets	Contains images, JavaScripts, and CSS files used by the Assisted Service application.
/META-INF	Contains the J2EE META-INF directory.

About the Web Services EAR File

The RESTful Web Services EAR file is located in the following directory, where *EDX_HOME* is the directory where you installed Oracle Billing Insight:

- **UNIX.** `EDX_HOME/J2EEApps/rs/weblogic/rs-app-7.1.ear`
- **Windows.** `EDX_HOME\J2EEApps\rs\weblogic\rs-app-7.1.ear`

About the Command Center Application EAR File

The Command Center EAR file is located in the following directory, where `EDX_HOME` is the directory where you installed Oracle Billing Insight:

- **UNIX.** `EDX_HOME/J2EEApps/commandcenter/weblogic/command-center-weblogic-7.1.ear`
- **Windows.** `EDX_HOME\J2EEApps\commandcenter\weblogic\command-center-weblogic-7.1.ear`

In general, you are not expected to modify this EAR file during deployment. Deploy the Command Center on a separate application server.

Debugging Oracle Billing Insight

Oracle Billing Insight produces various logging information to help you monitor activity and debug issues.

Oracle Billing Insight has three logging mechanisms:

- **Log4j.** Log4j is the main logging mechanism. Each EAR (application) requires different log4j files to avoid conflicting with each other.

CAUTION: Because of security concerns, update the `log4j_cc.xml` file to write Command Center log information to the Oracle Billing Insight database, not to a file. File appenders are not used. A *file appender* is a named entity that represents a specific output destination for messages. It is technically valid to write the Oracle Billing Insight Self-Service application and Assisted Service application log information to either the database or to files as specified in the `log4j.xml` and `log4j_csr.xml` files.
- **DB-logging.** Most Command Center jobs also use DB-logging for job-level information and log4j is still used to log API-level information. The DB-logging writes log information into DB tables and can be viewed from the Command Center. For details about Command Center message log files, see *Administration Guide for Oracle Billing Insight*.
- **Java-option-logging.** The logging is controlled by passing in a JVM `-D` option, to log debug-level information and mostly for development purposes. For information about using this option with payment functionality, see [“Using Payment Debug” on page 244](#).

In addition, in the majority of use cases, Oracle Billing Insight prints out the exception stack trace to the console or as part of the JSP error output page when an exception occurs. By default, Oracle WebLogic only prints error messages to the console. From the Oracle WebLogic console you can change the level of log messages to print more detailed information.

Viewing log4j Log Files

The Self-Service, Command Center, and Assisted Service applications maintain log4j files in the following directory:

- **UNIX.** `EDX_HOME/config`

■ **Windows.** *EDX_HOME\confi g*

You can configure the logging level for each application.

Table 7 lists the log4j debug tables generated by each application.

Table 7. Log4j Logging Files

Application	Log4j Files Maintained
Self-Service	<p>The Self-Service application maintains the following log4j debug files:</p> <ul style="list-style-type: none"> ■ hierarchy.log ■ reporting.log ■ umf.log ■ eBilling.log
Command Center	<p>The Command Center application maintains the following log4j debug files:</p> <ul style="list-style-type: none"> ■ log4j_eStatement.log ■ log4j_scheduler.log
Assisted Service	<p>The Assisted Service application maintains the log4j_csr.xml file.</p>

Configuring the Logging Level for Log4j Files

You can configure the logging level recorded to the log4j files.

To configure logging levels for log4j files

- 1 Open the configuration file for the application:

- **Self-Service.** log4j.xml
- **Assisted Service.** log4j_csr.xml
- **Command Center.** log4j_cc.xml

The files are located in the following directory. In the path, *EDX_HOME* is the directory where you installed Oracle Billing Insight:

- **UNIX.** *EDX_HOME/confi g*
- **Windows.** *EDX_HOME\confi g*

- 2 Set the File parameter for each application or module.

Displaying SQL Statements

One of the most useful debug features is to display the SQL statements issued to the Oracle Billing Insight database.

To view the Hibernate SQL statements

- 1 Open the persistence.xma.xml file for editing. This file is in the *EDX_HOME*/xma/config/modules directory. In this directory, *EDX_HOME* is the directory where you installed Oracle Billing Insight.

- 2 Edit the hibernate.show.sql property, changing the value from false to true:

```
<prop key=hibernate.show_sql>false</prop>
```

- 3 To be able to view the SQL binding values as well (the hibernate.show_sql property allows you to view the SQL statements only), edit the log4j files (log4j.xml, log4jcc.xml, and log4jccenter.xml) which are found in the *EDX_HOME*/config directory. Change the level for these two loggers to debug:

```
<logger name="org.hibernate.SQL" additivity="false">
<level value="TRACE"/> <appender-ref ref="cba-log"/>
</logger><logger name="org.hibernate.type" additivity="false">
<level value="error"/> <appender-ref ref="cba-log"/>
</logger>
```

- 4 These configurations apply to Hibernate-based DB access. One exception is reporting-related SQL statements, which are issued without using Hibernate. To view the report SQL statements and their binding values, add a Java -D option:

```
java -Ddatasource.debug=true
```

Accessing Oracle Billing Insight Data Dictionary and Javadoc

The Oracle Billing Insight Data Dictionary and API Javadoc are available in your product installation.

The Oracle Billing Insight Data Dictionary contains details about the OLAP and OLTP schema tables and is available in both PDF and HTML formats. The Data Dictionary files are located in the following directories:

- *EDX_HOME*/docs/datadictionary/html/BillingInsight70.zip
- *EDX_HOME*/docs/datadictionary/pdf/BillingInsight70_OLAP.pdf
- *EDX_HOME*/docs/datadictionary/pdf/BillingInsight70_OLTP.pdf

In the paths, *EDX_HOME* is the directory where you installed Oracle Billing Insight.

To access Oracle Billing Insight Javadoc

- 1 Unzip the apidoc.jar file found in the following directory.

- **UNIX.** *EDX_HOME*/docs/api
- **Windows.** *EDX_HOME*\docs\api

- 2 Open the index.html file.

Identifying the Installed Version of Oracle Billing Insight

The installed product version of Oracle Billing Insight is stored in the EDX_VERSION OLAP database table.

3

About Customizing Oracle Billing Insight

This chapter covers general information to get started customizing your application. It includes the following topics:

- [Best Practices for Customizing Oracle Billing Insight on page 27](#)
- [Non-Permitted Database Changes on page 28](#)
- [About User Interface Files on page 28](#)
- [Customizing the Default UI CSS on page 29](#)
- [About Custom JSP Pages and Action Classes on page 31](#)
- [About Velocity Templates on page 31](#)
- [About Customizing Reports on page 31](#)
- [Customizing the User Interface for Multiple Branding on page 31](#)
- [Localizing the User Interface on page 32](#)

Best Practices for Customizing Oracle Billing Insight

It is recommended that you follow these best practice guidelines when customizing the Self-Service or Assisted Service applications:

- Oracle Billing Insight provides a set of core functionality in the applications. A sample UI, included with the product, demonstrates these functions. The contract between Oracle Billing Insight core and the UI is a set of APIs. These APIs and the Java-docs are contained in the API JAR file of the EAR file. You must use these APIs for your customization purposes. Do not modify or bypass these APIs unless explicitly instructed in this guide.
- Because of the complexity of the Hierarchy Management UI, it is recommended that you try and keep your UI as close as possible to the sample hierarchy UI to reduce your workload.
- The functions exposed by the APIs exceed the ones demonstrated through the sample UI. Consult the API Java-docs and other topics of this guide for details. You can customize the Oracle Billing Insight application to take advantage of these functions.
- When you have to change existing Oracle Billing Insight files, such as a JSP or a Velocity template, you can work either on an existing file or copy it and work on the copy. The second method could be more time consuming but will save you more time when migrating. Keep the history of customization changes in a source control system.
- Never make any changes that could break the backward compatibility of the DB schema, such as changing the column type or renaming a column or a table.

- The Spring XMA configuration files are the core configuration files of Oracle Billing Insight and you must not modify them unless instructed in this document Oracle Billing Insight uses Spring to manage JavaBean creation and transactions, and also for the configuration of Hibernate. The Spring XMA configuration files are stored in the *EDX_HOME/xma* directory. Possible reasons to customize these files include:
 - To enable Hibernate show_sql.
 - To extend Hierarchy Management, such as adding a new link target type, reimplementing a hierarchy search interface such as *IAssignedObjectProvider*, inserting a new loader into the *OLTPProductionLoader* job, configuring a new event handler to handle hierarchy events, or configuring the hierarchy UI behavior. For more information about extending Hierarchy, see *Hierarchy Developer's Guide for Oracle Billing Insight*.
 - To configure batch report jobs, to send email, for example.

Non-Permitted Database Changes

To maintain operability of Oracle Billing Insight as well as the ability to upgrade to future releases, you must not alter or remove base objects in the following ways:

- Columns in base tables must not be altered in any way (changed, removed, or added).
- Columns in indexes must not be altered or removed.
- Tables must not be renamed or removed.
- Base triggers and sequences must not be renamed or removed.
- Base indexes must not be altered or removed.
- Base packages must not be altered or removed.
- Base procedures must not be altered or removed.
- Base functions must not be altered or removed.

About User Interface Files

UI customization can range from changing the look-and-feel or adding your own Struts action classes. The user interface-related files are located in the following Web application folders, packaged in the application EAR files:

- **The *_assets*.** Contains all images, CSS files, and scripts used in the application.
- **The *_templates*.** Template files for formatting and screen orientation.

Every JSP can extend any one of the templates. JSP files are stored in their respective module folders.

By default, all Oracle Billing Insight screens pick up their styles from a common file, *swan.css*, for Web rendering. This file is imported in all the templates. The print friendly view uses the *swan_print.css* file. The stylesheet defines the styles for all defined classes. You can define as many stylesheets as required, however, leave the class name the same as it is in *swan.css*.

The template files must import the corresponding customized CSS files as necessary, then the JSPs will use the new styles.

The Oracle Billing Insight UI is based on Tiles definitions. You can optionally modify the Tiles definitions file to use your own Tiles. The Tiles definitions are located in the `sel fservi ce-webl ogi c-7.1.ear/sel fservi ce-web-1.0-SNAPSHOT.war/WEB-INF/classes` directory.

In some cases, when you use the Oracle Billing Insight presentment engine to generate a report or a search page, the result of the query is not presented by JSP, instead, a set of Velocity templates are used. These templates are defined in the `EDX_HOME/templates/common/lib` and reporting directories. In the path, `EDX_HOME` is the directory where you installed Oracle Billing Insight. Do not touch the VM files in the lib directories. The files under reporting can be customized if necessary. However, most of the time you can customize reports using report XML files without touching the VM files.

About Repackaging EAR Files

Whenever you want to modify a JSP, add a new action class, or add an EJB, you must repackage the EAR file.

When repackaging the EAR file, make sure you do not remove existing components and only modify the components that are recommended as modifiable in this guide, such as JSP pages, CSS files, the `app-resources.jar` file, and so on.

Customizing the Default UI CSS

Oracle Billing Insight uses Cascading Style Sheets (CSS) as a mechanism for adding style, such as fonts, colors, and spacing, to Web documents. You can modify the default CSS file, `swan.css`, for use by all defined classes.

The user interface of Oracle Billing Insight uses industry standards, including consistent page layout, navigation bars, bread crumbs, and logically labeled controls to make a consistent and intuitive user experience. The use of Cascading Style Sheets ensures separation of style from presentation.

The page layout of the Oracle Billing Insight application consists of the following body areas:

- Pagewrap elements:
 - Top_page
 - Logo
 - Userlinks
 - Tabbar and tabs
 - Subtabbar and subtabs
 - Sidecontent
 - Quicklink (Header and Quicklinklist)
 - Reportcontext (Header and Reportcontextlist)

- Maincontent
 - Breadcrumb
 - Pageheading
 - Pagetabs (When applicable)
 - ErrorMessage
 - successmessage
 - Subtitle (Repeats at the top of each module)
 - Buttonbar downloadPrint (When applicable)
 - Buttonbar (When applicable)
 - Contextbox
 - Infomessage (When applicable)
 - Buttonbar (When applicable)
- Clearline
- Footer

Oracle Billing Insight calls the CSS file from the main templates, which are in the `_templates` directory. The JSP file names are:

- `simplelayout.jsp`
- `simplelayout1.jsp`
- `popupLayout.jsp`
- `paymentLayout.jsp`
- `dashBoardLayout.jsp`

To customize the default UI CSS

- 1 If you configured the dynamic CSS file feature during installation, then go to the following directory:

- **UNIX.** `EDX_HOME/confi g/_assets/`
- **Windows.** `EDX_HOME\confi g_assets/`

Otherwise, explode the Oracle Billing Insight EAR file, and go to the `/_assets` directory in the WAR file.

- 2 Copy and rename the default `swan.css` file to a new directory, such as `/_assets/MyDefaul t/MyDefaul t.css`.
- 3 Edit the new default CSS file for the preferred look and feel.
- 4 Add any new images linked in the CSS file to the new directory.

- 5 Modify the `CSS_REFERENCE` field in the `EDX_SYS_BRAND` table in the OLTP schema to point to the new default directory.

The default `swan.css` file is referenced by the default brand key value in the `CODE` field of the `EDX_SYS_BRAND` table.

- 6 Redeploy the EAR file, if necessary.

About Custom JSP Pages and Action Classes

It is possible to add your own UI components, such as JSP, JavaScripts and so on. After you create your own action class, you must modify the `struts.xml` file to register it.

About Velocity Templates

The Oracle Billing Insight reporting UI is based on Velocity templates, an open source project. The product offers a set of preconfigured templates to implement common UI features such as paging, sorting, charting, print-friendly, and download.

You can customize these preconfigured templates either by modifying them directly or by copying and then modifying. If you do copy and modify, then configure the report XML files to pick up your new templates.

About Customizing Reports

All of the report XML files are for the preconfigured Oracle Billing Insight user interface. The report XML files can be found in the following directory, where `EDX_HOME` is the directory where you installed Oracle Billing Insight:

- **UNIX.** `EDX_HOME/config/rpt`
- **Windows.** `EDX_HOME\config\rpt`

You can add your own report XML files, using either the default Velocity templates provided with Oracle Billing Insight or your own templates. For details on creating reports, see [Chapter 7, “Using the Reporting Engine.”](#)

Customizing the User Interface for Multiple Branding

Oracle Billing Insight supports multiple branding of the UI. You can create a different look-and-feel Cascading Spreadsheet (CSS) for each brand. Using Java APIs, you can set a brand code to each user's profile at enrollment. When a user logs into the Self-Service application and is authenticated, they are presented with the UI CSS for the brand code stored in their profile.

To customize the user interface for multiple branding

- 1 If you configured the dynamic CSS file feature during installation, then go to the following directory:
 - **UNIX.** `EDX_HOME/config/_assets/`
 - **Windows.** `EDX_HOME\config_assets/`Otherwise, explode the Oracle Billing Insight EAR file. For more information about updating CSS files, see [“Customizing the Default UI CSS” on page 29](#).
- 2 Create customized CSS files for each brand:
 - a Create a new subdirectory for each brand.
 - b Copy and rename the default `swan.css` file to use as a starting point for creating custom, brand-specific CSS files. Edit the new default CSS files for the preferred look and feel, and link any new images within the files.
 - c Place the new CSS files and any linked image files in the corresponding directories, for example:
 - `/_assets/Brand_1/Brand_1.css`
 - `/_assets/Brand_1/Brand_1_Logo1.gif`
 - `/_assets/Brand_1/Brand_1_Mascot1.gif`
 - `/_assets/Brand_2/Brand_2.css`
 - `/_assets/Brand_2/Brand_2_ImageSmall.gif`
 - `/_assets/Brand_2/Brand_2_ImageMed.png`
- 3 Populate the `EDX_SYS_BRAND` table with a record for each brand code you define. Include the brand code, device (NULL, which defaults to Web), brand description, and corresponding CSS file name and location.
- 4 Customize the user API to capture and set a brand key, or code, in each user’s profile when the user enrolls.

When a user logs in and is authenticated, Oracle Billing Insight reads the brand key stored in the user’s profile and references the `EDX_SYS_BRAND` table to find the corresponding CSS file to use for the session.

Localizing the User Interface

The Oracle Billing Insight is preconfigured to use U.S. English in the Self-Service, Assisted Service, and RESTful Web Services application interfaces. Translated resource bundle files are also provided for the following languages:

- Brazilian Portuguese
- Dutch
- German
- Italian

- Spanish (Worldwide)
- Turkish

To set one of these languages as the system default, see [“Changing the Default Language” on page 33](#).

Each user can set a preferred language for themselves which automatically appears each time that user logs in. Users can also choose a different language for an individual session.

You can localize your implementation for additional languages by copying and translating the English-language resource bundle (property) files and configuring other required files. For details, see [“Localizing the User Interface for Additional Languages” on page 33](#).

Changing the Default Language

You can change the default language that displays system-wide for the Self-Service, Assisted Service, and RESTful Web Services applications to any language for which translated resource bundles exist, including those provided with Oracle Billing Insight and any localized language file bundles you create.

The list of languages provided with Oracle Billing Insight are listed in [“Changing the Default Language” on page 33](#). To create additional language resource file sets, see [“Localizing the User Interface for Additional Languages” on page 33](#).

To change the default language displayed in the user interface

- 1 In the EDX_SYS_LANG database table, update the is_default field for the language (record) you want to use as the default. Specify the field value as 1 (Yes).
- 2 In the record for the language that you are removing as the default, specify 0 (No) in the is_default field.

NOTE: The language you remove as the default will still be available to individual users to choose as their personal default or to use for an individual session.

Localizing the User Interface for Additional Languages

You can add languages (other than those provided with Oracle Billing Insight) by copying and translating the required English-language resource bundle (property) files and configuring the additional files. Each language resource file is shared by the Self-Service, Assisted Service, and RESTful Web Services applications.

NOTE: Oracle Billing Insight comes with language resource files for U.S. English, Brazilian Portuguese, Dutch, German, Italian, Spanish (Worldwide), and Turkish. For details on using one of these languages, see [“Changing the Default Language” on page 33](#).

NOTE:

Each language requires the following set of files (shown for U.S. English):

- ApplicationMessages_en_US.properties

- ApplicationLabels_en_US.properties
- NotificationsEmail_en_US.properties
- NotificationsSMS_en_US.properties
- Period_en_US.properties
- CurrencyText_en_US.properties
- CommonFieldValidator_en_US.properties

To localize the user interface for additional languages

- 1 Verify that Oracle Billing Insight is installed and deployed.
- 2 For each new language, add a record in the EDX_SYS_LANG database table. For example, the following SQL statement adds traditional Spanish and sets it as the default language:

```
insert into edx_sys_lang(ID, code, name, is_default) values(1, 'es_ES', 'Español', 1);
```

where:

- *ID* is the ID you want to use as the primary key for this language.
- *code* is the language code. The format can be *language_country* (such as en_US, zh_CN, or ja_JP) or *language_country_variant*, such as en_US_Traditional_WIN. The *language* argument is a valid ISO-639 Language Code in two lower-case letters. The *country* argument is a valid ISO-3166 Country Code in two upper-case letters.

The *variant* argument is a vendor- or browser-specific code, for example: WIN for Windows, MAC for Macintosh, and POSIX for POSIX. The *variant* argument can have two parts, separated by an underscore. For example, the code for Traditional Spanish on Windows is es_ES_Traditional_WIN.

- *name* is the name of the language that appears when a user selects a language in the interface.
- *is_default* indicates whether this language is to be the default (0 is No, 1 is Yes).

If you set this new language the default, in the record for the language that you are removing as the default, specify 0 (No) in the *is_default* field.

- 3 In the CurrencyText.properties file, for each application and in each language, add mappings between the currency code and text for any currencies you want to use.

The currency text appears on reports and charts. The following values (for the American dollar, Chinese yuan, and Euro) are included in the file by default:

```
USD. CurrencyText=Dollars
CNY. CurrencyText=CNY
EUR. CurrencyText=Euro
```

- 4 Create new resource bundle property files for each new language:
 - a Make copies of the following language resource property files, one for each language:

- ❑ **ApplicationMessages_en_US.properties.** Message text, such as validation and error messages that appear in the user interface.
 - ❑ **ApplicationLabels_en_US.properties.** Text of tabs, labels, and titles that appear on the user interface Web pages.
 - ❑ **NotificationsEmail_en_US.properties.** Contains text strings used to compose email notifications.
 - ❑ **NotificationsSMS_en_US.properties.** Contains text strings used to compose SMS messages.
 - ❑ **Period.properties.** Contains monthly time periods that appear in lists in the user interface.
 - ❑ **CurrencyText.properties.** Contains the currency name to display in reports and charts for each language.
 - ❑ **CommonFieldValidator.properties.** Contains the validation rule for certain fields.
- b** Append the language code (typically in the format *_LanguageCode_Country*) to the new file names. Use the codes that you defined in the database table in [Step 2 on page 34](#).
- For example, for Spanish, the copy of the application resource messages file must be called *ApplicationLabels_es_ES.properties*. The files are located in the following directories:
- ❑ **UNIX.** *EDX_HOME/confi g/resourcebundl e/com/edocs/appl i cati on/resources*
 - ❑ **Windows.** *EDX_HOME\confi g\resourcebundl e\com\edocs\appl i cati on\resources*
- Place all new and updated property files in the same directories as the English language files.
- NOTE:** If you want to customize to the pre-configured notification content, then make those changes in the property files before creating copies for localization.

- 5** Translate the appropriate content in each new property file.
- 6** In the *ApplicationLabels_LanguageCode_Country.properties* file, for each language, update the file to customize the date, time, and number formats used in the user interface. Specifying custom values lets you override the default Java language formats. You can also add new patterns to the file.

To apply a different date format in a report (such as using the short date format instead of the medium format), update the report XML file with the pattern you prefer. The report XML files are found in the *EDX_HOME/confi g/rpt* directory. Also update the report XML files where you want to apply any new patterns.

Update the following code in the *ApplicationLabels_LanguageCode_Country.properties* files:

```
## Global Patterns used for reporting engine
#representation of integer
global.pattern.number.integer=#,##0
#representation of a decimal
global.pattern.number.decimal=#,##0.00
#representation of a percentage number
global.pattern.number.percent=#0.00%
#representation of amount value with currency
global.pattern.number.amount=\u00a4#,##0.00
```

```

global . pattern . number . amount2=\u00a4#, ##0. 00; \u00a4( #, ##0. 00)
global . pattern . number . amount3=#, ##0; ( #, ##0)
global . pattern . number . amount4=#, ##0. 00; ( #, ##0. 00)
global . pattern . date . short=M/d/yy
global . pattern . date . long=MMM/dd/yyyy
global . pattern . date . input=MM/dd/yyyy
global . pattern . date . j query=MM/dd/yy
global . pattern . time . short=HH: mm
global . pattern . time . long=HH: mm: ss
global . pattern . date . di sp=MMM dd, yyyy
global . pattern . date . expi ration=MM/yyyy
global . pattern . date . detai ls=MM/dd/yyyy hh: mm aaa
global . pattern . thousands=,
global . pattern . deci mal Pl aces=2
global . pattern . date . medi um=MM/dd/yyyy
global . pattern . date . ti me=MM/dd/yyyy HH: mm: ss
global . pattern . deci mal Poi nt=.
#The separator and order of hh, mm and ss can be changed, but letters hh, mm, ss
can not be changed.
global . pattern . durati on=hh: mm: ss
#The letter, separator and the order of the string can be changed. But the order
and separator need to be consistent with the value of global . pattern . durati on
global . pattern . durati onDesc=hh: mm: ss

```

NOTE: The amount3 and amount4 patterns display numbers that are not currency amounts.

- 7 Localize email and SMS notification templates by generating a template XML file for each language:
 - a Modify the template generator script. Update the value of *EDX_HOME*, using the full path names for your installation, in the generateNotificationTemplate.sh file, or the generateNotificationTemplate.cmd file on Windows. The files are in the following directories:
 - ❑ **UNIX.** *EDX_HOME/bin/notification/*
 - ❑ **Windows.** *EDX_HOME\bin\notification*

The template generator script generates an email notification template XML file based on the notification properties file for each language.
 - b Run the email and SMS template generator utility. You can run this tool in one of two ways:
 - **Generate all new language email and SMS templates in batch.** To generate a batch of XSL template files in the corresponding languages, create a text file that contains each language code on separate lines and place the file in the *EDX_HOME/bin/notification* directory (or the *EDX_HOME\bin\notification* directory on Windows):
 - ❑ **UNIX.** *./generateNotificationTemplate.sh -f filename -t templateType*
 - ❑ **Windows.** *generateNotificationTemplate.cmd -f filename -t templateType*

where:

 - ❑ *filename* is the name of the language code text file you created.

- *templateType* is the template type, which is used to generate different notifications according to the notification category. Specify either SMS or email.

For example, if you add the following content to a text file called `languages.txt`, and use the UNIX command `./ generateNoti fi cati onTempl ate. sh -f l languages -t email`, then the command generates two email templates called `template_ zh_CN.xml` and `template_ ja_JP.xml`.

- **Generate a single template for one new language.** Use the following commands to generate a single XSL template file for a new language:

- **UNIX.** `./generateNoti fi cati onTempl ate. sh -l code`
- **Windows.** `generateNoti fi cati onTempl ate. cmd -l code`

where:

- *code* is the language code.
- *templateType* is the template type, which is used to generate different notifications according to the notification category. Specify either SMS or email.

For example, to generate a Chinese email template file (called `template_zh_CN.xml`), use the following command:

- **UNIX.** `./generateNoti fi cati onTempl ate. sh -l zh_CN -t email`
- **Windows.** `generateNoti fi cati onTempl ate. cmd -l zh_CN -t email`

To generate a Chinese SMS template file (called `template_zh_CN.xml`), use the following example command:

- **UNIX.** `./generateNoti fi cati onTempl ate. sh -l zh_CN -t sms`
- **Windows.** `generateNoti fi cati onTempl ate. cmd -l zh_CN -t sms`

For SMS templates, Oracle Billing Insight names the file with the prefix `template_sms`. For email templates, the file name prefix is the word `template`. The new template files generated are saved in the `EDX_HOME/config/noti fi cati on/templ ates` directory.

- 8 For each new language, make copies of the following template files used for generating PDF reports, appending the locale code (`languageCode_Country`) to the new file names. The template files are in the `EDX_HOME/templ ate/pdf` directory (use back slashes (\) on Windows). Place the new files in the same directory.

- `PrintSummary.rtf`
- `StatementSummary.rtf`
- `telco_std_r1.rtf`
- `telco_std_r6.rtf`
- `telco_std_r13.rtf`

NOTE: Oracle Billing Insight comes preconfigured with a set of template files for American English (appended with the American English locale code, `_en_US`).

- 9 By default, Oracle Billing Insight automatically detects updates to the external resource files every five minutes. If you have turned this feature off in the global configuration file, then you must manually restart the server for the localization changes to take effect. For more information about automatic detection of changes to resource files, see ["Changing the Frequency of Automatic Resource File Updates" on page 38](#).

CAUTION: If you want to make any customizations to email notification text after localization, then make the changes to the notification property files for the language and regenerate the corresponding template file. Changes made directly to a template file will be lost if the template regeneration runs again for that language.

Changing the Frequency of Automatic Resource File Updates

Oracle Billing Insight automatically detects any changes made to external resource files, including localization and email content files. You can configure the frequency with which Oracle Billing Insight automatically detects these files. The default is every five minutes. You can optionally turn off the automatic detection feature, requiring you to restart the server if you make changes to resource bundles.

To configure the frequency of automatic detection of resource files

- 1 In the globalConfig.xma file, found in the `EDX_HOME\xma\config\modules` directory (or the `EDX_HOME/xma/config/modules` directory on Windows), specify the value of the `autoDetectExternalResourcePeriod` property, in minutes:

```
<property name="autoDetectExternalResourcePeriod"><value>5</value></property>
```
- 2 To disable the auto-detect functionality and require that the server restart to reload resources updates, set the parameter value to zero.
- 3 Restart the server.

Adding Your Own Custom Language Resource Bundle Files

In addition to the preconfigured language resource files provided with a language bundle in Oracle Billing Insight, you can implement additional customized resource bundle files you create, such as `customProperty.properties`.

To add your own defined resource bundle file

- 1 Place the new resource bundle file in the following location:
 - **UNIX.** `EDX_HOME/config/resourcebundle/com/edocs/application/resources`
 - **Windows.** `EDX_HOME\config\resourcebundle\com\edocs\application\resources`
- 2 Follow these steps for proper currency representation, which enables the currency symbols and currency text to display properly in report, payment, and mail notification content:

- a Populate the database. For each currency type, add the currency symbol to the CURRENCY_TYPE_SYMBOL column in the olap.EDX_RPT_CURRENCY_TYPE_DIM table.
- b Update the currency properties file. For each application and each language, add mappings between the currency code and text to the CurrencyText.properties file for any currencies you want to use. For example, the following values (for the American dollar, Chinese yuan, and Euro) are included in the file by default:

```
USD.CurrencyText=Dollar
CNY.CurrencyText=CNY
EUR.CurrencyText=Euro
```

- 3 Update the struts.xml file, located in the following directories, for both the Self-Service and Assisted Service applications:

- **UNIX.** For the Self-Service application:

```
EDX_HOME/J2EEApps/sel fservi ce/weblogi c/sel fservi ce-weblogi c-7. 1. ear/
sel fservi ce-web-1. 0-SNAPSHOT. war/WEB-INF/classes
```

For the Assisted Service application:

```
EDX_HOME/ J2EEApps/csr/weblogi c/csr-app-7. 1. ear/csr-web-1. 0-SNAPSHOT. war/
WEB-INF/classes
```

Windows. For the Self-Service application:

```
EDX_HOME\J2EEApps\selservi ce\weblogi c\selservi ce-weblogi c-
7. 1. ear\selservi ce-web-1. 0-SNAPSHOT. war\WEB-INF\classes\
```

For the Assisted Service application:

```
EDX_HOME\J2EEApps\csr\weblogi c\csr-app-7. 1. ear\csr-web-1. 0-SNAPSHOT. war\WEB-
INF\classes\
```

Change the value of struts.custom.i18n.resources to add the new resource file. The current value is:

```
<constant name="struts.custom.i18n.resources" value="com/edocs/appl i cati on/
resources/Appl i cati onLabel s, com/edocs/appl i cati on/resources/
Appl i cati onMessages, com/edocs/appl i cati on/resources/CommonFi el dVal i dator, com/
edocs/appl i cati on/resources/CurrencyText, com/edocs/appl i cati on/resources/
Period" />
```

- 4 Edit the global.xml file, located in the following directory:

- **UNIX.** *EDX_HOME*\confi g\rpt\common

- **Windows.** *EDX_HOME*\$confi g\rpt\common

Add a line with the following content:

```
<resourceBundl e name="com/edocs/appl i cati on/resources/enabl eMessageResources" />
```

Place it in the following section in the global.xml file:

```
<local i zer enabl eMessageResources="true" default tCode="0">
  <resourceBundl e name="com/edocs/appl i cati on/resources/Appl i cati onLabel s" />
  <resourceBundl e name="com/edocs/appl i cati on/resources/Appl i cati onMessages" />
```

```
>
  <resourceBundle name="com/edocs/appl i cati on/resources/CommonFi el dVal i dator"
/>
  <resourceBundle name="com/edocs/appl i cati on/resources/CurrencyText" />
  <resourceBundle name="com/edocs/appl i cati on/resources/Peri od" />
</l ocal i zer>
```


4

Customizing User Management

This chapter covers the public APIs available for customizing the Oracle Billing Insight user management functionality. It includes the following topics:

- [Deactivating and Reactivating the Bootstrap User on page 41](#)
- [Customizing the Assisted Service User Password Update Requirements on page 43](#)
- [Customizing the Administrator User Password Validation Rule on page 43](#)
- [Customizing End User and Agent Password Validation Rules on page 44](#)
- [Customizing User Enrollment on page 46](#)
- [Customizing the Validation Code Generator on page 47](#)
- [Customizing Account Lockout on page 48](#)
- [Customizing User Authentication and Authorization With Spring Security on page 49](#)
- [About Resource Authorization on page 51](#)
- [Adding a New User Role on page 55](#)
- [Adding a New Permission Set on page 56](#)
- [About Deleted Users on page 58](#)
- [Input File Format for the DefUsrAcctRelationship Job on page 59](#)
- [Input File Format for Migrating Consumer Users in Batch on page 59](#)
- [Input File Format for Deleting Consumer Users in Batch on page 61](#)

Deactivating and Reactivating the Bootstrap User

You can deactivate and reactivate the bootstrap user ID in the Assisted Service application and in the Command Center when needed. You control bootstrap user access to each of these application independently using SQL scripts.

The bootstrap user ID and password are created during the installation of Oracle Billing Insight, and enable you to create administrative users in the Command Center and Assisted Service applications.

The Payment Card Industry Data Security Standard (PCI DSS) requires that you disable the bootstrap user ID after creating administrator users in these applications.

Command Center Application

Follow these steps to deactivate or reactivate the bootstrap user ID in the Command Center application.

To deactivate or reactivate the bootstrap user ID in the Command Center application

- 1 Log into the Oracle Billing Insight instance using SQL*Plus, not as SYSDBA:

OLTP schema user_name/OLTP schema password@EBILL TNS name

where:

- *OLTP schema user_name* is the name of the OLTP schema user.
- *OLTP schema password* is the password of the OLTP schema user.
- *EBILL TNS name* is the TNS name for the Oracle Billing Insight instance.

- 2 To deactivate the bootstrap user, run the following command:

```
SQL>exec EDX_PKG_BOOTSTRAPUSER.DISABLE_CC_DEFAULT_ADMIN
```

```
SQL>commit;
```

To reactivate the bootstrap user, run the following command:

```
SQL>exec EDX_PKG_BOOTSTRAPUSER.ENABLE_CC_DEFAULT_ADMIN
```

```
SQL>commit;
```

Assisted Service Application

Follow these steps to deactivate or reactivate the bootstrap user ID in the Assisted Service application.

To deactivate or reactivate the bootstrap user ID in the Assisted Service application

- 1 Log on to the Oracle Billing Insight instance using SQL*Plus (not as SYSDBA):

OLTP schema username/OLTP schema password@EBILL TNS name

where:

- *OLTP schema username* is the name of the OLTP schema user.
- *OLTP schema password* is the password of the OLTP schema user.
- *EBILL TNS name* is the TNS name for the Oracle Billing Insight instance.

- 2 To deactivate the bootstrap administrator's ID, run the following command:

```
SQL>exec EDX_PKG_BOOTSTRAPUSER.DISABLE_CSR_DEFAULT_ADMIN
```

```
SQL>commit;
```

To reactivate the bootstrap administrator's ID, run the following command:

```
SQL>exec EDX_PKG_BOOTSTRAPUSER. ENABLE_CSR_DEFAULT_ADMIN
SQL>commi t;
```

Customizing the Assisted Service User Password Update Requirements

Oracle Billing Insight forces agent users of the Assisted Service application to change his or her password every 90 days, and the new password cannot be the same as any of the last 4 passwords used by the same user. You can change these values.

To customize the Assisted Service user password update requirements

- Open the security.xma.xml file, located in the following directory:
 - **UNIX.** *EDX_HOME*/xma/confi g/modul es/securi ty
 - **Windows.** *EDX_HOME*\xma\confi g\modul es\securi ty

Modify the values in the CSRPasswordManageRule JavaBean:

```
<bean i d="CSRPasswordManageRul e"
cl ass="com. edocs. common. securi ty. authenti cate. PasswordManageRul e"
scope="si ngl eton">
<property name="daysBeforeExpi ration"> <val ue>90</val ue> <!--User password wi ll
be expi red i n the gi ven days after created--></property><property
name="mi nUni queNumOfPwd"> <val ue>4</val ue> <!--New password can not be the same
as the l ast gi ven password used by the same user--></property>
</bean>
```

Customizing the Administrator User Password Validation Rule

A database administrator can customize the password validation rule for the Command Center administrator using a regular expression.

To customize the administrator user password

- 1 Log on to the Oracle Billing Insight Oracle Billing Insight database instance using SQL*Plus (not as SYSDBA):

```
OLTP schema username/OLTP schema password@EBILL TNS name
```

where:

- *OLTP schema username* is the name of the OLTP schema user.
- *OLTP schema password* is the password of the OLTP schema user.
- *EBILL TNS name* is the TNS name for the Oracle Billing Insight instance.

- 2 Enter the following command. In this command, *param_password_rule* is the regular expression with the new password rule you want to implement:

```
SQL>exec change_pwd_validate_rule(param_password_rule) ;
```

Customizing End User and Agent Password Validation Rules

You can customize the password rules for Self-Service application users and Assisted Service application users, or agents, passwords.

You can modify the strength of a password by customizing the regular expression rule in each use case where the user or agent enters a password in Oracle Billing Insight. You can specify different password validation rules for user and agent passwords. The default password rules requires that the password have at least one capital letter, one lowercase letter, one number, and no spaces.

You can also change the minimum and maximum password lengths, though the minimum password length cannot be less than 7 as required by the Payment Card Industry Data Security Standard (PCI DSS).

You must use the same password validation rule each time a user or an agent user enters his or her password.

To customize user or agent user passwords

- 1 To customize the regular expression rule for password input validation, follow these steps:
 - **User Passwords.** Update the value of the `common.field.pwd.rule` key defined in the `commonFieldValidator.properties` file. This file is in the `EDX_HOME\config\resourcebundle\com\edocs\application\resources\` directory (or the `EDX_HOME/config/resourcebundle/com/edocs/application/resources` directory on Windows), where *EDX_HOME* is the directory where you installed Oracle Billing Insight.
`common.field.pwd.rule=^(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9])[^\s]*$`
 - **Agent User Passwords.** Edit the following expression in the individual validation XML files for each type of agent user password you want to customize. For the validation file names, see [Table 8 on page 45](#).
`^(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9])[^\s]*$`
- 2 To customize the minimum and maximum length file for a password, edit the individual XML validation files for each type of password you want to customize. For the validation file names, see [Table 8 on page 45](#). In the `<field name="password">` section, update the `minLength` and `maxLength` parameter values.

Table 8 lists the name of the validation XML file to update for each type of password.

Table 8. Self-Service and Assisted Service User XML Files for Validating the Password

Password Type and Use Case	XML Validation File	XML File Location
User - Enrollment	SecurityInfoAction-validation.xml	<i>EDX_HOME\J2EEApps\selfservice\weblogic\selfservice-weblogic-7.1.ear\selfservice-web-1.0-SNAPSHOT.war\WEB-INF\classes\com\edocs\application\common\usermanagement\actions\</i>
User - Forgot Password and Reset Password	PasswordAction-Password_reset-validation.xml	<i>EDX_HOME\J2EEApps\selfservice\weblogic\selfservice-weblogic-7.1.ear\selfservice-web-1.0-SNAPSHOT.war\WEB-INF\classes\com\edocs\application\common\usermanagement\actions\</i>
User - Manage Profile	PasswordAction-Password_change-validation.xml	<i>EDX_HOME\J2EEApps\selfservice\weblogic\selfservice-weblogic-7.1.ear\selfservice-web-1.0-SNAPSHOT.war\WEB-INF\classes\com\edocs\application\common\usermanagement\actions\</i>
Agent (CSR) User - Agent Enrollment	CSREnrollAction-validation.xml	<i>EDX_HOME\J2EEApps\csr\weblogic\csr-app-7.1.ear\csr-web-1.0-SNAPSHOT.war\WEB-INF\classes\com\edocs\application\csr\profile\actions\</i>
Agent (CSR) User- Manage Agent Profile	PasswordAction-changePassword_update-validation.xml	<i>EDX_HOME\J2EEApps\csr\weblogic\csr-app-7.1.ear\csr-web-1.0-SNAPSHOT.war\WEB-INF\classes\com\edocs\application\csr\profile\actions\</i>
Agent (CSR) User - Agent Forgot Password and Reset Password	PasswordAction-ResetPwdConfirm_confirmPwd-validation.xml	<i>EDX_HOME\J2EEApps\csr\weblogic\csr-app-7.1.ear\csr-web-1.0-SNAPSHOT.war\WEB-INF\classes\com\edocs\application\csr\profile\actions\</i>

Customizing User Enrollment

You can customize the User Enrollment use case using XMA and APIs. You can configure the IUserService JavaBean in the userService.xma.xml file, located in the following directory:

- **UNIX.** *EDX_HOME*/xma/config/modules/services
- **Windows.** *EDX_HOME*\xma\config\modules\services

Using the User Enrollment API

When customizing user enrollment, you can call the enrollB2BUser or enrollB2CUser APIs to enroll business or consumer users, for example:

```
IUserService usrService=EBillingServiceFactory.getUserService();
usrService.enrollB2BUser(c_user, role, user.getUserProfile().getEmail());
```

Create a JavaBean called com.edocs.common.api.services.IUserEnrollProfile:

```
IUserEnrollProfile enrollProf = new UserEnrollProfile();
```

Use the following code to set the properties. This example is from com.edocs.application.common.usermanagement.actions.B2BEnrollAction:

```
enrollProf.setRole(role);
enrollProf.setCompanyId(companyId);
enrollProf.setServiceAgreementExtKey(saExtKey);
enrollProf.setAccountExtKeyList(acctExtKeyList);
enrollProf.setAdminEmail(adminUser.getUserProfile().getEmail());
```

The following code is used for enrolling a single business user:

```
usrService.enrollB2BUser(newUser, enrollProf, audit);
```

Using the Bulk Enrollment API

To use the bulk enrollment API, pass the input stream CSV file for bulk enrollment, the filename, import time, administrator user, and audit as shown in the following code. The method enrolls the users in the file and returns the success number:

```
IUserService usrService = EBillingServiceFactory.getUserService();
int succeedEnrolledUsersNum = usrService.enrollB2BUser(csvFile.getInputStream(),
csvFile.getFileName(), importTime, adminUser, audit);
```

Customizing the Validation Code Generator

Validation codes are used to create a validation link for user enrollment confirmation, reactivating a locked user, and other actions. The characters in the verification code and its expiration time are customizable.

To configure the validation code generator:

- 1 Open the security.xma.xml file, located in the following directory:
 - **UNIX.** `EDX_HOME/xma/config/modules/security`
 - **Windows.** `EDX_HOME\xma\config\modules\security`
- 2 Edit the following constraints as required, located in the section for the IVCodeGenerator JavaBean:
 - **length.** An integer that represents the length of the verification code. The default value is 7. The value must be equal to or larger than 7.
 - **exclude.** A string that contains the characters that cannot appear in the verification code.
 - **includeSpecial.** Whether the validation code can include special characters such as %\$(). Value can be true or false.
 - **minUppercase.** An integer, the validation code must contain at least this number of upper case letters.
 - **minLowercase.** An integer, the validation code must contain at least this number of lower case letters.
 - **minNumber.** An integer, the validation code must contain at least this number of digital characters.
 - **expirationTime.** An integer and a unit (D means day, H means hour, M means minute). For example, 4H means the verification code expires after four hours.

Using the Enrollment Validation API

The ISecurityProfileManager API provides the method `checkValidationCode` for validation code and security profile ID validation:

```
I SecurityProfileManager spManager = UserFactory.getSecurityProfileManager();
secProfile = spManager.checkValidationCode(secProfileId, validationCode);
If validate is expired, throw ValidationCodeExpiredException
//ISecurityXMAService provides the method getValidationCode() to get a validation
code.
LookupService lookup = LookupServiceFactory.getInstance();
ISecurityXMAService securityXMAService = (ISecurityXMAService)
lookup.getModule("security");
IValidationCode =
securityXMAService.createValidationCodeManager().getValidationCode();
```

Customizing Account Lockout

You can customize the maximum attempt thresholds in the Account Lockout use case.

Oracle Billing Insight locks a user account after a maximum number of attempts (5) to enter information during the following use cases:

- **Log In.** When a user tries to log into the application.
- **Forgot and Reset Password.** When a user tries to enter a user name, account number, or service number.
- **Forgot and Reset Password.** When a user tries to enter a security question or security answer.

By default, each of these activities uses the same threshold. You can specify one new threshold for all three activities, or set a different threshold value for each activity.

To configure the maximum attempt thresholds

- 1 Open the `user.xma.xml` file, located in the `EDX_HOME\xma\config\modules\umf` directory.
- 2 In the `IUserManager` JavaBean, three lockers are defined as properties under the tag, each associated with one action. All three lockers reference one locker, which means all three actions have the same threshold value (maximum number of attempts). You can specify one new value for the `max_attempts` property, or specify a different locker and configure the threshold for each activity:

```
<bean id="IUserManager"
class="com.edocs.common.umf.core.UserManager">
<property name="userManagerDao"> <ref local="userManagerDao"/> </property>
<property name="loginLocker"><ref local="locker"/> </property>
<property name="forgotPwdAccountLocker"> <ref local="locker"/> </property>
<property name="forgotPwdSecQstLocker"> <ref local="locker"/> </property>
</bean><bean id="locker" class="com.edocs.common.umf.core.Locker">
<property name="max_attempts" value="5"/></bean>
```

In the action layer, you can call the `hasActionThresholdReached` method to judge whether an action reached the maximum attempt threshold:

```
IUserService usrService=EBillingServiceFactory.getUserService();
usrService.hasActionThresholdReached(user, action);
```

The method `hasActionThresholdReached` adds the specific number of times attempted by 1, then compares the number of attempts with the maximum threshold. If the threshold is reached, the method returns true. If the threshold is not reached, then it returns false.

Customizing Reactivate Account Lockout

You can customize the Reactivate Account Lockout use case. An Assisted Service administrator user can reactivate a locked out account. In the action layer, you can call `reactivateAccount` API to reactivate an account:

```
IUserService usrService=EBillingServiceFactory.getUserService();
usrService.reactivateAccount(userId, audit);
```


These APIs use the following parameters:

- **userId**. User ID. The user's account will be reactivated.
- **audit**. Audit data of the reactivate user account action.

Customizing User Authentication and Authorization With Spring Security

Oracle Billing Insight uses the Spring Security framework for authentication and user access control. You can customize the Spring Security configuration for your implementation. Spring Security settings are maintained in the `spring-security.xml` file.

The following code defines the `UsernamePasswordAuthenticationFilter` filter that handles form authentication in the `spring-security.xml` file:

```
<security: http access-decision-manager-ref="accessDecisionManager">
  <security: logout logout-success-url="/nonAuth/logout.action"/> <security: access-
    denied-handler error-page="/accessDenied.action"/> <security: form-login login-
    page="/nonAuth/login.action" default-target-url="/reporting/reportStart.action"
    authentication-failure-url="/nonAuth/login.action?login_error=1"/>
</security: http>
```

Spring Security maintains a filter chain internally where each filter has a particular responsibility and filters can be added or removed from the configuration depending on which services are required. The ordering of the filters is important as there are dependencies between them. If you are using namespace configuration, then the filters are configured automatically.

The `form-login` element invokes the `UsernamePasswordAuthenticationFilter`. Spring Security adds the filter in the filter chain automatically, and adds an `LoginUrlAuthenticationEntryPoint` to the application context to provide authentication on demand.

`DaoAuthenticationProvider` from Spring Security is implemented for authentication. `DaoAuthenticationProvider` leverages a `UserDetailsService` in order to lookup the username, password and `GrantedAuthority`. `IBillingUserDetailsService` extends `UserDetailsService` and the implementation class implements `loadUserByUsername(String username)` method to provide the authentication mechanism. This method returns the `IBillingUserDetails` object.

About Authentication Request Processing

The preconfigured `AuthenticationManager` property in the `spring-security.xml` file processes each authentication request.

If authentication is successful, then Spring Security places the resulting `Authentication` object into the `Security Context Holder` and redirects the browser to the `defaultTargetUrl` property, which you can customize.

If authentication fails, then Spring Security places `AuthenticationException` into the `HttpSession` attribute indicated by `WebAttributes.AUTHENTICATION_EXCEPTION`, and displays an error page with a reason. This attribute contains well-known keys that are used to store Spring Security information in request or session scope. If authentication fails, the browser displays the authentication failure URL, then which you can also customize.

Customizing Authentication URLs

You can customize the default URLs for the following authentication pages:

- **Login Page.** The URL to display the Login page.
- **Default Login Target.** The URL for the page that displays to the user after logging in. The target can be an action, JSP, or HTML.
NOTE: If the user was prompted to log in after attempting to access a secured resource, then Spring displays the page for the originally requested URL.
- **Access Denied Page.** The URL for the Access Denied Page. If an authenticated user requests a page that he or she does not have authority to access, then Spring redirects the user to the Access Denied Page.
- **Log Out Target Page.** The destination URL that displays to the user after logging out.
- **Login Failure Target.** The URL for the page that displays to the user on login failure.

To customize Spring Security default URLs

- 1 Open the `spring-security.xml` file, located in the following directory:
 - **UNIX.** `EDX_HOME/confi g/securi ty/sel fservi ce`
 - **Windows.** `EDX_HOME\confi g\securi ty\selservi ce`
- 2 In the `security:http` section of the `spring-security.xml` file, edit the appropriate URL attribute, shown in the following table.

Feature URL	Spring Configuration Attribute	Mapping Details
Login Page	<code>login-page</code>	The <code>login-page</code> attribute maps to the <code>loginFormUrl</code> property of the <code>LoginUrlAuthenticationEntryPoint</code> .
Default Login Target	<code>default-target-url</code>	The <code>default-target-url</code> attribute maps to the <code>defaultTargetUrl</code> property of <code>UsernamePasswordAuthenticationFilter</code> code.
Access Denied Page	<code>error-page</code>	None.

Feature URL	Spring Configuration Attribute	Mapping Details
Log Out Target Page	logout-success-url	None.
Login Failure Target	authentication-failure-url	The authentication-failure-url attribute maps to the authenticationFailureUrl property of the UsernamePasswordAuthenticationFilter code.

About Resource Authorization

Authorization in the Oracle Billing Insight applications is based on user roles and associated resource permissions.

By default, each user is assigned one role. You can customize the application resources each role can access by customizing and assigning high-level sets of resource permissions, which are grouped and granted together.

You can add additional user roles and permission sets for your organization. For details, see:

- [“Adding a New User Role” on page 55](#)
- [“Adding a New Permission Set” on page 56](#)

When a user logs into the Assisted Service and Self-Service applications, Oracle Billing Insight validates the user’s role and displays only that functionality permitted for the permissions to which the user role is assigned. For business users, Oracle Billing Insight also displays only that billing data associated with the part of the hierarchy tree to which the user is assigned.

Oracle Billing Insight is preconfigured with the user roles in the azcfg.properties file, as shown in [Table 9](#).

Table 9. Preconfigured User Roles in the azcfg.properties File

User Role	Functional Area Access
Admin	A business user, associated with an individual company or parent company, who is responsible for managing company-related data, managing business and billing hierarchies and hierarchy assignments, and creating and managing users in the Self-Service application.
Manager	A business user, typically assigned to the hierarchy at the account level, who can see billing data from that position to the bottom in the hierarchy tree.
PayerManager	A business user who can make and manage payments on billing data from that position to the bottom in the hierarchy tree.
Subscriber	A business user, typically assigned at service agreement level, who can see data from that position to the bottom in the hierarchy tree.

Table 9. Preconfigured User Roles in the azcfg.properties File

User Role	Functional Area Access
User	A consumer end-user who uses the Self-Service application to make payments and manage their bills.
CSRAdministrator	An administrator of the Assisted Service application who can perform all of the application's functionality for assisting users, plus can create and manage other agents.
CSR	An Assisted Service application user, such as a customer service representative, who assists users, merchants, and managers. CSRs can view the same data available to a user and perform any function on their behalf using the Assisted Service application.
Merchant	A business owner user who can use the Assisted Service application to view various Return On Investment reports and monitor statistics of user activity and billing loads.

Oracle Billing Insight azpolicy.xml file is preconfigured with the permission sets described in [Table 10](#).

Table 10. Preconfigured Permission Sets Defined in the azpolicy.xml File

Permission Set	Functional Area Access
perm_Hierarchy	Use cases for viewing the user's hierarchy information.
perm_HierarchyAdmin	Use cases for creating and managing hierarchies for an organization.
perm_Payment	Use cases for making payments and managing payment accounts.
perm_CompanyGeneral	Use cases for viewing and updating company profile and contacts.
perm_CompanyAdmin	Use cases for managing company information, including viewing and updating all company hierarchies, settings and users.
perm_CSRGeneral	General Assisted Service application functionality for assisting customer users and managing an agent's personal profile.
perm_CSRAdmin	General Assisted Service application functionality plus the ability to create and manage other Assisted Service application users.
perm_Provisioning	Use cases for creating provisioning data in the Assisted Service application.
perm_Impersonation	Use case for impersonating a customer user from the Assisted Service application.
perm_Integration	Lets users of the Assisted Service application access the URL for integration with Siebel CRM.

Table 10. Preconfigured Permission Sets Defined in the azpolicy.xml File

Permission Set	Functional Area Access
perm_BudgetView	Use cases for viewing budgets.
perm_BudgetMgmt	Use cases for creating, managing, and viewing budgets.
perm_Hier_CSR	Defines the level of hierarchy data displayed to CSR users within the accessible use cases.
perm_Hier_Admin	Defines the level of hierarchy data displayed to ADMIN users within the accessible use cases.
perm_Hier_MANAGER, PAYERMANAGER	Defines the level of hierarchy data displayed to Manager and PayerManager users within the accessible use cases.
perm_Hier_Subscriber	Defines the level of hierarchy data displayed to Subscriber users within the accessible use cases.
perm_SysMetrics	Use cases for viewing system metric reports.
perm_AccountGroupReports	This is a context attribute that passes a value to determine whether a user can view billing account or group-related reports for the user's assigned hierarchy position.
perm_PostPay	This is a context attribute that passes a value to determine whether a user can view and choose post-payment notification preferences.

By default, all users are allowed to access functions of the Dashboard, Unbilled, Statements, and Analytics modules in the Self-Service application. No permission is checked for accessing those features. The hierarchy position where the user is assigned is considered both for billing and business hierarchies.

Table 11 shows which principal security roles are assigned to permissions as defined in the azpolicy.xml file for authorization by the Self-Service and Assisted Service applications.

Table 11. Permission to Principal Security Role Mappings in the azpolicy.xml File

Permission	ADMIN	MANAGER	PAYERMANAGER	SUBSCRIBER	B2C	CSR_ADMIN	CSR	CSRMERCHANT
perm_Hierarchy	Y	Y	Y	Y	N	Y	Y	N
perm_Hierarchy Admin	Y	Y	Y	N	N	Y	Y	N
perm_Payment	Y	N	Y	N	Y	Y	Y	N
perm_Company General	Y	Y	Y	N	N	Y	Y	N
perm_Company Admin	Y	N	N	N	N	Y	Y	N
perm_CSRGeneral	N	N	N	N	N	Y	Y	N
perm_CSRAdmin	N	N	N	N	N	Y	N	N
perm_Provisioning	N	N	N	N	N	Y	Y	N
perm_Impersonation	N	N	N	N	N	Y	Y	N
perm_Integration	N	N	N	N	N	Y	Y	N
perm_BudgetView	Y	Y	Y	Y	N	Y	Y	N
perm_BudgetMgmt	Y	Y	Y	N	N	Y	Y	N
perm_Hier_CSR	N	N	N	N	N	N	Y	N
perm_Hier_Admin	Y	N	N	N	N	N	N	N
perm_Hier_MANAGER, PAYERMANAGER	N	Y	Y	N	N	N	N	N
perm_Hier_Subscriber	N	N	N	Y	N	N	N	N
perm_SysMetrics	N	N	N	N	N	N	N	Y

Table 11. Permission to Principal Security Role Mappings in the azpolicy.xml File

Permission	ADMIN	MANAGER	PAYERMANAGER	SUBSCRIBER	B2C	CSR_ADMIN	CSR	CSRMERCHANT
perm_AccountGroupReports	Y	Y	N	N	N	Y	Y	N
perm_PostPay	Y	Y	Y	Y	Y	Y	Y	N

Adding a New User Role

You can add a new user role to define and control user access to resources in the applications. For details about preconfigured user roles and permissions in Oracle Billing Insight, see [“About Resource Authorization” on page 51](#).

To add a new user role for your organization

- 1 Add the new role to the OLTP.EDX_BSL_AUTH_SECROLE table. For the Type field, specify B2B, B2C, or CSR. Oracle Billing Insight does not use the other fields, which are optional.

User roles in this table are displayed in the role drop-down list.

- 2 Map the new user role to a principal role, which serves as an alias for the user role:

- a Open the azcfg.properties file, located in the following directory:

- **UNIX.** `EDX_HOME/config`
- **Windows.** `EDX_HOME\config`

- b Add a com.edocs.common.security.role-mappers.secl role statement mapping the new principal role (alias) with the new user role, as shown in the following preconfigured statements in the file:

```
com.edocs.common.security.role-mappers.secl role.ALL_USERS=Admin, User, PayerManager,
Manager, Subscriber, CSR, CSRAdminstrator
com.edocs.common.security.role-mappers.secl role.ADMIN=Admin
com.edocs.common.security.role-mappers.secl role.MANAGER=Manager
com.edocs.common.security.role-mappers.secl role.PAYERMANAGER=PayerManager
com.edocs.common.security.role-mappers.secl role.SUBSCRIBER=Subscriber
com.edocs.common.security.role-mappers.secl role.CSR=CSR
com.edocs.common.security.role-mappers.secl role.CSR_ADMIN=CSRAdminstrator
com.edocs.common.security.role-mappers.secl role.ALL_USERS_EXCLUDE_SUBSCRIBER=Admin,
User, PayerManager, Manager, CSR, CSRAdminstrator
com.edocs.common.security.role-mappers.secl role.B2C=User
com.edocs.common.security.role-mappers.secl role.CSRMERCHANT=Merchant
```

- c To map the new user role to any of the existing principal roles (assigning it the same resource permissions assigned to the principal role in the azpolicy.xml file), add it to the end of that mapping statement, using a comma to separate it from the other user roles in the line.
- 3 Assign the newly defined principle role to one or more existing resource permission sets:
 - a Open the azpolicy.xml file, located in the following directory:
 - **UNIX.** `EDX_HOME/config/azcfg/policy`

❏ **Windows.** *EDX_HOME\config\azcfg\policy*

- b** Add the new principal role (alias) to the list of values for each permission set that you'd like to grant the role, separated by a comma.

The following example shows the perm_CompanyGeneral permission group, which controls access to the Company tab functionality:

```
<permission>
<name>perm_CompanyGeneral</name>
<!-- Define the name of a resource. -->
<resource>com.edocs.common.security.authorization.az.permissions.EBillingPermission
</resource>
<rule>
<name>admin</name>
<!-- Defines the name of the rule -->
<type>SecurityRole</type>|
<!-- Type of the rule -->
<values>ADMIN, MANAGER, PAYERMANAGER, CSR, CSR_ADMIN</values>
<!-- The role(s) which can access this resource, comma separated. Note it can
be an alias defined in azcfg.properties -->
</rule>
</permission>
```

- 4** Also in the azpolicy.xml file you can optionally set permissions based on context attributes. The permission is granted by a rule based on attribute value, for example:

```
<permission>
<name>perm_PostPay</name>
<resource>com.edocs.common.security.authorization.az.permissions.EBillingPermission
</resource>
<rule>
<name>PostPay</name>
<type>ContextAttribute</type>
<matchKey>postPay</matchKey>
<values>true</values>
</rule>
</permission>
```

The following requirements must be met:

- ❏ The <type> must be ContextAttribute.
- ❏ In <matchKey>, specify the parameter passed into the permission engine, azEngine.
- ❏ Specify the granted parameter value in <values>.

Adding a New Permission Set

You can add new permissions in Oracle Billing Insight, creating a unique set of application resource privileges.

To add a new permission set

- 1 Open the azpolicy.xml file, located in the following directory:
 - **UNIX.** *EDX_HOME/confi g/azcfg/pol i cy*
 - **Windows.** *EDX_HOME\confi g\azcfg\pol i cy*
- 2 Add the new permission set authorization XML to the file, using the existing permission authorizations as a guide. Assign principal roles to grant access to those users.
- 3 Follow these steps to authorize access to application URLs:
 - a Open the security.xma.xml file, located in the following directory:
 - **UNIX.** *EDX_HOME/xma/confi g/modul es/securi ty*
 - **Windows.** *EDX_HOME\xma\confi g\modul es\securi ty*
 - b Add the new permission to the appropriate application URL mappings:
 - **Self-Service application.** Add the permission group to the AuthorizeMappingManager bean.
 - **Assisted Service application.** Add the permission group to the CSRAuthorizeMappingManager bean.

The bean XML in the security.xma.xml file is shown here:

```
<bean id="AuthorizeMappingManager"
class="com.edocs.common.securi ty. authori ze. Authori zeMappi ngManager"
scope="si ngl eton">
<property name="URLmappi ng"><val ue>/usermanagement/
CompanyProfi le_browse. acti on. *|/contact/Contact_browseCCL. acti on. *|/contact/
Contact_di spl ayUpl oadCCL. acti on. *|/contact/
Contact_upl oadCCL. acti on. *=perm_CompanyGeneral
.....
</val ue></property> </bean>
<bean id="CSRAuthori zeMappi ngManager"
class="com.edocs.common.securi ty. authori ze. Authori zeMappi ngManager"
scope="si ngl eton">
<property name="URLmappi ng"><val ue>
/CSR_Logi n. j sp. *|/i ndex. j sp. *|/val i date. acti on. *|/csrLogi n. acti on. *|/
veri fyEnrol lment. acti on|/secQuesti on. *. acti on|/ForgotPassword. acti on. *|/
ResetPwd. *. acti on. *|/ResetPwdConfi rm. *. acti on. *|/I logout. acti on. *|/
mi gratedCSRShow. acti on. *|/mi gratedCSRUser. *. acti on. *|/
mi gratedUserShow. acti on. *|/mi gratedUserProfi le. *. acti on. *|/
resetExpPwdConfi rm. *. acti on. *|/_assets/. *|/struts/. *|/
accessDeni ed. acti on. *=I S_AUTHENTI CATED_ANONYMOUSLY
.....
</val ue></property> </bean>
```

- 4 Follow these steps if you want to authorize a particular UI component:
 - a Surround the code or resources you want to authorize with a tag called <authz:authorize>. For example, the following code controls access to the Company Profile subtab:

```
<authz: authorize ifAnyGranted="perm_CompanyGeneral ">
<li><a href="{ctx}/{usermanagement}/CompanyProfile_browse.action"
title="<s: text name="global.mainNavTab1.subNav3"/>">
<s: text name="global.mainNavTab1.subNav3"/></a></li></authz: authorize>
```

- b** Separate multiple permissions by commas in the `ifAnyGranted` attribute. If any of the permissions in the list are granted, then the body of the tag is written.

The `authz:authorize` tag can have the following attributes:

- **ifAllGranted.** All the listed permissions must be granted for the tag to output its body.
- **ifAnyGranted.** Any of the listed permissions must be granted for the tag to output its body.
- **ifNotGranted.** None of the listed permissions must be granted for the tag to output its body.

- 5** Invoke the `com.edocs.common.api.security.authorize.IAccessManager` authorization in the Java code, for example:

```
IUser user = userSession.getUser();

//get an AccessManager instance

IAccessManager accessMgr =SecurityFactory.getAccessManager(user.getUserId());

Map ctxParameters = new HashMap();

Boolean
prepay=userSession.getAttribute("PREPAY")!=null?(Boolean)userSession.getAttribute("PREPAY"): false;

String[] values = {(Boolean)!prepay}.toString();

ctxParameters.put("postPay", values);

//do authorize by AZEngine

boolean isAuthorized= i saccessMgr.checkAccess("perm_PostPay", ctxParameters);
```

About Deleted Users

When a user is deleted in the Self-Service application, Oracle Billing Insight marks the user as deleted, but does not remove the user from the Oracle Billing Insight database. A deleted user will not appear in the application, and will eventually be removed by a batch purge job.

When user is deleted, the following resources owned by the user are also removed:

- Payments scheduled that are still pending
- Recurring payments
- Access to billing information
- Batch reports created by the user

Input File Format for the DefUsrAcctRelationship Job

To use the DefUsrAcctRelationship Command Center job to add or remove user-account relationships in Oracle Billing Insight in batch, create a pipe-delimited TXT data input file in the following format. For details on configuring and running the DefUsrAcctRelationship job, see *Administration Guide for Oracle Billing Insight*.

Table 12 shows the file format to use when creating the input file for the DefUsrAcctRelationship job.

Table 12. Input File Format for the DefUsrAcctRelationship Job

Position	Names	Maximum Length	Data Type	Required	Description
1	Username	255	VARCHAR2	Yes	User ID in Oracle Billing Insight
2	Billing System ID	255	VARCHAR2	Yes	Billing system ID
3	Account Number	255	VARCHAR2	Yes	Account number
4	Add or Remove Flag	1	CHAR	Yes	A - Add or R - Remove

Input File Format for Migrating Consumer Users in Batch

Oracle Billing Insight provides a SQL script for migrating consumer users from a billing system to Oracle Billing Insight in batch. To use this script, you must create a pipe-delimited TXT data input file in the following format. For details on running the consumer user migration script, see *Administration Guide for Oracle Billing Insight*.

Table 13 shows the file format to use when creating the input file for migrating consumer users in batch.

Table 13. Input File Format for Migrating Consumer Users in Batch

Position	Names	Maximum Length	Data Type	Required	Description
1	Username	36	VARCHAR2	Yes	User ID for Oracle Billing Insight
2	Email Address	255	VARCHAR2	No	The user's email address
3	Billing System ID	255	VARCHAR2	Yes	Billing system ID
4	Account Number	255	VARCHAR2	Yes	Account number

Table 13. Input File Format for Migrating Consumer Users in Batch

Position	Names	Maximum Length	Data Type	Required	Description
5	Creation Date	MM/DD/YYYY	DATE	No	The date the account was created
6	My Bill is Ready for Viewing Flag	1	CHAR	No	Y - Yes or N-No
7	My Payment Account Has Been Created, Deleted, or Updated Flag	1	CHAR	No	Y - Yes or N-No
8	My Payment is Due in (X) Days Flag	2	CHAR	No	Y - Yes or N-No
9	My Payment was submitted Flag	1	CHAR	No	Y - Yes or N-No
10	My Recurring Payment is Setup or Modified Flag	1	CHAR	No	Y - Yes or N-No
11	My Recurring Payment is Less Than the Total Amount Due (Threshold Exceeded) Flag	1	CHAR	No	Y - Yes or N-No
12	My Payment was Made Successfully Flag	1	CHAR	No	Y - Yes or N-No
13	My Payment Failed Flag	1	CHAR	No	Y - Yes or N-No
14	My Credit Card is About to Expire Flag	1	CHAR	No	Y - Yes or N-No
15	My Batch Report is Ready Flag	1	CHAR	No	Y - Yes or N-No
16	Flex Field 1	255	VARCHAR2	No	Flexible field
17	Flex Field 2	255	VARCHAR2	No	Flexible field
18	Flex Field 3	255	VARCHAR2	No	Flexible field
19	Flex Field 4	255	VARCHAR2	No	Flexible field
20	Flex Field 5	19	NUMBER	No	Flexible field

Input File Format for Deleting Consumer Users in Batch

Oracle Billing Insight provides a SQL script for deleting consumer users from Oracle Billing Insight in batch. To use this script, you must create a pipe-delimited TXT data input file in the following format. For details on running the script to delete consumer users, see *Administration Guide for Oracle Billing Insight*.

Table 14 shows the file format to use when deleting consumer users in batch.

Table 14. Input File Format for Deleting Consumer Users in Batch

Position	Names	Maximum Length	Data Type	Required	Description
1	User ID	40	VARCHAR2	Yes	User ID in Oracle Billing Insight

5

Customizing Billing Statements

This chapter covers APIs you can use to customize online billing statements. It includes the following topics:

- [About Split Billing Rules Management on page 63](#)
- [Updating Disputes in an External Customer Relationship Management System on page 64](#)
- [Defining Billing Periods on page 65](#)

The online statement feature reduces operational costs when subscribers adopt online statements instead of a printed one. Oracle Billing Insight can render a statement that looks similar to the paper statement. Taking advantage of the dynamic of the Web, Oracle Billing Insight can expand or collapse the amount of data displayed, drill down into details, show a more up-to-date statement, and display a previous statement.

About Split Billing Rules Management

The split-billing rules management feature enables providers to categorize business and personal mobile phone usage and expenses included on the same statement. Service providers define the split billing rules, and during the data load, Oracle Billing Insight uses these rules to categorize the expenses.

There are two ways to categorize mobile phone usage:

- All call records are automatically categorized at load time, based on the predefined rules. The rule ID is recorded in the EDX_RPT_SERVICE_DETAIL_FACT table for each service detail entry, so Oracle Billing Insight can track which split billing rule was applied when categorizing the service details.
- Users can manually change the categorization when viewing their statements online if any of the automatic categorization is not appropriate.

About the OLAP.EDX_RPT_ETL_PLUG_RULE Table

This table is where all the categorization rules used for split billing were specified. Upon billing data load processing, the rule execution loops through all the records in this table, following the path pointing to an individual rule specified in the RULE_STORED_PROC column, to find the corresponding stored procedure and execute it.

Rules are executed in the order specified in the RULE_EXEC_ORDER column.

If you have additional rules to execute, then you can develop a customized store procedure. Insert a new rule entry in the table, and the Oracle Billing Insight load process will apply the rule automatically.

A special entry (with `RULE_TYPE='MAN'`) in this table represents a manual categorization activity. When a user recategorizes certain service detail records in the application, the rule ID of the special entry will be recorded in `EDX_RPT_SERVICE_DETAIL_FACT` table, indicating that the user has manually categorized the service detail and the manual rule ID overwrites the previous split billing rule ID.

About the `OLAP.EDX_RPT_SPLIT_CATEGORY_TYPE` Table

This table records the category type used in split billing. Currently the category is either business or personal, (`CORPORATE CALLS/PERSONAL CALLS`). Choose one to be the default category using the `CATEGORY_DEFAULT_CODE` column, which represents a Boolean flag, indicating the default category type to be used when ambiguity occurs during the categorization process.

The last split billing rule (stored procedure) defined in `EDX_RPT_ETL_PLUG_RULE` categorizes all of the remaining uncategorized service detail records using the default category specified in this table.

If you have more categories to be used in the split-billing feature, then you can add a new entry to the table.

Oracle Billing Insight provides the following Split Billing Rules Management APIs:

- **ISplitBillingService.** The service API for split billing feature, used to retrieve a particular service detail fact record, and change the category on a particular transaction detail. It can also be used to retrieve the list of valid categories defined for categorizing transaction detail records. (API is in package `com.edocs.common.api.statement.splitbilling.ISplitBillingService`)
- **IServiceTransactionDetail.** Represents a single one single service detail fact record in `EDX_RPT_SERVICE_DETAIL_FACT` table. (API is in package `com.edocs.common.api.statement.IServiceTransactionDetail`)

Updating Disputes in an External Customer Relationship Management System

You can customize your implementation of Oracle Billing Insight to automatically update an external Customer Relationship Management (CRM) when a user creates a dispute.

To update an external CRM system with disputes

- 1 Customize the `com.edocs.common.api.webservice.connector.IDDisputeWSConnector` API to do the following:
 - Update your external CRM when a user creates a dispute in Oracle Billing Insight.
 - Send a confirmation to Oracle Billing Insight when the data has been received. Also mark the request complete in Oracle Billing Insight, returning an external ID and status.

- 2 Pass the following information to or from the external CRM system, in the following format.

Name	Required	Description
Biller ID	Yes	Billing system identifier
Billing Account Number	Yes	Billing account number for the payment transaction
Transaction (Statement) Number	Yes	The transaction submitted for dispute
Transaction (Statement) Date	Yes	The transaction date
Dispute Number	Yes	The Oracle Billing Insight dispute number
Disputed Amount	Yes	The amount disputed
Dispute Description	Yes	Reason for the dispute

Defining Billing Periods

To use Oracle Billing Insight to process bills, you must prepopulate the period and aggregate dimension database tables with billing period data. You must prepopulate the period tables as a new implementation, and then again periodically to add additional periods for ongoing use by the Self-Service application.

In the Oracle Billing Insight OLAP database, period dimension tables require an entry for each monthly billing period start date and end date. You use a stored procedure to prepopulate the billing tables, and you specify the billing start and end dates, number of years to populate, and other information. This information is used as a period key for partitioning statement fact records. Aggregate billing period dimension tables also require entries for each billing quarter and year. You must also use a stored procedure to synchronize this data in the OLTP database.

You can also run the script to define additional billing periods at any time as necessary for your implementation. Make sure the data is up-to-date in these tables.

Table 15 describes the period-related database tables in the Oracle Billing Insight database.

Table 15. Period-Related Database Tables

Database Table	Schema	Description
EDX_RPT_PERIOD_DIM	OLAP	Contains entries for each billing period start and end date.
EDX_RPT_AGGREGATE_PERIOD_DIM	OLAP	Contains quarterly and yearly aggregation master dimension data.

Table 15. Period-Related Database Tables

Database Table	Schema	Description
EDX_RPT_AGGREGATE_PERIOD_LINK	OLAP	Contains links between the aggregation table, EDX_RPT_AGGREGATE_PERIOD_DIM, and the entries in the period dimension table, EDX_RPT_PERIOD_DIM.
EDX_OMF_PERIOD	OLTP	Contains OLTP synchronization data for the entries in the OLAP period dimension table, EDX_RPT_PERIOD_DIM.

To define billing periods within the database

- For each language resource bundle you have defined in your implementation, update the Period.properties resource file (such as Period.en.us.properties) with the extended period string-value pairs. This file is located in the following directory:
 - UNIX.** `EDX_HOME/confi g/resourcebundl e/com/edocs/appl i cati on/resources`
 - Windows.** `EDX_HOME\confi g\resourcebundl e\com\edocs\appl i cati on\resources`
- Locate the `pr_period_dim_population.sql` script in the following directory:
 - UNIX.** `EDX_HOME/db/oracl e/ol ap/Scri pts`
 - Windows.** `EDX_HOME\db\oracl e\ol ap\Scri pts`
- Run the stored procedure, `populate_period_dim`, in the OLAP schema, specifying the input parameters described in [Table 16](#).

The following example shows the `populate_period_dim` procedure:

```
BEGIN
  popul ate_peri od_di m(p_fi rst_peri od_start_date => 01/03/2013,
    p_fi rst_peri od_end_date => 02/02/2013,
    p_number_of_years => 10,
    p_del ete_exi sti ng_enri es => 'Y',
    p_fi rst_quarter_start_date => 02/01);
END;
```

In the example, the parameters indicate the following:

- 01/03/2013.** Specifies that the start date of the first billing period is 01/03/2013, and indicates that all subsequent billing periods will start on the third day of each calendar month.
- 02/02/2013.** Specifies that the end date of the first billing period is 02/02/2013, and indicates that all subsequent billing periods will end on the second day of each calendar month. Do not leave a time gap between the period start and end date. The start and end dates must be contiguous, for example, if the end date of Mar-2015 is 03/02/2015 23:59:59, then the start date of Apr-2015 must be 03/03/2015 00:00:00.
- 10.** Specifies to populate ten years of billing periods, starting at 01/03/2013.

- **Y.** Specifies to delete all existing entries from period and aggregation dimension tables before adding new period entries.
 - **02/01.** Specifies that the first quarter start date is 02/01, indicating that the first quarter is 02/01 through 04/30, the second quarter is 05/01 through 07/31, and so on.
- 4 Synchronize the entries in the EDX_RPT_PERIOD_DIM table to the OLTP period table, EDX_OMF_PERIOD. Locate the pr_oltp_period_sync.sql script in the following directory, and run the stored procedure oltp_period_sync in the OLTP schema:
- **UNIX.** *EDX_HOME/db/oracle/oltp/Scripts*
 - **Windows.** *EDX_HOME\db\oracle\oltp\Scripts*

No input parameters are required for this procedure.

Table 16 describes the input parameters for the stored procedure, populate_period_dim, for populating billing period data in the OLAP schema.

Table 16. Input Parameters for the populate_period_dim Stored Procedure

Input Parameter	Data Type	Format	Description
P_FIRST_PERIOD_START_DATE	VARCHAR2	MM/DD/YYYY	Specifies the start date of the first billing period. Each subsequent billing period entry that the stored procedure adds to the period dimension table, EDX_RPT_PERIOD_DIM, will start on this day of the month.
P_FIRST_PERIOD_END_DATE	VARCHAR2	MM/DD/YYYY	Specifies the end date of the first billing period. Each subsequent billing period entry that the stored procedure adds to the period dimension table, EDX_RPT_PERIOD_DIM, will end on this day of the month.
P_NUMBER_OF_YEARS	NUMBER	None	Indicates the number of years of billing periods to populate in the billing period and aggregate dimension tables. For example, if you specify a value of 5, then the stored procedure adds 60 monthly billing period entries to the EDX_RPT_PERIOD_DIM table starting from the first period start date, P_FIRST_PERIOD_START_DATE. The procedure also adds 20 quarter billing entries and 5 yearly aggregation entries to the EDX_RPT_AGGREGATE_PERIOD_DIM table, starting from the first quarter start date, P_FIRST_QUARTER_START_DATE.

Table 16. Input Parameters for the populate_period_dim Stored Procedure

Input Parameter	Data Type	Format	Description
P_DELETE_EXISTING_ENTRIES	VARCHAR2	Y or N	<p>Specifies whether to delete all existing entries from the billing period dimension tables before adding new entries as specified by your input.</p> <p>Specify Y only if you are prepopulating dimension tables at the beginning of an implementation. This option deletes all entries from the tables before adding new ones.</p> <p>Specify N if the dimension tables are already populated and you want to add additional entries for the coming years as specified by your input. This option does not delete existing data from the tables.</p>
P_FIRST_QUARTER_ START_DATE	VARCHAR2	MM/DD	<p>The start date of the first billing quarter, used to populate quarter and yearly aggregation periods in the EDX_RPT_AGGREGATE_PERIOD_DIM table.</p> <p>After the first quarter, subsequent quarter entries that the stored procedure adds to the aggregation table will start at three-month increments from this day. For example, 01/01 indicates that the first quarter starts on 01/01 and ends on 03/31, the second quarter entry starts on 04/01 and ends on 06/30, and so on.</p> <p>For each yearly and quarterly aggregation based on the first quarter start date, the procedure also populates associated period keys in the aggregation period link table, EDX_RPT_AGGREGATE_PERIOD_LINK.</p>

6

Using and Customizing Email and SMS Notifications

This chapter covers customizing the notification feature in Oracle Billing Insight. It includes the following topics:

- [About Batch Email Notification Processing on page 69](#)
- [Notification Methods: Instant and Batch on page 71](#)
- [Email Notification Template Content for Business Users on page 74](#)
- [Email Notification Template Content for Consumer Users on page 101](#)
- [SMS Notification Template Content on page 130](#)
- [Customizing the Content of Notifications on page 134](#)
- [Customizing the Default Display Patterns Used in Notifications on page 135](#)
- [Configuring Messaging Properties on page 137](#)
- [Adding a Custom Message Provider on page 140](#)
- [Configuring the Secure Bill-Ready Notification Password on page 142](#)
- [Disabling SMS Notifications on page 142](#)

About Batch Email Notification Processing

Oracle Billing Insight interacts with the Oracle Billing Insight database to determine what email to send as part of batch processing. Each message composed is stored, then the email dispatcher takes the stored email and sends the messages based on selected external transport type, for example, SMTP.

Figure 2 shows an overview of email processing.

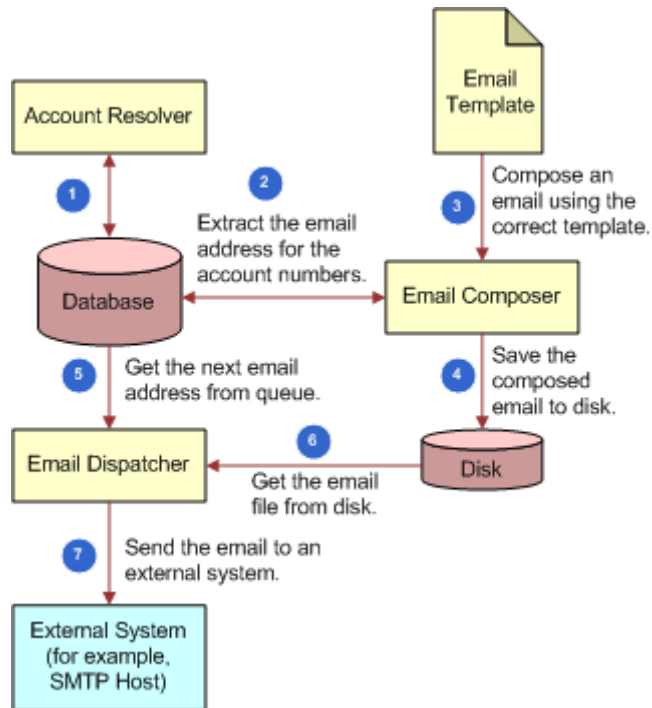


Figure 2. Email Processing

Oracle Billing Insight merges the message template with runtime information to create the email message. The messenger then calls the gateway, configured in the `globalConfig.xma.xml` file, to send the email message.

The email composer consists of three components that function together to group multiple account numbers by email address to roll-up messages. The components of the email composer perform the following functions:

- Group account numbers by send-to address
- Compose a group message based on a template
- Create a grouped message for a given queue

Figure 3 shows how email groups email messages for delivery.

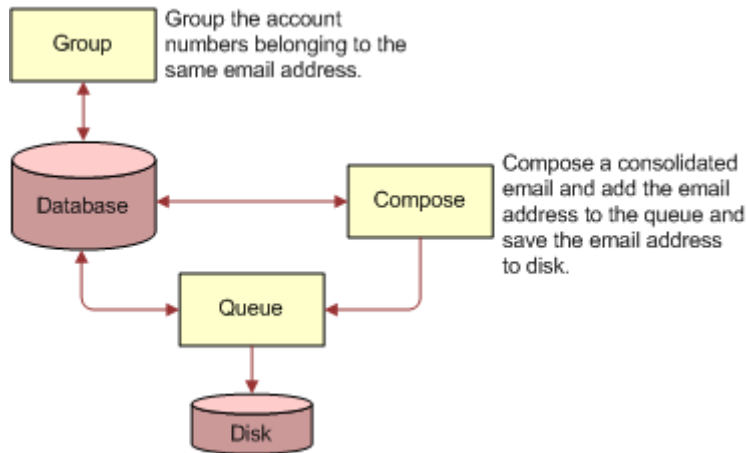


Figure 3. Email Grouping

Notification Methods: Instant and Batch

Some notification types are generated instantly and others are sent using a batch job in the Command Center. For details on running batch jobs, see *Administration Guide for Oracle Billing Insight*.

Table 17 indicates how each notification type is generated in the Oracle Billing Insight.

Table 17. Method of Notification

Notification Type	Edition	Generated Instantly or By Batch Job
Bill Ready	B2B and B2C	Batch
Bill Ready with PDF Attachment	B2B and B2C	Batch
Bill Ready with Daisy Attachment	B2B and B2C	Batch
Enrollment (Business User)	B2B	Instant
Enrollment (Consumer User)	B2C	Instant
Enrollment (Agent User)	B2B and B2C	Instant
Enrollment (Agent Administrator)	B2B and B2C	Instant
Enrollment (Organization Administrator)	B2B	Instant
Enrollment (Migrated Business User)	B2B	Instant
Enrollment (Migrated Consumer User)	B2C	Instant
Enrollment (Migrated Agent User)	B2B and B2C	Instant

Table 17. Method of Notification

Notification Type	Edition	Generated Instantly or By Batch Job
Validate Email	B2B and B2C	Batch
Job Alert Success	B2B and B2C	Batch
Job Alert Failure	B2B and B2C	Batch
Recurring Payment Confirmation	B2B and B2C	Batch
Recurring Payment Configuration Update	B2B and B2C	Batch
Recurring Payment Deleted	B2B and B2C	Batch
Recurring Payment Transaction Canceled	B2B and B2C	Batch
Successful Quick Payment	B2B and B2C	Batch
Quick Payment Failure	B2B and B2C	Batch
Payment Due	B2B and B2C	Batch
Payment Scheduled	B2B and B2C	Batch
Successful Payment	B2B and B2C	Batch
Payment Failure	B2B and B2C	Batch
Payment Canceled	B2B and B2C	Batch
Scheduled Payment Canceled	B2B and B2C	Batch
Refund Request Submitted	B2B and B2C	Instant
Statement Threshold	B2B and B2C	Batch
Payment Threshold	B2B and B2C	Batch
Credit Card Expiration	B2B and B2C	Batch
Payment Account Created Successfully	B2B and B2C	Instant
Payment Account Updated Successfully	B2B and B2C	Instant
Payment Account Deleted Successfully	B2B and B2C	Instant
Batch Report Ready	B2B and B2C	Batch
Batch Report Request Expiry	B2B and B2C	Batch
Login Password Changed	B2B and B2C	Instant
Security Question Changed	B2B and B2C	Instant
Forgot User ID	B2B and B2C	Instant
User Account Reactivated	B2B and B2C	Instant
Agent User Account Reactivated	B2B and B2C	Instant
Migrated User Account Reactivated	B2B and B2C	Instant

Table 17. Method of Notification

Notification Type	Edition	Generated Instantly or By Batch Job
Migrated Agent User Account Reactivated	B2B and B2C	Instant
Password Expired	B2B and B2C	Batch
User-Account Relationship Job Status	B2C	Batch
Email Bounceback	B2B and B2C	Batch
Secure Payment Option Selected	B2B and B2C	Instant
Payment - Bank Account Enrollment (Account Activation)	B2B and B2C	Batch
Payment - Bank Account Enrollment (Bank Account Number Update)	B2B and B2C	Batch
Payment - Bank Account Enrollment (Bank Routing Number Update)	B2B and B2C	Batch
Payment - Bank Account Enrollment (Bank Account Number and Routing Number Update)	B2B and B2C	Batch
Payment - Bank Account Enrollment (Bank Account Type Update)	B2B and B2C	Batch
Payment - Bank Account Enrollment (Bank Account Number and Bank Type Update)	B2B and B2C	Batch
Payment - Bank Account Enrollment (Bank Account, Routing Number, and Bank Type Update)	B2B and B2C	Batch
Payment Consolidator - Account Enrollment Job Errors	B2B and B2C	Batch
Payment Consolidator - Account Enrollment Response Job Errors	B2B and B2C	Batch
Payment Consolidator - Bill Summary Job Errors	B2B and B2C	Batch
Payment Consolidator - Bill Summary Acknowledgement Job Errors	B2B and B2C	Batch
Single Sign-On (SSO) Bulk Load Error	B2B and B2C	Batch

Email Notification Template Content for Business Users

This topic shows the content of the email templates preconfigured for each notification type for business users of Oracle Billing Insight. [Table 18](#) shows the template text provided for each notification type. (For the consumer notification content, see [“Email Notification Template Content for Consumer Users”](#) on page 101.)

Table 18. Notification Types and Associated Email Templates for Business Users

Email Notification Type	Email Template Content for Business Users
Bill Ready	From: <i>Admin@Example.com</i> Sent: <i>Date Time</i>
Bill Ready with PDF Attachment	To: <i>User</i> Subject: Your Statement is Ready
Bill Ready with Daisy Attachment	Dear <i>User</i> , View Statement Pay Now Your latest statement for the following billing account(s) are now available online. Billing Account Number ending in <i>Partial Account Number</i> Statement Date: <i>StatementDate</i> Statement Amount Due: <i>Amount Due</i> Payment Due Date: <i>Due Date</i> To view your statement, click <i>Oracle Billing Insight HTTPS URL</i> . This is an automatically generated email. Please do not reply to this message.
Enrollment (Business User)	From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled Dear <i>User</i> , Welcome! Your administrator has enrolled you in Oracle Billing Insight. Contact your company administrator at <i>Admin Email</i> to receive your assigned username. Then visit <i>Oracle Billing Insight HTTPS URL</i> to finish the enrollment process. This is an automatically generated email. Please do not reply to this message.

Email Notification Type	Email Template Content for Business Users
Enrollment (Agent User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome! Your administrator has enrolled you in Oracle Billing Insight.</p> <p>Contact your company administrator at <i>Admin Email</i> to receive your assigned username. Then visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the enrollment process.</p> <p>Log in now and you will be redirected to create your own personal password.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Enrollment (Agent Administrator)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome to Oracle Billing Insight.</p> <p>Now you can start managing other CSRs and assisting customers with Billing and Payment. Please visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the enrollment process.</p> <p>Log in now and you will be redirected to create your own personal password.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Enrollment (Organization Administrator)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome! Your administrator has enrolled you in Oracle Billing Insight.</p> <p>Contact your company customer service representative at <i>Admin Email</i> to receive your assigned username. Then visit <i>Oracle Billing Insight HTTPS URL</i> to finish the enrollment process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Enrollment (Migrated Business User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome to the Customer Care Application.</p> <p>Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the enrollment process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Enrollment (Migrated Agent User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome to the Oracle Billing Insight application.</p> <p>Please visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the enrollment process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Validate Email	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Please Finish Your Email Validation Process</p> <p>Dear <i>User</i>,</p> <p>Thank you for updating your email address for your online Oracle Billing Insight application.</p> <p>Please click on the link <i>Oracle Billing Insight HTTPS URL</i> to finish your email validation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Job Alert Success	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Job Completed Successfully</p> <p>Dear <i>User</i>,</p> <p>The following job completed successfully:</p> <p><i>Job Name Job Type Job Instance ID DDN</i></p>

Email Notification Type	Email Template Content for Business Users
Job Alert Failure	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Job Failure</p> <p>Dear <i>User</i>,</p> <p>The following job did not complete successfully:</p> <p><i>Job Name Job Type Job Instance ID DDN Exception</i></p>
Recurring Payment Confirmation	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Welcome to Automatic Bill Payment</p> <p>Dear <i>User</i>,</p> <p>You have successfully enrolled in the Automatic Bill Payment program for your online accounts.</p> <p>The first automatic bill payment will take place after your next billing cycle. You will receive a payment confirmation email when the payment is submitted.</p> <p>Account Number: <i>Account Number</i></p> <p>Payment Type: <i>Payment Type</i></p> <p>Amount: <i>Amount</i></p> <p>You can continue to make other payments on your account using the Quick Payment option on the Payments Menu. Click <i>Oracle Billing Insight HTTP URL</i> to view your e-bills anytime.</p> <p>Thank you for using Automatic Bill Payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Recurring Payment Configuration Update	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Automatic Bill Payment Settings Have Been Updated</p> <p>Dear <i>User</i>,</p> <p>The Automatic Bill Payment settings for your online accounts have been updated.</p> <p>Account Number: <i>Account Number</i></p> <p>Payment Type: <i>Payment Type</i></p> <p>Amount Due: <i>Amount Due</i></p> <p>Thank you for using Automatic Bill Payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Recurring Payment Deleted	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Automatic Bill Payment Participation Cancelled</p> <p>Dear <i>User</i>,</p> <p>This notification confirms that your participation in Automatic Bill Payment has been cancelled.</p> <p>Account Number: <i>Account Number</i></p> <p>Payment Type: <i>Payment Type</i></p> <p>Amount: <i>Amount</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using Automatic Bill Payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Recurring Payment Transaction Canceled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Recurring Payment Transaction Has Been Cancelled</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that your recurring payment schedule has been deactivated, due to your payment account information being incorrect, and payments can no longer be scheduled. Please log in to the Online Billing Application and make the necessary changes.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>Thank you for using Automatic Bill Payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Successful Quick Payment	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Thank You for Your Quick Payment</p> <p>Dear <i>User</i>,</p> <p>Thank you for submitting the following one-time Quick Payment.</p> <p>Your total transaction amount of <i>TotalAmount</i> was applied to the following account(s).</p> <p>Account Number: <i>Account Number</i></p> <p>Amount: <i>Amount</i></p> <p>Master Reference Number: <i>Reference Number</i></p> <p>Thank you for using Quick Payment. Visit us again to make your next online payment at <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Quick Payment Failure	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Quick Payment Failed</p> <p>A problem occurred during your one-time Quick Payment transaction. The following payment did not process successfully:</p> <p>Amount: <i>Amount</i></p> <p>Master Reference Number: <i>Reference Number</i></p> <p>Please visit <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Due	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have a Payment Due</p> <p>Dear <i>User</i>,</p> <p>You have a payment due for each of the following accounts:</p> <p>Account: <i>Account</i></p> <p>Due Date: <i>Due Date</i></p> <p>Due Amount: <i>Amount Due</i></p> <p>You have additional bills; only <i>Cutoff Number</i> accounts have been displayed above.</p> <p>Thank you for using online payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment Scheduled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Automatic Payment Schedule Confirmed</p> <p>Dear <i>User</i>,</p> <p>Automatic payments have been scheduled for the following accounts:</p> <p>Account: <i>Number</i></p> <p>Due Date: <i>Due Date</i></p> <p>Amount Due: <i>Due Amount</i></p> <p>Amount Paid: <i>Configured Amount</i></p> <p>Outstanding Balance: <i>Balance Due</i></p> <p><i>Statement Credit</i> has been credited to your account.</p> <p>Thank you for using online payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Successful Payment	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Thank You for Your Payment</p> <p>Dear <i>User</i>,</p> <p>Your payments for the following accounts completed successfully.</p> <p>Your total transaction amount of <i>TotalAmount</i> was applied to the following account(s).</p> <p>Account: <i>Number</i></p> <p>Amount Due: <i>Amount Due</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment Failure	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Payment Did Not Complete Successfully</p> <p>Dear <i>User</i>,</p> <p>Your online payment for the following accounts has failed.</p> <p>Your Total transaction amount of <i>TotalAmount</i> was applied to the following account(s).</p> <p>Account: <i>Account Number</i></p> <p>Amount Due: <i>Amount Due</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Canceled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Payment Transaction Has Been Canceled</p> <p>Dear <i>User</i>,</p> <p>A payment transaction that has been entered has been recently canceled:</p> <p>Your Total transaction amount of <i>TotalAmount</i> was applied to the following account(s).</p> <ul style="list-style-type: none"> ■ Account: <i>Account Number</i> ■ Due Date: <i>Due Date</i> ■ Amount Due: <i>Amount Due</i> ■ Amount Canceled: <i>Amount Canceled</i> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Scheduled Payment Canceled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Scheduled Payment Has Been Canceled</p> <p>Dear <i>User</i>,</p> <p>A payment that has been scheduled was recently canceled for the following accounts:</p> <p>Your Total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <ul style="list-style-type: none"> ■ Account: <i>Account Number</i> ■ Due Date: <i>Due Date</i> ■ Amount Due: <i>Amount Due</i> ■ Amount Canceled: <i>Amount Canceled</i> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p>
Refund Request Submitted	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Refund Request has been Submitted</p> <p>Dear <i>User</i>,</p> <p>This is to inform you that your Refund Request has been submitted:</p> <p>Your Total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <ul style="list-style-type: none"> ■ Account Number: <i>Account Number</i> ■ Refunded Amount: <i>Amount Refunded</i> ■ Master Reference Number: <i>Number</i> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Statement Threshold	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: 1st Notice - Your Payment Threshold Exceed</p> <p>Dear <i>User</i>,</p> <p>Your most recent statement has exceeded the payment threshold you have set for yourself.</p> <p>Account: <i>Account Number</i></p> <p>Amount Due: <i>Amount Due</i></p> <p>Configured Amount: <i>Configured Amount</i></p> <p>Please log into your Oracle Billing Insight application at <i>Oracle Billing Insight HTTPS URL</i> and make the necessary changes.</p> <p>Thank You</p>
Payment Threshold	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Final Notice - Your Payment Threshold Exceed</p> <p>Dear <i>User</i>,</p> <p>This email is to notify you that the amount due exceeds the threshold you set for automated payment for the following accounts:</p> <p>Account: <i>Account Number</i></p> <p>Amount Due: <i>Amount Due</i></p> <p>Configured Amount: <i>Configured Amount</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Credit Card Expiration	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Card is About to Expire</p> <p>Dear <i>User</i>,</p> <p>The following card account used for online payments is about to expire:</p> <p>Credit Card Number: <i>Credit Card Number</i></p> <p>Expiration Date: <i>Expiration Date</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Account Created Successfully	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Payment Account Created Successfully</p> <p>Dear <i>User</i>,</p> <p>The following payment account has been created successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>Thank you for using online payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Account Updated Successfully	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Payment Account Updated Successfully</p> <p>The following payment account below has been updated successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment Account Deleted Successfully	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Payment Account Deleted Successfully</p> <p>Dear <i>User</i>,</p> <p>The following payment account has been deleted successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Batch Report Ready	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Batch Reporting Status</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you of the status of your batch report:</p> <p>Report Name: <i>Report Name</i></p> <p>Create Date: <i>Create Date</i></p> <p>Start Date: <i>Start Date</i></p> <p>End Date: <i>End Date</i></p> <p>Status: <i>Status</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Batch Report Request Expiry	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Batch Report Request Expiry</p> <p>Dear <i>User</i>,</p> <p>The following batch requests have expired, please log into the Oracle Billing Insight application and recreate the batch requests if applicable.</p> <p>Thank You</p> <p>Request Name: <i>Request Name</i></p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Login Password Changed	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Password Credentials Have Been Updated</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that your password has just been recently updated. If you are unaware of this action please contact a customer service representative immediately at <i>csrPhoneNumber</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p> <p>Your Security Credentials Have Been Updated</p>
Security Question Changed	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Password Credentials Have Been Updated</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that either your security question and/or answer has just been recently updated. If you are unaware of this action please contact a customer service representative immediately at <i>csrPhoneNumber</i>.</p> <p>Thank You for using the Oracle Billing Insight Application.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Forgot User ID	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your User ID</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that your User ID is <i>XXXX</i>. If you are unaware of this action please contact a customer service representative immediately at <i>csrPhoneNumber</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p> <p>Refund Request has been Submitted</p>

Email Notification Type	Email Template Content for Business Users
User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated. You can now continue managing and paying your bills online using the online Customer Care Application.</p> <p>Why wait! Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the reactivation process.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Agent User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated; you can once again access your online Customer Care Application.</p> <p>Please visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the reactivation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Migrated User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated. You can now continue managing and paying your bills online using the online Customer Care Application.</p> <p>Why wait! Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the reactivation process.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Migrated Agent User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated; you can once again access your online Customer Care Application.</p> <p>Please visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the reactivation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Password Expired	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Administrator Password is Expiring</p> <p>Dear <i>User</i>,</p> <p>Your account password will expire in <i>Number</i> days. Please set a new password.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Email Bounceback	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Invalid Email Address</p> <p>Dear <i>User</i>,</p> <p>Thank you for updating your email address for your online Customer Care application.</p> <p>Please click on the link <i>Secure Validate Email</i> to finish your email validation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Secure Payment Option Selected	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Information about receiving your Bill Summary via PDF</p> <p>Dear <i>User</i>,</p> <p>Your have successfully chosen to receive PDF attachments option.</p> <p>Please save this E-Mail notification for future references.</p> <p>When your monthly billing statement has been received within the Oracle Billing Insight application a Bill Summary via PDF notification will be sent to your inbox. Within this notification there will be a PDF attachment that when you click upon it, you will be prompted for a password.</p> <p>This password consists of the last four digits of your billing account number and your five-digit postal code with no space between them.</p> <p>After opening the PDF attachment you will be able to review your bill summary information.</p> <p>You will be able to visit the online Oracle Billing Insight application anytime to review all statement details, statement history, payments accounts and transactions by clicking <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Account Activation)	<p>From: Admin@Example.com Sent: Date Time To: User Subject: Bank Account Enrollment Notification</p> <p>Dear User,</p> <p><i>If the activation is successful:</i></p> <p>Your payment account that ends in <i>AccountNumber</i> has been successfully activated.</p> <p><i>If the activation is not successful:</i></p> <p>There has been a problem activating your payment account that ends in <i>Partial Account Number</i></p> <p><i>One of the following causes: R01: Insufficient Funds; R02 Account Closed; R03 No Account/Unable to Locate Account; R04 Invalid Account Number; R05 Reserved; R06 Returned per ODFI's Request; R07: Authorization Revoked by Customer (adjustment entries); R08: Payment Stopped or Stop Payment on Item; R09: Uncollected Funds; R10: Customer Advises Not Authorized; Item Is Ineligible, Notice Not Provided, Signatures Not Genuine, or Item Altered (adjustment entries); R11: Check Truncation Entry Return (Specify) or State Law Affecting Acceptance of PPD Debit Entry Constituting Notice of Presentment or PPD Accounts Receivable Truncated Check Debit Entry; R12: Branch Sold to Another DFI; R14: Representative Payee Deceased or Unable to Continue in that Capacity; R15: Beneficiary or Account Holder (Other Than a Representative Payee) Deceased; R16: Account Frozen; R17: File Record Edit Criteria (Specify); R20: Non-Transaction Account; R21: Invalid Company Identification; R22: Invalid Individual ID Number; R23: Credit Entry Refused by Receiver; R24: Duplicate Entry; R29: Corporate Customer Advises Not Authorized; R31: Permissible Return Entry (CCD and CTX only); R33: Return of XCK Entry;</i></p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Bank Account Number Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank account number update is successful:</i></p> <p>Your Bank Account Number has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p><i>If the bank account number update is not successful:</i></p> <p>Your current Bank Account Number is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Bank Routing Number Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank routing number update is successful:</i></p> <p>Your Bank Routing Number has been changed.</p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Old Bank Routing Number was: <i>Old Routing Number</i></p> <p><i>If the bank routing number update is not successful:</i></p> <p>Your current Bank Routing Number is out of date.</p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Current Bank Routing Number is: <i>Old Routing Number</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Bank Account Number and Routing Number Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank account and routing number update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Old Bank Routing Number was: <i>Old Routing Number</i></p> <p><i>If the bank account and routing number update is not successful:</i></p> <p>Your current Bank Account Information is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Current Bank Routing Number is: <i>Old Routing Number</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Bank Account Type Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank account type update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your old Bank Type was: <i>Old Payment Type</i></p> <p><i>If the bank account type update is not successful:</i></p> <p>Your current Bank Account Type is out of date.</p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your current Bank Type is: <i>Old Payment Type</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Bank Account Number and Bank Type Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p><i>If the bank account number and type update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your old Bank Type was: <i>Old Payment Type</i></p> <p><i>If the bank account number and type update is not successful:</i></p> <p>Your current Bank Account Information is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>New Bank Type is: <i>New Payment Type</i></p> <p>Current Bank Type is: <i>Old Payment Type</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment - Bank Account Enrollment (Bank Account, Routing Number, and Bank Type Update)	<p>From: <i>Admin@Example.com</i></p> <p>Sent: <i>Date Time</i></p> <p>To: <i>User</i></p> <p>Subject: Bank Account Enrollment Notification</p> <p><i>If the bank account, routing number, and bank type update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Old Bank Routing Number was: <i>Old Routing Number</i></p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your old Bank Type was: <i>Old Payment Type</i></p> <p><i>If the bank account, routing number, and bank type update is not successful:</i></p> <p>Your current Bank Account Information is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Current Bank Routing Number is: <i>Old Routing Number</i></p> <p>New Bank Type is: <i>New Payment Type</i></p> <p>Current Bank Type is: <i>Old Payment Type</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment Consolidator - Account Enrollment Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors/Rejections while executing the Account Enrollment Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job or rejected by Biller/Consolidator:</p> <p>Job Name: <i>Job Name</i></p> <p>Task Name: <i>Task Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p>Input File Name: <i>File Path</i></p>
Payment Consolidator - Account Enrollment Response Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors while executing the Account Enrollment Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job:</p> <p>Job Name: <i>Job Name</i></p> <p>Task Name: <i>Task Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p>Response File Name: <i>Response File Name</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>Please refer to <i>Log File Path</i> for more information.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment Consolidator - Bill Summary Job Errors	<p>From: <i>Admin@Example.com</i></p> <p>Sent: <i>Date Time</i></p> <p>To: <i>User</i></p> <p>Subject: Errors while executing the Bill Summary Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job:</p> <p>Job Name: <i>Job Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>Please refer to <i>Log File Path</i> for more information.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Business Users
Payment Consolidator - Bill Summary Acknowledgement Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors/Rejections while executing the Bill Summary Acknowledgment Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job or rejected by the payment consolidator:</p> <p>Job Name: <i>Job Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p><i>Rejected Number</i> entries rejected by consolidator.</p> <p><i>If all accounts were rejected:</i></p> <p>All entries rejected by Biller/Consolidator.</p> <p>Please refer to <i>Log File Path</i> for more information.</p> <p><i>Or</i></p> <p>Please refer to <i>Reject File Path</i> for more information about the rejected accounts.</p> <p>This is an automatically generated email. Please do not reply to this message</p>
Single Sign-On (SSO) Bulk Load Error	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: SSO Bulk Load Error Notification</p> <p>Dear <i>Administrator</i>:</p> <p>This notification is to inform you of the status of:</p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p>Input File Name: <i>File Name</i></p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>Please see log file.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Template Content for Consumer Users

This topic shows the content of the email templates preconfigured for each notification type for consumer users of Oracle Billing Insight. (For the content of email notifications for business users, see [“Email Notification Template Content for Business Users” on page 74.](#))

Table 19 shows the template content for each notification type.

Table 19. Notification Types and Associated Email Templates for Consumer Users

Email Notification Type	Email Template Content for Consumer Users
Bill Ready	From: <i>Admin@Example.com</i>
Bill Ready with PDF Attachment	Sent: <i>Date Time</i>
Bill Ready with Daisy Attachment	To: <i>User</i>
	Subject: Your Statement is Ready
	Dear User,
	Your latest statement for the following billing account(s) are now available online.
	Billing Account Number ending in <i>Partial Account Number</i>
	Statement Date: <i>Statement Date</i>
	Statement Amount Due: <i>Amount Due</i>
	Payment Due Date: <i>Due Date</i>
	Service Number: <i>Service Number</i>
	Additional statements have also been loaded, but only <i>Cutoff Number</i> accounts are being displayed within this e-mail.
	To view your statement, click <i>Oracle Billing Insight HTTPS URL</i> .
	This is an automatically generated email. Please do not reply to this message.

Email Notification Type	Email Template Content for Consumer Users
Enrollment (Consumer User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome! You have successfully enrolled in the Oracle Billing Insight Application.</p> <p>You can now start managing and paying your bills online. Why wait! Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the enrollment process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Enrollment (Agent User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome! Your administrator has enrolled you in the Oracle Billing Insight Application.</p> <p>Contact your company administrator at <i>Admin_Email</i> to receive your assigned username. Then visit <i>Oracle Billing Insight HTTPS CSR URL</i> to finish the enrollment process.</p> <p>Log in now and you will be redirected to create your own personal password.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Enrollment (Agent Administrator)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome to the Oracle Billing Insight Application.</p> <p>Now you can start managing other CSRs and assisting customers with e-billing. Please visit <i>Oracle Billing Insight HTTPS CSR URL</i> to finish the enrollment process.</p> <p>Log in now and you will be redirected to create your own personal password.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Enrollment (Migrated Consumer User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome to the Oracle Billing Insight application. You can now start managing and paying your bills online.</p> <p>Why wait! Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the enrollment process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Enrollment (Migrated Agent User)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have Been Enrolled</p> <p>Dear <i>User</i>,</p> <p>Welcome to the Oracle Billing Insight application.</p> <p>Please visit <i>Oracle Billing Insight HTTPS CSR URL</i> to finish the enrollment process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Validate Email	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Please Finish Your Email Validation Process</p> <p>Dear <i>User</i>,</p> <p>Thank you for updating your email address for your online Oracle Billing Insight application.</p> <p>Please click on the link <i>Oracle Billing Insight HTTPS URL</i> to finish your email validation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Job Alert Success	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Job Completed Successfully</p> <p>Dear <i>User</i>,</p> <p>The following job completed successfully:</p> <p><i>Job Name Job Type Job Instance ID DDN</i></p>
Job Alert Failure	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Job Failure</p> <p>Dear <i>User</i>,</p> <p>The following job did not complete successfully:</p> <p><i>Job Name Job Type Job Instance ID DDN Exception</i></p>

Email Notification Type	Email Template Content for Consumer Users
Recurring Payment Confirmation	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Welcome to Automatic Bill Payment</p> <p>Dear <i>User</i>,</p> <p>You have successfully enrolled in the Automatic Bill Payment program for your online accounts.</p> <p>The first automatic bill payment will take place after your next billing cycle. You will receive a payment confirmation email when the payment is submitted.</p> <p>Account Number: <i>Account Number</i></p> <p>Payment Type: <i>Payment Type</i></p> <p>Amount: <i>Amount</i></p> <p>You can continue to make other payments on your account using the Quick Payment option on the Payments Menu. Click <i>Oracle Billing Insight HTTP URL</i> to view your e-bills anytime.</p> <p>Thank you for using Automatic Bill Payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Recurring Payment Configuration Update	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Automatic Bill Payment Settings Have Been Updated</p> <p>Dear <i>User</i>,</p> <p>The Automatic Bill Payment settings for your online accounts have been updated.</p> <p>Account Number: <i>Account Number</i></p> <p>Payment Type: <i>Payment Type</i></p> <p>Amount Due: <i>Amount Due</i></p> <p>Thank you for using Automatic Bill Payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Recurring Payment Deleted	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Automatic Bill Payment Participation Cancelled</p> <p>Dear <i>User</i>,</p> <p>This notification confirms that your participation in Automatic Bill Payment has been cancelled.</p> <p>Account Number: <i>Account Number</i></p> <p>Payment Type: <i>Payment Type</i></p> <p>Amount: <i>Amount</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using Automatic Bill Payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Recurring Payment Transaction Canceled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Recurring Payment Transaction Has Been Cancelled</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that your recurring payment schedule has been deactivated, due to your payment account information being incorrect, and payments can no longer be scheduled. Please log in to the Online Billing Application and make the necessary changes.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i></p> <p>Thank you for using Automatic Bill Payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Successful Quick Payment	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Thank You for Your Quick Payment</p> <p>Dear <i>User</i>,</p> <p>Thank you for submitting the following one-time Quick Payment.</p> <p>Your total transaction amount of <i>TotalAmount</i> was applied to the following account(s).</p> <p>Account Number: <i>Account Number</i></p> <p>Amount: <i>Amount</i></p> <p>Master Reference Number: <i>Reference Number</i></p> <p>Thank you for using Quick Payment. Visit us again to make your next online payment at <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Quick Payment Failure	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Quick Payment Failed</p> <p>A problem occurred during your one-time Quick Payment transaction. The following payment did not process successfully:</p> <p>Amount: <i>Amount</i></p> <p>Master Reference Number: <i>Reference Number</i></p> <p>Please visit <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Due	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: You Have a Payment Due</p> <p>Dear <i>User</i>,</p> <p>You have a payment due for each of the following accounts:</p> <p>Account: <i>Account</i></p> <p>Due Date: <i>Due Date</i></p> <p>Due Amount: <i>Amount Due</i></p> <p>You have additional bills; only <i>Cutoff Number</i> accounts have been displayed above.</p> <p>Thank you for using online payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Scheduled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Automatic Payment Schedule Confirmed</p> <p>Dear <i>User</i>,</p> <p>Automatic payments have been scheduled for the following accounts:</p> <p>Account: <i>Number</i></p> <p>Due Date: <i>Due Date</i></p> <p>Amount Due: <i>Due Amount</i></p> <p>Amount Paid: <i>Configure Amount</i></p> <p>Outstanding Balance: <i>Balance Due</i></p> <p><i>Statement Credit</i> has been credited to your account.</p> <p>Thank you for using online payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Successful Payment	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Thank You for Your Payment</p> <p>Dear <i>User</i>,</p> <p>Your payments for the following accounts completed successfully.</p> <p>Your total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <p>Account: <i>Number</i></p> <p>Amount Due: <i>Due Amount</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Failure	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Payment Did Not Complete Successfully</p> <p>Dear <i>User</i>,</p> <p>Your online payment for the following accounts has failed.</p> <p>Your Total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <p>Account: <i>Number</i></p> <p>Amount Due: <i>Due Amount</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Canceled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Payment Transaction Has Been Canceled</p> <p>Dear <i>User</i>,</p> <p>A payment transaction that has been entered has been recently canceled:</p> <p>Your Total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <ul style="list-style-type: none"> ■ Account: <i>Account Number</i> ■ Due Date: <i>Due Date</i> ■ Amount Due: <i>Amount Due</i> ■ Amount Canceled: <i>Amount Canceled</i> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Scheduled Payment Canceled	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Scheduled Payment Has Been Canceled</p> <p>Dear <i>User</i>,</p> <p>A payment that has been scheduled was recently canceled for the following accounts:</p> <p>Your Total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <ul style="list-style-type: none"> ■ Account: <i>Account Number</i> ■ Due Date: <i>Due Date</i> ■ Amount Due: <i>Amount Due</i> ■ Amount Canceled: <i>Amount Canceled</i> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p>

Email Notification Type	Email Template Content for Consumer Users
Refund Request Submitted	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Refund Request has been Submitted</p> <p>Dear <i>User</i>,</p> <p>This is to inform you that your Refund Request has been submitted:</p> <p>Your Total transaction amount of <i>Amount</i> was applied to the following account(s).</p> <ul style="list-style-type: none"> ■ Account Number: <i>Account Number</i> ■ Refunded Amount: <i>Amount Refunded</i> ■ Master Reference Number: <i>Number</i> <p>Please continue visiting <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Statement Threshold	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: 1st Notice - Your Payment Threshold Exceed</p> <p>Dear <i>User</i>,</p> <p>Your most recent statement has exceeded the payment threshold you have set for yourself.</p> <p>Account: <i>Number</i></p> <p>Amount Due: <i>Due Amount</i></p> <p>Configured Amount: <i>Configured Amount</i></p> <p>Please log into your Oracle Billing Insight application at <i>Oracle Billing Insight HTTPS URL</i> and make the necessary changes.</p> <p>Thank You</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Threshold	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Final Notice - Your Payment Threshold Exceed Reached</p> <p>Dear <i>User</i>,</p> <p>This email is to notify you that the amount due exceeds the threshold you set for automated payment for the following accounts:</p> <p>Account: <i>Number</i></p> <p>Amount Due: <i>Due Amount</i></p> <p>Configured Amount: <i>Configured Amount</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Card Expiration	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Card is About to Expire</p> <p>Dear <i>User</i>,</p> <p>The following card account used for online payments is about to expire:</p> <p>Credit Card Number: <i>Credit Number</i></p> <p>Expiration Date: <i>Expiration Date</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Account Create	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Payment Account Created Successfully</p> <p>Dear <i>User</i>,</p> <p>The following payment account has been created successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>Thank you for using online payment.</p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Account Update	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Payment Account Updated Successfully</p> <p>The following payment account below has been updated successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Account Delete	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Payment Account Deleted Successfully</p> <p>Dear <i>User</i>,</p> <p>The following payment account has been deleted successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Batch Report Ready	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Batch Reporting Status</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you of the status of your batch report:</p> <p>Report Name: <i>Report Name</i></p> <p>Create Date: <i>Create Date</i></p> <p>Start Date: <i>Start Date</i></p> <p>End Date: <i>End Date</i></p> <p>Status: <i>Status</i></p> <p>Please continue visiting <i>Oracle Billing Insight HTTP URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message</p>
Batch Report Request Expiry	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Batch Report Request Expiry</p> <p>Dear <i>User</i>,</p> <p>The following batch requests have expired, please log into the Oracle Billing Insight application and recreate the batch requests if applicable.</p> <p>Thank You</p> <p>Request Name: <i>Request Name</i></p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Login Password Changed	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Password Credentials Have Been Updated</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that your password has just been recently updated. If you are unaware of this action please contact a customer service representative immediately at <i>csrPhoneNumber</i>.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p> <p>Your Security Credentials Have Been Updated</p>
Security Question Changed	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Password Credentials Have Been Updated</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that either your security question and/or answer has just been recently updated. If you are unaware of this action please contact a customer service representative immediately at <i>csrPhoneNumber</i>.</p> <p>Thank You for using the Oracle Billing Insight Application.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Forgot User ID	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your User ID</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that your User ID is <i>XXXX</i>. If you are unaware of this action please contact a customer service representative immediately at <i>csrPhoneNumber</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p> <p>Refund Request has been Submitted</p>

Email Notification Type	Email Template Content for Consumer Users
User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated. You can now continue managing and paying your bills online using the online Customer Care Application.</p> <p>Why wait! Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the reactivation process.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Agent User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated; you can once again access your online Customer Care Application.</p> <p>Please visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the reactivation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Migrated User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated. You can now continue managing and paying your bills online using the online Customer Care Application.</p> <p>Why wait! Please visit <i>Oracle Billing Insight HTTPS URL</i> to finish the reactivation process.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Migrated Agent User Account Reactivated	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Account Has Been Reactivated</p> <p>Dear <i>User</i>,</p> <p>Your account has been reactivated; you can once again access your online Customer Care Application.</p> <p>Please visit <i>Oracle Billing Insight CSR HTTPS URL</i> to finish the reactivation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Password Expired	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Your Administrator Password is Expiring</p> <p>Dear <i>User</i>,</p> <p>Your account password will expire in <i>Number</i> days. Please set a new password.</p> <p>Thank you for using online payment.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
User-Account Relationship Job Status	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Account Relationship Upload Status</p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you of the status of your Business to Consumer User Account Relationship job:</p> <p>Job Name: <i>Job Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p>Input File Name: <i>File Name</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Warning Number</i> entries processed with warnings for your review.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>This is automatically generated email. Please do not reply to this email address.</p>
Email Bounceback	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Invalid Email Address</p> <p>Dear <i>User</i>,</p> <p>Thank you for updating your email address for your online customer care application.</p> <p>Please click on the link <i>Secure Validate Email</i> to finish your email validation process.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Secure Payment Option Selected	<p>From: <i>Admin@Example.com</i></p> <p>Sent: <i>Date Time</i></p> <p>To: <i>User</i></p> <p>Subject: Information about receiving your Bill Summary via PDF</p> <p>Dear <i>User</i>,</p> <p>Your have successfully chosen to receive PDF attachments option.</p> <p>Please save this E-Mail notification for future references.</p> <p>When your monthly billing statement has been received within the Oracle Billing Insight application a Bill Summary via PDF notification will be sent to your inbox. Within this notification there will be a PDF attachment that when you click upon it, you'll be prompted for a password.</p> <p>This password consists of the last four digits of your billing account number and your five-digit postal code with no space between them.</p> <p>After opening the PDF attachment you will be able to review your bill summary information.</p> <p>You will be able to visit the online Oracle Billing Insight application anytime to review all statement details, statement history, payments accounts and transactions by clicking <i>Oracle Billing Insight HTTPS URL</i>.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Account Activation)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the activation is successful:</i></p> <p>Your payment account that ends in <i>AccountNumber</i> has been successfully activated.</p> <p><i>If the activation is not successful:</i></p> <p>There has been a problem activating your payment account that ends in <i>AccountNumber</i></p> <p><i>One of the following causes: R01: Insufficient Funds; R02 Account Closed; R03 No Account/Unable to Locate Account; R04 Invalid Account Number; R05 Reserved; R06 Returned per ODFI's Request; R07: Authorization Revoked by Customer (adjustment entries); R08: Payment Stopped or Stop Payment on Item; R09: Uncollected Funds; R10: Customer Advises Not Authorized; Item Is Ineligible, Notice Not Provided, Signatures Not Genuine, or Item Altered (adjustment entries); R11: Check Truncation Entry Return (Specify) or State Law Affecting Acceptance of PPD Debit Entry Constituting Notice of Presentment or PPD Accounts Receivable Truncated Check Debit Entry; R12: Branch Sold to Another DFI; R14: Representative Payee Deceased or Unable to Continue in that Capacity; R15: Beneficiary or Account Holder (Other Than a Representative Payee) Deceased; R16: Account Frozen; R17: File Record Edit Criteria (Specify); R20: Non-Transaction Account; R21: Invalid Company Identification; R22: Invalid Individual ID Number; R23: Credit Entry Refused by Receiver; R24: Duplicate Entry; R29: Corporate Customer Advises Not Authorized; R31: Permissible Return Entry (CCD and CTX only); R33: Return of XCK Entry;</i></p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Bank Account Number Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank account number update is successful:</i></p> <p>Your Bank Account Number has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p><i>If the bank account number update is not successful:</i></p> <p>Your current Bank Account Number is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Bank Routing Number Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank routing number update is successful:</i></p> <p>Your Bank Routing Number has been changed.</p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Old Bank Routing Number was: <i>Old Routing Number</i></p> <p><i>If the bank routing number update is not successful:</i></p> <p>Your current Bank Routing Number is out of date.</p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Current Bank Routing Number is: <i>Old Routing Number</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Bank Account Number and Routing Number Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank account and routing number update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Old Bank Routing Number was: <i>Old Routing Number</i></p> <p><i>If the bank account and routing number update is not successful:</i></p> <p>Your current Bank Account Information is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Current Bank Routing Number is: <i>Old Routing Number</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Bank Account Type Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p>Dear <i>User</i>,</p> <p><i>If the bank account type update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your old Bank Type was: <i>Old Payment Type</i></p> <p><i>If the bank account type update is not successful:</i></p> <p>Your current Bank Account Type is out of date.</p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your current Bank Type is: <i>Old Payment Type</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Bank Account Number and Bank Type Update)	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Bank Account Enrollment Notification</p> <p><i>If the bank account number and bank type update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your old Bank Type was: <i>Old Payment Type</i></p> <p><i>If the bank account number and bank type update is not successful:</i></p> <p>Your current Bank Account Information is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>New Bank Type is: <i>New Payment Type</i></p> <p>Current Bank Type is: <i>Old Payment Type</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment - Bank Account Enrollment (Bank Account, Routing Number, and Bank Type Update)	<p>From: Admin@Example.com Sent: Date Time To: User Subject: Bank Account Enrollment Notification</p> <p><i>If the bank account, routing number, and bank type update is successful:</i></p> <p>Your Bank Account Information has been changed.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Old Bank Account Number was: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Old Bank Routing Number was: <i>Old Routing Number</i></p> <p>Your new Bank Type is: <i>New Payment Type</i></p> <p>Your old Bank Type was: <i>Old Payment Type</i></p> <p><i>If the bank account, routing number, and bank type update is not successful:</i></p> <p>Your current Bank Account Information is out of date.</p> <p>New Bank Account Number is: <i>New Payment Account</i></p> <p>Current Bank Account Number is: <i>Old Payment Account</i></p> <p>New Bank Routing Number is: <i>New Routing Number</i></p> <p>Current Bank Routing Number is: <i>Old Routing Number</i></p> <p>New Bank Type is: <i>New Payment Type</i></p> <p>Current Bank Type is: <i>Old Payment Type</i></p> <p>Please login to change your profile.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Consolidator - Account Enrollment Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors/Rejections while executing the Account Enrollment Job - <i>Job Name</i> with Job Instance Id <i>Job Instance ID</i></p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job or rejected by Biller/Consolidator:</p> <p>Job Name: <i>Job Name</i> Task Name: <i>Task Name</i> Job Instance ID: <i>Instance ID</i> Input File Name: <i>Input File Name</i></p> <p><i>Success Number</i> entries processed successfully. <i>Failed Number</i> entries failed to be processed. <i>Rejected Number</i> entries rejected by Biller/Consolidator.</p> <p>Please refer to <i>Log File Name</i> for more information. This is automatically generated email. Please do not reply to this email address.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Consolidator - Account Enrollment Response Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors while executing the Account Enrollment Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job:</p> <p>Job Name: <i>Job Name</i></p> <p>Task Name: <i>Task Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p>Response File Name: <i>Response File Name</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>Please refer to <i>Log File Path</i> for more information.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>
Payment Consolidator - Bill Summary Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors while executing the Bill Summary Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job:</p> <p>Job Name: <i>Job Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>Please refer to <i>Log File Path</i> for more information.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

Email Notification Type	Email Template Content for Consumer Users
Payment Consolidator - Bill Summary Acknowledgement Job Errors	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: Errors/Rejections while executing the Bill Summary Acknowledgment Job - <i>Job Name</i> with job instance ID <i>Job Instance ID</i></p> <p>Dear <i>User</i>,</p> <p>This notification is to inform you that one or more records were not successfully processed when running the following job or rejected by the payment consolidator:</p> <p>Job Name: <i>Job Name</i></p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p><i>Success Number</i> entries processed successfully.</p> <p><i>Failed Number</i> entries failed to be processed.</p> <p><i>Reject Number</i> entries rejected by consolidator.</p> <p><i>If all accounts were rejected:</i></p> <p>All entries rejected by Biller/Consolidator.</p> <p>Please refer to <i>Log File Path</i> for more information.</p> <p><i>Or</i></p> <p>Please refer to <i>Reject File Path</i> for more information about the rejected accounts.</p> <p>This is an automatically generated email. Please do not reply to this message</p>
Single Sign-On (SSO) Bulk Load Error	<p>From: <i>Admin@Example.com</i> Sent: <i>Date Time</i> To: <i>User</i> Subject: SSO Bulk Load Error Notification</p> <p>Dear <i>Administrator</i>:</p> <p>This notification is to inform you of the status of:</p> <p>Job Instance ID: <i>Job Instance ID</i></p> <p>Input File Name: <i>File Name</i></p> <p><i>Failed Number</i> entries failed to be processed.</p> <p>Please see log file.</p> <p>This is an automatically generated email. Please do not reply to this message.</p>

SMS Notification Template Content

This topic shows the content of the SMS templates preconfigured for each type of notification in Oracle Billing Insight. [Table 20](#) shows the template text provided for each notification type.

Table 20. Notification Types and SMS Templates

SMS Notification Type	SMS Template Content
Bill Ready	From: <i>Admin@Example.com</i> Subject: A New Statement is Available Online
Bill Ready with PDF Attachment	A new statement has been generated and is now available online.
Bill Ready with Daisy Attachment	Billing Account Number ending in <i>Partial Account Number</i> Additional statements have also been loaded, but only <i>Cutoff Number</i> accounts are being displayed. To view your statement, go to <i>Oracle Billing Insight HTTPS URL</i> .
Recurring Payment Confirmation	From: <i>Admin@Example.com</i> Subject: Welcome to Automatic Bill Payment You have successfully enrolled in the Automatic Bill Payment program for your online billing accounts. The first automatic bill payment will take place after your next billing cycle. You will receive a payment confirmation message when the payment has been submitted.
Recurring Payment Configuration Update	From: <i>Admin@Example.com</i> Subject: Your Automatic Bill Payment Settings Have Been Updated An Automatic Bill Payment has been recently updated for at least one of your online billing accounts.
Recurring Payment Delete	From: <i>Admin@Example.com</i> Subject: Automatic Bill Payment Participation Cancelled This notification confirms that your participation in Automatic Bill Payment has been cancelled.
Successful Quick Payment	From: <i>Admin@Example.com</i> Subject: Thank You for Your Payment Thank you for submitting an online payment Account Number: <i>Account Number</i> in the Amount of <i>Payment Amount</i> Please use this reference number: <i>Reference Number</i> To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i> .

SMS Notification Type	SMS Template Content
Quick Payment Failure	<p>From: <i>Admin@Example.com</i> Subject: Payment Has Failed</p> <p>A problem has occurred with your Payment transaction <i>Reference Number</i> that was in the amount of <i>Amount</i>.</p> <p>To view your statement go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Quick Payment Cancel	<p>From: <i>Admin@Example.com</i> Subject: Your Payment Transaction Has Been Canceled.</p> <p>Your payment transaction Account Number <i>Account Number</i> in the amount of <i>Configured Amount</i> has recently been canceled.</p> <p>To view payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Scheduled Payment Cancel	<p>From: <i>Admin@Example.com</i> Subject: Your Scheduled Payment Has Been Canceled</p> <p>A scheduled payment transaction Account Number <i>Account Number</i> in the amount of <i>Configured Amount</i> has recently been canceled.</p> <p>To view payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Quick Payment Updated	<p>From: <i>Admin@Example.com</i> Subject: Your Payment Transaction Has Been Updated</p> <p>Your payment transaction Account Number <i>Account Number</i> in the amount of <i>Configured Amount</i> has recently been updated.</p> <p>To view payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Scheduled Payment Update	<p>From: <i>Admin@Example.com</i> Subject: Your Scheduled Payment Has Been Updated</p> <p>A scheduled payment transaction has recently been updated</p> <p>Account Number: <i>Account Number</i></p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Payment Due	<p>From: <i>Admin@Example.com</i> Subject: Payment Due</p> <p>You have a payment that is due on an outstanding billing account.</p> <p>Account: <i>Account Number</i></p> <p>You may have additional accounts that have a payment due, as only <i>Cutoff Number</i> billing accounts have been displayed.</p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>

SMS Notification Type	SMS Template Content
Payment Scheduled	<p>From: <i>Admin@Example.com</i> Subject: Automatic Payment Schedule Confirmed</p> <p>A payment has automatically been scheduled for a billing account that is coming due.</p> <p>Account: <i>Account Number</i></p> <p>You may have additional payments scheduled automatically, as only <i>Cutoff Number</i> billing accounts have been displayed.</p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Successful Payment	<p>From: <i>Admin@Example.com</i> Subject: Thank you for Your Payment</p> <p>Thank you. Your online payment has been processed successfully.</p> <p>Account: <i>Account Number</i></p> <p>You may have additional payments that have been processed successfully, as only <i>Cutoff Number</i> billing accounts have been displayed.</p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Payment Failure	<p>From: <i>Admin@Example.com</i> Subject: Your Payment Did Not Complete Successfully</p> <p>Sorry to inform you but your payment submitted online has failed.</p> <p>Account: <i>Account Number</i></p> <p>You may have additional payments that have failed, as only <i>Cutoff Number</i> billing accounts have been displayed.</p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Statement Threshold	<p>From: <i>Admin@Example.com</i> Subject: 1st Notice - Your Payment Threshold Exceed</p> <p>Your latest statement loaded online has exceeded the payment threshold set for online payments.</p> <p>Account: <i>Account Number</i></p> <p>Configured Amount: <i>Configured Amount</i></p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>

SMS Notification Type	SMS Template Content
Payment Threshold	<p>From: <i>Admin@Example.com</i> Subject: Final Notice - Your Payment Threshold Exceed</p> <p>Final Notice: This is to notify you that the amount due on a billing account exceeds the threshold set for automated online payments.</p> <p>You may have additional statements that have exceeded their threshold, as only <i>Cutoff Number</i> billing accounts have been displayed.</p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Credit Card Expiration	<p>From: <i>Admin@Example.com</i> Subject: Your Credit Card is About to Expire</p> <p>Your online payment account using a credit card for online payments is about to expire.</p> <p>Credit Card Number: <i>Card Number</i></p> <p>Expiry Date: <i>Expiration Date</i></p> <p>To view your payment activity go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Payment Account Created Successfully	<p>From: <i>Admin@Example.com</i> Subject: Payment Account Created Successfully</p> <p>The following payment account(s) have been created successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>To view your payment accounts go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Payment Account Updated Successfully	<p>From: <i>Admin@Example.com</i> Subject: Payment Account Updated Successfully</p> <p>The following payment account(s) have been updated successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>To view your payment accounts go to <i>Oracle Billing Insight HTTPS URL</i>.</p>
Payment Account Deleted Successfully	<p>From: <i>Admin@Example.com</i> Subject: Payment Account Updated Successfully</p> <p>The following payment account(s) have been deleted successfully:</p> <p>Account Name: <i>Account Name</i></p> <p>To view your payment accounts go to <i>Oracle Billing Insight HTTPS URL</i>.</p>

SMS Notification Type	SMS Template Content
Batch Report Ready	From: <i>Admin@Example.com</i> Subject: Batch Report Status Your batch report has completed. <i>Report Status</i> To view your batch reports go to <i>Oracle Billing Insight HTTPS URL</i> .
Batch Report Request Expiry	From: <i>Admin@Example.com</i> Subject: Batch Report Request Expiry A batch request has expired. Please recreate the batch request if applicable. To view your batch reports go to <i>Oracle Billing Insight HTTPS URL</i> .

Customizing the Content of Notifications

Oracle Billing Insight provides email and SMS notification templates for each type of message it supports. You can customize some notification content for your organization by updating the notification properties file contained in the application resource bundles and regenerating the template XSL files using the automated template generator. The notification properties file contains the strings and text used in the composition of messages.

Oracle Billing Insight can send notifications to users based on enrollment events as well as various Self-Service lifecycle events, such as when a new bill is ready for viewing online, a payment is due, scheduled, sent, and so on. Notifications are classified into batch and instant notifications, based on whether the notification is generated by a batch job defined and run in the Command Center or a user's action. You customize email and SMS notifications separately.

If you have localized your Oracle Billing Insight applications, then be sure to update the corresponding notification property file for each language (if appropriate). For more information about localization, see [“Localizing the User Interface” on page 32](#).

To customize the content of email or SMS notifications

- 1 Verify that Oracle Billing Insight is installed and deployed.
- 2 Edit the notification resource properties file, such as `NotificationsEmail_en_US.properties` or `NotificationsSMS_en_US.properties`, with your customizations. The files are located in the following directory:
 - **UNIX.** `EDX_HOME/config/resourcebundle/com/edocs/application/resources/`
 - **Windows.** `EDX_HOME\config\resourcebundle\com\edocs\application\resources`
- 3 Generate a new email or SMS template XML file:
 - a Edit the template generator script for your implementation, located in the following directory. The template generator script generates an email or SMS notification template XML file based on the notification properties file.

- ❑ **UNIX.** `EDX_HOME/bin/notification/generateNotificationTemplate.sh`
 - ❑ **Windows.** `EDX_HOME\bin\notification\generateNotificationTemplate.cmd`
- b** Update the value of `EDX_HOME` and using the full path names for your installation.
- c** Run the template generator script. To generate the email template file, `template_en_US.xml`, run the following command:
 - ❑ **UNIX.** `./generateNotificationTemplate.sh -l en_US -t sms`
 - ❑ **Windows.** `generateNotificationTemplate.cmd -l en_US -t sms`

To generate the SMS template file, `template_sms_en_US.xml`, run the following command:

 - ❑ **UNIX.** `./generateNotificationTemplate.sh -l en_US -t sms`
 - ❑ **Windows.** `generateNotificationTemplate.cmd -l en_US -t sms`
- 4** By default, Oracle Billing Insight automatically detects updates to the external resource files every five minutes. If you have turned this feature off in the global configuration file, then you must manually restart the server for the notification changes to take effect. For more information about automatic detection of changes to resource files, see [“Changing the Frequency of Automatic Resource File Updates” on page 38](#).

Customizing the Default Display Patterns Used in Notifications

You can change the default display patterns Oracle Billing Insight uses to display the following data in email and SMS notifications:

- Date
- Time
- Date and time
- Integers
- Decimals (double)
- Amounts

The default notification display patterns are defined in the `notification.cfg.xma.xml` file. The valid display patterns are defined in the `ApplicationLabels_LanguageCode_Country.properties` file, as shown:

```
## Global Patterns used for reporting engine
#representation of integer
global.pattern.number.integer=#,##0
#representation of a decimal
global.pattern.number.decimal=#,##0.00
#representation of a percentage number
global.pattern.number.percent=#0.00%
#representation of amount value with currency
global.pattern.number.amount=\u00a4#,##0.00
global.pattern.number.amount2=\u00a4#,##0.00;\u00a4(,##0.00)
```

```
global . pattern . number . amount3=#, ##0; (#, ##0)
global . pattern . number . amount4=#, ##0.00; (#, ##0.00)
global . pattern . date . short=M/d/yy
global . pattern . date . long=MMM/dd/yyyy
global . pattern . date . input=MM/dd/yyyy
global . pattern . date . jquery=MM/dd/yy
global . pattern . time . short=HH: mm
global . pattern . time . long=HH: mm: ss
global . pattern . date . di sp=MMM dd, yyyy
global . pattern . date . expi ration=MM/yyyy
global . pattern . date . detai ls=MM/dd/yyyy hh: mm aaa
global . pattern . thousands=,
global . pattern . deci mal PI aces=2
global . pattern . date . medi um=MM/dd/yyyy
global . pattern . date . ti me=MM/dd/yyyy HH: mm: ss
global . pattern . deci mal Poi nt=.
#The separator and order of hh, mm and ss can be changed, but letters hh, mm, ss can
not be changed.
global . pattern . durati on=hh: mm: ss
#The letter, separator and the order of the string can be changed. But the order and
separator need to be consistent wi th the value of global . pattern . durati on
global . pattern . durati onDesc=hh: mm: ss
```

The defaults set in the notification.cfg.xma.xml are used in all Oracle Billing Insight applications: Self-Service, Assisted Service, Command Center, and Web Services. For example, you can change the default date pattern from global.pattern.date.medium (MM/dd/yyyy), to global.pattern.date.long (MMM/dd/yyyy).

To customize the default display patterns used in email notifications

- 1 Edit the notification.cfg.xma.xml file, located in the following directory:
 - **UNIX.** `EDX_HOME/xma/confi g/com/edocs/common/noti fi cati on\`
 - **Windows.** `EDX_HOME\xma\confi g\com\edocs\common\noti fi cati on\`
- 2 Edit the appropriate values in the patternConfig property in the <bean id="GlobalConfigurationBean" JavaBean. Specify one of the display patterns defined in the ApplicationLabels_LanguageCode_Country.properties file:

```
<property name="patternConfi g"><map merge="default t">
<entry key="default tDatePattern"><val ue>gl obal . pattern . date . medi um</val ue>
</entry>
<entry key="default tTi mePattern"><val ue>gl obal . pattern . ti me . l ong</val ue>
</entry>
<entry key="default tDateTi mePattern"><val ue>gl obal . pattern . date . ti me</val ue>
</entry>
<entry key="default tI ntPattern"><val ue>gl obal . pattern . number . i nteger</val ue>
</entry>
<entry key="default tDoubl ePattern"><val ue>gl obal . pattern . number . deci mal </val ue>
</entry>
```



```
<entry key="default tAmountPattern"> <val ue>gl obal . pattern. number. amount</val ue>
</entry>
</map> </property>
```

Configuring Messaging Properties

You can configure several properties that control email and Short Message Service (SMS) text message delivery for your organization.

You can configure the following properties:

- **Global notification type settings.** The settings for each notification type, which indicate whether to send or suppress the notification type for all applicable users or allow individual users to choose whether to receive the notification.
- **Maximum email queue threads.** The maximum number of message threads to create when sending email and SMS messages. The default is 10 threads.
- **Maximum queue elements per thread.** Email messages are sent in batches, by thread. The maximum number of messages that each thread must send in each batch. The default is 30 messages.
- **Queue dispatcher sleep period.** The time period, in seconds, that the dispatcher must sleep between sending email and SMS messages, to allow other threads to complete sends before removing queued messages. The default is 5 seconds.
- **Queue hanging timeout period.** The time period, in seconds, that the dispatcher must wait before deciding that the email host or SMS gateway is not responding and queue messages. The default period is 15 seconds.
- **Queue storage directory.** The directory used to temporarily store undeliverable email and SMS messages. By default, the queue storage directory is:
 - **UNIX.** `EDX_HOME/mail queue/`
 - **Windows.** `EDX_HOME\mail queue\`

To configure messaging properties

- 1 Edit the notification.cfg.xma.xml file, located in the following directory:
 - **UNIX.** `EDX_HOME/xma/config/com/edocs/common/notification`
 - **Windows.** `EDX_HOME\xma\config\com\edocs\common\notification`

- 2 To specify global settings for each notification type valid in your edition of Oracle Billing Insight, specify the following values under the preference bean for each notification type. Each notification type is defined as a bean under the notificationPreferenceConfig property in the NotificationPreferenceManagerTarget bean.

Notification Configuration Property	Description
sms	<p>The preference setting for an SMS notification of this message type:</p> <ul style="list-style-type: none"> ■ True. All applicable users receive the SMS notification type. Individual users cannot set a preference. ■ False. The SMS notification type is not generated globally. Individual users cannot set a preference. ■ notSet. No global setting is specified. Individual users can set their own preference.
selectedByDefault	<p>The preference setting for email notification of this message type:</p> <ul style="list-style-type: none"> ■ True. All applicable users receive the email notification type. Individual users cannot set a preference. ■ False. The email notification type is not generated globally. Individual users cannot set a preference. ■ notSet. No global setting is specified. Individual users can set their own preference.
ADAOnly	This preference setting indicates that this notification is displayed only in the Self-Service application when the user is ADA enabled.
resource	<p>The resource bundle of the message shown on the notification preference UI page in the Self-Service application.</p> <p>This resource bundle is used to automatically generate the notification preference page. If a new notification without a special preference parameter, such as numDaysBeforeDueDate, is later added to the page, then the JSP does not have to change. If this property is not specified, then the preference will not appear in the UI.</p>
additionalParams	Additional parameters required by this notification template, such as numDaysBeforeDueDate in paymentDueNotification, which you can be extended.

Notification Configuration Property	Description
emailCheckBoxTitle	To support ADA, this property adds the title of email check boxes to the user notification configuration page in the Self-Service application.
smsCheckBoxTitle	To support ADA, this property adds the title of SMS check boxes to the user notification configuration page in the Self-Service application.

For example, the following code in the notification.cfg.xma.xml file shows where you specify the global properties for the bill-ready notification type:

```

bean id="NotificationPreferenceManagerTarget"
class="com.edocs.common.notification.core.NotificationPreferenceManager"
scope="singleton" lazy-init="default" autowire="default" dependency-
check="default">
- <!-- notification preference setting, this configuration can enable/disable
certain notification for all users on specified channel
-->
- <property name="notificationPreferenceConfig">
- <list merge="default">
- <bean id="BillNotificationConfigBean"
class="com.edocs.common.notification.config.PreferenceBean" lazy-init="default"
autowire="default" dependency-check="default">
- <!-- Notification preference name
-->
- <property name="name">
<value>newStmtAvailabl</value>
</property>
- <!-- sms notification preference setting. Its value specifies enable/disable
this message sending for certain channel.
-->
- <property name="sms">
<value>notSet</value>
</property>
- <!-- email notification preference setting. Its value specifies enable/disable
this message sending for certain channel.
-->
- <property name="selectedByDefault">
<value>notSet</value>
</property>
- <!-- Resource bundle of message which is shown in E-Billing notification
preference UI page. It is used to automatically generate notification preference
page.
-->
- <property name="resource">
<value>user.message.notification.billReady</value>
</property>
- <property name="emailCheckBoxTitle">
<value>global.label.billReady</value>

```

```
</property>
- <property name="smsCheckBoxTitle">
<value>global .label .bill ReadySMS</value>
</property>
</bean>
```

Only those settings for notification types valid in your edition are recognized. For information about valid email notification types for business users, see [“Email Notification Template Content for Business Users” on page 74](#). For information about valid email notification types for consumer users, see [“Email Notification Template Content for Consumer Users” on page 101](#).

- 3 To update any of the following email notification properties, specify the value under the corresponding property.

Property Name	Email Notification Property Function
mailQueueThreadMax	The maximum number of concurrent email queue threads.
mailQueueElementsPerThread	The maximum number of queue elements for each thread. Each outgoing email thread receives a batch of notification records from the Messenger_Queue_Table database table, then sends the notifications. This parameter is the notification record count for a batch.
mailQueueDispatcherSleepPeriod	The queue dispatcher sleep period, in seconds. When all email threads are running and there more notification records must still be sent, then this function defines the period the dispatcher waits for a free thread to process the next batch of notification records.
mailQueueHangingTimeout	The queue delay timeout period.
mailQueueStorageDirectory	The queue storage directory, used by the Notifier job. The NotificationComposer task composes a collection of email and saves it to a temporary path. The NotificationDispatcher task loads the email file in this path then sends the notifications. Only a single level directory for this path is supported.
maxEntriesPerUser	The maximum number of entries that can be consolidated.

Adding a Custom Message Provider

Use the following procedure to add a custom message provider.

To add a custom message provider

- 1 Create a new message type and accompanying class to override the com.edocs.common.api.notification.AbstractMessage class.

- 2 Initialize the new class with all the information necessary to send a message of that type using the custom messaging provider.

It is not necessary to include static information that does not vary by individual message. For example, an email address is necessary if you are using an SMTP provider and the address changes for every message. However, the SMTP hostname does not change with every message.

If an existing implementation has all the necessary information, then you can use that implementation without modification, except that you must provide a unique message by calling the `setMessageType` method on the object after object creation. The class `com.edocs.common.notification.extensions.InternalMessage` provides a reference implementation.

- 3 Create a new transport class with the logic for sending messages using the new message provider, overriding the `com.edocs.common.notification.extensionsapi.AbstractTransporter` class.

This transport class contains the methods for sending the message. All of the information necessary for this class is available from the `IMessage` object.

- 4 Add a JavaBean definition to the `notification.cfg.xma.xml` file. Give the file a name that ends with Bean, such as `CustomTransportBean`.

This name (without the Bean part) is used inside the `NotificationService` class to return the correct messenger, for example: `IMessenger messenger = MessengerFactory.getMessenger(CustomTransport)`.

The `notification.cfg.xma.xml` file is located in the following directory:

- **UNIX.** `EDX_HOME/xma/config/com/edocs/common/notification\`
- **Windows.** `EDX_HOME\xma\config\com\edocs\common\notification\`

- 5 In the JavaBean definition, add all the properties required by the transporter to use the messaging provider, such as the SMTP name. Do not include items like the email address, which the message object provides.

The class `com.edocs.common.notification.extensions.TrueTransporter` and the JavaBean definition for `TrueTransporterBean` in the `notification.cfg.xma.xml` file provide a reference implementation.

- 6 Create a new `NotificationService` class that determines when to use the transport class, which you wrote in [Step 3](#), which overrides the `com.edocs.common.api.notification.INotificationService` class.

`INotificationService` has two methods for sending instant messages and batch messages. These methods decide which transport to use based on the `IMessage` object being passed in. You must supply the logic to call the `MessengerFactory` with the transport in [Step 3](#) as the transport type, for the types of Messages you want that transport to send.

- 7 After you create the `NotificationService` class, add or modify the JavaBean called `NotificationService` to the `notification.cfg.xma.xml` file. `NotificationServiceFactory` looks up the JavaBean called `NotificationService` from the XML file.

The class `com.edocs.common.notification.core.NotificationService` and the JavaBean definition for `NotificationService` in the `notification.cfg.xma.xml` file provide a reference implementation.

Configuring the Secure Bill-Ready Notification Password

The bill-ready email notification is sent to users with a PDF attachment that is preconfigured with an open password. Oracle Billing Insight provides an interface you can customize to require a password to open the notification.

To configure a password to open the PDF in a bill-ready notification

- 1 Edit the `statement.xma.xml` file, located in the following directory:
 - **UNIX.** `EDX_HOME/xma/config/modules/statement`
 - **Windows.** `EDX_HOME\xma\config\modules\statement`
- 2 Create a custom class specifying the password requirement for customers opening the PDF file attached to bill-ready notifications.
- 3 Under the `notificationPasswordProvider` bean ID in the `statement.xma.xml` file, replace the default class, `com.edocs.common.statement.provider.NotificationPasswordProvider`, with the name of your custom class.
- 4 Implement the interface, replacing the default implementation class in the `INotificationPasswordProvider` bean.

Disabling SMS Notifications

You can optionally disable SMS notifications for your organization.

To disable SMS notifications

- 1 Open the `globalConfig.xma.xml` file, located in the following directory:
 - **UNIX.** `EDX_HOME/xma/config/modules`
 - **Windows.** `EDX_HOME\xma\config\modules`
- 2 Set the `enableSMS` property to `N` to disable SMS functionality, for example:

```
<property name="enableSMS"><value>N</value></property>
```

You can reset this property to `Y` to enable SMS notifications at any time.

7

Using the Reporting Engine

This chapter covers using the Reporting Engine feature in Oracle Billing Insight. It includes the following topics:

- [Reporting Engine Features on page 143](#)
- [Components Used by the Reporting Engine on page 148](#)
- [Core Reporting Features on page 184](#)
- [Customizing the Reporting Engine on page 192](#)
- [Customizing Threshold Values for Batch Reporting on page 195](#)
- [Customizing Charts on page 197](#)
- [Configurable Chart Properties on page 198](#)
- [Customizing the Statement Summary Chart on page 206](#)
- [Changing the Local Storage Location for the HTML Report Formatting Files on page 207](#)

Reporting Engine Features

The Reporting Engine is used for much more than just reporting. The Reporting Engine can present any data you can retrieve from any data sources, such as RDBMS or CSV files.

Possible use cases supported by the Reporting Engine include:

- Viewing statements and invoices
- Analytic reports, such as the top 10 most expensive calls
- Cost center reports (hierarchy report), such as cost summary by cost centers
- Reports, such as most frequent users or logging analysis
- Address book
- Email content composition

The Reporting Engine offers great tools to help you implement these use cases. It uses XML to describe how you want to present a report. Then the Reporting Engine does the rest of the work for you, including retrieving data from data source, formatting, and then presenting the data to the user through Velocity templates.

The Reporting Engine is designed to do the following:

- **Use XML Files:** Create an XML file to describe the report you want to create. The Reporting Engine automatically generates that report for you, in variety of formats, such as HTML or CVS.

- **Have an extendable, customizable UI:** You can extend the Reporting Engine to support any UI customization. The Reporting Engine uses Velocity templates, which is a powerful reporting tool based on Model-View-Controller (MVC) technology.
- **Be maintainable:** The Reporting Engine is MVC-based and offers the best separation of presentation logic and business logic, which makes it maintainable.

The following features are offered by the Reporting Engine:

- **Multiple data sources.** The Reporting Engine connects to multiple data sources, including SQL data source, object data source, and DSV data source.
- **Prompts.** Prompts allow you to select desired data from data source.
- **Interactive sorting.** Sorting can be case sensitive or insensitive.
- **Interactive grouping.** Data is grouped by a particular column's values.
- **Calculator operations.** Summary, Boolean, minimal, average and count operations are supported.
- **Charting.** This feature supports bar, stack bar charts, line, and pie charts.
- **Template.** Template-based presentation for both Web-based and non-Web based applications.
- **Formatting.** Support is provided for locale based format for numeric values and dates.
- **Print friendly view.** This feature allows you to generate a print friendly view for printing.
- **CSV download.** CSV download lets you download the report in CSV format.
- **XML download.** XML download lets you download the report in XML format.
- **PDF download.** PDF download lets you download the report in PDF format. PDF format is not generated automatically for all the reports. You must create an RTF template file for particular reports to generate in PDF format.
- **Paging.** Pages through a large set of data.
- **Custom report.** Custom reports allow users to create their own reports and save them for later retrieval.
- **Internationalization.** Standard Java resource bundle based internationalization.
- **Drilldown and Breadcrumb links.** The Report engine offers a way to drill down to different reports and drill back through breadcrumb links.
- **Seamless integration with Struts and Tiles.** The Reporting Engine is not tied to a particular presentation framework, but offers excellent support for Struts and Tiles.
- **Batch report.** When it takes a long time to generate a report online, you can use the batch report feature to send a request which will be processed offline.
- **Unlimited Paging.** If the data source has too many rows and there is a performance issue retrieving all the rows, then the Reporting Engine can retrieve them in batches. The paging through these batches is seamless and retrieving result sets in batches is invisible to the user.

The new Velocity version has not been tested with Oracle Billing Insight. Also, the default velocity.properties file has been changed for Oracle Billing Insight. The velocityCount starts from 0 instead of the default 1, and the templates can be loaded as file and also as class.

You cannot define your own data source. One way to get around this is to retrieve your data as a list of objects and then use the Object Datasource feature to present it through the Reporting Engine.

You cannot extend report XML to add your own custom tags.

Reporting Engine Architecture

Figure 4 shows the Reporting Engine architecture based on the UML component model.

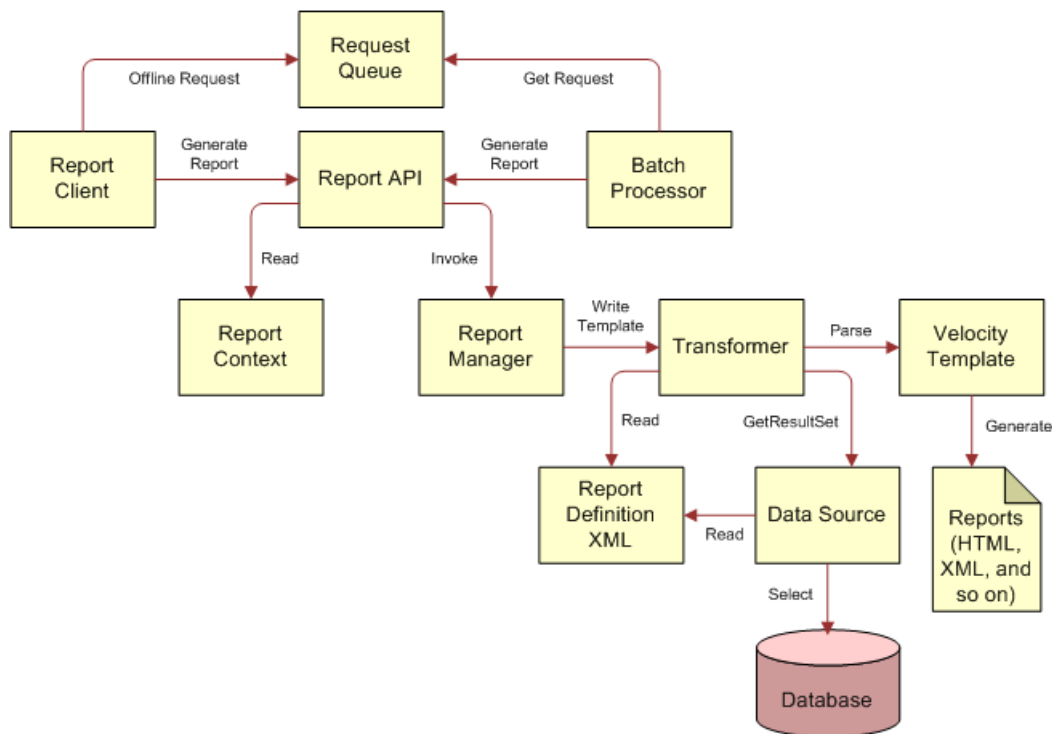


Figure 4. Reporting Engine Architecture

The overall Reporting Engine architecture follows the MVC model. The data source is the model, the report manager, transformer and report XML are the controller, and the template is the view.

- **Report Client.** This client calls the Report API to generate reports. The client can be a Web Client, such as JSP and Servlet or Struts and Tiles, or it can be a regular standalone application.
- **Report API.** This is a set of APIs that the reporting client can use to generate a report. For information about how this API works, go to the Oracle Billing Insight Javadoc as described in ["Accessing Oracle Billing Insight Data Dictionary and Javadoc" on page 25](#).

- **Report Context.** The Report context is used by the Report Client to exchange information with the Report Engine. It includes the information passed from the client that is used to bind the SQL query parameters and parse the templates. For example, the context can contain user session information, such as login name, current role and organization level. Or it can contain report input information, such as the date range used to generate reports. All the objects in the context can be accessed using Velocity templates.
- **Request Queue.** This queue holds all offline batch report requests. Users can generate reports immediately, or they can request that the reports be generated offline. Offline reports send email notification when the reports are ready. The Request Queue is a JMS queue, and holds all offline report requests.
- **Batch Processor.** The processor retrieves offline report requests from the Request Queue, and sends them to the Report Engine for processing. The batch processor is a batch job that runs in Command Center.
- **Report Manager.** The Report Manager is the central controller of the report engine. It receives reporting requests from the client, and invokes the appropriate data source and transformer to perform the desired processing.
- **Data source.** This item represents the data source. The data source can be an SQL statement, an Object or a CSV file.
- **Transformer.** The transformer transforms the query result from presentation, and applies a set of computations on it, including sorting, grouping, paging, aggregation (summary, average, Boolean, minimal, count), and formatting. The transformer can also cache the data retrieved from data source so that the operations can be performed in the cache data (which reduces Oracle Billing Insight database accesses).
- **Velocity Template.** Templates are used to generate desired report output views. The templates are based on Velocity, and can generate any text reports, such as HTML or CSV. However, it is not currently possible to use Velocity to generate binary reports.
- **Report Definition XML.** Report XML files control how reports are generated. To create your own report, create a report definition in a report XML file. You can have multiple report XML files, and each report XML file can define multiple reports.

Reporting Engine Object Model

Figure 5 shows the Reporting Engine object model. Only the main objects are shown.

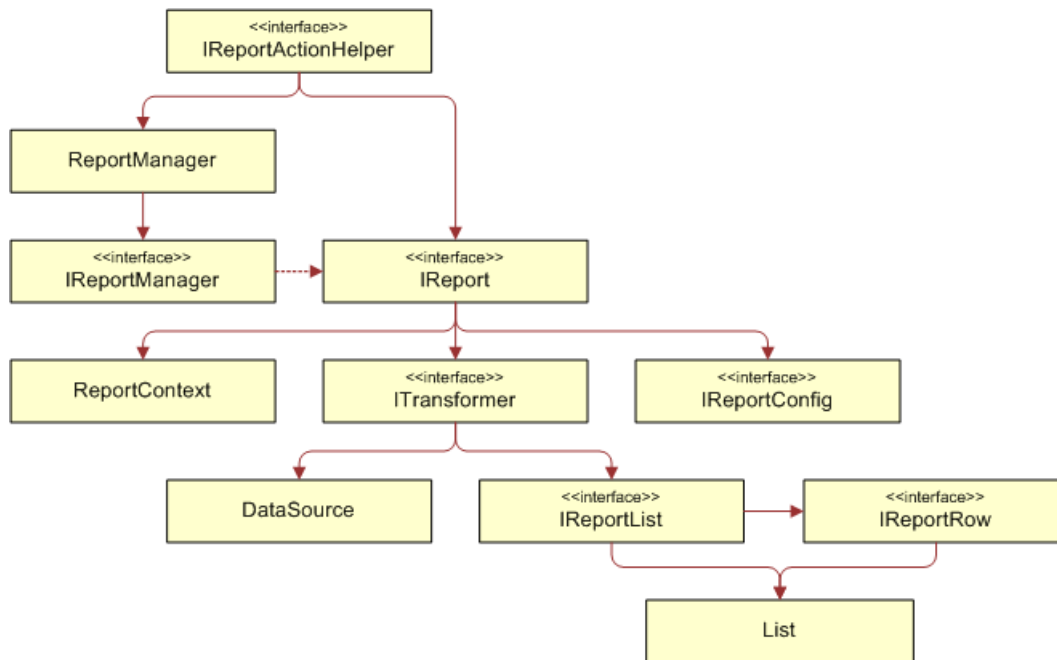


Figure 5. Reporting Engine Object Model

- **ReportActionHelper.** This class was designed to be called by the Servlet or Struts action class. It performs a number of tasks, such as parsing the request parameters, and then does the sorting, paging, and so on. It returns an **IReport** object, that you can use to render a report, or manipulate further before rendering it. Though it is possible to avoid using this class by using other APIs, it is strongly recommended that you use this class to reduce your customization work.
- **ReportManager.** Use this class to get an instance of **IReportManager**.
- **IReportManager.** This is the entry class to the Reporting Engine APIs. For example, to get an instance of **IReport** and other objects.
- **ReportContext.** This class is a Map, which allows the Reporting Engine client to pass information to the Reporting Engine. For example, the binding values to SQL “?” parameters, and the objects used in Velocity templates.
- **IReportConfig.** This interface represents the report XML definition. For example, the SQL used to query, instructions to bind the report context objects to the SQL, instructions to format the report. There are a set of Config objects related to this class that represent the report XML elements. For more information about this API, go to the Oracle Billing Insight Javadoc as described in [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#).
- **ITransformer.** This object represents the transformer defined in the report XML. It offers a set of APIs that manipulate the format, such as format value, write the template, and so on.

- **DataSource.** This API is not a public. It represents the datasource defined in the report XML, and allows you to retrieve report data from that data source.
- **IReportList and IReportRow.** The report data retrieved from DataSource is represented as IReportList, which is a java.util.List. IReportList includes a list of IReportRow objects, which represents rows in a report. The objects in IReportRow are basic Java objects, such as Integer, Double, String, Date, and so on. For more details, please check Java APIs of Reporting Engine.

The object model of Reporting Engine is straightforward. Figure 6 shows how the Reporting Engine objects interact with each other to generate a report.

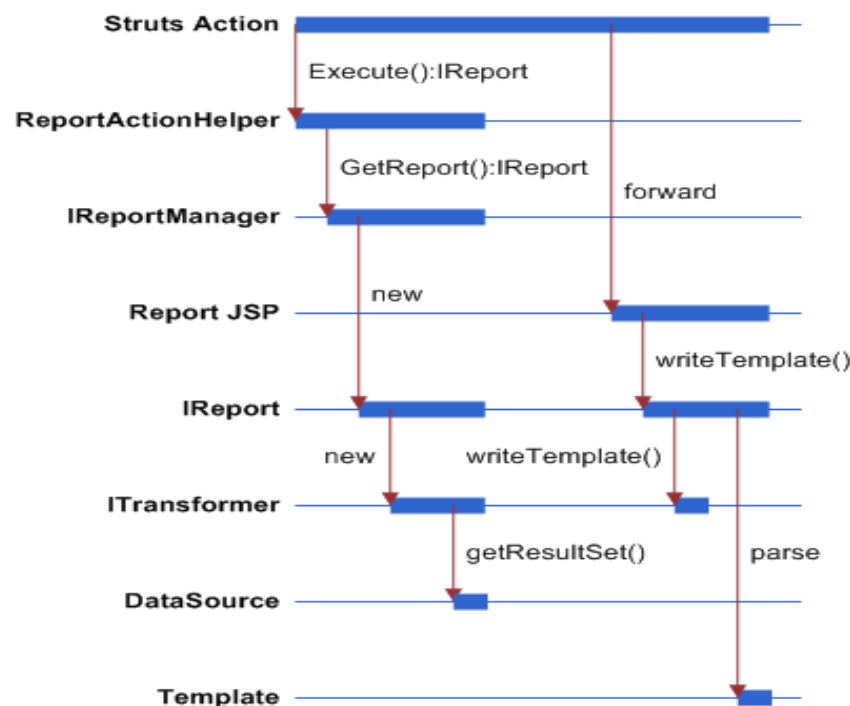


Figure 6. Reporting Engine Object Interaction

For more information on how to write action class and report JSP pages, see [“Customizing the Reporting Engine” on page 192](#).

Components Used by the Reporting Engine

The Reporting Engine uses the following components:

- Reporting XML
- Templates

■ Reporting API

Using the Report List Properties File

The report list properties file, `reportList.properties`, includes the list of report XML files to be loaded into the report engine. You must have your report XML file defined in this file. The file format is:

```
name=xml_file_path
```

In this definition, `name` must be unique for each report XML and the XML file must be either under `EDX_HOME` or on the class path.

In the following example, the `telco.xml` file is in the `EDX_HOME/config/rpt/` directory or on the class path. In the path, `EDX_HOME` is the directory where you installed Oracle Billing Insight:

```
telco_xml=config/rpt/telco.xml
```

Configuring Batch Reporting

You can specify the following parameters for batch reporting in Oracle Billing Insight:

- **batchReport.failTries.** This parameter specifies the number of times to retry the batch report job after a transaction timeout exception.
- **batchReport.processingTimeOut.** This parameter specifies the time out period after which a user can delete a batch report request with `PROCESSING` status.

For information about integrating Oracle Billing Insight with Oracle WebCenter or other content management systems, see *Integration Guide for Oracle Billing Insight*.

To configure batch reporting

- Update the appropriate parameters in the `globalConfig.properties` file, located in the `EDX_HOME/config/rpt` directory (or the `EDX_HOME\config\rpt` directory on Windows):

```
batchReport.failTries=3
    #The unit is hour.
batchReport.processingTimeOut=12
```

Related Topics

[Changing the Local Storage Location for the HTML Report Formatting Files on page 207](#)

Report XML

The report XML describes how to generate a report. The report XML includes the `dataSource` section and the `transformer` section. The `dataSource` section describes how to retrieve data from data source, and the `transformer` manipulates the data before sending it to the template.

Samples of report XML files are located in the *EDX_HOME/config/rpt* directory. To get a complete list of all the valid report XML elements and attributes, see the *report.xsd* file, under the *EDX_HOME/config/rpt* directory. In the path, *EDX_HOME* is the directory where you installed Oracle Billing Insight.

The following topics describe some of the main features of the Reporting Engine and explain how to use report XML to implement them.

Element <reports>

This is the root element of report XML. This element can include <report>, <localizer>, <prompts> and <templates> elements. The following XML shows that structure:

```
<reports>
  <templates>...</templates>
  <localizer>...</localizer>
  <prompts>...</prompts>
  <report>...</report>
</reports>
```

Element <localizer>

This element defines how the localization of the reports will be done. For details, see [“Internationalization and Localization of Reporting” on page 187](#) for more information.

The <localizer> element has the attributes described in [Table 21](#).

Table 21. Attributes for <localizer>

Name	Required	Description
enableMessageResources	No	This attribute allows you to use Struts MessageResource to look for the resource bundles for reports. This means you can use the same copy of resource bundle files defined in a struts config file without reloading another copy of it. The default is true.
defaultCode	No	<p>This attribute enables you to define the default behavior if a resource is not found.</p> <ul style="list-style-type: none"> ■ A value of 0 means to use the key as the default value. ■ A value of 1 means to use Struts notion of “<i>Text<locale>.<key>Text</i>” <p>A value of -1 means to throw an exception.</p>

Localizer can include <resourceBundle> as its child elements.

Element <resourceBundle>

This element specifies one resource bundle property file name to be used for report localization. For more information about localization, see [“Localizing the User Interface” on page 32](#).

For example:

```
<resourceBundle name="config/l10n/message" />
```

This example means the property file, `config/l10n/message_<locale>.properties`, located in the *EDX_HOME* directory (the directory where you installed Oracle Billing Insight) is used for localization.

Element <prompts>

The <prompts> element has the same format as the one defined under <dataSource>. However, because it is defined at the global level, it can be shared and referenced by other reports. This significantly reduces duplication of the report XML contents, and makes it easier to maintain report XML files. For more details, see the <prompts> definitions in [“Element <dataSource>” on page 152](#).

Element <templates>

This element allows you define a list of global templates that can be included and parsed into other templates. For example, the `paging.vm` is used to generate paging UI and could be included by other templates, like `report_body.vm`.

For example, to define a template:

```
<templates>
  <template id="paging.vm" name="template/common/reporting/paging.vm"/>
</templates>
```

This example means there is a template named `paging.vm`, located in the *EDX_HOME*/template/common/reporting/ directory.

Then you can include the `paging.vm` from another template like this:

```
#parse ($transformerConfig.getTemplateName("paging.vm"))
```

The method `transformerConfig.getTemplateName("paging.vm")` returns this template, `paging.vm`, from the *EDX_HOME*/template/common/reporting/paging.vm directory.

NOTE: If you have a template that has the same ID defined inside the transformer element, then the ID in transformer takes precedence over the what is in the global template list. This allows an individual transformer to use its own template.

Element <template>

This element defines a global template, which has following attributes described in [Table 22](#).

Table 22. Attributes for the <template> Element

Name	Required	Description
ID	Yes	The ID is a unique identifier among all the global templates. Note you can use the same ID for the transformer template ID. In this case, the transformer template takes precedent of the global one.
name	Yes	The name attribute is the full class path name of the template.

Element <report>

This element defines a report. A report can include zero or more <dataSource> elements, one or more <transformer>s, and zero or one of <customList>, <printList> and <downloadList>.

```
<report id="reportId" name="MyReport">
  <downloadList>...</downloadList>
  <printList>...</printList>
  <customList>...</customList>
  <dataSource>...</dataSource>
  <transformer>...</transformer>
</report>
```

The <report> element has two attributes:

- **id.** The ID identifies this report. All the reports defined in the report XML files in reportList.properties must have unique IDs. This ID must start with an alphabetic character, and can include numbers and underscores.
- **Name.** This is the name of the report. This name is used to search the report bundle to get a localized version of the report name. For example, in the Report List page, the names of reports are from this attribute.

Element <dataSource>

This element defines how to retrieve data from the data source.

```
<dataSource id="" uri="jdbcJNDI:edx.report.databasePool">
  <query dynamic="true">
    </query>
  <columns><column id="" type="" /></columns>
  <inputBindings><inputBinding /></inputBindings>
</dataSource>
```

The data retrieved from the data source is represented as a List of Lists of simple Java objects, such as Strings, Date/Time/Timestamp or Numbers. The data does not use a two-dimensional array because a List of Lists gives you the potential to increase its size if required, and Velocity does not support accessing array elements through the [] operator.

The <dataSource> element has following attributes:

- **id:** A unique ID identifies this data source in this report. You must define it even there is only one data source. It is not required that the ID be unique across all reports. This ID must start with an alphabetic character, and can include numbers and underscores.
- **uri:** A Universal Resource Identifier identifies the location of the data source. Oracle Billing Insight supports three data sources: SQL data source, object data source, and DSV data source. This example focuses on the SQL data source. For information about object data sources, see [“Object Data Source” on page 190](#) and for DSV data source, see [“DSV Data Source” on page 191](#).

For an SQL data source, there are three URIs:

- **jdbcJNDI:<dataSource_JNDI_NAME>.** The jdbcJNDI indicates that this is a JDBC data source identified by its JNDI name. For example, jdbcJNDI:edx.report.databasePool means there is a JNDI data source named edx.report.databasePool.
- **jdbcRef:<dataSource_REF_NAME>.** The jdbcRef indicates that this is a JDBC data source identified by its local reference name, either defined in the web.xml or ejb-jar.xml file. For example, you can have an entry similar to this in the web.xml file:

```
<resource-ref>
  <res-ref-name>jdbc/rptDataSource</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>
```

With this entry, you can use following URI: jdbcRef:jdbc/rptDataSource. You must also resolve this local reference through the weblogic.xml file or another vendor-specific XML file.

- **jdbcDirect:<jdbc_config_property_file_class_path>.** The jdbcDirect means that there is no connection pool and the Reporting Engine must make a direct JDBC connection to the Oracle Billing Insight database. You must specify the class path to the DB config file. For example, jdbcDirect:config/db/jdbcConfig.properties. For the format of the config file, look at the sample jdbcConfig.properties file coming with the product. Avoid using this URI if your application can access a connection pool.

This element can include <query>, <inputBindings>, <prompts> and <columns> elements.

Element <query>

The <query> element defines the query used to retrieve data from the data source. The <query> element applies to an SQL data source but not to an object data source.

```
<query dynamic="false" maxRows="1000"> <![CDATA[select name, amount from summary where user_id=? ]]>
</query>
```

The value for <query> is enclosed in a CDATA topic, which can include any SQL.

The question mark in the SQL means that a variable must be resolved (bound) before the SQL can be executed. Variables are resolved through the <inputBindings> element.

Table 23 describes the attributes of the <query> element.

Table 23. Attributes for <query> Element

Name	Required	Description
dynamic	No	This attribute indicates whether to parse this SQL as a Velocity template before execution. This allows you to use a Velocity template to generate a SQL dynamically. For information about how to write dynamically generated SQL, see “Dynamic SQL” on page 186 . The value can be True or False. The default value is False.
maxRows	No	This attribute indicates the Boolean number of rows will be retrieved from the data source. The value must be an integer. The default value is 1000.

Element <inputBindings>

This element defines a list of input bindings that are used to bind the SQL variables defined in the <query> element. It has no attribute, and includes an <inputBinding> element.

Element <inputBinding>

This element defines a single input binding. There are two kinds of bindings: objects and prompts. The order of the <inputBinding> elements is the same as the order of the SQL variables. That means the nth <inputBinding> is used to bind the nth SQL variable. Object binding means binding an object or its property to an SQL variable.

For example:

```
<inputBinding object="bean" property="userId" />
```

This means there is an object called bean in the report context, this object is a JavaBean, and it has a property named userId. The value returned by bean.getUserId() will be used to bind the SQL variable. Usually, the JavaBean is a Struts Action Bean object. If the object returned by the property is a Collection, then each element in the Collection will be used for binding.

```
<inputBinding object="myObject" />
```

In this case, there is no property defined, so myObject is not assumed to be a JavaBean. If the myObject is not a Collection, then myObject is used to bind to the SQL variable directly. If the myObject is a Collection, then each element in the myObject Collection will be used to bind to the SQL variables in its natural order in the collection. This latter case is very useful where the number of SQL variables is dynamic, such as a name in a (?...?) clause. For more information about using dynamic SQL, see [“Dynamic SQL” on page 186](#).

Prompt binding is a special case of object binding. Prompt binding means that the binding object is from the user prompt, which allows you to bind the value of the prompt to a SQL variable.

```
<inputBinding object="form" property="<bean_property>" prompt="<prompt_id>" />
```

You can use a map-backed Action Bean also. For example, the ReportForm from the Oracle Billing Insight application is a map-backed form. It has map-methods, such as `getParameter(String name)` and `setParameter(String name, Object value)`. You can use this syntax in a property or prompt attribute:

- `<inputBinding object="form" property="parameter(callType)" />`
- `<inputBinding object="form" property="parameter(callType)"
prompt="parameter(callType)" />`

Table 24 describes the attributes of the `<inputBinding>` element.

Table 24. Attributes for the inputBinding Element

Name	Required	Description
object	Yes	<p>The name of the object in the report context used for binding. This object must be put into report context.</p> <p>In the case of prompt binding, the Reporting Engine automatically retrieves the prompt value from the prompt form, and puts this object into the report context. The ReportActionHelper class puts the value of the prompt into the context with that name.</p> <p>In the non-prompt case, the caller of Report engine must put this object into context.</p>
property	No	<p>This attribute is optional. When it appears, it means the object is a JavaBean and the value of the property of this JavaBean is used to bind SQL variable.</p> <p>If this property is not there, then it means the object identified by object attribute is used for binding.</p> <p>NOTE: A map-backed property is supported, such as <code>parameter(callType)</code>.</p>
prompt	No	<p>This attribute indicates that this input binding is from a prompt, and the value of it must be the ID of the prompt defined in the <code><prompts></code> element.</p>

NOTE: The object name for the prompt form is fixed to form and you must use `object=form` for prompt.

Element <prompts>

This element defines an HTML form whose input is used for data source input bindings. Each input field in the form is called a prompt. You configure where the prompt gets its original data (from the Oracle Billing Insight database or from a fixed value list), and how it will be presented by the report XML. The Reporting Engine builds the report prompt (input) UI, which is fully customizable (it uses a template to generate the UI).

To control the look and feel of prompts, reporting uses a technique similar to tiles. Layout format is controlled by layout.vm, and prompt rendering is controlled by prompt.vm.

The <prompts> element has a list of prompt blocks. Each block is separated by that dark blue bar at the top, and you can define a label for each blue bar. Inside each block, you can define a list of groups, where each group has a list of prompts. Each prompt group acts like <tr> in an HTML table, and all prompts within a prompt group display horizontally in a row. Each prompt must belong to a group. Prompts can be HTML input or a plain label. In the preceding example UI, Data range is a group with two prompts: the start date and end date. Usage type is another group that has two prompts: usage type and call type.

The <prompts> definition used to generate the example UI is:

```
<prompts id="prompts1" formName="reportForm" action="report.action" method="post"
  templateID="layout.vm"><block><group label="Date Range: "><text id="fromDate"
  size="12" value="1/1/2004" imgSrc="_assets/images/calendar.gif" label="From: "
  labelPosition="top"/>
  <text id="toDate" size="12" value="12/1/2004"
  imgSrc="_assets/images/calendar.gif" label="To: " labelPosition="top"/>
</group><group>
<select id="parameter(usageType)" report="prompt_usageType"
displayColumnId="usage_type_name" valueColumnId="usage_type_key"
value="2" label="Usage Type: "/>
<select id="parameter(callType)" report="prompt_callType"
displayColumnId="call_type_name" valueColumnId="call_type_key" value="2"
label="Call Type: "/><image name="display" src="_assets/images/display.gif" />
</group><group label="Billing Reports"><select report="prompt_reportList"
value="first" name="reportId" onChange="cleanupHiddenValues()" /></group></block>
</prompts>
```

You can define <prompts> under <reports> and it will be global. To refer to a global <prompts> from inside <dataSource>, use the following:

```
<prompts id="billingPrompts"/>
```

This expression means that there is a global <prompts> whose ID is billingPrompts. If the same <prompts> is used across multiple data sources, then global <prompts> helps you to maintain only one copy of it.

The <prompts> element has the attributes described in [Table 25](#).

Table 25. Attributes for the <prompts> Element

Name	Required?	Description
id	Yes	A unique ID is used to identify this prompts list in this data source. Oracle Billing Insight supports one prompts element for each data source.
label	No	The label displayed for the entire set of prompts.
formName	No	The name of the HTML form and default reportForm. This attribute is only useful if you want to use JavaScript to manipulate the form.

Table 25. Attributes for the <prompts> Element

Name	Required?	Description
action	Yes	The action of the HTML form. Use report.action for the action because it is used as the default. If you change the action name defined in Struts config XML, then you must search all your JSP pages and Velocity templates to replace it.
method	No	The default is post.
templateID	Yes	The template ID specifies the layout template ID. The template must be either defined in corresponding transformer's <templates> or in the global <templates>.
enctype	No	The encryption type.
onReset	No	The name of JavaScript being called when Reset is called on the HTML form.
onSubmit	No	The name of JavaScript being called when Submit is called on the HTML form.

The <prompts> elements contain one or more <block> elements.

Element <block>

This is an optional element. If you do not define it, then you can define group directly under <prompts>, and all the groups will be put, implicitly, under a block. You can define a label for a block and the label will be displayed in the blue bar of the prompt.

Element <group>

This element defines a group of prompts. This group of prompts will be displayed horizontally in one line. Different groups of prompts will be displayed vertically.

The <group> element has the attributes described in [Table 26](#).

Table 26. Attributes for the <group> Element

Name	Required?	Description
label	No	The label displays at the beginning of the each prompt group.
description	No	The description displays for the rollover question mark.

There are eight types of prompts, which correspond to input types in an HTML form (except Label).

Some supported HTML forms are: text, check box, select, radio, image, submit, reset and label. Image, submit, reset, label are purely for HTML form rendering and manipulation. Their values are not used for report SQL input bindings. Check box, select, radio and text can be used for SQL input bindings.

Attributes for prompt related configuration in XML file, most of attributes are from an HTML form, others are required by the Report Engine.

The <group> element can include one of the following attributes: <checkBox>, <select>, <radio>, <text>, <image>, <label>, <submit> and <reset>.

Element <select>

This element defines a select prompt. A select prompt allows you to select one or more values from a list of values. A select prompt must associate with a report whose result set is used to populate the select list. For example:

```
<select id="parameter(Call Type)"
report="prompt_Call Type"
displayColumnId="Call_type_name"
valueColumnId="Call_type_key"
value="2"
label="Call Type: "/>
```

A select list requires two types of information: display values and actual values. The display values are for displaying, and the actual values are for querying. For example, you can display May 2010, but use an internal value 5 for a query. For example:

```
<select><option value="5">May 2010</option></select>
```

To render the preceding UI, get the options values and display names from the associated reports. [Table 27](#) describes the select options.

Table 27. Attributes for the <select> Element

Name	Required?	Description
id	Yes	<p>Identifies this prompt in this prompts list. The ID is used as the name of the input prompt in the HTML forms, which means that it determines which Action Bean property is used to hold this input value. In the example, the billPeriod property of Action Bean holds the value of the select box.</p> <p>If there is no corresponding property in the Action Bean (if it is a map-backed form), you can use the Parameter property (a map-backed property) to get the value into the ActionForm.</p> <p>The following example creates a prompt for call type, which is not a property of ActionForm:</p> <pre><inputBinding object="form" property="parameter(callType)" prompt="parameter(callType)"/></pre> <p>The prompt is declared as:</p> <pre><select id="parameter(callType)" label="Call Type:"></pre> <p>NOTE: When using <code>parameter(calltype)</code> as id (and therefore the HTML input file name), JavaScript might not recognize the name. In that case, you might want to extend your Action Beanimplementation to be a regular JavaBean property, which allows you to use <code><select id="callType" ></code>.</p>
label	No	The label of this prompt. Used for display.
labelPosition	No	<p>Display the label position against the prompt. Top, bottom, left, and right are supported:</p> <ul style="list-style-type: none"> ■ Top. The label is at the top of the prompt. ■ Bottom. The label is at the bottom of the prompt. ■ Left. The label is to the left of the prompt. ■ Right. The label is to the right of the prompt.
size	No	Size of the HTML input field.
report	Yes	The ID of the report, whose result set will be used to populate the Select element. The report can load data from the Oracle Billing Insight database or it can load from a DSV data source which is useful if the data in the list is fixed.

Table 27. Attributes for the <select> Element

Name	Required?	Description
displayColumnId	No	The column ID of the report, whose values will be used as the display names of the <option> fields of <select> list. The first column of the report is used when displayColumnId is not specified.
valueColumnId	No	The column ID of the report, whose values will be used as the values of the <option> fields of <select> list. The second column of the report is used when valueColumnId is not specified.
value	No	The default value for the <select> list. It can be: <ul style="list-style-type: none"> ■ first, using the first value in the valueColumnId column of the report ■ last, using the first value in the valueColumnId column of the report An integer N, such as 1 or 2, which indicates the nth value in valueColumnId column of the report. Note the index starts from 1.
multiple	No	Specifies that multiple items can be selected. The value can be True or False. The default value is False.
onBlur	No	Name of JavaScript being called for onBlur event.
onChange	No	Name of JavaScript being called for onChange event.
onClick	No	Name of JavaScript being called for onClick event.
onFocus	No	Name of JavaScript being called for onFocus event.

The report used to generate <prompt> must meet the following requirements:

- Have two columns: one column for display, and another for prompt value. The display column ID must match the displayColumnId attribute defined, and the value column ID must match the valueColumnId attributed defined. If the report only has only one column, then you can have both displayColumnId and valueColumnId point to the same column.
- The report ID of the prompt report must match the report attribute defined.
- You can format the prompt display names by using pattern attribute of column element of the report.

Element <checkBox>

The checkBox prompt allows you to print the prompt values in a list of check boxes. For example:

```
<checkBox id="billPeriod" label="Bill Period: "
report="prompt_billPeriod" onClick="alter('onClick') "
displayColumnId="bill_period_name"
valueColumnId="bill_period_key"
value="last"/>
```


In the example, the bill period prompt is defined as a set of check boxes, where you can check one or more bill periods. The display names and values of bill period come from the `prompt_billPeriod` report. The `<checkBox>` element has the same attributes as `<select>`, except `multiple` does not apply. For information about using the `<select>` element, see ["Element <select>" on page 158](#). You can think of the `checkBox` element as just another view presenting the same prompt, similar to a multiple-select list. The data retrieved from data source for the `<checkbox>` element must be either `true` or `false`.

Element <radio>

This prompt presents a list of radio buttons, only one of which can be selected.

```
<radio id="billPeriod" label="Bill period: "
report="prompt_billPeriod"
onClick="alert('onClick') "
value="last" />
```

In the example, the bill period prompt is defined as a set of radio buttons, where you can only check one of the bill periods. The display names and values for bill period come from the `prompt_billPeriod` report.

The `<radio>` has the same attributes as `<select>`, except `multiple` does not apply. See `<select>` for more information. In fact, you can just think radio as another view of presenting the same prompt. `<radio>` is like a single-select list.

The data retrieved from the data source used for `<radio>` must be either `true` or `false`, and only one can be `true`.

Element <text>

This element allows you to define a text box and use the user-entered value as the prompt value.

```
<prompt id="billPeriod" label="Bill period: ">
  <text report="prompt_billPeriod"
    maxLength="10"
    onBlur="alert('onBlur') "
    onChange="alert('onChange') "
    onFocus="alert('onFocus') "
    onSelect="alert('onSelect') "
    size="10"
    value="06/2004" />
</prompt>
```

In the text prompt, `size` attribute determines the width of the prompt.

Element <image>

This element allows you to define an image. For example this usage creates an image submit button:

```
<image name="display" src="_assets/images/display.gif" />
```

NOTE: The `<image>` element is different from the `` HTML tag.

Table 28 describes the attributes for the <image> element.

Table 28. Attributes for the <image> Element

Name	Required?	Description
name	Yes	The display name of the image.
scr	Yes	The image src.
align	No	Left or right.

Element <label>

This element defines text to display in the form. For example:

```
<label name="ccc_totl_lbl" value=" and " />
```

Table 29 describes the attributes for the <label> element.

Table 29. Attributes for the <label> Element

Name	Required?	Description
name	No	Not used.
Value	Yes	The text to be displayed as it is on the screen

Element <reset>

This element displays an HTML reset button. For example:

```
<reset name="reset" value="reset" />
```

Table 30 describes the attributes for the <reset> element.

Table 30. Attributes for the <reset> Element

Name	Required?	Description
name	Yes	Name of the reset button.
value	Yes	The display value of the reset button.
onClick	No	JavaScript to invoke.

Element <submit>

This element displays an HTML submit button. For example:

```
<submit name="submit" value="ok" />
```

Table 31 describes the attributes for the <submit> element.

Table 31. Attributes for the <submit> Element

Name	Required?	Description
name	Yes	Name of the submit button.
value	Yes	The display value of the submit button.
onClick	No	The JavaScript to invoke.

Element <columns>

This element, under <dataSource>, defines the list of columns retrieved from the data source. As described previously, the data retrieved from the data source is a two-dimensional matrix with rows and columns. For an SQL query, the rows are the rows from the SQL table, and the columns are the SQL table columns. Most of the transformer operations, such as sorting, grouping and calculation, are based on the types of the columns. Only the type of the column is important, not the definition of the column. For example, you can summarize if the type is Number. It does not matter if the definition is Air Fee or Toll Charge. That is the primary reason to use a List of Lists of objects to present all the data.

You must define all the columns retrieved from the data source in this element, in the same order as the data source. For example, if you are using a SQL data source, the order of selected columns from Select must be the same as the order defined in the XML element. The same is true for object data sources.

Element <column>

This element describes the column retrieved from the data source. You must define the type of the column in this element. The order of <column> elements must be the same as the order of columns retrieved from the data source and for each column in the data source, you must have one of this XML element defined for it.

The element <column> includes the attributes described in Table 32.

Table 32. Attributes for the <column> Element

Name	Required?	Description
ID	Yes	Uniquely identifies this column in the data source.
type	Yes	Type of column. The legal types are all simple Java object types. A column can be sorted if its type is java.lang.Comparable. or it can take a calculator operation (aggregation), if its type is java.lang.Number.
default	No	This attribute indicates the default value for this column if the value returned from data source is null.

Column types can be one of the following:

- **Java.lang.Object.** A generic type. Avoid using this if you want to sort or format on the column. Use a more specific type instead.
- **Java.lang.Double.** A double value, which can be sorted and aggregated.
- **Java.lang.Float.** A float value, which can be sorted and aggregated.
- **Java.lang.Integer.** An integer, which can be sorted and aggregated.
- **Java.lang.Long.** A Long value, which can be sorted and aggregated.
- **Java.lang.Short.** A Short value, which can be sorted and aggregated.
- **Java.lang.BigDecimal.** A BigDecimal, which can be sorted and aggregated.
- **Java.lang.String.** A String value, which can be sorted.
- **Java.sql.Date.** A Date value (a Date has no time information). It can be sorted.
- **Java.sql.Time.** A Time value (a Time has no date information). It can be sorted.
- **Java.sql.Timestamp.** A Timestamp value, which includes both date and time information. It can be sorted.
- **Java.lang.Boolean.** A Boolean value, which can be sorted.
- **Java.lang.Byte.** A Byte value, which can be sorted.

Attributes can be one of the following:

- **Number.** The default value is parsed as a Number string using the `parseXXX` method on the corresponding Java class. For example, use `Double.parseDouble()` if it is a double. It can only include digits and decimal point.
- **Timestamp.** You must supply the default value formatted as `yyyy-mm-dd hh:mm:ss`.
- **Date.** You must supply the default value formatted as `yyyy-mm-dd`.
- **Time.** You must supply the default value formatted as `hh:mm:ss`.
- **String.** The default value is used as it is.
- **Boolean.** The default value can be true or false.

Element <transformer>

This element defines a transformer for this report. A report can include zero or more transformers. Transformer is key element of the report engine. It is responsible for transforming the data retrieved from data source into a format suitable for presentation.

The <transformer> element has the attributes described in [Table 33](#).

Table 33. Attributes for the <transformer> Element

Name	Required?	Description
Id	Yes	Uniquely identifies this transformer in this report. Note, you are allowed to have two transformers with same ID if they are from different reports.
datasourceId	No	The ID of the data source where the transformer gets data. Note, a transformer is not required to have a data source. If a transformer does have a data source, then the Reporting Engine is usually used as a pure Template engine, and no meaningful data transformation is done inside transformer. That means that all the reporting functionality, such as sorting and paging, will not apply. For example, in the Telco.xml file, the transformer with report_header.vm defined has no data source.
pageSize	No	This attribute enables paging and defines the number of rows that will be displayed in one page. All the data will be presented in one page if this attribute is not specified.
viewType	No	This attribute sets the view type. The value can be portrait, landscape, or tree. The default value is portrait.
displayReport	No	This attribute determines whether the report portion of the UI is displayed. The values can be true or false. The default value is true.
displayPrompts	No	This attribute determines whether the prompts portion of the UI is displayed. The values can be true or false. The default value is true.
showTitle	No	This attribute determines whether the title portion of the UI is displayed. The values can be true or false. The default value is true.
showItems	No	This attribute determines whether the (Number of Items) portion of the UI is displayed. The values can be true or false. The default value is true.
showHeader	No	This attribute determines whether the header portion of the UI is displayed. The values can be true or false. The default value is true.
hiddenResultCount	No	This attribute determines whether the report table and chart are displayed. If the report result size is less than or equal to the value of the hiddenResultCount attribute, then the table and chart are hidden. The default value is -1.
divClass	No	This attribute sets the CSS class for <div class="contentbox">. The default value is empty, which uses contentbox.

Table 33. Attributes for the <transformer> Element

Name	Required?	Description
tableClass	No	This attribute sets the CSS class for <table="rpt">.
Background	No	This attribute sets the background colors. By default, the background shows different colors for even and odd rows. You can set this to blank.
chartPosition	No	This attribute sets the chart position relative to the report table. The value can be one of the following: top, bottom, left, right, topInside, bottomInside, leftInside, rightInside. The default value is top.
collectionId	No	For Web Services, the name of a list, or group, of objects in a report.

Element <columns>

This element defines a list of columns for the transformer. You are not required to define a column in the transformer for each column in the data source. It is not necessary that the order of columns in the transformer match the order of the columns in the data source. However, following those two rules will make your code easier to maintain.

This XML element has no attribute and contains <column> elements.

Element <column>

This XML element defines a column for the transformer. The transformer will render the columns in a table format. This is one of the most important XML element.

```
<column
  id="myColumnId"
  name="Column Name"
  hidden="false"
  sortable="true"
  defaultSort="true"
  caseSensitiveSort="true"
  pattern="MM/dd/yyyy"
  link="report.action?reportId=myReport&#x26;parameter(myColumnId)=$col "
  localize="true"/>
```

Table 34 lists all the attributes for the <column> XML element.

Table 34. Attributes for the <column> Element

Name	Required?	Description
id	Yes	This ID must match one of the IDs defined in the data source.
name	Yes	The name of the column. The name is localized and presented as the table column name.
hidden	No	A template string. The parsing result is either true or false.

Table 34. Attributes for the <column> Element

Name	Required?	Description
sortable	No	This attribute defines whether this column is sortable. If true, then the template generates a URL link for this column. The value can be True or false. The default is False.
defaultSort	No	This attribute defines whether to sort this column when the report generates. The value can be True or false. The default is false.
caseInsensitiveSort	No	This attributes defines whether you want a case-insensitive sort when the column type is java.lang.String. The value can be True or false. The default is false.
onlineOnly	No	This attribute defines whether this column shows on the Web page only, and not in the CSV download file. The value can be True or false. The default is false.
downloadOnly	No	This attribute defines whether this column shows only in CSV download file only, and not on the Web page. The value can be True or false. The default is false.

Element <link>

This element allows you to define a drilldown link, which can also be defined as an attribute of the <column> element. The benefit of using it as an attribute is that you can wrap the content in CDATA without escaping the special characters.

Element <templates>

This element includes a list of template elements. It has no attributes, and includes only one element, `template`.

Element <template>

This element defines one template used by the transformer. A transformer can define one or more templates and each template represents a presentation view. For example, you can define one template for HTML, one for XML and another for CSV. You specify which view or template to use to render the UI by passing the template ID through `Ireport.writeTemplate()`.

```
<templates>
<template
  id="HTML_TEMPLATE"
  name="template/common/reporting/report_body.vm"/>
</templates>
```

Table 35 describes the attributes for the <template> element.

Table 35. Attributes for the <template> Element

Attribute	Required	Description
id	Yes	The ID identifies this template inside this transformer. An ID must be unique to this transformer.
name	Yes	The class path of the template name. Because the class loader loads the template by default, this template must exist on the classpath (such as on the WEB-INF/classes directory or packaged into a JAR file). For example, if your template is located under the template/templ/my.vm directory and that is on the class path, then you must use the template/temp/my.vm directory as the name.
localize	No	True or false. True means this template is localized. There is one template for each locale, and the report engine finds the correct template based on the locale. For example, the email template has a lot of static text, therefore define one template for each locale and specify this attribute as true to associate the correct template for each locale.

For information about creating PDF templates, see [“Creating a PDF Template for Reporting” on page 178](#).

Element <groups>

This element allows you to group the data retrieved from a data source into groups, where each group is presented inside a table. For example, you might want to group on all types, so that all the local calls are presented in one table, and international calls are presented in another table. Only single column grouping is supported.

You can define multiple groups. You can define one of them as default grouping, so when the data is retrieved from the data source, it will be grouped by that default grouping. Call `Ittransformer.group()` in your calling program to switch to another group.

This element has no attributes, and can include the <group> element.

Element <group>

This XML element defines a single group. The <column> element defines the columns you want to group on. You can only define one column. For example:

```
<group id="group_by_type" default="true"><column id="type"/></group>
```


Table 36 describes the <group> element attributes.

Table 36. Attributes for the <group> Element

Name	Required	Description
id	Yes	Defines a unique ID that identifies this group in this transformer. The group ID must only be unique among the groups defined in this transformer.
default	Optional	This flag indicates that this group is the default one, so when data is retrieved from data source, the data will be grouped (only one group can be default). The data will not be grouped if there is no default group defined. The default is False.

Element <column>

This <column> element is defined as part of the <group> element, and identifies the column where grouping will happen. It has the attributes described in Table 37.

Table 37. Attributes for the <column> Element

Name	Required	Description
id	Yes	This is the column ID defined in data source. This ID must match the ID of the column of the data source where you want the grouping to happen.

Element <calculator>

This element defines a calculator for the report. The calculator can perform a set of operations, for example: summarize (subtotal), average, Boolean and minimal. The operations are grouped together into an operation group. calculator contains one or more <operationGroup> elements. For example:

```
<calculator>
  <operationGroup name="Total">
    <operation type="sum" columnName="Charges" />
    <operation type="sum" columnName="taxes" />
  </operationGroup>
  <operationGroup name="Average">
    <operation type="ave" columnName="Charges" />
    <operation type="ave" columnName="taxes" />
  </operationGroup>
</calculator>
```

For example, the Reporting Engine generates a table similar to the example in [Table 38](#).

Table 38. Example of Table Generated by the Reporting Engine

Invoice Number	Charges	Taxes
12345	10.01	0.23
23456	12.11	1.03
Total	22.12	1.26
Average	11.06	0.63

Element <operationGroup>

This element defines a group of operations. Different operations in the group must operate on different columns, but it is not required that groups have the same operation types. That is, you can mix sum with avg in the same operation group.

In general, do not define an operation on the first visible column of the table. That column will be used to display the name of the operationGroup. However, if it is necessary to define an operation on the first visible column, then you can change the report_body.vm by replacing the operationGroup name with the operation value you define.

The <operationGroup> element has one attribute, name, which is described in [Table 39](#).

Table 39. Attributes for the <operationGroup> Element

Name	Required	Description
name	Yes	The name of this group of operations. The default template, report_body.vm, presents the group in the first column of the operation row of the table.

This element can contain one or more <operation> elements.

Element <operation>

The <operation> element defines a single calculator operation on a single column. It has the attributes described in [Table 40](#).

Table 40. Attributes for the <operation> Element

Name	Required	Comments
type	Yes	<p>The type of operation:</p> <ul style="list-style-type: none"> ■ summary. Finds the summary of all the values of the column identified by columnId attribute. ■ avg. Finds the average of all the values of the column identified by columnId attribute. ■ max. Finds the Boolean value of all the values of the column identified by the columnId attribute. ■ min. Finds the minimal value of all the values of the column identified by the columnId attribute. <p>count. Finds the total number of rows. In this case, columnId is optional.</p>
columned	Yes	The ID of the column that the operation will apply to.

Element <charts>

This element allows you to define one or more charts for a single transformer. For example:

```
<charts><chart id="c1"
type="BAR_VERT_CLUST"
style="config/chart/vertical_bar_chart.properties"
chartTitle="global.title.accountBillingOverview"
xAxisTitle="global.label.accounts"
yAxisTitle="global.label.dollars">
<datasets>
<dataset><column id="Total"/></dataset>
</datasets>
<xlabel><column id="Billing_Account"/></xlabel>
</chart><chart id="c2"
type="PIE"
style="config/chart/pie_chart.properties"
chartTitle="global.title.plan">
<datasets>
<dataset><column id="total"/></dataset>
</datasets>
<xlabel><column id="rate_plan"/></xlabel>
<compress threshold="2" label="global.label.other" append="true"/>
</chart></charts>
```

Element <chart>

This element defines a single chart for this transformer. Oracle Billing Insight supports two chart types: Bar chart and Pie chart. The data of the chart must come from the columns of the data source.

The <chart> element includes the attributes described in [Table 41](#).

Table 41. Attributes for the <charts> Element

Name	Required	Description
id	Yes	Uniquely identifies this chart among all the charts defined in this transformer. Note, you can use the same chart IDs in different transformers.
type	Yes	The type of the chart. Oracle Billing Insight supports the following types of chart: <ul style="list-style-type: none"> ■ BAR_VERT_CLUSTER. Vertical bar chart. ■ BAR_HORIZ_CLUSTER. Horizontal bar chart. ■ BAR_VERT_STACK. Vertical stack bar chart. ■ BAR_HORIZ_STACK. Horizontal stack bar chart. ■ PIE. Pie chart. ■ LINE. Line chart.
style	Yes	Path to the name of the DVT chart properties file. For information on configuring the properties file for DVT charting, see "Customizing Charts" on page 197 .
chartTitle	No	Defines the title of the chart.
xAxisTitle	No	The title of the X-axis.
yAxisTitle	No	The title of the Y-axis.

The <chart> elements also include following two elements: <datasets> and <xlabel>.

Element <datasets>

This element allows you to define multiple data sets used to draw the chart. Only one dataset for each chart is supported.

Element <dataset>

This element defines a data set used for charting. A data set must come from the column of the data source. Currently, you can only define one column for on dataset. It has no attributes and contains one element: <column>.

Element <column>

This element defines the column whose values will be used as the data set for DVT charting. For example, for the BAR_VERT_CLUST chart, the dataset is used for the Y-axis values. For PIE, the dataset is used for the pie chart data.

The <column> element of Report XML includes one attribute, which is described in [Table 42](#).

Table 42. Attributes for the <column> Element

Name	Required	Description
id	Yes	The ID of the column where the chart will get its data. The type of the column must be a number.

Element <xlabel>

This element defines the values for the x-axis. The x-label must come from the data source column. It has no attributes, and contains one element: <column>. You can only define one column for each x-label.

Element <column>

This element defines the column used for the x-label. The values of the column are used for the x-axis values. This element only includes one attribute, which is described in [Table 43](#).

Table 43. Attributes for the <column> Element

Name	Required	Description
id	Yes	The ID of the column where the chart will get its x-axis values.

Element <downloadList>

This element defines a list of downloads available for this report. For example, you can define XML, CVS, and PDF downloads. For each download, the template generates a download link. You can define multiple downloads for one report. For example:

```
<downloadList name="Download"><download
  name="Download CSV"
  type="csv"
  description="CSV download"
  templateId="CSV_TEMPLATE" />
</downloadList>
```

The <downloadList> element has one attribute, which is described in [Table 44](#).

Table 44. Attributes for the <downloadList> Element

Name	Required	Description
name	No	The name of this downloadList. Depending on your template, you can use this name for different purposes. For example, you can build a list of downloads and use this name as the name of the list.

Element <download>

The <download> element defines one download for the report. It has the attributes described in [Table 45](#).

Table 45. Attributes for the <download> Element

Name	Required	Description
type	Yes	The type of the download. You can name any type you want. The type is used as the download file extension. For example, use csv for CSV download and use xml for XML download.
name	No	The name of the download, which depends on the template. It can be shown as a URL link or as a list item.
description	No	Description of the download. Currently, the description is not used by template, but you can modify the template to use it in a pop-up help window.
templateId	Yes	The template ID used to generate the download of the report. It is possible that the same template ID list appears in multiple transformers. If so, then the templates will be parsed and appended together in the order of the templates defined in XML.

Element <printList>

This element defines a list of print friendly links available for this report. Though it is possible, it is unlikely you will define more than one print friendly. For each print friendly, a print friendly link will be generated through the template.

For example:

```
<printList name="Print friendly"><print
name="Print friendly"
description="print friendly account details"
templateId="PRINT_TEMPLATE" />
</printList>
```

The <printlist> element has one attribute, which is described in [Table 46](#).

Table 46. Attributes for the <printList> Element

Name	Required	Description
name	No	The name of this printList. It is not used by current template

Element <print>

The <print> element defines one print-friendly view for the report. It has the attributes described in [Table 47](#).

Table 47. Attributes for the <print> Element

Name	Required	Description
name	No	The name of the print-friendly view. The default template renders it as a URL link.
description	No	Description of the print-friendly view. Currently this is not used, but you can modify the template to use it for a pop-up help window.
templateId	Yes	The template ID used to generate the print-friendly report. It is possible that the same template ID can appear in multiple transformers, so all these templates will be parsed and appended together, in the order of the templates defined in XML.

Element <customList>

This element defines a list of custom reports available for this report. Though possible, it is unlikely that you must define more than one custom report. For each custom report, a custom report link will be generated through the template.

For example:

```
<customList name="Customize"><custom
name="Customize"
description="Create a custom report for contract call details"
reportId="telco_cust_std_r4" />
</customList>
```

The <customList> element has one attribute, which is described in [Table 48](#).

Table 48. Attributes for the <customList>

Name	Required	Description
name	N	The name of this customList. It is not used by current template.

Element <custom>

The <custom> element defines one custom report for the current report. Each custom report must be itself defined as a report. This tag is used to build a link to that custom report. It has the attributes described in Table 49.

Table 49. Attributes for the <custom> Element

Name	Required	Description
name	No	The name of the custom report. The default template renders it as a URL link.
description	No	Description of the custom report. Currently it is not used by template but you can modify template to use it for a pop-up help window.
reportId	Yes	The report ID of the report used to define the custom report: the custom report itself is a report and you must define it as a report.

Using Report Templates

All the report UIs are generated through Velocity templates. For information about how the Velocity templates work, see

<http://velocity.apache.org/>

Oracle Billing Insight has changed some of the default Velocity templates. The most important one is that inside for each loop, the \$velocityCount variable starts from 0 instead of the default 1.

Oracle Billing Insight offers a set of example templates that generate useful UIs. These templates are very generic, are not tied to a particular application, and can be used as the base for your customization work.

The templates are all defined in the *EDX_HOME*/template/common directory. In the directory, *EDX_HOME* is the directory where you installed Oracle Billing Insight.

The lib subdirectory includes some Velocity MACRO library files and the reporting subdirectory includes report template files.

Table 50 explains the libraries that are included with the report package.

Table 50. Libraries Included with the Reporting Package

Name	Description
Lib/report_library.vm	This file defines some common MACROs used by the Reporting Engine. You must use it as it is.

Table 51 explains the templates that are included with the report package.

Table 51. Templates Included with the Reporting Package

Name	Description
Common/report_header.vm	This is the header part of the report. Note, this is not the header of the tiles. The tile header is usually the Navigation Tabs. The report header usually includes the report name and the download, print friendly, and custom report links.
Common/report_body.vm	This template renders the table associated with the transformer. Because a report can define multiple transformers, the template can be parsed multiple times for a report.
Common/paging.vm	This template renders the paging navigation part, which has previous, forward buttons for a user to page through the report.
Common/layout.vm	This template is used to define the layout of the prompts of the report.
Common/promt.vm	This template renders each individual prompt of the report.
Common/promt.vm	This template renders each individual prompt of the report.
Common/csv.vm	This template renders the CSV format of a report. The current CSV format does not consider the case of how to escape the special characters like a comma. You must write code to handle that case.
Common/print.vm	This template renders the print friendly format of a report.
Common/custom_report.vm	The template is used for custom reporting. It displays the custom report detail and allows you to type a name for the report to save into the Oracle Billing Insight database.
common/batch_report.vm	The template is used for batch reporting. It displays the batch report detail and lets you type a name for the report to save into the Oracle Billing Insight database.
common/xml.vm	This template renders the XML format of a report.
common/rsxml.vm	This template renders the XML format for the REST Web service call for a report.
pdf/PrintSummary.rtf	This template renders PDF format for the Print Summary quick link on pages displayed to consumer users.
pdf/StatementSummary.rtf	This template renders PDF format for the Statement Summary report.
pdf/telco_std_r1.rtf	This template renders PDF format of the Account Billing Overview report.

Table 51. Templates Included with the Reporting Package

Name	Description
pdf/telco_std_r6.rtf	This template renders the PDF format of Total Cost by Plan report.
pdf/telco_std_r13.rtf	This template renders the PDF format of Service Details report.

Creating a PDF Template for Reporting

Oracle Billing Insight provides preconfigured PDF templates for reporting (listed in [Table 51 on page 177](#)). You can also define additional templates to present other reports in PDF.

To create a PDF template for reporting

- 1 Download and install Oracle Business Intelligence(BI) Publisher at <http://www.oracle.com/technetwork/middleware/bi-publisher/downloads/index.html>
- 2 Download the existing or new report in XML format and use this file to create a new PDF template (an RTF) file. See the Template Builder for Word Tutorial in Oracle BI Publisher for assistance creating the RTF template. Training is also available on the Oracle Technology Network.
- 3 Using the toolbar in Oracle BI Publisher, load the report XML file.
- 4 Use Microsoft Word formatting and Oracle BI Publisher insert functions to add fields, tables, charts, and conditional formatting to the template appropriate for your organization. Save the file as an RTF.
- 5 After you create the template file *filename.rtf*, copy the file to *EDX_HOME/template/pdf/filename_//_CC.rtf* from the *EDX_SYS_LANG* table in the OLTP schema . In the template file, *filename* is the name of the report template. In the directory where you copy the template file, *//_CC* is your default system language code.
- 6 If the application supports multiple languages, then you can translate your template and name it using the different language code. For example, if you support Spanish (*es_ES*) as well as American English (*en_US*), then create a template called *filename_en_US.rtf* and create a translated template called *filename_es_ES.rtf*.
- 7 In *EDX_HOME/config/rpt/*.xml* , add the following line in *<downloadlist>* tag for the new report:

```
<download name="global.dropdown.pdf" type="pdf" description="PDF download"
templatel d="PDF_TEMPLATE"/>
```
- 8 Add the following line in the *<templates>* tag for the report. In the line, *//_CC* is the default language.

```
<template id="PDF_TEMPLATE" downloadable="true" name="template/pdf/
filename_//_CC.rtf"/>
```

NOTE: You do not have to specify additional languages in the XML. Oracle Billing Insight finds the template that corresponds with the selected language.

Predefined Context Variables

When you call the `IReportActionHelper.execute()` method to generate reports, the Reporting Engine puts a list of predefined context variables into the report context, which are then available to the Velocity template.

Table 52 lists some of the variables that you can use. If the overwrite flag is Y, then you can pass a variable with the same name through `ReportContext` to overwrite the default values set by `ReportActionHelper`.

Table 52. Predefined Context Variables

Name	Type	Over write?	Description
form	Action Form	No	This is the struts Action Bean object currently being processed.
gifDir	String	Yes	The directory where the image files used by report are saved, for example, the paging arrow images. The default is <code>_assets/images</code> .
link	String	No	This is the URL link base of this page and it is equivalent to the html <code><base></code> tag. The default value is: <code>HttpServletRequest.getContextPath() + HttpServletRequest.getServletPath()</code> The URL is similar to <code>http://host:port/<web-root>/report.action</code>
user	IUser of UMF	Yes	The current user logged in. The <code>IUser</code> object is passed as a session variable, <code>USER_PROFILE</code> . If the object is not in the session, then Oracle Billing Insight does not put it in the context. The user variable is for query purposes only and its absence does not affect the functionality of the reporting. For example, for some reason, you might not use UMF <code>IUser</code> and you can use your own user object. NOTE: Some templates, like <code>report_header.vm</code> , might expect <code>IUser</code> to get the user name and if you do not supply <code>IUser</code> , then the template might not display the user name properly.

Table 52. Predefined Context Variables

Name	Type	Over write?	Description
contact Profile	IContactProfile of UMF	Yes	The contactProfile is a profile of IUser named as contact_profile. Oracle Billing Insight uses it to retrieve the user's first name and last name and is currently only used in report_header.vm. The absence of this information does not affect the function of Reporting Engine.
locale	String	Yes	The default value set by ReportActionHelper is the from http session: session.getAttribute("org.apache.struts.action.LOCALE"). NOTE: This locale is put into session by the Struts framework.
reportId	String	No	The report ID of current report.
transformer	ITransformer	No	You can use transformer object to do work such as formatting data. Never call ITransformer.writeTemplate() in the template.
reportConfig	IReportConfig	No	Represents the report configuration.
dataSourceConfig	IDataSourceConfig	No	Represents the data source configuration.
dataSource ColumnConfigs	A list of IDataSource ColumnConfig	No	Represents the list of data source column configurations.
transformerConfigs	ITransformerConfig	No	Represents the transformer configuration.
transformer ColumnConfigs	A list of ITransformer ColumnConfig	No	Represents the list of transformer column configurations.
operationGroup Configs	A list of IOperation GroupConfig	No	Represents the list of operation groups defined inside calculator for the transformer.
chartConfigs	A list of IChartConfig	No	Represents the list of chart configurations for the transformer.
templateConfigs	A list of ITemplateConfig	No	Represents the list of template configurations for the transformer.

Table 52. Predefined Context Variables

Name	Type	Over write?	Description
rowlist	IReportList	No	Represents the original data retrieved from the data source. The data could be sorted and so the order could be changed. Though you cannot overwrite this variable, you can certainly change the content of the list.
groupSet	Set	No	To support grouping, the transformer maintains a Map of List objects. In the case of no grouping, there is only one entry in the map, the key is the report name, and the value is the List returned from data source. In the case of grouping, the original list from the data source is regrouped into multiple lists. Each list has the same group value, and the group value becomes the map key. This variable is looped through in report_body.vm to build the HTML table.
dataMap	Map	No	This is the map of group keys to the List as described previously.
reportContext	ReportContext	No	The ReportContext object used to generate reports. NOTE: Note: You cannot overwrite reportContext, but you can change the content.
URLEncoder	URLEncoder	No	This is a wrapper class around java.net.URLEncoder, because Velocity cannot invoke a static method directly through class name, and java.net.URLEncoder does not have a constructor. Use this class to encode the parameter values you passed through the URL.

Integration with Struts and Tiles

The Reporting Engine can be used with any presentation framework. However, because Oracle Billing Insight is based on Struts and Tiles, the Reporting Engine has special extensions to help it integrate with Struts and Tiles. This topic describes that integration.

Struts Action Class

The Struts action class performs the following processing:

```

ReportContext ctx = new ReportContext()
ctx.put(...) //put whatever your stuff used in template
IReportActionHelper helper = ReportManager.getReportActionHelper()
IReport report = helper.execute(ctx, form, request, response); //IReport will be in
session
return mapping.findForward("page.reports.report");

```

The Struts action class creates a ReportContext object which you can put your own objects into. These objects can then be used in report templates. Then it calls IReportActionHelper.execute() method to get an IReport object. If this is the first time to access the report, then a new IReport object will be created. If this is a sorting or paging operation, then the IReport object cached in the session will be returned. In case a new IReport object is necessary, the report data will be retrieved from the DataSource defined in the report XML of this reportId.

Next it calls IReportActionHelper.execute() method to get an IReport object. If this is the first time accessing the report, then a new IReport object will be created. If this is a sorting or paging operation, then the IReport object cached in the session will be returned. In case a new IReport object is necessary, the report data will be retrieved from the DataSource defined in the report XML of this reportId.

For the last action of this class, control is forwarded to the tile, page.reports.report, which is defined in the tiles definition file.

Tiles Definition

Tiles are defined in the tiles-defs-*ModuleName*.xml file, such as tiles-defs-reporting.xml, in the WAR file of the EAR file.

```

<definition name="page.reports.report" extends="simpleLayout_1">
  <put-attribute name="pageName" value="Billing Report"/>
  <put-attribute name="leftBar" value="/_includes/sidebar_left_analytics.jsp"/>
  <put-attribute name="pageDesc" value=""/>
  <put-attribute name="header" value="/_includes/header_analytics.jsp"/>
  <put-attribute name="footer" value="/_includes/footer_relative.jsp"/>
  <put-attribute name="subtab" value="/_includes/subtab_billing.jsp"/>
  <put-attribute name="body" value="/reporting/report.jsp"/>
</definition>

```

The key to this tile is that the body tile is report.jsp, which generates the main body of reporting UI.

Report.jsp

The report.jsp page is used to render the view. In fact, there is almost no HTML code in this page. Instead, this page just invokes the Velocity template engine to parse the templates:

```

IReport report = (IReport)request.getSession().getAttribute(reportId);
IReport.writeTemplate(jspWriter, templateId);
//template is the one defined in report xml and default to "HTML_TEMPLATE"

```

The Reporting Engine goes through the Transformers defined in the report XML for this reportId and for each transformer, parsing the template whose ID matches templateId. Note a transformer will be ignored if it has no template with a matching templateId defined in the transformer configuration of the report XML.

The matching templates will be parsed in the same order as defined in the report XML, and the results will be written back into JSPWriter sequentially.

Reporting API

The reporting API offers an interface to interact with the Reporting Engine. These APIs manage common reporting features, such as sorting, grouping and paging. They also offer report clients the flexibility to customize reporting.

The reporting API is not tied to a particular presentation framework. You can use struts and tiles or servlets and JSP to access it. However, you could find that using struts and tiles is the easiest way to implement your own reporting UI, because that is the default presentation framework used for the reporting UI of Oracle Billing Insight.

The core reporting APIs are: ReportContext, IReportManager, IReport, ITransformer, IReportConfig and ReportActionHelper. For more information about reporting APIs, go to the Oracle Billing Insight Javadoc as described in [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#).

ReportContext is the carrier of information between the reporting caller and the Reporting Engine. ReportManager is a factory that gets an instance of IReportManager. IReportManager is the factory for IReport objects. IReport represents a report defined in XML. ITransformer represents the transformer defined inside a report in XML. IReportConfig represents the configuration information in XML.

The following example shows how to generate a report:

```
ReportContext context = new ReportContext();
context.put("form", StrutsActionBean);
IReportManager rptmgr = ReportManager.getInstance();
IReport rpt = rptmgr.getReport("reportId", context);
Rpt.writeTemplate("templateId", Writer);
```

In the example, a Struts Action Bean is put into the reportContext, which means this object is available to the Velocity template. You can use the syntax of \$form.name in the Velocity template. Assume there is a name property in the form.

After you get an instance of IReportManager, call its getReport method to get a report. The report must match the one defined in the report XML. It will return an object that represents the report defined in the XML with the same reportId.

After you get an instance of IReport, it calls the writeTemplate() method to parse the Velocity template identified by templateId in the report XML, and writes the content into a Writer output. This method loops through all the transformers in the report and calls transformer.writeTemplate(). If the same template IDs appear in different transformers, then multiple templates can be parsed and the content of the parsed templates will be appended together in the order in which they appear in the report configuration XML.

You can also call the individual APIs of `ITransformer` to do sorting, grouping, or paging. However, it is tedious to call these APIs: they are usually used for back-end based applications. For the common UI features, such as sorting, grouping, and paging, the reporting API offers a Web helper class, `ReportActionHelper`, to shield you from the low-level APIs. This class is a facade to the Report Engine APIs. In most cases, your struts action must call this helper class instead of calling the lower-level reporting APIs. However, you can always access the report APIs directly if you want to. The action used by the product, `Com.edocs.app.reporting.actions.ReportAction`, calls this helper class. You can similarly do this in your action class.

Core Reporting Features

This topic describes some of the most important features of Reporting Engine, and how to use them in your application.

Sorting Feature of the Reporting Engine

Sorting is a built-in feature of the report engine. It is available when you use the `ReportActionHelper` class from your action class. With the reporting XML and template, enabling sorting is like configuring a transformer's column. For example:

```
<column sortable="true" .../>
```

Only single column sorts are supported. The sorting is done in-memory, to eliminate accesses to the data source.

Set the column attribute `sortable` to `true`. The Reporting Engine reads the configuration, instructs the template to generate a sort-able link for the corresponding table column name, and the `ReportActionHelper` class calls the `ITransformer.sort()` API.

When a column is defined as sort-able, the `report_body.vm` template renders the column of the HTML table with a URL link. For example:

```
<a
href="$link?sortColumn=$x&reportId=$reportId&transformerId=$transformerConfig.id&currentSortColumn=$currentSortColumn&ascending=$ascending&currentGroup=$groupId"
>
```


Table 53 describes the parameters in the URL.

Table 53. Sorting Parameters in the URL

Parameter	Description
\$link	The URL context base, which is set to <code>http://host:port/<web-context>/report.action</code> in the ReportActionHelper class. In the class, <web-context> is the Web context you defined in your EAR file.
SortColumn=\$x	This is the column index of the column being sorted in the transformer configuration.
Reported=\$reported	The report ID of the report.
TransformerId=\$transformerConfig.id	The ID of the transformer currently being sorted
CurrentSortColumn=\$currentSortColumn	This is the current column being sorted in this transformer.
Ascending=\$ascending	Defines the sort order, true or false.
CurrentGroup=\$groupIndex	Not used but can be used for grouping.

The Web component must process the URL request, and calls the `ITransformer.sort()` method to sort the column. The Helper class, ReportActionHelper does this work for you.

Just call the ReportActionHelper in your struts action. It processes this request and calls `ITransformer.sort()` to sort the column, then reorders the newly sorted report for you.

Paging Feature of the Reporting Engine

Paging is a built-in feature of the Reporting Engine. Use the ReportActionHelper class and the default templates (or templates based on the defaults) to access that function. The main paging template is `paging.vm`, which is included in `report_body.vm`.

Paging is enabled when

- You specify `pageSize` for transformer in report XML.
- The following is set: `<transformer <pageSize="20" />`

Because reports are loaded and cached in the user session, paging is done on cached data. This method of paging does not scale when there are a large number of rows of data. For that case, you must limit the number of rows retrieved using the `maxRows` attribute of the `<query>` element.

Dynamic SQL

Some situations require you to generate SQL dynamically. For example, you might have a report that searches the call details. One of the criteria is the call date. You want to search for call date equals a particular date, or you want to search for call dates between a start date and end date. Because the where clause is different for these two search cases, without dynamically generated SQL, you would be forced to write two reports with two SQL clauses. Dynamically generated SQL can solve this problem. The where clause of the SQL statement can be generated based on the current operation (equal or between), so only one report is required.

The Reporting Engine allows you to write an SQL query in a Velocity template, so that the SQL query will be parsed before it is executed. You must set the dynamic attribute of <query> to true. For example:

```
<query dynamic="true"> <![CDATA[
  select * from my_table where #if ($equal) date = ? #else date >= ? and date <= ?
#end]]></query>
```

The variable \$equal is set by the caller through the IreportActionCallback interface. It is true if the user selects the date equal operation, and false if the user chooses the date between operations.

NOTE: The number of question marks is different based on operation types: one for equal and two for between. To solve this problem, the report engine supports binding a Collection object to question marks. The report engine loops through the Collection and binds each element to question marks.

The following example shows how to bind:

```
<input binding object="form" property="parameter(dateList)" />
```

The method form.getParameter(dateList) returns a list of Date objects, and each date in the list is bound to the question marks in the query. The caller of Reporting Engine is responsible for collecting the list of dates and passing them to ActionForm.setParameter(dateList, dateList). (This assumes that Action Bean is as map-backed form, and has a pair of setParmeter and getParameter methods).

Another common use case is to generate the in operation in a where clause. The number of question marks is based on the size of a Collection object.

For example, if you have a list of categories saved in a List, and want to generate a where clause, then use

```
Where category in (?, ?, . . . , ?, ?)
```

In this clause, the number of question marks is the size of the List.

When doing the input binding, there is only one List, but loops through the List to set the question marks in the SQL as appropriate. This ensures that the number of question marks match the number of variables passed in.

There is a macro to help you generate the number of question marks based on the collection size:

```
#macro getSQLVariableIgnoreNull ($list $columnName)
```

The macro generates the list.size() number of question marks. For example:

```
select * from my_table where date in getSQLVariableIgnoreNull($dateList "date")
```

If the dateList size is 2, and it is Oracle database, then the result is:

```
select * from my_table where date in (NVL(?, date), NVL(?, date))
```

In this clause, NVL means ignore this question mark if the value is null.

Internationalization and Localization of Reporting

Resource bundles are used to support internationalization. This topic discusses internationalization for Velocity templates.

Because the Reporting Engine uses Velocity templates, you cannot take advantage of the JSP <message> tag or the Struts internationalization framework. Instead, the Reporting Engine has its own internationalization mechanism specially designed for Velocity templates, which has following features:

- Allows a user to specify any resource bundle, just like Struts config does.
- Allows a user to format a string as Y does, for example, My name is {0}.
- Provides a seamless integration with Struts if it is used. For example, sharing the same resource bundle.
- Offers a better way to handle default messages than Struts. In Struts, a resource that is not found is either returned as null or as:

```
Text<local e><resource_key>Text
```

With the Reporting Engine, you can configure it to return the key itself when the value of the key is not found.

For internationalization of reporting, you translate the following text:

- Regular text on the report user interface.
- Some text coming from the data source.
- Chart, title and amount format, and so on.
- Date format, number format, and so on.

Resource Bundle Definition

The resource bundle files used by the Reporting Engine templates are defined in the report XML files under the <reports> tag. The following example comes with Oracle Billing Insight, and is defined in the telco_global.xml file.

```
<localizer enableMessageResources="true" defaultCode="1">
  <resourceBundle name="com/edocs/app/reporting/resources/ApplicationResources" />
</localizer>
```

Follow these naming guidelines for localizing resource bundles:

- You must use "/" instead of "." in the name of the resource bundle, which differs from Struts message resource.

- The `<localizer>` tag defines how text will be localized. You can define multiple `<resourceBundle>` tags. Each `resourceBundle` tag defines a resource bundle file, and its name is defined by name attribute.

When the Reporting Engine searches for the resource bundle, it first checks whether this bundle exists as a file under `EDX_HOME` (the directory where you installed Oracle Billing Insight), or the current directory if `EDX_HOME` is not defined. If the file does not exist, then the Reporting Engine searches for the bundle as a class.

The attribute `enableMessageResources` enables you to use Struts `MessageResource` to search for a resource.

The attribute `defaultCode` enables you to define the default behavior if a resource is not found. 0 means to use the key as the default value. A value of 1 means to use Struts notion of *"Text<locale>.<key>Text"* and -1 means throw an exception. The default value for the attribute `defaultCode` is 0.

The search order for finding a resource is:

- 1 If the attribute `enableMessageResources` is true, and the Struts `MessageResource` does exist (it might not exist for a non-Struts application), then search the resource from Struts `MessageResource`, and return the resource if it is found.
- 2 For each resource bundle defined in `resourceBundle`, load the bundle as either file or class, and then search the resource in the order it appears, return if found.
- 3 If nothing is found, then use `defaultCode` described previously.

Follow these guidelines when defining a resource bundle:

- If you check the resource bundle name in the struts configuration file, then you will notice that the same file, `com/edocs/app/reporting/resources/ApplicationResources`, is defined in both the Struts and report XML files. The only difference in the definitions is the file separators. Reporting uses a back slash (`/`) and Struts uses a period (`.`). The same file is in two locations in order to support batch reporting. A batch job is not a Web application, so it does not have access to Struts `MessageResource`. This is also true if you are using the Reporting Engine at the EAR level. For example, you can generate an email message from an MDB event handler or from an EJB. However, if you are using Struts, and you using the Reporting Engine for online applications only (not batch reporting), then do not define a `resourceBundle`, because the online Web application can always find resources from `MessageResource`.
- Because the same resource is defined twice, both Struts and the Reporting Engine load the same resource bundle and cache them (twice). Usually, this is not a problem, because a resource bundle file is small. However, if you do want to reduce memory usage, then you can put all the template related resources into one file. Or, you can be more selective by putting only the batch report, email, and AR-related resources into one file, and load it by using the `resourceBundle` tag in report XML.
- It is recommended that you define the resource bundle as a flat file under the `EDX_HOME` directory, which lets you modify the file and reload it using this URL, without restarting the server:

`http://localhost:7001/selfservice/reporting/reloadReportConfig.jsp`

- If you want to use a struts message source, which is loaded from the classpath, then you can disable it during the development stage by setting `enableMessageResource` to false and loading a resource bundle from file system.
- Set the `defaultCode` to 1 to find all the text not being internationalized properly. You might want to set it to 0 for demonstration purposes.

Localization of Report Text

The localization of text in report is done through the `#localize` macro, which is defined in `reporting_library.vm`. It is defined as:

```
#macro (localize $name)
```

For example, in your template, you can call this macro as follows. This expression searches the report bundle to find a key with a value that matches name:

```
#localize("name")
```

All the texts defined in the `report.xml` file are treated as resource bundle keys. For example, report names and column labels. In the report template files, all the texts are localized through the `#localize` macro.

Localization of Report Data from a Data Source

By default, the text data retrieved from data source is not localized. You must turn on this option. In this case, the text data from data source will be used as keys to search reporting resource bundles.

The localization of data from data source is done through the `localize` attribute of transformer column configuration in report XML.

```
<column id="call type" localize="true" />
```

The column data retrieved from the Oracle Billing Insight database will be localized.

Localization of Charts

The chart components (chart title, labels and data) are localized by the `ITransformer.writeChart()` method. The chart title is searched as a regular resource bundle name. Label and data are localized if the `localize` attribute is set to true for the corresponding columns.

Locale

To support internationalization, you must pass the `Locale` object to `ReportContext` by calling `setLocale()`. If `ReportContext` does not have a locale defined, then when you call the `IReportActionHelper.execute()` method, it puts the Struts locale object in session.

Dynamic Localization

Velocity is used to support localization. Velocity acts similar to the way `java.text.MessageFormat` does, and achieves the same result. The Reporting Engine parses the resource value as a Velocity template, whose resource key ends with `.vm`, and returns the parsed value. For example,

```
rpt.test.vm=My name is $name.
```

Object name must come from the report context. This feature can make any text in your report dynamic. For example, on the account detail page, to display the report title as Account detail for `<account_number>` instead of the default text, define the report title as a `.vm` resource bundle. In the following example, `accountNumber` is from the Struts ActionForm:

```
rpt.accountDetail.title=Account detail for $form.accountNumber
```

Object Data Source

Because you might not have access to the Oracle Billing Insight database, the Reporting Engine provides an API to get back a list of Objects, which can be presented in a table with paging or sorting. The Reporting Engine offers an Object data source to provide that feature.

The object data source is defined as:

```
<dataSource id="ds1" uri="object:reportList">
  <columns>
    <column id="id" type="java.lang.String"/>
    <column id="name" type="java.lang.String"/>
  </columns></dataSource>
```

This example states that there is an object called `reportList` in `ReportContext`, and you must put that object into `ReportContext` before calling `IReportActionHelper`. This object can either be a `List` (`java.util.List`), `List` of objects, or a `List` of `JavaBean` Objects. If the object is a `List` of `List` of objects, then it is assumed that the objects in the inner list are basic Java objects, such as `String` or `Integer`. The objects must also match the types defined in the `dataSource` column.

Usually, the object is a `List` of `JavaBean` objects. For example, as shown in the example XML, `reportList` is a `List` of `IReportConfig` objects. (For more information about APIs, see the Oracle Billing Insight Javadoc as described in ["Accessing Oracle Billing Insight Data Dictionary and Javadoc" on page 25](#).) The Reporting Engine uses reflection to get the property values of the `JavaBeans`, whose property names match the column IDs defined in the example XML, and converts this `List` of `JavaBeans` into a `List` of `Lists` of `JavaBean` property objects (more precisely, into a `IReportList` of `IReportRow` objects). It is also assumed that the `JavaBean` properties are basic Java types. In the example, for each `IReportConfig` object in the list, the report engine calls `IReportConfig.getId()` and `IReportConfig.getName()`, and converts the `List` of `IReportConfig` objects into an `IReportList` object. Each element in `IReportList` is an `IReportRow` object. Each `IReportRow` includes two elements, the report IDs and the report names.

Then define the rest of the report XML, including transformers, as usual.

The object data source enables the Reporting Engine to connect to other data sources currently not directly supported by the Reporting Engine. For example, you might have a CORBA interface that retrieves financial data from a legacy system. You can still use the report engine to present the data, as long as you can convert the data into a `List` of `Lists` of objects.

DSV Data Source

This feature allows you to read a delimiter separated string as a data source. The URI format of this data source is as follows:

```
"dsv:inline:,:|"
```

In the data source, dsv stands for Delimiter Separated Values. Inline means that the data can only be embedded in the report XML (support is not available for reading data from a file). The comma is the column separator, and the pipe-delimiter (|) is a line separator, as shown in the following example:

```
<dataSource id="ds" uri="dsv:inline:,:|">
<query><![CDATA[0,Business|1,Personal]]></query>
<columns>
<column id="value" type="java.lang.Integer"/>
<column id="name" type="java.lang.String"/>
</columns></dataSource>
```

The data source will be transferred into an IReportList with two IReportRows. The first row has values of 0 and Business and the second row has values of 1 and Personal. You can use this data source to implement the split-billing feature. For example, you can generate a list for call details and allow the user to change a call from personal to business or conversely.

Drilldown and Breadcrumb Link

The Reporting Engine allows you to build a breadcrumb link while you are drilling down from report to report.

To build drilldown link, define a <link> for a transformer column:

```
<report id="testrpt0">
<transformer id="tr1" dataSourceId="ds1">
<column id="invoice_number" name="Invoice number" >
<link title="Drill down to the invoice detail."><![CDATA[
report.action?reportId=testrpt1c&invoiceNumber=$row.get(1)&parameter(parentNode)
=root]]></link></column> </transformer>
</report>
```

The <link> element instructs the Reporting Engine to build a drilldown link for each account number. You must construct the link, which must point to another report. The link will be parsed as a Velocity template.

This link also has a title attribute, which allows you add an HTML title to the link. In most browsers, the title will be displayed as popup help.

When you click an account number, you will drilldown to testrpt1 report. However, by default, there is no breadcrumb link built to allow you to go back to the testrpt0 report. To enable the breadcrumb link, add enableDrillUp=true to the column definition:

```
<report id="testrpt0">
<transformer id="tr1" dataSourceId="ds1">
<column id="invoice_number" name="Invoice number" "enableDrillUp"=true >
```

```
link title="Drill down to the invoice detail."><![CDATA[  
report.action?reportId=testrpt1c&invoiceNumber=$row.get(1)&parameter(parentNode)=root ]]></link>  
</column></transformer></report>
```

When this flag is set to true, and you drilldown from testrpt0 to testrpt1, there will be a breadcrumb link in the testrpt1 view which allows you to go back to the testrpt0 report.

Currently, you must drill down from one report ID to another report ID, but the breadcrumb link will not work if you try to drilldown to the same report. This feature makes sense when you are viewing the same report but drilldown through hierarchy.

Customizing the Reporting Engine

This topic describes how to customize the Reporting Engine. The examples use Struts and Tiles for the presentation framework, but the same techniques can be used for any other Web presentation framework.

You might want to customize the Reporting Engine to add the following features:

- Write your own Report XML
- Modify report templates
- Extending Reporting Engine through Reporting API

Write Your Own Report XML

The first step in creating your own report is to create your own report XML. Each report XML is project-specific. The best way to start is to use existing report as a base for your modifications

CAUTION: The Reporting Engine has a DTD, but is not used to validate the report XML. Therefore, make sure you do not miss required attributes or XML elements.

You can create one report XML, which includes all the reports for your project, or you can create one XML file for each report. Remember to register all your report XML files in the reportList.properties file, and to give each XML file a unique name.

After creating your own report XML, you can test it through the default template. Name your report ID with a prefix of telco_std, which will cause it to be loaded into the standard billing report list of Oracle Billing Insight.

CAUTION: Make sure that each report has a unique name across all the reports in all report XMLs, or else a latter one will overwrite the previous one.

Customize the Report Template

After you have created a report XML and familiarize yourself with how the report engine renders the report, you might want to customize the report template to generate the look and feel of your project.

A set of templates are provided with the report product. To customize them, make a copy of each template, put it into a new template directory, and change your report XML to point to the new directory.

You can add new objects into the report context (and thereby, the Velocity context) using the `IReportActionCallback` interface. But do not to overwrite the existing context variables. One technique is to use a special prefix (a underscore character, for example) for your custom context variables.

The CSS for the reporting HTML is defined in a file called `skin.css`. You can modify this file to change the CSS of the report UI.

Write Your Own Action Classes and ReportBean

Write your own Action class and action bean for your reports. Use the `ReportActionHelper` class to take care of common issues such as sorting and paging.

When writing your own action class, you must call the `ReportActionHelper.execute()` method. See [“Integration with Struts and Tiles” on page 181](#) for details about how to invoke this method.

When defining your own Struts ActionForm, you can make the form map-based, which allows you to pass any parameter into the Reporting Engine without explicitly adding a set of get and set methods. The only downside to this method is that a map-based property cannot be passed into JavaScript for client side validation.

For example, you can define two map methods: `public Object getParameter(String name)` and `void setParameter(String name, Object value)`. To use these parameters in an HTML form or URL, use a notion similar to the following:

```
"parameter(contractNumber)=123456"
```

This expression passes the contract number to struts, which calls `setParameter()` on your Action Bean to put the `contractNumber` into the map. This parameter can either be used as an SQL data source input binding or used in template.

To retrieve the parameter as an inputBinding, use:

```
<inputBinding object="form" property="parameter(contractNumber)" />
```

To retrieve the parameter from the template, use:

```
$form.getParameter("contractNumber").
```

Packaging

You can package your Struts action classes as usual at the WAR level. For struts beans, if you are not using batch reporting, then you can package them at the WAR level, but if you do use batch reporting, then the beans must be accessible by non-Web components such as the Common Center batch report job. In that case, you must package your report forms at the EAR level. For example, make them part of the `reporting-ext.1.2.1.jar` file.

You must register your report XML files in the `reportList.properties` file, and put the report XML files in the `EDX_HOME/config/rpt` directory. In the directory, `EDX_HOME` is the directory where you installed Oracle Billing Insight.

However, it is possible to put the report XML files under the other subdirectories of `EDX_HOME`.

Hiding Report Columns and Manipulating IReport

After you call `IReportActionHelper` and get back an `IReport` object, you can manipulate the object before forwarding it to `report.jsp`.

For example, to hide some columns based on certain conditions, get the `IReportConfig` object from `IReport`, find the `ITransformerColumnConfig` of the corresponding columns, and set the `Hidden` attribute based on your conditions.

Unlimited Paging

By default, the Reporting Engine retrieves 1000 rows from the data source. You can configure the number of rows the Reporting Engine retrieves (`maxRows` or `fetchSize`) in the report XML file.

The following sample code shows how to configure the size in the sample `report.xml` file:

```
<transformer id="tr1" ...><paging fetchSize="2000"/></transformer>
```

To retrieve all the rows from the data source without impeding performance, you can use unlimited paging. *Unlimited paging* enables the Reporting Engine to get the result set in batches and allows users to page across multiple batches. A *fetch* is one batch.

Unlimited paging retrieves result set rows in multiple fetches on demand when the user requests them. The user pages through the result set like regular paging. If the requested page is not in the current fetch, then the Reporting Engine gets the next fetch from the data source. However, all the intricacies of checking whether the requested page is in the current fetch, if not getting next the fetch, are hidden from the user.

Fetch size is the number of result set rows in one fetch. You can configure fetch size and page size in report XML. The following sample XML demonstrates how you can enable unlimited paging and to define the fetch size.

```
<transformer id="tr1" pageSize="20" ...><paging unlimited="true" fetchSize="5000"/></transformer>
```

The Reporting Engine supports unlimited paging for SQL data source and object data source. If unlimited paging is enabled, then sorting and calculator are not supported because it is necessary to sort and apply calculator operations for all the result set across all the fetches rather than current fetch.

Unlimited Paging for SQL Data Source

For the SQL data source you define the query as usual. The Reporting Engine embeds this query with `select count(*)` to get the size of the total result set.

Unlimited Paging for Object Data Source

For the object data source to get the result set in batches, the data source provides the Reporting Engine with a call back method which retrieves the data from start position to end position. For this purpose, the Reporting Engine expects an object which implements the call back interface in report context rather than result set object. That means, for regular paging, you put result set list or array of objects in the report context and for unlimited paging, you put an object which implements call back interface. This call back interface is called `IReportObjectResultSet` and has the following methods:

```
public Object getResultSet (ReportObjectSearchCriteria objectSearchCriteria);
public int getResultSetSize();
```

`ReportObjectSearchCriteria` object has the start position and end position of a fetch.

The object you put in the result set must implement `getResultSet` and `getResultSetSize()`.

The `getResultSet(ReportObjectSearchCriteria objectSearchCriteria)` method returns result set rows from start position to end position defined in `objectSearchCriteria`.

The `getResultSetSize()` method returns size of the complete result set that data source returns. If you do not know the result set size, then you can return `IReportObjectResultSet.unknownResultSetSize`.

Reloading Report XML and Templates without Restarting the Server

If you change the report XML, then you can use the following URL to reload. In the URL, `localhost:7001` is the name and port number for your local host:

`http://localhost:7001/selfservice/reporting/reloadReportConfig.jsp`

When you change the Velocity templates, the Velocity engine loads the templates automatically. However, because of browser caching issues, you must restart the server or clean up the browser cache.

If you put the resource bundle files under the `EDX_HOME` directory and load them through `<localizer>`, then the resource bundle can also be reloaded with the preceding URL.

The URL does not work in a clustered environment because it only refreshes the cache in one JVM.

Customizing Threshold Values for Batch Reporting

Oracle Billing Insight sets batch threshold values for Account Billing Overview and the Service Billing Overview reports by default. To implement this feature for other reports, you must specify the report threshold value, which determines the number of result set lines above which a report must process in batch mode instead of as an online download. Each report type uses this threshold value as follows:

- **CSV.** The report threshold value is the maximum number of output lines

■ **PDF.** A percentage of the batch threshold value (the default is 10%)

■ **XML.** A percentage of the batch threshold value (the default is 20%)

For example, if the CSV report threshold is set to 3,000 result set lines, then a PDF threshold value set at 10% must process in batch mode when it has 300 or more result set lines

Table 54 shows the report XML file name and the report ID, found in the XML file, that you use to set batch report thresholds.

Table 54. Report XML File Names and IDs

Report Name in UI	Report XML File Name	Report ID in XML
Account Billing Overview	telco_billing_account.xml	telco_std_r1
Account Billing Trend	telco_billing_account.xml	telco_std_r9
Statement Billing Overview	telco_billing_account.xml	telco_std_r5
Service Billing Overview	telco_billing_contract.xml	telco_std_r3
Service Billing Trend	telco_billing_contract.xml	telco_std_r11
Service Details	telco_billing_contract.xml	telco_std_r13
Total Cost by Plan	telco_billing_contract.xml	telco_std_r6
Find Calls	telco_billing_contract.xml	telco_find_call
Highest Spending Services	telco_topX.xml	telco_topX_r13
Highest Spending Services by Service Agreement	telco_topX.xml	telco_HighestSpendingServicesBySA_topX
Most Expensive Calls	telco_topX.xml	telco_topX_r7
Longest Calls	telco_topX.xml	telco_topX_r12
Most Frequently Called Numbers	telco_topX.xml	telco_topX_r10
Most Frequently Called Numbers by Service Agreement	telco_topX.xml	telco_FreqNumberBySA_topX
Most Frequently Called Numbers by Service Agreement Detail	telco_topX.xml	telco_FreqNumberBySADetail_topX
Most Frequently Called Destinations	telco_topX.xml	telco_topX_r15
Most Frequently Called Destinations by Service Agreement	telco_topX.xml	telco_FreqDestBySA_topX
Most Frequently Called Destinations by Service Agreement Detail	telco_topX.xml	telco_FreqDestBySADetail_topX
Most Frequently Called Countries	telco_topX.xml	telco_topX_r16

Table 54. Report XML File Names and IDs

Report Name in UI	Report XML File Name	Report ID in XML
Most Frequently Called Countries by Service Agreement	telco_topX.xml	telco_FreqCountryBySA_topX
Most Frequently Called Countries by Service Agreement Detail	telco_topX.xml	telco_FreqCountryBySADetail_topX

To specify a batch reporting threshold for a report

- 1 Add the following code before the <dataSource> tag in the report XML file, specifying the threshold value and the XML report ID:

```
<batchCriteria threshold="10" reportId="batchCriteria_accounts">
  <query></query><inputBindings name="input_bindings"></inputBindings>
</batchCriteria>
```

- 2 You can also set the stopThreshold attribute. When you set the stopThreshold attribute, if the testing report result meets the stopThreshold value, then the report engine withholds the report and displays the following message:

The scope of the report you have requested is too large. Reduce the number of periods, change your filter criteria, or change your hierarchy position.

Customizing Charts

Some Oracle Billing Insight reports use Data Visualization Tools (DVT) for charts. You can customize the format of DVT charts, specifying size, fonts, color, and so on for each chart type. The properties for each report are stored in individual property files. The properties in each file apply to all report charts of that type.

You can also create additional property files with alternate formats to associate with particular reports. You can associate only one property file with a chart type at any time, however. For details about which reports can be presented as charts, see *Self-Service Application Guide for Oracle Billing Insight*. For details on which properties are configurable for each type of chart available, see [“Configurable Chart Properties” on page 198](#).

To customize the DVT charts

- 1 To customize the formatting properties for a particular DVT chart, edit the property file associated with the particular chart type. The following chart property files are in *EDX_HOME/confi g/chart* directory:

DVT Chart Type	Property File
Vertical bar chart	vertical_bar_chart.properties
Horizontal bar chart	horiz_bar_chart.properties
Vertical stack bar chart	vertical_stack_bar_chart.properties
Horizontal stack bar chart	horiz_stack_bar_chart.properties
Pie chart	pie_chart.properties
Line chart	line_chart.properties

- 2 You can create an alternate property file for a chart type and associate it with the chart type (replacing the default property file for the chart). Edit the chart section of XML in the *EDX_HOME/confi g/rpt/fi lename.xml* file. In the file name, *filename* is the name of the report. Replace the name of the properties file with the new one in the style statement as shown in the following table. Specify only one property file for each chart type at a time.

DVT Chart Type	Chart Type as Indicated in the Report XML	Corresponding XML Style Statement Where You Specify the Chart Property File Name
Vertical bar chart	type="BAR_VERT_CLUST"	styl e="confi g/chart/verti cal _bar_chart. properti es"
Horizontal bar chart	type="BAR_HORI Z_CLUST"	styl e="confi g/chart/hori z_bar_chart. properti es"
Vertical stack bar chart	type="BAR_VERT_STACK"	styl e="confi g/chart/verti cal _stack_bar_chart. properti es"
Horizontal stack bar chart	type="BAR_HORI Z_STACK"	styl e="confi g/chart/hori z_stack_bar_chart. properti es"
Pie chart	type="PI E"	styl e="confi g/chart/pi e_chart. properti es"
Line chart	type="LI NE"	styl e="confi g/chart/li ne_chart. properti es"

Configurable Chart Properties

You can configure many properties of each Oracle DVT chart type available with Oracle Billing Insight.

For instructions on updating the properties for a chart, see ["Customizing Charts" on page 197](#).

Bar Chart and Stack Bar Chart Properties

Table 55 shows the configurable properties for the following types of charts:

- Vertical bar chart
- Horizontal bar chart
- Vertical stack bar chart
- Horizontal stack bar chart

Table 55. Configurable Properties Horizontal and Vertical Stack Bar Charts

Property Description	Property Name	Values or Units
Chart width	width	Pixels
Dashboard width	dashboardWidth	Pixels
Chart height	height	Pixels
Dashboard height	dashboardHeight	Pixels
Three-dimensional effect	3D	True or False
Gradient effect	gradient	True or False
Background color	backgroundColor	Transparent or a hex color code
Legend display location	legend	Auto, Top, Bottom, Left, Right, or None
Legend background color	legendBGColor	Transparent or a hex color code
Legend border color	legendBorderColor	Transparent or a hex color code
Legend font color	legendFontColor	Hex color code
Legend font type	legendFont	Style name
Legend font size	legendFontSize	Number
Legend dashboard font size	dashboardLegendFontSize	Number
Bold on legend	legendBold	True or False
Italic on legend	legendItalic	True or False
Underline on legend	legendUnderline	True or False
Legend alignment	legendAlignment	Right, Left, or Center
Series colors	seriesColors	String of hex color codes separated by a comma
Color by group	colorByGroup	True or False

Table 55. Configurable Properties Horizontal and Vertical Stack Bar Charts

Property Description	Property Name	Values or Units
Color of plot background	plotBGColor	Transparent or a hex color code
Color of plot border	plotBorderColor	Transparent or a hex color code
Alignment of data labels	dataLabels	Above, Center, or None
Number of digits to display to the right of the decimal point	dataDecimalDigit	Whole number
Maximum bar width	maxBarWidth	Pixels
Color of title font	titleFontColor	Hex color code
Font style of title	titleFont	Style name
Font size of title	titleFontSize	Points
Font size of dashboard title	dashboardTitleFontSize	Points
Bold on title	titleBold	True or False
Italics on title	titleItalic	True or False
Underline on title	titleUnderline	True or False
Alignment of title	titleAlignment	Left, Right, or Center
Color of X axis title	xAxisTitleColor	Hex color code
Font style on X axis title	xAxisTitleFont	Style name
Font size of X axis title	xAxisTitleFontSize	Points
Font size of X axis dashboard title	dashboardXAxisTitleFontSize	Points
Bold on X axis title	xAxisTitleBold	True or False
Italics on X axis title	xAxisTitleItalic	True or False
Underline on X axis title	xAxisTitleUnderline	True or False
Alignment of X axis title	xAxisTitleAlignment	Left, Right, or Center
Color of X axis label	xAxisLabelColor	Hex color code
Font on X axis label	xAxisLabelFont	Style name
Font size of X axis label	xAxisLabelFontSize	Points
Font size of X axis dashboard label	dashboardXAxisLabelFontSize	Points
Bold on X axis label	xAxisLabelBold	True or False
Italics on X axis label	xAxisLabelItalic	True or False
Underline on X axis label	xAxisLabelUnderline	True or False

Table 55. Configurable Properties Horizontal and Vertical Stack Bar Charts

Property Description	Property Name	Values or Units
Alignment of X axis label	xAxisLabelAlignment	Left, Right, or Center
Color of X axis line	xAxisLineColor	Hex color code
Display X axis grid	xGrid=false	True or False
Color of X axis grid	xGridColor	Hex color code
Color of Y axis title	y1AxisTitleColor	Hex color code
Font of Y axis title	y1AxisTitleFont	Style name
Font size of Y axis title	y1AxisTitleFontSize	Number of points
Font size of Y axis dashboard title	dashboardY1AxisTitleFontSize	Number of points
Bold on Y axis title	y1AxisTitleBold	True or False
Italics on Y axis title	y1AxisTitleItalic	True or False
Underline on Y axis title	y1AxisTitleUnderline	True or False
Alignment of Y axis title	y1AxisTitleAlignment	Left, Right, or Center
Color of Y axis label	y1AxisLabelColor	Hex color code
Font on Y axis label	y1AxisLabelFont	Style name
Font size of Y axis label	y1AxisLabelFontSize	Points
Font size of Y axis dashboard label	dashboardY1AxisLabelFontSize	Points
Bold on Y axis label	y1AxisLabelBold	True or False
Italics on Y axis label	y1AxisLabelItalic	True or False
Underline on Y axis label	y1AxisLabelUnderline	True or False
Alignment of Y axis label	y1AxisLabelAlignment	Left, Right, or Center
Color of Y axis line	y1AxisLineColor	Hex color code
Show Y axis Grid	y1Grid	True or False
Color of Y axis grid	y1GridColor	Hex color code
Number of markers per row	nMarkersPerRow	Whole number

Pie Chart Properties

Table 56 shows the configurable properties for pie charts.

Table 56. Configurable Properties for Pie Charts

Property Description	Property Name	Values or Units
Three-dimensional effect	3D	True or False
Gradient effect	gradient	True or False
Width of chart	width	Pixels
Height of chart	height	Pixels
Width of dashboard	dashboardWidth	Pixels
Height of dashboard	dashboardHeight	Pixels
Background color	backgroundColor	Transparent or a hex color code
Legend display location	legend	Auto, Top, Bottom, Left, Right, or None
Legend background color	legendBGColor	Transparent or a hex color code
Legend border color	legendBorderColor	Transparent or a hex color code
Legend font color	legendFontColor	Hex color code
Legend font type	legendFont	Style name
Legend font size	legendFontSize	Number
Legend dashboard font size	dashboardLegendFontSize	Number
Bold on legend	legendBold	True or False
Italic on legend	legendItalic	True or False
Underline on legend	legendUnderline	True or False
Legend alignment	legendAlignment	Left, Right, or Center
Number of markers per row	nMarkersPerRow	Whole number
Series colors	seriesColors	String of hex color codes separated by a comma
Color by group	colorByGroup	True or False
Color of plot background	plotBGColor	Transparent or a hex color code
Color of plot border	plotBorderColor	Transparent or a hex color code

Table 56. Configurable Properties for Pie Charts

Property Description	Property Name	Values or Units
Alignment of data labels	dataLabels	Above, Center, or None
Number of digits to display to the right of the decimal point	dataDecimalDigit	Whole number
Color of title font	titleFontColor	Hex color code
Font style of title	titleFont	Style name
Font size of title	titleFontSize	Points
Font size of dashboard title	dashboardTitleFontSize	Points
Bold on title	titleBold	True or False
Italics on title	titleItalic	True or False
Underline on title	titleUnderline	True or False
Alignment of title	titleAlignment	Left, Right, or Center
Type of pie slice label	sliceLabelType	<ul style="list-style-type: none"> ■ Percent. The percentage value. ■ Series. The name of the series. ■ Value. The value of each slice. ■ Series_percent. The series name and percentage of the slice.
Position of pie slice label	sliceLabelPosition	Inside, Outside_without_feeler, Outside_with_feeler, or None.
Color of pie slice label	sliceLabelColor	Hex color code
Font style of pie slice label	sliceLabelFont	Style name
Font size of pie slice	sliceLabelFontSize	Number
Bold on pie slice label	sliceLabelBold	True or False
Italics on pie slice label	sliceLabelItalic	True or False
Underline on pie slice label	sliceLabelUnderline	True or False

Line Chart Properties

Table 57 shows the configurable properties for line charts.

Table 57. Configurable Properties for Line Charts

Property Description	Property Name	Values or Units
Width of chart	width	Pixels
Width of dashboard	dashboardWidth	Pixels
Height of chart	height	Pixels
Height of dashboard	dashboardHeight	Pixels
Three-dimensional effect	3D	True or False
Gradient effect	gradient	True or False
Background color	backgroundColor	Transparent or a hex color code
Legend display location	legend	Auto, Top, Bottom, Left, Right, or None
Legend background color	legendBGColor	Transparent or a hex color code
Legend border color	legendBorderColor	Transparent or a hex color code
Legend font color	legendFontColor	Hex color code
Legend font type	legendFont	Style name
Legend font size	legendFontSize	Number
Legend dashboard font size	dashboardLegendFontSize	Number
Bold on legend	legendBold	True or False
Italic on legend	legendItalic	True or False
Underline on legend	legendUnderline	True or False
Legend alignment	legendAlignment	Left, Right, or Center
Series colors	seriesColors	String of hex color codes separated by a comma
Color by group	colorByGroup	True or False
Color of plot background	plotBGColor	Transparent or a hex color code
Color of plot border	plotBorderColor	Transparent or a hex color code
Alignment of data labels	dataLabels	Above, Center, or None

Table 57. Configurable Properties for Line Charts

Property Description	Property Name	Values or Units
Number of digits to display to the right of the decimal point	dataDecimalDigit	Whole number
Color of title font	titleFontColor	Hex color code
Font style of title	titleFont	Style name
Font size of title	titleFontSize	Points
Font size of dashboard title	dashboardTitleFontSize	Points
Bold on title	titleBold	True or False
Italics on title	titleItalic	True or False
Underline on title	titleUnderline	True or False
Alignment of title	titleAlignment	Left, Right, or Center
Color of X axis title	xAxisTitleColor	Hex color code
Font style on X axis title	xAxisTitleFont	Style name
Font size of X axis title	xAxisTitleFontSize	Points
Font size of X axis dashboard title	dashboardXAxisTitleFontSize	Points
Bold on X axis title	xAxisTitleBold	True or False
Italics on X axis title	xAxisTitleItalic	True or False
Underline on X axis title	xAxisTitleUnderline	True or False
Alignment of X axis title	xAxisTitleAlignment	Left, Right, or Center
Color of X axis label	xAxisLabelColor	Hex color code
Font on X axis label	xAxisLabelFont	Style name
Font size of X axis label	xAxisLabelFontSize	Points
Font size of X axis dashboard label	dashboardXAxisLabelFontSize	Points
Bold on X axis label	xAxisLabelBold	True or False
Italics on X axis label	xAxisLabelItalic	True or False
Underline on X axis label	xAxisLabelUnderline	True or False
Alignment of X axis label	xAxisLabelAlignment	Left, Right, or Center
Color of X axis line	xAxisLineColor	Hex color code
Display X axis grid	xGrid=false	True or False
Color of X axis grid	xGridColor	Hex color code
Color of Y axis title	y1AxisTitleColor	Hex color code

Table 57. Configurable Properties for Line Charts

Property Description	Property Name	Values or Units
Font of Y axis title	y1AxisTitleFont	Style name
Font size of Y axis title	y1AxisTitleFontSize	Number of points
Font size of Y axis dashboard title	dashboardY1AxisTitleFontSize	Number of points
Bold on Y axis title	y1AxisTitleBold	True or False
Italics on Y axis title	y1AxisTitleItalic	True or False
Underline on Y axis title	y1AxisTitleUnderline	True or False
Alignment of Y axis title	y1AxisTitleAlignment	Left, Right, or Center
Color of Y axis label	y1AxisLabelColor	Hex color code
Font on Y axis label	y1AxisLabelFont	Style name
Font size of Y axis label	y1AxisLabelFontSize	Points
Font size of Y axis dashboard label	dashboardY1AxisLabelFontSize	Points
Bold on Y axis label	y1AxisLabelBold	True or False
Italics on Y axis label	y1AxisLabelItalic	True or False
Underline on Y axis label	y1AxisLabelUnderline	True or False
Alignment of Y axis label	y1AxisLabelAlignment	Left, Right, or Center
Color of Y axis line	y1AxisLineColor	Hex color code
Show Y axis Grid	y1Grid	True or False
Color of Y axis grid	y1GridColor	Hex color code

Customizing the Statement Summary Chart

You can customize the vertical bar chart shown on the Statement Summary page in the Self-Service application. The properties for this chart are maintained in the `statement.properties` file. For details about the configurable properties for the Statement Summary chart, see [“Bar Chart and Stack Bar Chart Properties” on page 199](#).

NOTE: The dashboard height and width properties are not used with the Statement Summary chart.

To customize the Statement Summary vertical bar chart

- Edit the `statement.properties` file, located in the `EDX_HOME/conf/g/chart` directory (use back slashes (\) on Windows).

Changing the Local Storage Location for the HTML Report Formatting Files

Oracle Billing Insight requires that the HTML report formatting files be stored locally. You can change the local file save location for the HTML formatting files. The default storage location is the `EDX_HOME/output/reportapp` directory. If your applications are deployed in a distributed environment, the file storage path you specify must be a shared folder.

Oracle Billing Insight is preconfigured with the HTML formatting files. Oracle Billing Insight automatically generates report subfolders under the root path, named by date.

To change the default local storage location for the HTML report formatting files

- 1 Open the `reporting.batch.xma.xml` file, located in the following directory:
 - **UNIX.** `EDX_HOME/xma/config/com/edocs/common/reporting`
 - **Windows.** `EDX_HOME\xma\config\com\edocs\common\reporting`
- 2 In the `reporting.batch.xma.xma` file, specify the path parameter value, starting with the correct slash (/ or \) for your platform, in the `BatchFileConfig` bean.

The following code shows the default `BatchFileConfig` bean definition:

```
<bean id="BatchFileConfig"
class="com.edocs.common.reporting.batch.file.BatchFileConfigBean"
scope="singleton">
<!-- use this property to override the default base, the system property for
edx.home
<property name="rootDir"><value>C:/oracle/edocs/data</value></property> -->
<property name="path"> <value>/output/reportapp</value></property>
```

- 3 If you need to change the install, or `EDX_HOME` directory, then edit the `rootDir` parameter. Specify an absolute path, such as `/oracle/edocs/data`, and do not end the `rootDir` value with a slash.

8

Payment Processing

This chapter describes how Oracle Billing Insight processes certain payment activities. It includes the following topics:

- [About Check Processing on page 209](#)
- [About Credit Card Processing on page 216](#)
- [About Recurring Payment Processing on page 222](#)
- [About Payment Auditing on page 240](#)
- [Using Payment Debug on page 244](#)
- [Input File Format for Canceling Payments for Consumer Users in Batch on page 244](#)

About Check Processing

This topic describes how Oracle Billing Insight supports check payments through the ACH gateway.

Example of Adding a Check Account at Enrollment

This topic shows an example of adding a check account. You might use this feature differently, depending on your business model. When a user adds a new bank account, Oracle Billing Insight verifies the account information before the account can be used to make a payment. This process can take a few days or the customer can choose to use auto verify, where the bank account can be used immediately without account verification, until a payment is rejected.

The following actions occur when a new user specifies a checking account during enrollment:

- 1 A new customer enrolls for check payment services by completing an enrollment form in the user interface. Oracle Billing Insight saves the information in the `payment_accounts` table with an enrollment status of `pnd_active`.
- 2 The `pmtSubmitEnroll` job runs to submit the enrollment information to the payment gateway. It changes the enrollment status to `pnd_wait`. If the check cannot be submitted, then the status is changed to `Failed`.

For ACH only, `pmtSubmitEnroll` sends customer enrollment information, which is contained in a zero amount check called a prenote, to an ACH payment gateway for verification. To send a prenote, the `pmtSubmitEnroll` job creates a zero amount check with status of `prenote_scheduled`, and immediately inserts the check into the `EDX_PMT_TRANSACTION` table with a status of `prenote_processed`. This means that the status `prenote_scheduled` is transitory, and so is not visible in the `EDX_PMT_TRANSACTION` table. A summary report is created, which can be viewed from the Command Center.

- 3 After receiving the customer enrollment information, the ACH payment gateway responds with a return file only if there are errors in the customer enrollment information. If there are no errors, then ACH does not send a return file, or any other form of acknowledgement.
- 4 The `pmtConfirmEnrollment` job runs. This job updates the status of the customer enrollment status to `Active` if there are no problems after a specified number of days (by default, three days).

If the payment enrollment information is not correct, then the `pmtConfirmEnrollment` job updates the customer enrollment status to `bad_active`. An exception report is created, which can be viewed from the Command Center.
- 5 The customer might optionally receive an email about enrollment status from the `pmtNotifyEnroll` job.

Check Account Enrollment Status Workflow

Figure 7 shows the status changes that a new check account goes through for enrollment, depending on customer actions and the `pmtSubmitEnroll` and `pmtConfirmEnroll` jobs. The status is kept in the `account_status` field in the `payment_accounts` table.

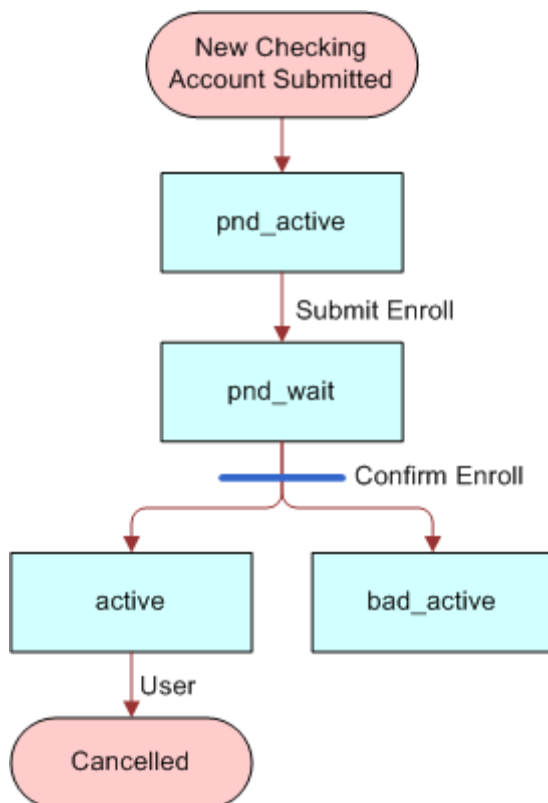


Figure 7. Check Account Enrollment Status Workflow

Table 58 describes the new check account status.

Table 58. New Check Account Status

Enrollment Status	Description
pnd_active	A new check account is enrolled, pending approval.
pnd_wait	The check account has been sent to the bank for verification.
active	The check account has been activated for payment.
bad_active	The check account failed to be activated.

Check Payment Transaction Workflow

Check Payment Transaction Workflow, shown in Figure 8, processes the typical ACH check payment transaction cycle (excluding transfers between the ODFI, ACH operator and RDFI).

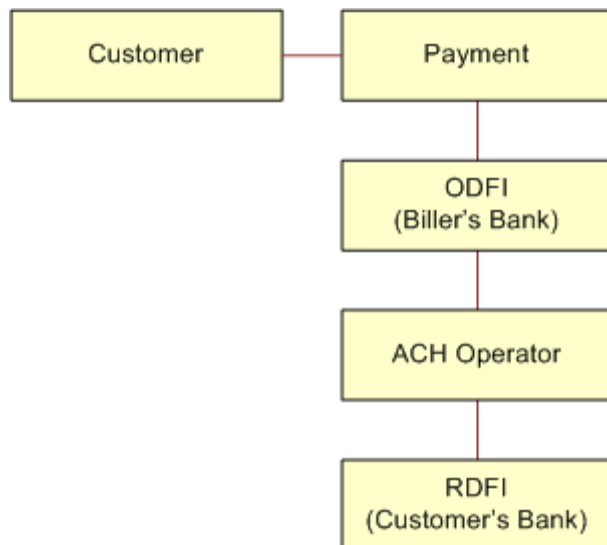


Figure 8. Check Payment Transaction Workflow

This workflow performs the following actions:

- 1 A customer logs in and schedules a new payment. This step inserts a check into the Oracle Billing Insight database with a status of scheduled.

If the customer later cancels the payment, then the check status is changed to cancelled, but the payment remains in the Oracle Billing Insight database for the customer to view as a cancelled payment.

- 2 The pmtCheckSubmit job runs. This step selects all the checks that are due for payment, creates a batch file of selected checks, and sends the batch file to the payment gateway (ODFI). It also changes the status of each selected check to Processed in the Oracle Billing Insight database.

If the check cannot be submitted, then the status is changed to Failed. A summary report log file is generated, which can be viewed from Command Center.

- 3 The payment gateway (ODFI) processes the received check payment through the ACH operator to the RDFI. In this step, if there is an error clearing the check, then ACH creates a file containing a code that indicates why the check was returned, and sends the file to Oracle Billing Insight.

- 4 The pmtCheckUpdate job runs. This step changes the status of the check from Processed to Paid if there is no return file, and five business days (default) have passed.

If the payment gateway returns the check, then the pmtCheckUpdate job updates the check's status to Returned, and saves the reason code in the txn_err_msg field of the EDX_PMT_TRANSACTION table. An exception report is generated to summarize the information in the returned file, which can be viewed from Command Center.

If there is an error other than returned, then the pmtCheckUpdate job changes the check status to Failed.

- 5 If configured, then the pmtPaymentReminder job sends email to the customer about the status of the check payment.

Check Payment Status Workflow

Check Payment Status Workflow updates the check payment status at different stages of check payment processing.

Figure 9 shows the states that a check can be in, and the jobs that change the state.

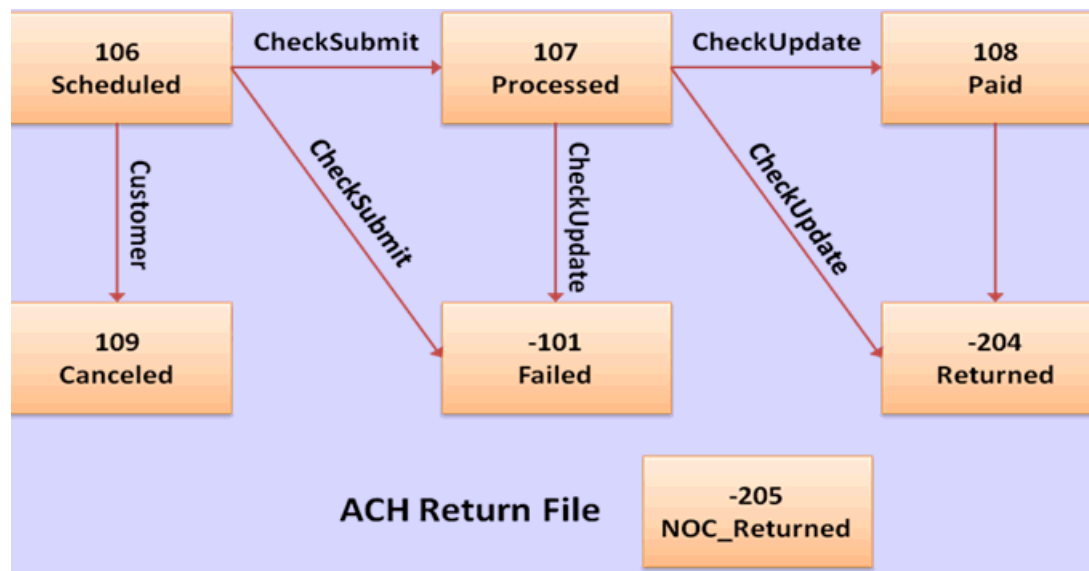


Figure 9. Check Payment Status Workflow

Table 59 lists the statuses that can occur during a check payment transaction cycle. The values in parentheses () are the values saved in the Oracle Billing Insight database

Table 59. Check Payment Transaction Status

Transaction Status	Description
Scheduled(106)	A customer scheduled a new check payment.
Processed(107)	Oracle Billing Insight Payment processed a check and sent it to the ACH gateway.
Paid(108)	ACH paid or cleared a check.
Cancelled(109)	The customer cancelled a check.
Failed(-101)	ACH failed to pay a check failed for a reason other than returned.
Returned(-204)	ACH returned a check.
noc_returned(-205)	This customer's payment account information must be changed.

Automated Clearing House (ACH) Return Codes

This topic describes the codes and other data used with Automated Clearing House (ACH) Network fund transfers. Additional information about ACH and change codes are available at

<http://www.nacha.org>

Supported SEC Codes

For ACH, Oracle Billing Insight supports the following SEC Codes (Standard Entry Class Codes):

- **Web.** Internet initiated entry (default for Oracle Billing Insight). Debit entries are originated (single or recurring) from a customer's account using web based authorization.
- **PPD.** Pre Arranged Payment and Deposit Entry. Under PPD the following types are included:
 - **Direct Deposit.** The credit application transfers funds into the customer's account.
 - **Preauthorized Bill Payment.** This is a debit application, where billers transfer electronic bill payment entries through the ACH network.
- **CTX.** Corporate Trade Exchange. Supports multiple addenda records based on ANSI ASC X12 standards. This code can be used either with the credit or debit application.

ACH Change Codes (NOC)

Table 60 lists some of the ACH change codes (also known as NOC codes) that might appear in the returns file after running the pmtCheckUpdate job if previously valid payment information is now incorrect or out-of-date.

Table 60. ACH Change Codes

Code	ACH Change Code Description
C01	Incorrect DFI Account Number.
C02	Incorrect Routing Number.
C03	Incorrect Routing Number and Incorrect DFI Account Number.
C05	Incorrect Transaction Code.
C06	Incorrect DFI Account Number and Incorrect Transaction Code.
C07	Incorrect Routing Number, Incorrect DFI Account Number, and Incorrect Transaction Code.

ACH Return Codes

Table 61 lists some of the ACH return codes that might appear in the returns file after running the pmtCheckUpdate job.

Table 61. ACH Return Codes

Code	ACH Return Code Description
R01	Insufficient Funds.
R02	Account Closed.
R03	No Account or Unable to Locate the Account.
R04	Invalid Account Number.
R05	Reserved.
R06	Returned at the request of ODFI.
R07	Authorization Revoked by Customer (adjustment entries).
R08	Payment Stopped or Stop Payment on Item.
R09	Uncollected Funds.
R10	Customer Advises Not Authorized, Item Is Ineligible, Notice Not Provided, Signatures Not Genuine, or Item Altered (adjustment entries).
R11	Check Truncation Entry Return (Specify) or State Law Affecting Acceptance of PPD Debit Entry Constituting Notice of Presentment or PPD Accounts Receivable Truncated Check Debit Entry.
R12	Branch Sold to Another DFI.

Table 61. ACH Return Codes

Code	ACH Return Code Description
R14	Representative Payee Deceased or Unable to Continue in that Capacity.
R15	Beneficiary or Account Holder (Other Than a Representative Payee) Deceased.
R16	Account Frozen.
R17	File Record Edit Criteria (Specify).
R20	Non-Transaction Account.
R21	Invalid Company Identification.
R22	Invalid Individual ID Number.
R23	Credit Entry Refused by Receiver.
R24	Duplicate Entry.
R29	Corporate Customer Advises Not Authorized.
R31	Permissible Return Entry (CCD and CTX only).
R33	Return of XCK Entry.

Additional information about these and additional ACH return codes are available at

<http://www.nacha.org/>

NOC Transactions

When a prenote is returned with a NOC, `TXN_MESSAGE` is populated with NOC information formatted as `NOC_CODE::NEW_ADDENDA_INFO::OLD_ADDENDA_INFO`

where:

- `NOC_CODE` is the three-character code returned.
- `NEW_ADDENDA_INFO` is the NOC information returned from ACH, which can include the corrected account number, routing and account type.
- `OLD_ADDENDA_INFO` is the existing addenda information.

ACH Effective Date

The Skip nonbusiness days for batch effective entry date field on the Payment Settings page for an ACH check payment gateway controls how the effective entry date is calculated when the ACH batch file is created by `pmtCheckSubmit`.

If the field is set to Yes, then nonbusiness days are not taken into consideration. The effective entry date is set to the payment date that the customer specified when scheduling the payment.

If the field is set to No, then nonbusiness days are skipped, and the effective entry date is the next business day following the computed date. Payment checks the scheduled payment date to see whether it is on or before the end of today. If so, then the computed date is the customer-scheduled date plus one. If not, then the computed date is the customer-scheduled date.

Nonbusiness days are weekend days and the U.S. Federal holidays.

ACH Settlement Date

The ACH settlement date is not written to the ACH batch file by pmtCheckSubmit. That date is added by the ACH Operator when the payment is settled.

ACH Addenda Records

Payment supports ACH addenda records, which means you can append a list of addenda records after an entry detail record in an ACH file. Addenda records are biller-specific, so customization is required to support this feature. Theoretically, you can put any information into an addenda record, such as the invoices of a payment. To add addenda records, you must write a plug-in for the pmtCheckSubmit job. For more information about supporting ACH addenda records, create a service request (SR) on My Oracle Support. Alternatively, you can phone Oracle Global Customer Support directly to create a service request or get a status update on your current SR. Support phone numbers are listed on My Oracle Support.

About Credit Card Processing

Credit card payments are supported for immediate and future (scheduled) payments. Card payments require two steps:

- **Authorization.** Authorization verifies the customer account and puts a hold on the account for the amount of the payment.
- **Settlement.** Settlement occurs when the payment is made.

Oracle Billing Insight Payment performs authorization and settlement in one transaction.

Credit card payments require an agreement with a credit card gateway to process credit card transactions. A cartridge for PayPal Payflow Pro is provided with Oracle Billing Insight, which requires signing up with PayPal Payflow Pro payment services. The Chase Paymentech Orbital Payment Gateway cartridge has been tested for use with Oracle Billing Insight. In addition, you can create cartridges to support other payment processors. For help with cartridges, contact your Oracle sales representative to request assistance from Oracle's Professional Services.

Paymentech Authorization Transactions

For account verification, Paymentech accepts Visa and Mastercard zero dollar transactions, however, American Express and Discover cards are \$0.01 transactions.

Credit Card Payment Status

Table 62 lists the statuses that can occur during a payment transaction cycle. The values in parentheses () are the values saved in the Oracle Billing Insight database.

Table 62. Payment Status

Transaction Status	Description
Scheduled (106)	A customer has scheduled a new credit card payment.
Settled (108)	The credit card payment was authorized and settled successfully.
Failed-authorized (-104)	A credit card payment failed during authorization.
Cancelled (109)	A credit card payment was cancelled by the customer.
Failed (-101)	A credit card payment failed because of network problems. This state occurs only for instant payments. For scheduled payments or recurring payments, the state stays scheduled if there is a network problem, so that it will be tried again. It is not necessary for Oracle Billing Insight to retry an instant payment. The user sees the error message and can optionally retry the payment.

Credit Card Payment Transactions

Figure 10 shows the entities involved in a credit card payment transaction.

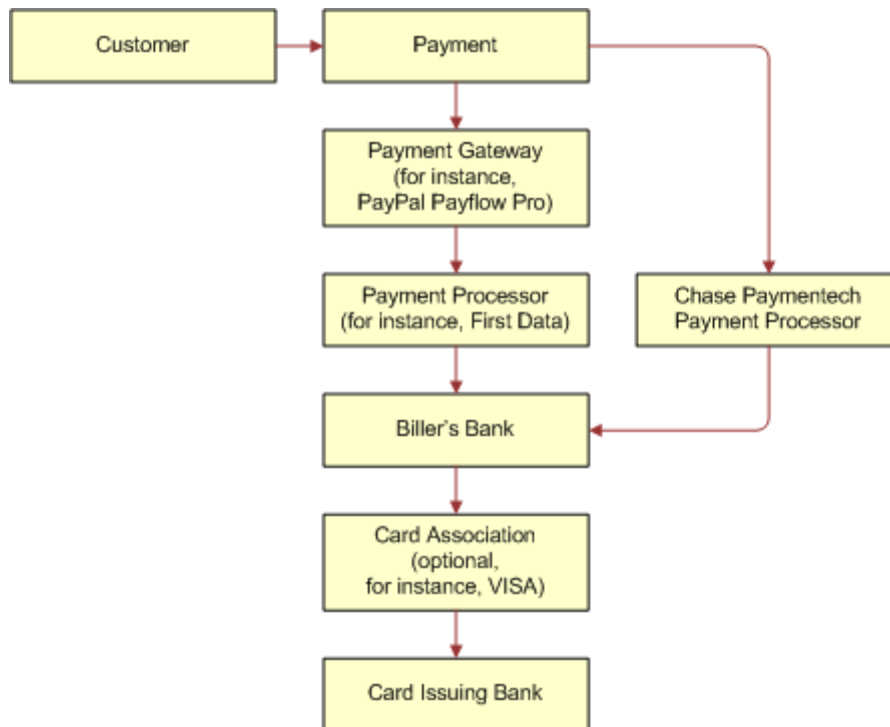


Figure 10. Credit Card Payment Transactions

Because credit card processing is real-time and not batch-based, the life cycle of credit card is simpler than check processing. Credit card processing typically goes through the following steps:

- 1 A user enters a credit card number and other card-related information.
- 2 The card information is sent to the card-issuing bank for authorization. Authorization only guarantees that the money is available at the time of authorization.
- 3 The merchant issues a settlement request to issuing bank so that the money can be transferred, usually after fulfillment (sending out ordered goods). For bill payments, the biller does not send out ordered goods, so authorization and synchronization are combined into one operation. A credit card payment is settled at the same time it is authorized.

Instant Credit Card Payments

The following code shows the status values for an instant credit card payment. For instant payments, there is no scheduled state:

- Settled
- Failed-authorize
- Failed

An instant card payment processes as follows:

- 1 A user submits an instant credit card payment from the UI.
- 2 Oracle Billing Insight sends the payment to the credit card cartridge in real time.
- 3 If the card is authorize and settled, then the credit card state is set to Settled.
- 4 If the card failed to authorize, then the state is set to Failed_authorize.
- 5 If there is a network problem, then the state is set to Failed.
- 6 The card is inserted into the EDX_PMT_TRANSACTION table.
- 7 The result of the transaction is presented to the user.
- 8 The pmtPaymentReminder job runs and (optionally) sends email to users who have made an instant payment.

Scheduled Credit Card Payments

The following code shows the status values for a scheduled credit card payment:

- Scheduled
- Cancelled
- Settled
- Failed-authorize
- Scheduled

Scheduled credit card payments process is as follows:

- 1 A card payment is scheduled by the customer through the user interface, and the payment is marked as Scheduled in the EDX_PMT_TRANSACTION table.

Before the scheduled card payment is processed by pmtCreditCardSubmit, the user can modify or cancel it.

- 2 When the `pmtCreditCardSubmit` job runs, it selects all credit card payments that are scheduled to be paid at the time the job runs, opens a connection to the credit card payment gateway, and starts making payments. The Number of days before a credit card's pay date for it to be submitted parameter determines how many days in the future to look when selecting payments to be made.

If `ICreditCardPlugIn` has been implemented in Payment Settings, then this job modifies the card payments that are scheduled to be paid, or takes other actions related to the selected card payments. Functions in the plug-in are called before and after credit card payment processing. For more information about the `pmtCreditCardSubmit` job and its plug-in, see *Administration Guide for Oracle Billing Insight*. For help with configuring job plug-ins, contact your Oracle sales representative to request assistance from Oracle's Professional Services.

- 3 The credit card gateway sends the transactions to the credit card processor. The credit card processor either authorizes and settles the payment, or rejects it. The results are returned to the credit card gateway, which forwards the results to the `pmtCreditCardSubmit` job.
- 4 The `pmtCreditCardSubmit` job changes the status of the credit card payment in the Oracle Billing Insight database depending on the transaction status returned by the credit card processor, and optionally sends email to the customer about the status of the payment.
 - If the card is authorized and settled, then the credit card status is set to Settled.
 - If the card fails to authorize, then the status is set to Failed_authorize.
 - If there is a network problem, then the status remains Scheduled.
- 5 The `pmtPaymentReminder` job runs and (optionally) sends email to users about the status of their scheduled payment.

User Options

The user interface to Oracle Billing Insight Payment can offer a variety of credit card payment options. Some of those options require that fields be configured in Payment Settings for a credit card payment gateway.

Using PayPal Payflow Pro as a Payment Gateway

A cartridge for PayPal Payflow Pro is provided with Oracle Billing Insight. Before configuring a PayPal Payflow Pro credit card payment gateway, you must obtain a digital certificate through PayPal Payflow Pro.

You must configure PayPal Payflow Pro as your payment gateway in the Command Center. For details on configuring a gateway with Oracle Billing Insight, see *Administration Guide for Oracle Billing Insight*.

Using Chase Paymentech Orbital Payment Gateway

The cartridge for Chase Paymentech Orbital Payment Gateway has been tested for use with Oracle Billing Insight.

You download the required Chase Paymentech files as part of the application server configuration process. For more information about installing Chase Paymentech, see *Installation Guide for Oracle Billing Insight*.

You must configure Chase Paymentech as your payment gateway in the Command Center. For details on configuring Paymentech with Oracle Billing Insight, see *Administration Guide for Oracle Billing Insight*.

Using CyberSource Payment Gateway

The cartridge for CyberSource Payment Gateway has been tested for use with Oracle Billing Insight.

You must configure CyberSource as your payment gateway in the Command Center. For details on configuring CyberSource with Oracle Billing Insight, see *Administration Guide for Oracle Billing Insight*.

Address Verification Service

Address Verification Service (AVS) reduces the risk of fraudulent transactions by verifying that the credit card holder's billing address matches the one on file at the card issuer. The address is optional and does not affect whether the payment is accepted or rejected. However, using an address might get a lower rate from the card issuer.

A merchant, also known as the biller, submits the AVS request through the payment process directly to the specific credit card association, such as PayPal Payflow Pro, for address comparison. If AVS is turned on by the System Administrator, then address information is passed into PayPal Payflow Pro as part of the PayPal Payflow Pro request. PayPal Payflow Pro then contacts the credit card issuing bank and passes along the address information.

The credit card issuing bank verifies the credit card address information on record matches the address information from PayPal Payflow Pro. The credit card issuing bank then replies back to PayPal Payflow Pro whether information matched (address and postal code are checked during AVS). Y means yes, N means no, and X means a match cannot be determined. PayPal Payflow Pro then accepts or rejects (voids) the transaction based on the filter set through Oracle Billing Insight Payment (for both street address and postal code).

There is also a filter option to set the international AVS code to determine whether the AVS response was international, U.S. or could not be determined. Some credit card issuing banks require city and state verification as well. Oracle Billing Insight Payment does not handle these by default, but the `pmtCreditCardSubmit` job has a plug-in to allow custom code to pass in the AVS values.

If Oracle Billing Insight Payment does not send the address information to PayPal Payflow Pro, or the system administrator did not turn on AVS, and the AVS check level is set to Full, then the transaction fails. If the card issuer address is sent to the payment gateway but the address does not match the information on the gateway, then the gateway can send an AVS code. If an AVS code is received, then Oracle Billing Insight Payment logs the AVS code in the audit tables.

Turning AVS On or Off by Transaction

PayPal Payflow Pro and the Chase Paymentech Orbital Payment Gateway support turning AVS on or off by transaction. However, the lower capability Payflow Link can perform this function. You also must set up the AVS level as part of your PayPal Payflow Pro agreement. When setting up the account with PayPal Payflow Pro, the merchant must specify the level of AVS check: full, medium or light. For additional information on setting up PayPal Payflow Pro, see the PayPal Payflow Pro documentation.

When Oracle Billing Insight passes the address information, PayPal Payflow Pro accepts or rejects the transaction based on the AVS check level. Note that the AVS check level is specified once during merchant account setup and applies to all transactions for that merchant. During setup, the customer (merchant) also must specify to PayPal Payflow Pro that he or she uses AVS (through Oracle Billing Insight) for transactions.

About Recurring Payment Processing

Oracle Billing Insight provides two types of recurring payments for check and credit card:

- **A recurring payment.** A recurring payment allows a customer to schedule a payment amount that is fixed or for the entire amount due from a bill. The payment can be scheduled to be paid on a certain date of the month.
- **An automatic payment.** An automatic payment allows a customer to schedule a payment of a fixed amount or for the entire amount due from a bill, to be made a certain number of days before due date. Automatic payments of the entire amount due can also be made, if the amount due is less than a specified amount.

Both recurring and automatic payments are designated as recurring payments by the NACHA 2009 specification. NACHA 2009 defines a payment as recurring when the account manager (Oracle Billing Insight) keeps the account information in a database.

Recurring payments can be modified or cancelled at any time before the payment is scheduled.

Recurring payment allows a customer to make payments automatically, based on the amount and pay date. There are five kinds of recurring payments:

- Amount due and before due date, for example, pay the entire amount due two days before the due date.
- Amount due and fixed pay date, for example, pay minimal amount due on day 31 of each month.
- Fixed amount and before the due date, for example, pay \$100 one day before the due date.
- Fixed amount and fixed pay date, for example, pay \$100 on the first day of each month.
- Amount due up to a fixed amount, and send email if over that fixed amount.

Amount defines how much the recurring payment is going to pay for each payment. The amount can be a fixed amount or the amount due. The amount due is that of the loaded bill.

Pay date defines when each payment is going to be cleared (money transfers). The pay date can be a fixed date or a date before it is due.

For monthly payments, if day 29, 30, or 31 is selected, and that day does not exist for a particular month, then the pay date defaults to the last day of that month. For example, specifying day 31 of each month ensures that payments are made at the last day of each month.

The *effective period* defines when a recurring payment starts and ends. A payment is made if its pay date is within the effective period (inclusive). If the pay date is after the end date of the effective period, then the recurring payment is deactivated. By default, a recurring payment only starts tomorrow. This is done so that all bills that arrive up to and including today are considered paid, so recurring payment must not pay these bills a second time.

Each bill has a unique ID called the *bill ID*, which is from the statement number of each statement record.

Oracle Billing Insight prevents a bill from being paid twice.

After a user creates a recurring payment, that user is not permitted to change the payment amount from fixed to amount due, or to change the pay date from fixed to before due date, or conversely. When a recurring payment starts (which is when the first recurring payment has been made), the start date of the recurring payment cannot be modified.

CAUTION: Recurring payment supports only one customer account for each biller. Recurring payment does not support multiple customer accounts with a single biller.

Recurring payments consist of actions at the front-end (UI) and back end (Command Center jobs). The UI allows a user to insert, update, and delete a recurring payment, and the back end `pmtRecurPayment` job makes the payment.

The recurring payment feature involves a great deal of business logic.

Recurring Payment Assumptions

The recurring payment feature assumes that the bill balances are accumulative. The bill for this billing cycle includes the balance of the bill from previous billing cycle, and the later bill has a due date after that of the previous bill.

Recurring payment also assumes that each bill has a date indicating the chronological order of bills. This is usually the date when the bill arrives. For example, the column `UPDATE_DATE` in the `EDX_ACCOUNT_LATEST_STATEMENT` table can be used to indicate the chronological order of arriving bills. In the case of external billing software, other dates such as statement date can be used for this purpose. When recurring payment synchronizes with the Command Center or other billing software, it must retrieve the latest bill issued between the `last_process_time` and current time. This chronological date of bills (`UPDATE_DATE` or statement date) is used to guarantee that functionality.

Negative and Zero Bill Amounts

If a bill has a negative balance, then no payment is made. Instead, recurring payment assumes that this credit will roll into the balance of next bill. However, a zero dollar payment will be made if the balance is zero.

Due Dates

If a recurring payment is not a fixed date and fixed amount, then it must have a due date. The due date is used to decide which bill is the latest one to pay. You must load the due date or a date to use that is equivalent to the due date.

Tables Updated by Recurring Payments

The amount field in the recurring_payments table records the amount when you do one of the following:

- Specify the recurring payment to pay fixed amount
- Pay up to this amount

The recurring_payments table does not save bill summary or payment information.

Oracle Billing Insight pulls bill summary information from the EDX_ACCOUNT_LATEST_STATEMENT table and saves it into the payment_bill_summaries table. The pmtRecurringPayment job populates the payment_bill_summaries and bill_id of the recurring_payments tables.

Payment information is scheduled into the EDX_PMT_TRANSACTION table for a check or credit card. The recurring_payments table is updated with the payment_id.

Recurring Payment Transaction Cycle

Recurring payment information is saved into the recurring_payments table.

By default, Oracle Billing Insight uses the latest available bill when submitting the payment to the payment gateway. You can configure each payment gateway to only synchronize once, which reduces processing. The setting, Whenever Job Runs, can be changed to, Only After the Current Bill is Scheduled, which causes Oracle Billing Insight to synchronize only once when the bill is scheduled.

The pmtRecurringPayment job retrieves bills, makes payments (check or credit card), and sends email notifications for recurring payments. The job performs two actions:

- Retrieves the latest bill for a recurring payment that a customer set up through the UI. This process is called *synchronization*. A recurring payment can only be synchronized with the Oracle Billing Insight database if it is associated with a bill and the amount to pay is the amount due or the pay date is before the due date. A recurring payment with fixed amount and fixed date will not be synchronized with the Oracle Billing Insight database, which means there is no bill information associated with this recurring payment.

- Schedules payments (inserts a payment with status of scheduled in the EDX_PMT_TRANSACTION table so that the payments will be processed. This process is called *scheduling*. A payment is scheduled zero days before the pay date (by default). The number of days can be changed by changing the Number of days before pay date to schedule the payment field in the job configuration. This delay allows the customer to modify or cancel this payment before the payment is processed by the pmtCheckSubmit or pmtCreditCardSubmit jobs.

The parameter, When to Synchronize Recurring Payment with Statements, belongs to the pmtRecurringPayment job.

Table 63 shows the columns that are updated in the recurring_payments table by the pmtRecurringPayment job.

Table 63. Columns Updated in recurring_payments table by the pmtRecurringPayment Job

Column Name in the recurring_payments Table	Description
bill_scheduled	Y or N: Determines whether the current bill associated with the recurring payment has been scheduled (inserted) into the EDX_PMT_TRANSACTION table. It is always N for a fixed amount and fixed pay date.
Status	Active or Inactive: This status is calculated internally. It indicates whether the recurring payment has ended, because either the pay date is after the end date, or the number of payments has reached the maximum allowed.
last_process_time	The last synchronization time. To improve performance, only bills whose Update Date falls between last_process_time and the current job running time (inclusive) are synchronized. By default, last_process_time is set to the start_date of the effective period when the recurring payment is created, which means all bills whose Update Dates are before start_date will not be synchronized.
last_pay_date	The pay date of last payment made. It is set to 01/01/1970 if the recurring payment has not started yet.
next_pay_date	The pay date of next payment. It is calculated based on start_date, last_pay_date and pay_interval.
bill_id	A foreign key reference to a row in the payment_bill_summaries table. Use bill_id to retrieve the latest bill information paid by the recurring payment. It might be null if there is no such bill.
curr_num_payments	Current number of payments made.

TIP: No payment is inserted into the EDX_PMT_TRANSACTION table when a recurring payment is created by the user. Payments are inserted by the pmtRecurringPayment job.

User Interface Updates to the Recurring Payments Table

The recurring payment UI lets users insert, update, or delete a recurring payment and view a list of recurring payments. Retrieving and deleting recurring payments from the Oracle Billing Insight database is straightforward. The UI prevents a user from updating a recurring payment from fixed date to before due date or conversely.

The activities performed in the user interface update the data in the recurring_payments table as described in [Table 64](#).

Table 64. Changes in the Recurring Payments Table

Column Name	Comment
AMOUNT_TYPE and AMOUNT	<p>These two columns record how the payment amount is generated. They are only updated through the UI and are used by back-end jobs to calculate how much to pay. The valid values of AMOUNT_TYPE are:</p> <ul style="list-style-type: none"> ■ Fixed amount. Pay a fixed amount and the amount value is specified by AMOUNT column. ■ Amount due. Pay amount due on the bill and, AMOUNT column is not used (null). ■ Upto amount. Pay the amount due if it is less than the value of the AMOUNT column. Otherwise, pay the value of AMOUNT and send a notification.
PAY_INTERVAL DAY_OF_PAY_INTERVAL MONTH_OF_PAY_INTERVAL	<p>These three columns record how the payment date is generated. They are only updated through the UI, and are used by back-end jobs to calculate when to pay. The valid PAY_INTERVAL value is Monthly. User-specified to make payments monthly. The day of month is specified by DAY_OF_PAY_INTERVAL. The MONTH_OF_PAY_INTERVAL is irrelevant.</p>

Table 64. Changes in the Recurring Payments Table

Column Name	Comment
START_DATE END_DATE CURR_NUM_PAYMENTS MAX_NUM_PAYMENTS STATUS	<p>These columns determine when to start the recurring payment and when to stop it. START_DATE, END_DATE and MAX_NUM_PAYMENTS can only be updated through the UI.</p> <p>START_DATE is required, but you set only one of the END_DATE (end by that date) or MAX_NUM_PAYMENTS (end when this number of payments is made).</p> <p>The recurring payment STATUS is active when it is created and it has not reached either END_DATE or MAX_NUM_PAYMENTS. When one of them is reached, the STATUS is changed to inactive and the recurring payment will never take effect again.</p> <p>If END_DATE is chosen, if the pay date for the next bill to be paid is later than or the same as the start date, and sooner or the same as the end date, then the bill will be paid. The STATUS is set to inactive if the next pay date is later than the end date.</p> <p>If MAX_NUM_PAYMENTS is chosen, then the STATUS is changed to inactive when CURR_NUM_PAYMENTS reaches MAX_NUM_PAYMENTS.</p>
LAST_PAY_DATE	This is the pay date of the last bill. It is set to 01/07/1970 when recurring payment is created to indicate that there is valid information.
NEXT_PAY_DATE	This is the pay date of the next bill. When the recurring payment job runs, it schedules a payment with a pay date of NEXT_PAY_DATE. Note, NEXT_PAY_DATE is calculated based on LAST_PAY_DATE and PAY_INTERNAL.

Recurring Payment Job

The pmtRecurringPayment job gets bills and then schedules payments. The first process is called *synchronization* and the second process is called *scheduling*.

Recurring Payment Synchronization

During the synchronization process, the job retrieves a list of recurring payments to be synchronized, and then tries to get the bills for the recurring payments from the Command Center. [Figure 11](#) illustrates the recurring payment synchronization process.

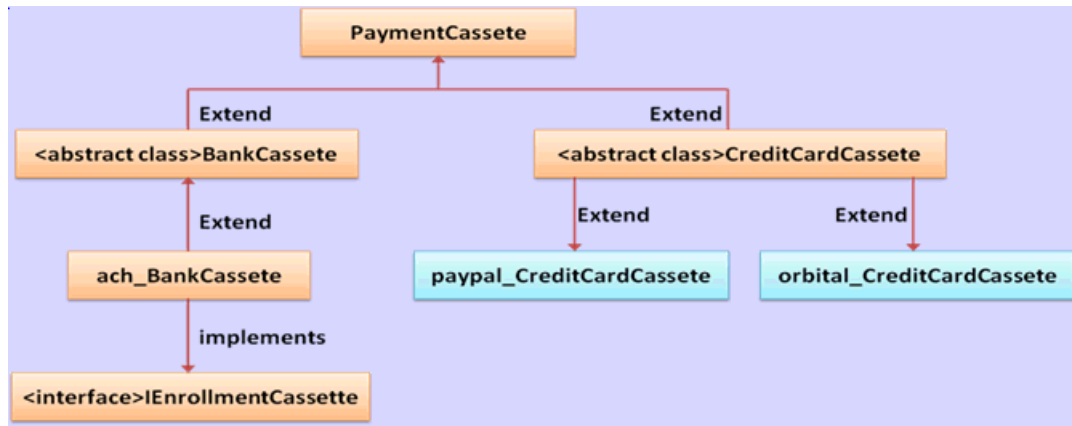


Figure 11. The Recurring Payment Synchronization Process

[Figure 12](#) shows the synchronization.

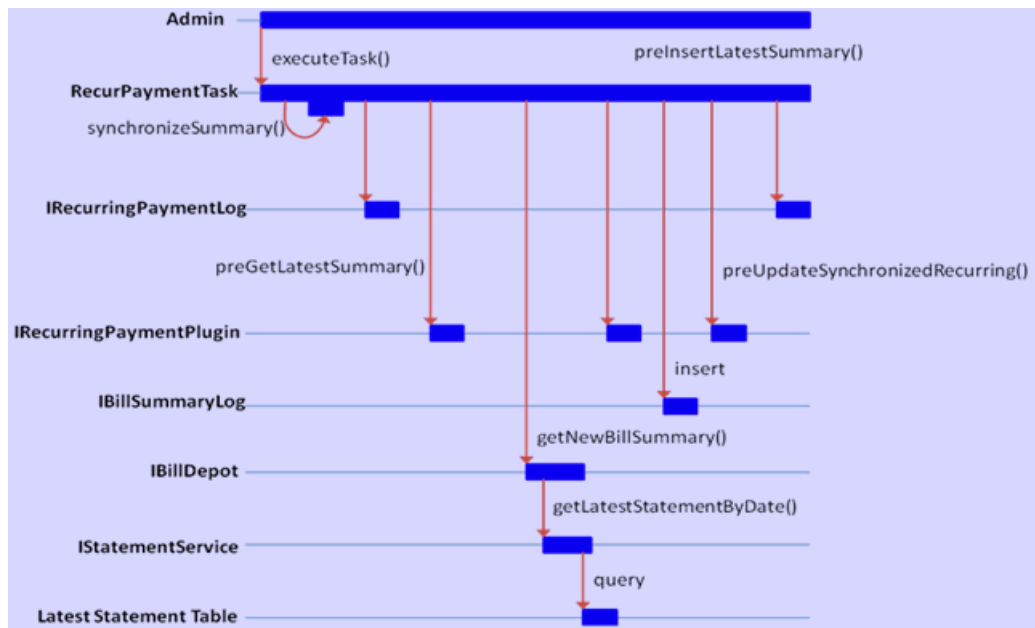


Figure 12. Recurring Payment Synchronization

Recurring Payment Scheduling

Recurring payment scheduling processes as follows:

- 1 Calls `RecurPaymentTask.isValidBillSummar()` to validate the latest retrieved bill summary. The latest bill summary could be ignored if it has no bill due date, or if the recurring payment is based on amount due but the bill summary has no amount due.
- 2 Now a valid bill summary exists. If the payment to the previous bill summary is still in scheduled status, then the following occurs:
 - Calls `RecurPaymentTask.cancelScheduledPayment()` to cancel this payment. The reason to cancel it is that the new bill summary just retrieved must include the balance of this scheduled bill, cancel the payment so that it will not pay the same bill twice.
 - Calls `RecurPaymentTask.modifyLastPayDate()`. If a recurring payment has a fixed pay date, but the amount is based on amount due, then it is necessary to back date the last pay date because the previous bill payment has been cancelled. Failing to do so will cause the current new bill being paid in next pay interval, not the current one. For example, assume that current bill cycle is October, the previous bill was retrieved on October 10 and is scheduled to pay on October 15. As a result, the `last_pay_date` and `next_pay_date` of the recurring payment are updated to October 15 and November 15, respectively. On October 11, a new bill is retrieved and the payment is scheduled. If Oracle Billing Insight does not back up the `last_pay_date`, then the new bill will be scheduled to pay on November 15. But in this case, it is necessary to pay the bill on October 15 because it is still in the October billing cycle. To fulfill this goal, go back date the `last_pay_date` to Sep 15 so the `next_pay_date` will be calculated as October 15, which will be used as the pay date for the new bill.
- 3 Calls `RecurPaymentTask.insertNewBillAndUpdateRecurring()`, which inserts the retrieved new bill and updates recurring payment accordingly.
 - Calls `IRecurringPaymentPlugIn.preInsertLatestSummary()` before inserting the bill summary in the `payment_bill_summaries` table.
 - If `PRE_INSERT_LATEST_SUMMARY_REJECT` is returned from the plug-in, then calls `RecurPaymentTask.updateRecurringPaymentOnly()` and return.
 - Calls `IBillSummaryLog.insert()` to insert this new bill summary.
 - If `IBillSummaryLog.insert()` generates the `DuplicateKeyException` indicating that this bill is already in the Oracle Billing Insight database, then calls `RecurPaymentTask.updateRecurringPaymentOnly()`.
 - Sets the `bill_scheduled` flag to N if the payment amount is not negative, or Y if it is negative. This means that no credit or reversal will be issued from recurring payment, and the credit appears as part of the next bill.
 - Sets the `bill_id` of the recurring payment to the one of the new bill summary.
 - Calls `IRecurringPaymentPlugIn.preUpdateSynchronizedRecurring()`.
 - If `PRE_UPDATE_SYNCHRONIZED_RECURRING_REJECT` is returned from the plug-in, then calls `RecurPaymentTask.updateRecurringPaymentOnly()` and return.

- Calls `IRecurringPaymentLog.update()` to update the recurring payment. The following table lists the information updated.

Column	Value
<code>last_pay_date</code>	In the case where the pay date is fixed, but amount is based on amount due, <code>last_pay_date</code> could be moved one <code>pay_interval</code> back if a scheduled payment is cancelled because a new bill arrives. Otherwise, <code>last_pay_date</code> will stay the same.
<code>next_pay_date</code>	<code>Next_pay_date</code> will be updated in <code>RecurringPaymentUtil.calculateInternal()</code> . In the case of fixed pay date, it will be updated based on <code>last_pay_date</code> . In the case of before due, it will be updated based on the due date of the new bill.
<code>status</code>	Because <code>next_pay_date</code> is changed, the status could be changed to inactive if <code>next_pay_date</code> falls after <code>end_date</code> .
<code>bill_id</code>	It is set to the <code>bill_id</code> (the primary key of the bill record) of the bill being inserted into the <code>payment_bill_summaries</code> table.
<code>bill_scheduled</code>	The <code>bill_scheduled</code> flag is set to N if the payment amount is not negative, Y if it is negative.
<code>last_process_time</code>	Set to the current time.

Recurring Payment Scheduling Workflow

Recurring Payment Scheduling Workflow schedules recurring payments for processing with the pmtRecurringPayment job. During scheduling processing, the pmtRecurringPayment job retrieves a list of recurring payments to be scheduled, and then schedules them, as shown in Figure 13.

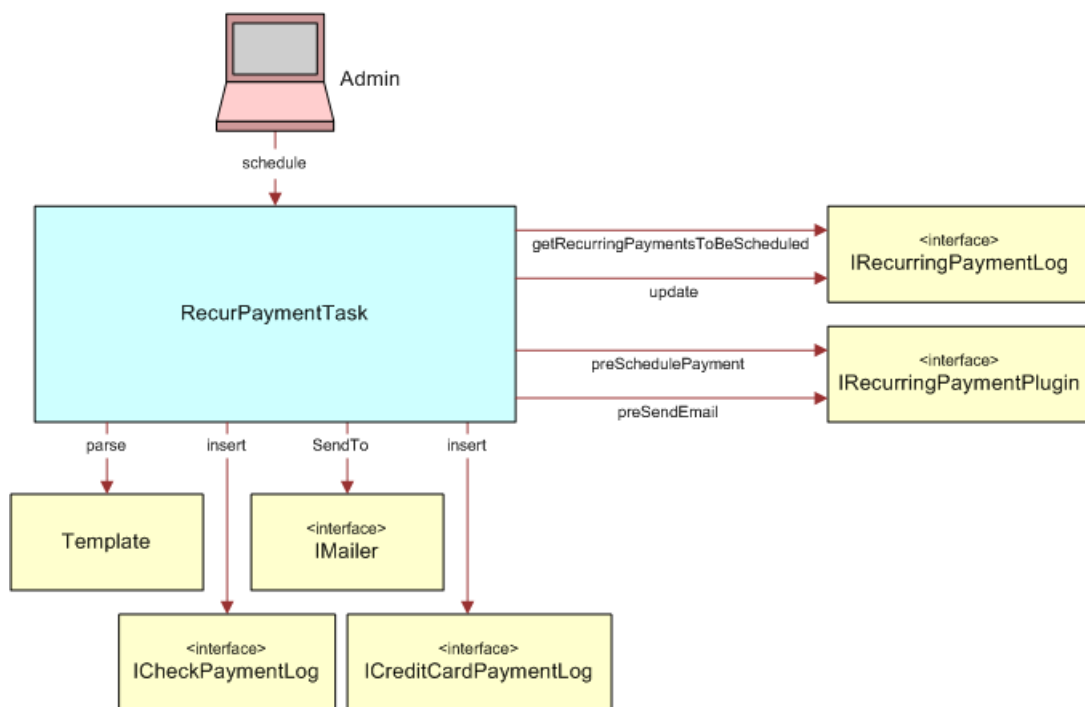


Figure 13. Recurring Payment Scheduling Workflow

Figure 14 shows the action sequence:

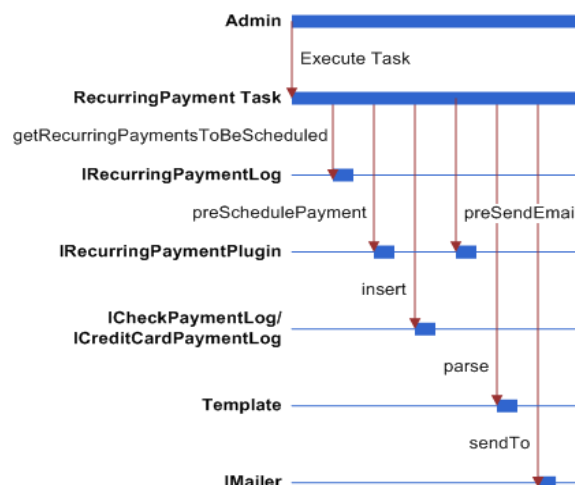


Figure 14. Recurring Payment Scheduling Action Sequence

Example of Scheduling the Amount Due Before the Due Date

This topic shows an example of how a recurring payment processes for amount due before the due date. You might use this feature differently, depending on your business model.

Example of How to Schedule the Amount Due Before the Due Date

The following steps describe an example of how to schedule the amount due before the due date:

- 1 On date 04/09/2012, a customer with account number acct1111 creates a recurring payment. The amount is amount due, the pay date is one day before due date, the start date is 04/10/2012, and the end date is 06/10/2012.

Column Name in the recurring_payments Table	Value
payer_account_number	acct1111
bill_scheduled	Y
status	active
last_process_time	04/10/2012. The value is the same as the start date.
last_pay_date	01/01/1970. It has not been paid yet.
next_pay_date	01/01/3000. This future date ensures there is no due date available yet.
bill_id	Null
max_num_payments	2147483647. This large number means the recurring payment will only be deactivated when the pay date is after the end date.

- 2 The pmtRecurringPayment job runs on 04/10/2012 23:59:00PM. The job searches the recurring_payments table to find all recurring payments whose bill_scheduled is Y and status is Active. It finds the example recurring payment and then asks Command Center to return all bills whose account number is acct1111 and whose UPDATE_DATE is between 04/10/2012 (last_process_time) and 04/10/2012 23:59:00PM (job run time). Two bills, bill2 and bill3, are returned. pmtRecurringPayment then finds the bill with latest due date bill3. bill2 is ignored because only the latest bill is paid.
- 3 After finding the latest bill from Command Center, pmtRecurringPayment checks whether the BILL_CYCLE_END_DATE of this bill is after the BILL_CYCLE_END_DATE of the bill used in the last payment (last bill information can be retrieved from payment_bill_summaries using the bill_id). If not, then this is an old bill and must not be paid. In this case, because there is no last payment, bill3 is paid.

- 4 Oracle Billing Insight inserts bill3 into the payment_bill_summaries table and the recurring_payment table is recalculated as shown in the following table.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	N, means this bill has not been paid or scheduled
status	active, because next_pay_date is within the effective period
last_process_time	04/10/2012 23:59:00PM, changes to job run time
last_pay_date	01/01/1970, unchanged
next_pay_date	05/14/2012, one day before the due date, 05/15/2012
bill_id	bill3

- 5 If the pmtRecurringPayment job runs between 04/11/2012 and 05/10/2012, then nothing happens to this recurring payment because synchronization and scheduling do not happen. The table remains unchanged.
- 6 On 05/11/2012 11:59:00PM, three days before next_pay_date, pmtRecurringPayment runs again. The recurring payment mentioned previously will not be synchronized, because its bill_scheduled is N. However, it will be scheduled. pmtRecurringPayment finds all recurring payments whose bill_scheduled is N, status is Active and next_pay_date is equal to or before 05/14/2012 (05/11/2012 + 3 days). The previously mentioned recurring payment is picked up and a payment is inserted into the EDX_PMT_TRANSACTION table. The amount of the payment is \$100.00, and the pay date is 05/14/2012. After this, the recurring payment table is changed to:

Column Name	Value
payer_account_number	acct1111
bill_scheduled	Y, means this bill has been paid
status	Active because next_pay_date is within the effective period
last_process_time	04/10/2012 23:59:00PM, unchanged because there was no synchronization
last_pay_date	05/14/2012, change to check's pay date
next_pay_date	05/14/2012, unchanged
bill_id	bill3
payment_id	Points to the new payment_id inserted into the EDX_PMT_TRANSACTION table.

The customer can now view the payment from Future Payments in the example interface. He or she can update or cancel the scheduled payment if desired.

- 7 On 05/12/2012 23:59:00PM, pmtRecurringPayment runs again and finds bills whose UPDATE_DATE is between 04/10/2012 11:59:00PM and 05/12/2012 23:59:00PM. No bills exist, and the Last Process Time is updated to 05/12/2012 23:59:00PM. Everything else remains the same.
- 8 On 05/13/2012, the ETL runs again and inserts a new bill, bill4. The bill detail information is obtained from the EDX_ACCOUNT_LATEST_STATEMENT table from OLTP..

ACCOUNT_NUMBER	STATEMENT_NUMBER	UPDATE_DATE	TOTAL_AMOUNT_DUE	STATEMENT_DUE_DATE
acct1111	bill1	03/10/2012	100.01	04/15/2012
acct1111	bill2	04/10/2012	50.00	04/25/2012
acct1111	bill3	04/10/2012	100.00	05/15/2012
acct1111	bill4	05/13/2012	80.00	06/15/2012

- 9 On 05/13/2012 23:59:00PM, the pmtRecurringPayment job runs again. It contacts Command Center and retrieves bills whose UPDATE_DATE are between 05/12/2012 23:59:00PM and 05/13/2012 23:59:00PM. bill4 is retrieved and the recurring_payments table is updated as shown in the following table.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	N means this bill has not been paid
status	Inactive because next_pay_date is beyond the effective period
last_process_time	05/15/2012 23:59:00PM, changes to job run time
last_pay_date	05/14/2012, unchanged
next_pay_date	06/14/2012, one day before due date, 06/15/2012
bill_id	bill4

After synchronization, the recurring payment is deactivated, and is never synchronized or scheduled again.

Example of Scheduling the Amount Due on a Fixed Pay Date

This topic shows an example of how a recurring payment processes for the amount due scheduled on a fixed pay date. You could use this feature differently, depending on your business model.

Example of How to Schedule the Amount Due with a Fixed Pay Date

The following steps describe an example of how to schedule the amount due with a fixed pay date:

- 1 On 04/09/2012, a customer with account number acct1111 creates a recurring payment. The amount is amount due, the pay date is day 31 of each month, the start date is 04/10/2012, and the recurring payment stops after 10 payments.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	Y
status	active
last_process_time	04/10/2012
last_pay_date	01/01/1970
next_pay_date	4/30/2012. This is the first available pay date after 04/10/2012 (because there is no April 31).
bill_id	Null
end_date	01/01/3000. The end date is so far in the future that the recurring payment will only be deactivated when the number of payments reaches maximum allowed.
curr_num_payments	0. No payments have been made yet.

The Bill table has the following values:

ACCOUNT_NUMBER	STATEMENT_NUMBER	UPDATE_DATE	TOTAL_AMOUNT_DUE	STATEMENT_DUE_DATE
acct1111	bill1	03/10/2012	100.01	04/15/2012
acct1111	bill2	04/10/2012	50.00	04/25/2012
acct1111	bill3	04/10/2012	100.00	05/15/2012

Even though the pay date is not related to the due date, DueDate must still be indexed because it is used to decide which bill is the latest.

- 2 The pmtRecurringPayment job runs on 04/10/2012 23:59:00PM. bill3 is found in the latest statement table and inserted into the payment_bill_summaries table. The recurring_payments table is recalculated as shown in the following table.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	N, this bill has not been paid.
status	Active, curr_num_payments is less than max_num_payments.
last_process_time	01/01/1970. This is unchanged.
last_pay_date	01/01/1970. This is unchanged.

Column Name	Value
next_pay_date	04/30/2012. There is no April 31.
bill_id	bill3
curr_num_payments	0

- 3 On 04/27/2012, three days before next_pay_date, pmtRecurringPayment runs again. There is no synchronization (bill_scheduled is N), but a payment is inserted into the EDX_PMT_TRANSACTION table. The amount of the check is \$100.00 and its pay date is 04/30/2012. The recurring payment table is changed as shown in the following table.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	Y, means this bill has been paid.
status	Active, curr_num_payments is less than max_num_payments.
last_process_time	04/10/2012 23:59:00PM: not changed because there has been no synchronization.
last_pay_date	04/30/2012. This was changed to next_pay_date.
next_pay_date	05/31/2012. This was changed to next available pay date.
bill_id	bill3
payment_id	Points to the new payment_id inserted into the EDX_PMT_TRANSACTION table.
curr_num_payments	1

- 4 Repeat steps 2, 3 and 4 until curr_num_payments reaches 10. At step 4 of the tenth payment, the status changes to Inactive.

If no bills arrive for a month, then next_pay_date is automatically moved to next month. For example, if there is no bill for April, then the next_pay_date is automatically moved from 04/30/2012 to 05/31/2012 when the current job run time is May 1.

Example of Scheduling a Fixed Amount Before the Due Date

This topic shows an example of how a recurring payment processes for a fixed amount scheduled before the due date. You could use this feature differently, depending on your business model.

Example of How to Schedule a Fixed Amount Before the Due Date

The following steps describe an example of how to schedule a fixed amount before the due date:

- 1 On 04/09/2012, a customer with account number as acct1111 creates a recurring payment from the UI. The amount is \$50, the pay date is one day before the due date, the start date is 04/10/2012 and the recurring payment stops after 10 payments.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	Y
status	active
last_process_time	04/10/2012
last_pay_date	01/01/1970
next_pay_date	01/01/300
bill_id	Null
end_date	01/01/3000. The end date is so far in the future that the recurring payment will only be deactivated when the number of payments reaches the maximum allowed.
curr_num_payments	0. No payments have been made yet.

Index table entries are as follows.

ACCOUNT_NUMBER	STATEMENT_NUMBER	UPDATE_DATE	STATEMENT_DUE_DATE
acct1111	bill1	03/10/2012	04/15/2012
acct1111	bill2	04/10/2012	04/25/2012
acct1111	bill3	04/10/2012	05/15/2012

Amount due is not required for this case.

- 2 The pmtRecurringPayment job runs on 04/10/2012 23:59:00PM, after running the load and after the new bill has been inserted. In this case, bill3 is found in the latest statement table and inserted into the payment_bill_summaries table. bill3 details are retrieved from the EDX_ACCOUNT_LATEST_STATEMENT table in OLTP and inserted into the payment_bill_summaries table. The recurring_payments table is recalculated as shown in the following table.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	N. This bill has not been paid.

Column Name	Value
status	Active, curr_num_payments is less than max_num_payments.
last_process_time	04/10/2012 23:59:00P. This changes to job run time.
last_pay_date	01/01/1970. The value is unchanged.
next_pay_date	05/14/2012. This date is one day before the due date, 05/15/2012.
bill_id	bill3
curr_num_payments	0

- 3 On 05/11/2012, three days before next_pay_date, pmtRecurringPayment runs again. There is no synchronization (because bill_scheduled is N), but a payment is inserted into the EDX_PMT_TRANSACTION table. The amount of the payment is \$50.00 and its pay date is 05/14/2012. The recurring_payments table is changed as follows.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	Y means this bill has been paid.
status	Active, next_pay_date is not after end_date.
last_process_time	04/10/2012 23:59:00PM. This value is unchanged because there was no synchronization.
last_pay_date	05/11/2012. This value changed to next_pay_date.
next_pay_date	05/11/2012. This value is changed. The next bill is not known.
bill_id	bill3
payment_id	Points to the new payment_id inserted into the EDX_PMT_TRANSACTION table.
curr_num_payments	1

- 4 These steps repeat until next_pay_date is after end_date, when status changes to inactive.

Example of Scheduling Fixed Amount and Fixed Pay Date

This topic shows an example of how a recurring payment processes for a fixed amount scheduled on a fixed pay date. You could use this feature differently, depending on your business model.

Example of How to Schedule a Fixed Amount with a Fixed Pay Date

The following steps describe an example of how to schedule a fixed amount with a fixed pay date:

- 1 On 04/09/2012, a customer with account number acct1111 creates a recurring payment. The amount is \$50 and the pay date is day 1 of each month. The recurring payment starts at 04/10/2012 and ends at 06/10/2012. The columns in the recurring_payments table are updated as follows.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	N
status	active
last_process_time	04/10/2012
last_pay_date	01/01/1970
next_pay_date	05/01/2012
bill_id	Null
end_date	06/10/2012
curr_num_payments	0. No payments have been made yet.

- 2 On 04/28/2012, three days before next_pay_date, pmtRecurringPayment runs again. There is no synchronization (bill_scheduled is always N) but a payment is inserted into the EDX_PMT_TRANSACTION table. The amount of the check is \$50.00 and its pay date is 05/01/2012. The columns in the recurring_payments table are updated as shown in the following table.

Column Name	Value
payer_account_number	acct1111
bill_scheduled	N. This bill has been paid.
status	Active, next_pay_date is not after end_date.
last_process_time	04/10/2012. This value is unchanged because there was no synchronization.
last_pay_date	05/01/2012. This value changed to next_pay_date.
next_pay_date	06/01/2012. This value changed to the next available pay date.
bill_id	Null
payment_id	Points to the new payment_id inserted into the EDX_PMT_TRANSACTION table.
curr_num_payments	1

Repeat Step 2 until next_pay_date is after end_date. Then the status changes to Inactive.

About Payment Auditing

Oracle Billing Insight Payment audits some Oracle Billing Insight Payment jobs to track a variety of transaction failures. Audits are kept for actions taken through the UI, as well as jobs.

Payment Jobs That Are Audited

The following jobs write to the audit tables:

- **pmtCheckSubmit.** Writes the following audited information:
 - Payments that failed during submission.
 - Encryption exceptions.
- **pmtPaymentReminder.** Writes payment reminders that were not sent, including:
 - Regular payment reminders that failed to send, for any reason, such as bad email address.
 - Check payment email that failed to send, for any reason, such as encryption error, bad email address.
 - Credit card payment email failed to send, for any reason, such as encryption error or bad email address.

- **pmtCreditCardSubmit**. Writes credit card payments that failed to submit, for example, because of encryption errors, invalid credit card information (such as invalid account) or network errors.
- **pmtIntegrator (AR)**. Writes check and credit card payments that were not written to the AR file, such as because of encryption errors or file write errors.
- **pmtRecurringPayment Job**. Check and credit card payments that failed.
- **pmtCheckSubmit and pmtCreditCardSubmit**. Submits check and credit card payments.

UI Actions That Are Audited

Lists successful and unsuccessful payments along with a reason code.

The UI actions that trigger an audit entry are:

- Create Recurring Payment
- Update Recurring Payment
- Delete Recurring Payment
- Create Schedule Payment
- Create Instant Payment
- Cancel Future Payment (Credit Card Payments)
- Update Future Payment (Credit Card Payments)
- Cancel Future Payment (Check Payment)
- Update Future Payment (Check Payment)
- Create Check Account
- Edit Check Account
- Delete Check Account
- Create Credit Card Account
- Edit Credit Card Account
- Delete Credit Card Account

Audit Database

The Oracle Billing Insight database supports auditing.

Columns for Audit

The EDX_PMT_TRANSACTION_HISTORY table has the new columns. The history tables have all the columns that the base table (EDX_PMT_TRANSACTION) has, plus the columns listed in [Table 65](#).

Table 65. Additional Columns in History Tables

Column Name	Comments
audit_operation	Defined in corresponding constant tables
audit_status	Defined in corresponding constant tables
audit_reason	Description of the audit
job_id	Pwc job ID
job_name	User-given job name
time_stamp	The record insertion time. For example: 1/18/2004 11:47:38 AM

New Tables

The following tables are based on the table name with _history at the end. They have all the columns in the base table, plus the new columns listed in [Table 65 on page 242](#) to support auditing.

- payment_accounts_history
- payment_bill_summaries_history
- payment_reminder_history
- recurring_payments_history

Audit Table Constants

Table 66 lists the tables that have audit information and the names of the corresponding code tables that explain the numeric codes for audit columns. See the tables in your Oracle Billing Insight database for the code descriptions.

Table 66. Audit Table Constants

Constant Table Name	History Table Name
EDX_PMT_TRANSACTION_CONST	EDX_PMT_TRANSACTION_HISTORY
RECURRING_PAYMENT_CONST	RECURRING_PAYMENT_HISTORY
PAYMENT_BILL_SUMMARIES_CONST	PAYMENT_BILL_SUMMARIES_HISTORY
PAYMENT_ACCOUNT_CONST	PAYMENT_ACCOUNT_HISTORY
PAYMENT_REMINDERS_CONST	PAYMENT_REMINDERS_HISTORY

Job Name Entries

User job names are combined with a shortened version of the task name to keep database entries manageable. The name of the job given by the user is combined with a shortened version of the task name as follows:

<job name given by the Admin>-<shorten task name>

Table 67 shows the shortened name for each job.

Table 67. Job Name Entries

Task Name	Shortened Task Name
CheckSubmitTask	ChkSubTsk
CheckUpdateTask	ChkUpdTsk
PaymentIntegratorTask	PmtIntTsk
CreditCardExpNotifyTask	CCExpNTsk
CreditCardSubmitTask	CCSubTsk
CreditCardUpdateTask	CCUpdTsk
ConfirmEnrollTask	ConEnrTsk
NotifyEnrollTask	NotEnrTsk
RecurPaymentSchedulerTask	RcuSchTsk
RecurPaymentSynchronizerTask	RcuSynTsk
PaymentReminderTask	PmtRmdTsk
SubmitEnrollTask	SubEnrTsk
CustomTask	CustomTsk

Using Payment Debug

You can turn on the payment debug flag to view details about payment processing.

To turn on the Payment debug flag

- Configure your application server to use `-Dcom.edocs.payment.debug=true` as part of the JVM starting option. For example, for Oracle WebLogic on UNIX, update the `startWebLogic.sh` file to add the following option to the Java command:

```
j java -Dcom.edocs.payment.debug=true ...
```

Related Topic

[“Debugging Oracle Billing Insight” on page 23](#)

Input File Format for Canceling Payments for Consumer Users in Batch

Oracle Billing Insight provides a SQL script for canceling payments configured for consumer users in batch. To use this script, you must create a pipe-delimited TXT data input file in the following format. For details on running the script to cancel user payments, see *Administration Guide for Oracle Billing Insight*.

[Table 68](#) shows the file format to use when canceling payments configured for consumer users in batch.

Table 68. Input File Format for Canceling Payments for Consumer Users in Batch

Position	Names	Maximum Length	Data Type	Required	Description
1	Billing System ID	20	VARCHAR2	No	The ID of the biller's system in Oracle Billing Insight
2	Billing Account Number	20	VARCHAR2	Yes	The user's billing account number

9

Customizing Payment

This chapter covers the tasks required to customize the Payment module in Oracle Billing Insight. It includes the following topics:

- [About Payment Plug-Ins on page 245](#)
- [Customizing Oracle Billing Insight Payment Template Files on page 254](#)
- [Generating Accounts Receivables \(A/R Files\) on page 265](#)
- [Customizing the Payment Amount Format on page 271](#)
- [Packaging Oracle Billing Insight Payment Custom Code on page 271](#)
- [About the APIs for Implementing Custom Payment Cassette Cartridges on page 272](#)
- [Implementing Custom Payment Cassette Cartridges on page 274](#)
- [Configuring Convenience Fee Payments on page 277](#)
- [Handling Multiple Payee ACH Accounts on page 279](#)
- [Using Payment APIs on page 280](#)
- [Configuring International Bank Routing on page 281](#)
- [Customizing Anonymous Payment Validation on page 282](#)
- [Updating the Back-End Billing System With a Top-Up Payment on page 283](#)

About Payment Plug-Ins

The Oracle Billing Insight Payment plug-in is a callback, which allows you to add code to extend the functionality of Oracle Billing Insight. The following payment plug-ins are included with Oracle Billing Insight:

- **IAchPlugIn.** This plug-in is for the ACH cartridge when submitting checks to ACH and processing the return file provided by ACH. For details on creating a custom ACH plug-in, see [“Creating an ACH Plug-In” on page 252](#).
- **ICreditCardPlugIn.** This plug-in is provided for use with all credit cards when submitting to a credit card gateway. For details on creating a custom credit card plug-in, see [“Creating a Credit Card Plug-In” on page 252](#).
- **IPaymentReminderPlugIn.** This plug-in is provided for use with the pmtPaymentReminder job. For details on creating a custom the payment reminder plug-in, see [“Creating a Payment Reminder Plug-In” on page 253](#).
- **IRecurringPaymentPlugIn.** This plug-in is provided for use with the pmtRecurPayment job. For details on creating a custom recurring payment plug-in, see [“Creating a Recurring Payment Plug-In” on page 254](#).

Each plug-in comes with a default implementation. It is recommended that you derive your plug-in from the default implementation to ensure that future updates to the plug-in do not break your code.

Job Plug-Ins

Table 69 lists the plug-ins available for the payment jobs.

Table 69. Payment Job Plug-Ins

Job	Plug-In Code
pmtCheckSubmit	DefaultAchPlugin.java
	AddendaCheckSubmitPlugIn.java
	A sample implementation is included in Oracle Billing Insight.
pmtCreditCardSubmit	DefaultCreditCardPlugin.java
pmtPaymentReminder	PaymentReminderPlugIn.java
pmtRecurringPayment	RecurringPaymentPlugIn.java
	SampleRecurringPlugIn.java
	A sample implementation is included in Oracle Billing Insight.

Check Submit Job with Plug-In Workflow

Figure 15 shows the workflow for the pmtCheckSubmit job when using the IAchPlugin plug-in.

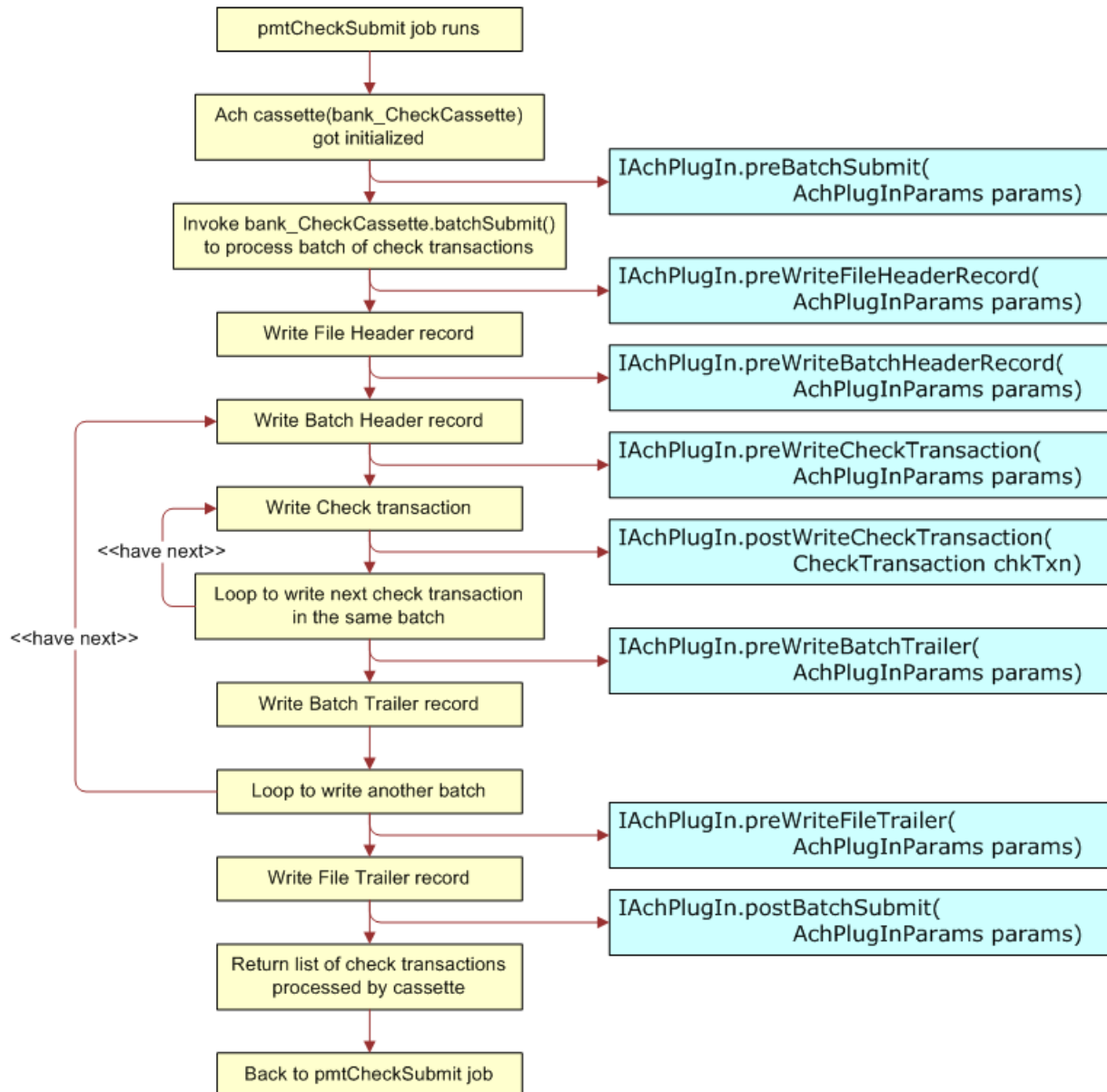


Figure 15. Workflow for the pmtCheckSubmit job with the IAchPlugin Plug-In

Instant Credit Card Payment with Plug-In Workflow

Figure 16 shows the workflow of the ICreditCardPlugin plug-in when a user submits an instant payment.

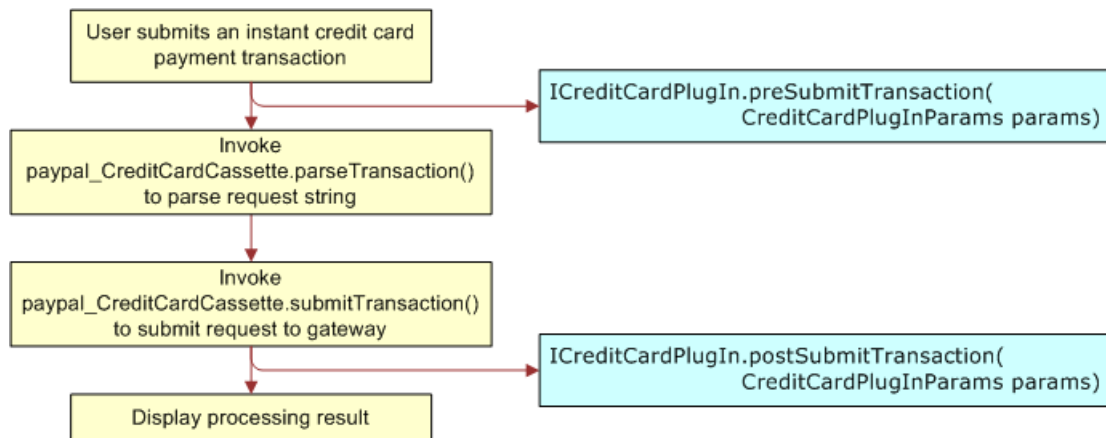


Figure 16. Workflow of the ICreditCardPlugin Plug-In When an Instant Credit Card Payment is Submitted

Credit Card Batch Payment Job with Plug-In Workflow

Figure 17 shows the workflow for the credit card batch payment processing job, `pmtCreditCardSubmit`, with the `ICreditCardPlugIn` plug-in.

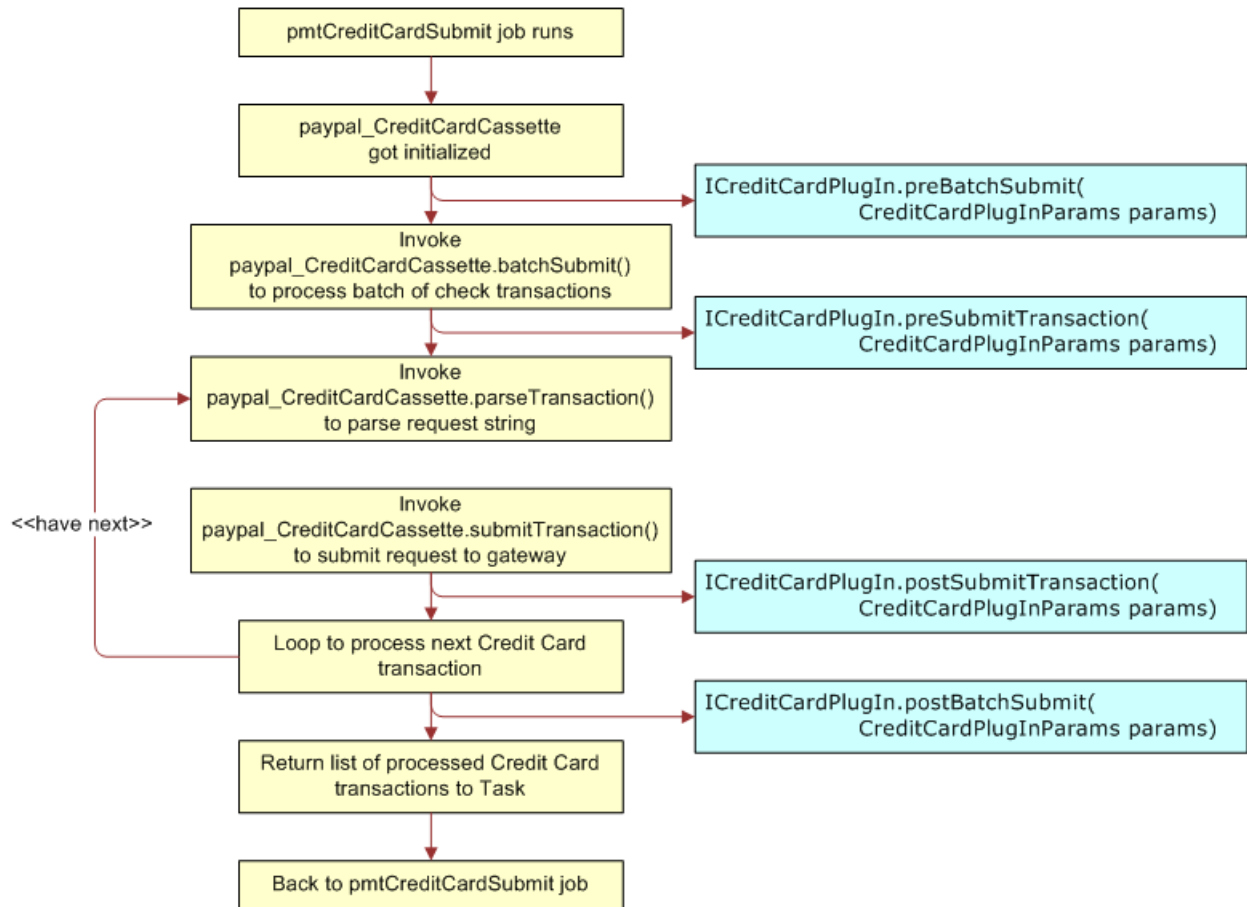


Figure 17. Workflow for the `pmtCreditCardSubmit` Batch Job with the `ICreditCardPlugIn` Plug-In

Recurring Payment Job with Plug-In Workflow

Figure 18 shows the workflow of the recurring payment job and how the plug-in works.

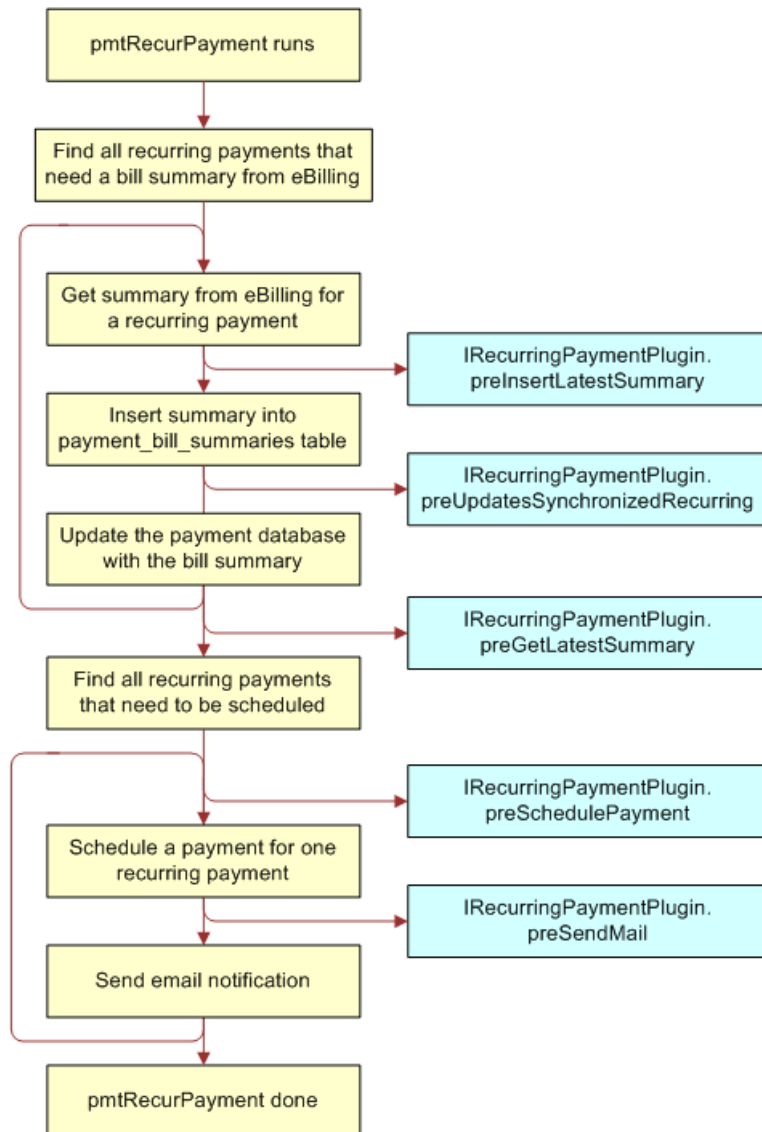


Figure 18. Recurring Payment Job with Plug-In Workflow

Payment Reminder Job with Plug-In Workflow

Figure 19 shows the workflow for the pmtPaymentReminder job plug-in.

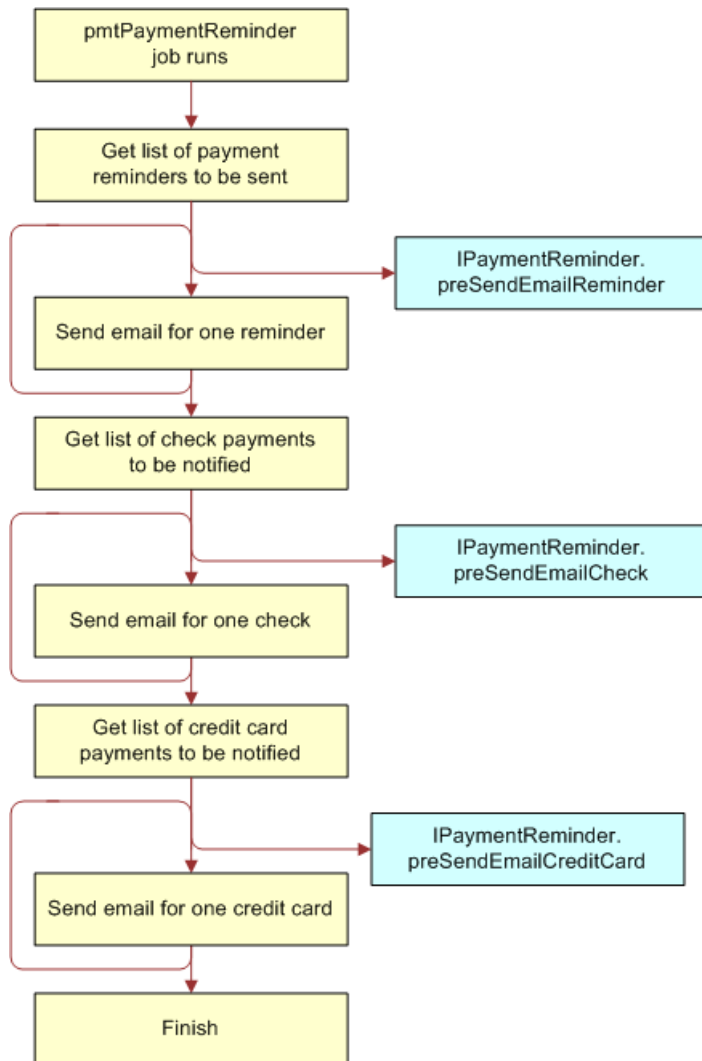


Figure 19. Payment Reminder Job Plug-In Workflow

Creating an ACH Plug-In

The ACH cartridge supports a plug-in to modify ACH file generation. When the `pmtCheckSubmit` job runs for ACH, it calls the methods of the implementation of `IAchPlugIn` during numerous events. (You specify the implementation in the Command Center Payment Settings.) The default implementation is `DefaultAchPlugIn`, which implements two methods to extend functionality while generating an ACH file.

You can use the `pmtCheckSubmit` plug-in to do the following:

- Change the default name of the ACH file.
- Change the format of the ACH file that the `pmtCheckSubmit` job generates.
- Create a remittance file in addition to the standard ACH file.
- Deny a check or change the default information put into the ACH file.
- Write addenda records for ACH. The implementation called `AddendaCheckSubmitPlugIn` gets the invoice information of a payment and writes them as addenda records.

Create your own implementation to accomplish these tasks. See [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#) for information about writing an implementation of `IAchIn`.

To create an ACH plug-in

- 1 Derive your implementation from the default implementation `DefaultAchPlugIn`.
- 2 Overwrite the methods whose behavior you want to change.
- 3 When compiling the class, include the `payment-1.0-SNAPSHOT.jar` and `payment_client.jar` files in your Java classpath.
- 4 Package this class into the `payment_custom.jar` file of each EAR file.

See [“Packaging Oracle Billing Insight Payment Custom Code” on page 271](#) for information about redeploying EAR files.

- 5 Change the payment settings in the Command Center to point to your new class.

For more information about changing payment settings in the Command Center, see *Administration Guide for Oracle Billing Insight*.

Related Topics

[Check Submit Job with Plug-In Workflow on page 247](#)

Creating a Credit Card Plug-In

The credit card plug-in is invoked from both the front end when an instant credit card payment is made and the back end when the credit card submit job runs. This plug-in allows you to audit the credit card payment, deny it, or change the payment request sent to the credit card gateway. The default implementation of `ICreditCardPlugIn`, `DefaultCreditCardPlugIn`, does nothing. You must write an implementation.

To create a credit card plug-in

- 1 Derive your implementation from `DefaultCreditCardPlugIn`.
- 2 Overwrite the methods for which you want to change the default behavior.
- 3 When compiling the class, include the `payment-1.0-SNAPSHOT.jar` and `payment_client.jar` files in your Java class path.
- 4 Package this class into the `payment_custom.jar` file of each EAR file.
- 5 Change the payment settings of the data definition file (DDN) in the Command Center to use the new plug-in implementation.
- 6 Command Center.

For more information about updating payment settings in the Command Center, see *Administration Guide for Oracle Billing Insight*.

Related Topics

[Instant Credit Card Payment with Plug-In Workflow on page 248](#)

[Credit Card Batch Payment Job with Plug-In Workflow on page 249](#)

Creating a Payment Reminder Plug-In

The payment reminder plug-in is invoked when the `pmtPaymentReminder` job runs. The `pmtPaymentReminder` job creates the following payment reminders:

- Regular payment reminders
- Check status notifications
- Credit card status notifications

Each of these tasks has a corresponding plug-in. Use the `com.edocs.common.payment.tasks.reminder.IPaymentReminderPlugIn` API. The default plug-in implementation, `com.edocs.commonpayment.tasks.reminder.PaymentReminderPlugIn`, does nothing.

To create a payment reminder job plug-in

- 1 Derive your implementation class from `PaymentReminderPlugIn`.
- 2 Overwrite the methods that you want to change the behavior of.
- 3 When compiling the class, include the `payment-1.0-SNAPSHOT.jar` and `payment_client.jar` files in your Java class path.
- 4 Package this class into the `payment_custom.jar` file of each EAR file.
- 5 Update the `pmtPaymentReminder` job configuration to use the new class.

For more information about changing job settings in the Command Center, see *Administration Guide for Oracle Billing Insight*.

Related Topics

[Payment Reminder Job with Plug-In Workflow on page 251](#)

Creating a Recurring Payment Plug-In

The recurring payment plug-in is called when the `pmtRecurPayment` job runs. You can use this plug-in to prevent a recurring payment from being scheduled based on business rules. Or, you can extract some statement detail fields from the latest statement table, and put them into the payment being scheduled.

The default plug-in implementation, `RecurringPaymentPlugIn`, does nothing. The file `SampleRecurringPlugIn.java` provides an example implementation.

To create a recurring payment plug-in

- 1 Derive your implementation class from `RecurringPaymentPlugIn`.
- 2 Overwrite the method that you want to change the behavior of.
- 3 When compiling the class, include the `payment-1.0-SNAPSHOT.jar` and `payment_client.jar` files in your Java class path.
- 4 Package this class into the `payment_custom.jar` file of each EAR file.
- 5 Update the `pmtRecurPayment` job configuration in the Command Center to use the new class.

For more information about changing job settings in the Command Center, see *Administration Guide for Oracle Billing Insight*.

Related Topics

[Recurring Payment Job with Plug-In Workflow on page 250](#)

Customizing Oracle Billing Insight Payment Template Files

Oracle Billing Insight provides a template engine to generate text messages, such as email, ACH files, and A/R files. This topic describes how to use Oracle Billing Insight Payment templates to customize those text messages.

Oracle Billing Insight Payment Template Engine

The Oracle Billing Insight Payment templates provide a generic template mechanism based on Java reflection. The template engine generates custom text output based on the templates. Similar to JSP, the template engine replaces the special placeholders inserted into the text file with the values of Java objects. For more detailed API documentation, see [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#) for details on accessing *Oracle Billing Insight* Javadoc.

The Template engine hosts a pool of objects in its context in the form of a hash table. You can refer to the variables in that context by their names. For example, there is a `Check` object whose name is `check`. You can refer to that object as `%check%`. This means replace `%check%` with the string returned from `check.toString()`. This is true for all Java objects except `java.util.Date`, where `getTime()` is called and inserts a long value that is the number of milliseconds since January 1, 1970, 00:00:00 GMT. If a method returns void, then nothing prints.

The content of the message consists of text plus resolved placeholders. Placeholders are Java variables, which are Payment hosted objects including their attributes and methods.

Enclose all template variables with two percent signs (`%%`). To escape `%`, use `%%`. For example, `%%40` means `%40`.

In addition to referring to variables, you can also access an object's public fields and methods. The valid reference is `%name.field%`, `%name.method(param1, param2, ...)%`. Each parameter to a method can be `name`, `name.field`, or `name.method(param1, param2, ...)`. The number of parameters is unlimited and an arbitrary level of method nesting is allowed (nesting means that a method's return value is used as a parameter when calling another method). For example, suppose there are two objects in contexts: *buf* which is a `StringBuffer`, and *str* which is a `String`. The following references are valid:

```
%buf%, %buf.append(str)%, %buf.append(str.toString())%
```

A static field or method can be accessed directly without instantiating an object. For example, `java.lang.Integer` has a static field called `MIN_VALUE` and a static method called `parseInt`. You can refer to them as `%java.lang.Integer.MIN_VALUE%` or `%java.lang.Integer.parseInt("12.34")%`.

All variables must be preset by calling `putToContext` on the Template class. Some variables are already set by Oracle Billing Insight Payment which you can use directly. But you can also put your own variables into the context:

```
%template.putToContext("buf", new java.lang.StringBuffer())%
```

This means to put a new `StringBuffer` object called *buf* into the template context. You can then refer to this object by its name:

```
%buf.append("abc")%
```

This appends "abc" to the end of the `StringBuffer`'s value.

The Oracle Billing Insight Payment engine has some limitations. It cannot do math operations, such as `x plus y`. You must call a Java method to do math operations. Another limitation is that it does not allow you to concatenate method calls, for example: `%variable.method().method() %`. You must write your own Java method to do method concatenation.

Included with the Oracle Billing Insight Payment package, there are a few utility classes to help you overcome the weakness of Oracle Billing Insight Payment Template Engine. These classes are:

```
com.edocs.common.payment.util.DecimalUtil
com.edocs.common.payment.util.DateUtil
com.edocs.common.payment.util.StringUtil.
```

One useful method in `StringUtil` is `concat`, which you declare and use as follows:

```
public static String concat(String s1, String s2, String s3)
%com.edocs.common.payment.util.StringUtil.concat(s1, s2, s3)%
```

Remember, you cannot do `%s1.concat(s2).concat(s3)%` inside a template, instead, you must call this function from template:

```
%com.edocs.common.payment.util.StringUtil.concat(s1, s2, s3)%.
```

Another useful method is `format()` from `DateUtil` class. This method helps format a `Date` object into different display formats. For example: `%com.edocs.common.payment.util.DateUtil.format("MMM dd, yyyy", check.getPayDate())%` formats a check's pay date to display as "Jan 01, 2000." For a complete list of possible date formats, please check the JDK document about `java.text.SimpleDateFormat`.

When writing customized Java code, it is strongly recommended that you use static methods as frequently as possible, so you can call them directly from a template without creating an instance of that object first. For example, by default, the individual ID field of an ACH entry detail field is populated with the customer's account number using `%check.getPayerAcctNumber()%`. The returned result is 16 bytes long, but the account number is 15 bytes, so you must truncate the retrieved value.

To create a Java class to do truncation and enable it in the Oracle Billing Insight Payment Template Engine

- 1 Write a Java class:

```
package com.edocs.common.ps;
public class MyUtil {
    public static String truncate(String s){
        return s.substring(1);
    }
}
```

- 2 Compile the class and put it into the `Payment_custom.jar` of each EAR file, then redeploy the EAR files.
- 3 Refer to this class in a template as follows:

```
%com.edocs.common.ps.MyUtil.truncate(check.getPayerAcctNumber())%
```

Oracle Billing Insight Payment Reminder Template

Oracle Billing Insight Payment reminder messages are generated based on the following notification types, and are defined in the notification template file:

- `PaymentScheduledNotification`
- `PaymentSuccessNotification`
- `PaymentFailureNotification`

`PaymentReminder.txt`, which resides in the `EDX_HOME/confi g/noti fi cati on/templ ates/` directory (the `EDX_HOME\onfi g\noti fi cati on\templ ates` directory on Windows). In the path, `EDX_HOME` is the directory where you installed Oracle Billing Insight.

This template is used for email notifications for processed, returned or failed payments.

Enrollment Notification Template

The enrollment notification template notifies customers about active and bad-active payment accounts and NOC returns. Enrollment reminder messages are generated based on the notification type `PaymentNotifyEnrollNotification`, which is defined in the notification template file.

This template is used for ACH. The parameter "isACH" is for ACH. If there are no payment gateways for ACH, then you can remove that topic from the template file.

Each Oracle Billing Insight Payment account is sent an individual email. Oracle Billing Insight Payment supports multiple payment accounts. If a customer has multiple payment accounts, then there could be more than one email message sent for each customer.

The variables described in [Table 70](#) apply to ACH.

Table 70. ACH Variables

ACH Variable	Type	Description
isACH	Boolean	True indicates this is an ACH notification.
success	Boolean	Success means this account has been activated successfully.
errCode	String	ACH return code, if the transaction failed.

The variables described in [Table 71](#) apply to ACH NOC returns.

Table 71. ACH NOC Return Variables

ACH NOC Variable	Type	Description
isNOC	Boolean	True indicates this is an NOC return.
isC01, isC02, isC03, isC05, isC06, isC07	Boolean	True indicates the returned NOC codes.
isAutoUpdate	Boolean	Returns the state of the <code>com.edocs.common.payment.cassette.ach.autoUpdatNOC</code> flag, which is configured on the Payment Settings page from the Command Center.
newPaymentAccount	String	New payment account number.
oldPaymentAccount	String	Old payment account number.
newRouting	String	New payment routing number.
oldRouting	String	Old payment routing number.
newPaymentType	String	New payment account type.
oldPaymentType	String	Old payment account type.

Recurring Payment Schedule Notification Template

When recurring payment schedules a payment, email notification messages are generated from the notification type `PaymentScheduledNotification`, which is defined in the template file.

Customizing ACH Templates

The ACH records of interest are in File Header, Batch Header, Entry Detail for PPD, Addenda and return for PPD, Batch Trailer and File Trailer. ACH fields can be mandatory, required, or optional. The contents of mandatory fields are fixed and must not be customized. Required fields are usually defined by the receiving bank, and can be customized for different banks. Optional fields can be customized, also.

By default, `secCode` is set to `WEB` to be compliant with the ACH 2001 format. However, you can change the SEC code based on the requirements of a biller's bank by editing the `batchHeader_template.xml` file.

[Table 72](#) describes some ACH fields. These fields can be customized upon a biller's request. The `pmtCheckSubmit` jobs running date is referred to as `Today`.

Table 72. ACH Fields

Field Name	Location	Description
Company Descriptive Date	8 th field in batch header, optional	Default is set to <code>Today</code> . The date <code>pmtCheckSubmit</code> is running.
Effective Entry Date	9 th field in batch, required	The date when checks in the batches are to be cleared. This is a suggested date from ACH, but the date that checks are cleared can vary. All checks with the same pay date will be put into one batch. The effective entry date might not always be the pay date. The default setting for effective entry date is: If the pay date is tomorrow or earlier, then it is the earliest business date after today. If the pay date is after tomorrow, then it is the earliest business date after the pay date (including the pay date).
Individual ID	7 th field in PPD entry detail, optional or required	By default set to the customer's account with the biller. Because this field is 15 bytes, the length of customer's account must not exceed 15 bytes. If the customer account is longer than 15 bytes, then either the field will not be populated, or you must truncate this field using Java code or the Java classes provided by Oracle Billing Insight.
Individual Name	8 th field in PPD entry detail. Required	By default set to the check's payment ID. Payment ID is the primary key on the <code>EDX_PMT_TRANSACTION</code> table. It can be used to map a returned check back to the one in Oracle Billing Insight database.

The templates for ACH are XML files, which describe the format of each ACH record, such as the start position, length, and so on. There are two sets of templates: one to generate ACH files, and another to parse ACH return files.

The first set of templates is used to generate the following ACH files:

- fileHeader_template.xml
- batchHeader_template.xml
- entryDetail_template.xml
- batchTrailer_template.xml
- Trailer_template.xml

When an ACH file is generated, check information is pulled from the database and then populated into the content of the XML files by replacing the template variables. The resulting XML file is transferred into an ACH file according to the format specified by the XML tags. The generic format of an XML tag is:

```
<amount pos="30" len="10" fmt="N" fract="2">%
```

where:

- *amount* is the name of the tag
- pos="30" is the start position
- len="10" is the length of the field
- fmt="N" is the format of the field
- fract="2" is the number of digits after the decimal point if the format (fmt) is N (numerical)

Table 73 through Table 77 list the template variables that are predefined in the Oracle Billing Insight Payment Template Engine. These variables are used to populate the content of the templates.

Table 73 describes the template variables that all templates use.

Table 73. Global Template Variables

Global Variable Name	Type	Description
template	com.edocs.common.util.template.Template	The template engine.
stringUtil	com.edocs.common.payment.util.StringUtil	Makes calling the static methods of StringUtil easier. Instead of using: %com.edocs.common.payment.util.StringUtil.concat("a","b","c")% use: %StringUtil.concat("a", "b", "c")%
decimalUtil	com.edocs.common.payment.util.DecimalUtil	Provides decimal number manipulations.
dateUtil	com.edocs.common.payment.util.DateUtil	Provides date manipulation methods and a calendar, which includes all U.S. holidays.
batch	com.edocs.common.payment.IPaymentBatch	The payment summary report, which you can view through the Command Center.
config	com.edocs.common.payment.config.IPaymentConfig	Payment setting information.
attributeName	com.edocs.common.payment.config.AttributeName	Payment setting parameter names. Use this parameter with the variable config to get payment setting information.

Table 74 describes the template variables that File Header uses.

Table 74. File Header Variables

Variable Name	Type	Description
fileCreateDate	java.util.Date	Creation date of the ACH file.
fileCreateTime	java.util.Date	Creation time of the ACH file.
fileIdModifier	java.lang.String	ACH file modifier, A to Z and 0 to 9.

Table 75 describes the template variables that Batch Header uses.

Table 75. Batch Header Variables

Variable Name	Type	Description
curPayDate	java.util.Date	The pay date of checks in the batch. All the checks in the same batch have the same pay date.
companyDescData	String	From Payment Settings.
companyDescDate	Date	Defaults to Today. To use another date, you must call a static Java method.
batchNumber	int	Starts from 1. This identifies the batches in the ACH.
batchEffectiveEntryDate	Date	Identifies the batches in the ACH.

Table 76 describes the template variables that Entry Detail uses.

Table 76. Entry Detail Variables

Variable Name	Type	Description
check	com.edocs.common.api.payment.ICheck	All check payment information, including the trace number.
addenda Record Indicator	int	Indicates whether there is addenda record for entry detail. 0=No, 1=Yes.

Table 77 describes the template variables that Batch Trailer uses.

Table 77. Batch Trailer Variables

Variable Name	Type	Description
batchEntryHash	String	See the ACH documentation.
batchEntryAddendaCount	int	Number of entries in the batch.
batchDebitAmount	String	Total debit amount in the batch.
batchCreditAmount	String	Always zero.
blockCount	int	See the ACH documentation.
totalEntryHash	String	See the ACH documentation.
totalEntryAddendaCount	int	Total number of entries in the file.
totalDebitAmount	String	Total debit amount in the file.

Matching a Check in the ACH Return to the Database

Return files are parsed by the return templates:

- fileHeader_return_template.xml
- batchHeader_return_template.xml
- entryDetail_return_template.xml
- addenda_return_template.xml
- batchTrailer_return_template.xml
- fileTrailer_return_template.xml

The format of these files is similar to the format of the submit template. For example:

```
<i ndi vi dual Name pos="55" l en="22" fmt="AN" target="%check. setPaymentId(?)%"></i ndi vi dual Name>
```

This code retrieves the part of the text from positions 55 to 77, puts it into a variable called ? and then calls `check.setPaymentId()` to set `payment_id` for the check. The template executes the template statement specified by XML tag "target" only.

When a check is returned from the ACH network, Oracle Billing Insight Payment matches it to that check in the database and marks it as returned. ACH modifies several fields in the return file. Oracle Billing Insight Payment populates one or more unchanged fields with identification information to help in matching them with a check in the database. Consult the ACH documentation for information about which fields are not changed.

The return template retrieves the error return code from the addenda record and then tries to reconstruct the payment ID or gateway payment ID to match a check in the database. If Oracle Billing Insight cannot populate the payment ID into the ACH file, then it uses the gateway payment ID, which is a concatenation of a few check payment fields that can identify a check.

By default, Oracle Billing Insight Payment populates the `payment_id` of the check into the individual name field to create the ACH file. The following line in the `entryDetail_template.xml` file populates the payment ID into an individual name:

```
<i ndi vi dual Name pos="55" l en="22" fmt="AN">%check. getPaymentId()%</i ndi vi dual Name>
```

The following line in the `entryDetail_return_template.xml` file extracts the payment ID:

```
< i ndi vi dual Name pos="55" l en="22" fmt="AN" target="%check. setPaymentId(?)%"></i ndi vi dual Name >
```

The following line in the `addenda_return_template.xml` file extracts the return error code:

```
<returnReasonCode pos="4" l en="3" target="%check. setTxnErrMsg(?)%"></returnReasonCode>
```

Payment then changes the status of the check to returned and updates this check in the database using its `payment_id`.

If the individual name is required for another task, for example, the check account name (which is the first 22 bytes), then follow these steps to use gateway payment ID.

To use the gateway payment ID

- 1 Modify the entryDetail_template.xml file to populate individual name with account name. Change:

```
<i ndi vi dual Name pos="55" len="22" fmt="AN">%check.getPaymentId()%</i ndi vi dual Name>
```

to:

```
<i ndi vi dual Name pos="55" len="22"
fmt="AN">%StringUtil.substring(check.getAccountName(), 0, 22)%</i ndi vi dual Name>
```

- 2 Modify the entryDetail_return_template.xml file so that payment ID will not be set for a returned check. Change:

```
<i ndi vi dual Name pos="55" len="22" fmt="AN" target=' %check.setPaymentId(?)%'></i ndi vi dual Name>
```

to:

```
<i ndi vi dual Name pos="55" len="22" fmt="AN"></i ndi vi dual Name>
```

- 3 Because payment ID cannot be used to match checks, use the gateway payment ID instead. Gateway payment ID is the ID generated by the template that submitted the ACH file to ACH. This template generates a unique ID based on the information submitted to ACH. This ID must contain information that will not be changed by ACH in the return file. The Oracle Billing Insight engine uses the gateway payment ID to find a match in the database.

In very rare circumstances, more than one match might be found. In that case, the match with the latest creation time is used. The following example discusses several ways to generate the gateway payment ID. Oracle Billing Insight Payment generates a trace number and puts that into the entry detail record. By default, the trace number starts at 00000000 and increases by one for each check until it reaches 99999999. After this point, the numbering restarts at 00000000. It is possible to get a duplicate trace number (after 10 million checks). However, because the Oracle Billing Insight Payment engine always chooses the payment with the latest date, the correct check will be matched. You can use both the trace number and individual ID (customer account number) to identify a payment and use them for the gateway payment ID.

Example 1: Unchanged ACH Trace Number

In the following example, it is assumed that the ACH or Bank will return both the original trace number and individual ID to Oracle Billing Insight:

- 1 At the beginning of the entryDetail_template.xml file, find the following code:

```
<ACH_6>
%<*>%
%check.setGatewayPaymentId(com.edocs.common.payment.util.StringUtil.concat(check.getPayerAcctNumber(), "_", check.getTxnNumber()))%
```

```
%< /*>%
```

This statement is commented out in the template, using %< *>% and %< /*>%. Removing the comment tags enables the statement.

The trace number is stored as txnNumber in the check object. This statement concatenates the customer account number, a "_", and trace number as the gateway payment ID. The setGatewayPaymentId method returns void, so nothing will print out. (If it did return a value, then that would print, which would ruin the format of the XML file.) After running pmtCheckSubmit, check the gateway payment ID in the EDX_PMT_TRANSACTION table, which is the concatenation of the individual ID and the trace number that are written into the entry detail record.

- 2 Next, Payment retrieves the original trace number from the return file, and sets it as the gateway payment ID. In the addenda_return_template.xml, find this code:

```
<traceNumber pos="80" len="15" fmt="N"
target1=' %check.setGatewayPaymentId(txnNumber)%'
target2=' %check.setGatewayPaymentId(stringUtil.concat(payerAcctNumber, "_",
txnNumber))%'></traceNumber>
```

Rename target2= to target, which will reconstruct the gateway payment ID based on the returned customer account number and trace number. Template variable payerAcctNumber has been set in the entryDetail_return_template.xml file and txnNumber has been set before this line in the addenda_return_template.xml file by calling template.putToContext.

- 3 Test this setting using a real return file, and verify that the check's status has been updated to -4 in the EDX_PMT_TRANSACTION table.

Example 2: Modified ACH Trace Number

If the individual ID is not returned as it was set, then you can try to use other information, such as individual name combined with trace number. If only the trace number can be used for gateway payment ID, then use that as follows.

To use only the trace number for gateway payment ID

- 1 At the beginning of the entryDetail_template.xml file, find the following code:

```
%< *>%
%check.setGatewayPaymentId(check.getTxnNumber())%
%< /*>%
```

Remove the comment tags to enable the statement.

- 2 In the addenda_return_template.xml file, rename target1 to target to enable using trace number as gateway payment ID:

```
<traceNumber pos="80" len="15" fmt="N"
target1=' %check.setGatewayPaymentId(txnNumber)%'
target2=' %check.setGatewayPaymentId(stringUtil.concat(payerAcctNumber, "_",
txnNumber))%'></traceNumber>
```


Generating Accounts Receivables (A/R Files)

It is often necessary to synchronize Payment with a biller's A/R software. Payment sends A/R files periodically to a biller's A/R software, which includes the payments being made through Payment. The format of the file varies among billers. To support this function, Payment has the pmtARIntegrator job, which uses a template and XML/XSLT to generate output in a variety of file formats.

The pmtARIntegrator job queries the database to get proper payments, and then writes the payments into a flat file or an XML file using the Payment Template Engine. The XML file can be further transformed into other format by using XSLT.

The default implementation of the pmtARIntegrator job performs the following steps:

- 1 Queries the database to get a list of check or credit card payments, or both. The query is defined in arQuery.xml file, which finds all the check and credit card payments where the payee_id matches the current job DDN, the status is 8 (paid) and flexible_field_3 is N.
- 2 Invokes the process() method of the default implementation of com.edocs.common.payment.tasks.ar.IARPaymentIntegrator, which is com.edocs.common.payment.tasks.ar.SampleARPaymentIntegrator. In this method, ARPaymentIntegrator writes the payments into a flat file or XML file using the Payment Template Engine. There are two templates provided by Payment:
 - **arFlat_template.txt**. Generates a flat A/R file
 - **arXML_template.txt**. Generates an XML file

The output file name is: ar_yyyyMMddHHmmssSSS.extension, where extension matches the extension of the template file.
- 3 Inside the process() method, if the output is an XML file, then SampleARPaymentIntegrator can optionally apply an XSLT file against the output file to transform it into another format. The transformed file name is: ar_trans_yyyyMMddHHmmssSSS.extension, where extension is defined by the pmtARIntegrator job configuration.
- 4 Inside the process() method, SampleARPaymentIntegrator updates flexible_field_3 of both check and credit card payments to Y, and writes that to database. This ensures these payments will not be processed again by the next run of pmtARIntegrator.

Customizing the arQuery.xml File

The SQL queries used by the pmtARIntegrator job are defined in an XML file, arQuery.xml, which is provided by the default Payment installation. The arQuery.xml file is based on Oracle XMLQuery technology.

CAUTION: XMLQuery supports paging, but this feature must not be used for this job.

Most of the A/R file creation is done by an implementation class of the interface com.edocs.common.payment.tasks.ar.IARPaymentIntegrator. This adaptor interface provides maximum flexibility for customizing this job. The default implementation is com.edocs.common.payment.tasks.ar.SampleARPaymentIntegrator.

Before the query is executed in the database, the job invokes the `getMap()` method of `IARPaymentIntegrator`, which gets a list of objects that are used to replace the variables “?” defined in the SQL query of the `arQuery.xml` file. For more information about `IARPaymentIntegrator`, see [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#) for details on accessing *Oracle Billing Insight Javadoc*.

The default `IARPaymentIntegrator` implementation, `SampleARPaymentIntegrator`, uses this `arQuery.xml` file for database query:

```
<?xml version="1.0" encoding="UTF-8"?>
<query-spec>
  <data_source_type>SQL</data_source_type>
  <query name="checkQuery">
    <sql-stmt><![CDATA[select payment_id, payer_account_number, amount, pay_date,
status, payee_id from EDX_PMT_TRANSACTION where payee_id = ? and status = 108 and
arFlag = 'N' and PAYMENT_TYPE='check' ]]></sql-stmt>
    <param name="payee_id" type="java.lang.Integer" position="1"/>
    <!--param name="last_modify_time" type="java.sql.Timestamp" position="2" /-->
  </query>
  <query name="creditCardQuery">
    <sql-stmt><![CDATA[select payment_id, payer_account_number, amount, pay_date,
status, payee_id from EDX_PMT_TRANSACTION where payee_id = ? and status = 108 and
arFlag = 'N' and PAYMENT_TYPE='ccard' ]]></sql-stmt>
    <param name="payee_id" type="java.lang.Integer" position="1"/>
  </query>
```

The following queries are defined:

- **checkQuery.** Queries check payments
- **creditCardQuery.** Queries credit card payments

The `creditCardQuery` gets all the successful payments (`status=8`) of the current payee (biller or DDN of current job) from the relevant Oracle Billing Insight Payment tables. They both use `arFlag` as a flag to prevent a payment from being sent to the A/R job twice. This flag is initially set to N when the payment is created. After the A/R job runs, the `SampleARPaymentIntegrator` changes the flag to Y.

When using `arFlag` as an A/R flag, you can create an index for it to increase performance.

Each of the queries in the `arQuery.xml` file has an SQL variable (“?”) that must be resolved before the query can be sent to the database. The A/R job calls the `getMap()` method of `IARPaymentIntegrator` to get a Map of query variables, and uses their values to replace the “?”s in the query. The names of the Map elements match those defined in the param tags of the query tags.

For example, the default `arQuery.xml` file has the param tag:

```
<param name="payee_id" type="java.lang.Integer" position="1"/>
```

To support this, define a Map element whose name is `payee_id` and whose value (which must be an Integer, and contains the DDN reference number) replaces the question mark (?) with `payee_id` in the query:

```
select payment_id, payer_account_number, amount, pay_date, status, payee_id from
EDX_PMT_TRANSACTION where payee_id = ? and status = 108 and arFlag = 'N' and
PAYMENT_TYPE=' check'
```

The query result set will be transferred to a list of checks (ICheck objects) for checkQuery, credit cards (ICreditCard objects) for creditCardQuery and then pass that list to the process() method of IARPaymentIntegrator.

CAUTION: The XML Query object supports paging, but do not use this feature for A/R query.

You can modify this file to use different queries.

Querying Case Study

The new requirement for this example is to retrieve all payments whose status is returned or paid between 5:00PM today (the job run date) and 5:00PM yesterday (yesterday's job run date).

To try a query case study

- 1 Change the arQuery.xml file for checkQuery:

```
<query name="checkQuery">
<sql-stmt><![CDATA[select * from EDX_PMT_TRANSACTION where payee_id=? and status
in (108, -104) and last_modify_time >= ? and last_modify_time < ? and
PAYMENT_TYPE=' check' ]]> </sql-stmt>
<param name="payee_id" type="java.lang.Integer" position="1"/>
<param name="min_last_modify_time" type="java.sql.Timestamp" position="2"/>
<param name="max_last_modify_time" type="java.sql.Timestamp" position="3"/>
</query>
```

TIP: Use java.sql.Timestamp instead of java.util.Date.

- 2 Change the arQuery.xml file for creditCardQuery. Because you are adding more question marks to the query, override the getMap() method of the default ARPaymentIntegrator:

```
package com.edocs.common.ps.ar;
import java.util.*;
import com.edocs.common.payment.util.DateUtil;
public class MyARIntegrator extends ARPaymentIntegrator
{ /**Override this method to populate the SQL variables in arQuery.xml
*/public Map getMap(ARPaymentIntegratorParams payIntegratorParam,
String objectFlag) throws Exception
{ //call super class because need to get the payee_id value
Map map = super.getMap(payIntegratorParam, objectFlag) //no need to check
objectFlag because we actually populate the //same values for both checkQuery and
creditCardQuery Date today = new Date(); today = DateUtil.dayStart(today); //set
to 00:00:00AM Date today5 = DateUtil.addHours(today, 17); //set to 05:00:00PM
Date yesterday5 = DateUtil.addHours(today, -7) ;//set to 05:00:00PM of yesterday
map.put("min_last_modify_time", DateUtil.toSqlTimestamp(yesterday5));
map.put("max_last_modify_time", DateUtil.toSqlTimestamp(today5))}
```

- 3 To make the cutoff time configurable instead of fixed at 5:00PM, use the flexible configuration fields of the A/R job, which are passed in as part of `ARPaymentIntegratorParams`. For more information about `ARPaymentIntegratorParams`, see [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#) to access the Javadoc.
- 4 Compile your class using the `payment_client.jar` and `payment-1.0-SNAPSHOT.jar` that comes with Oracle Billing Insight, package the compiled class into the payment EAR files, and redeploy the EAR files.
- 5 Log into the Command Center and change the configuration of the A/R job to use the new implementation of the `IARPaymentIntegrator`, `com.edocs.common.ps.ar.MyARIntegrator`.

For more information about using the Command Center, see *Administration Guide for Oracle Billing Insight*.

Customizing the `arFlat_template.txt` File

Payments returned by the `arQuery.xml` file are written to an A/R file using an Oracle Billing Insight Payment template file. Two templates come with Oracle Billing Insight:

- **`arFlat_template.txt`**. Generates a flat A/R file
- **`arXML_template.xml`**. Generates an XML A/R file

The `arFlat_template.txt` file generates a sample flat A/R file. If this file includes most of your required data, but the format is not what you want, then you can edit the template file to generate your own format. For more information about using the Template class, see [“Accessing Oracle Billing Insight Data Dictionary and Javadoc” on page 25](#).

The A/R job using `arFlat_template.txt` does the following:

- Loops through the list of check and credit card payments to print out their details.
- Calculates the totals for check debits, check credits, credit card debits and credit card credits (reversals).

Customizing the `arXML_template.xml` File

The `arXML_template.xml` file generates the same information as `arFlat_template.txt`, but in XML format. After creating the XML file, you can use XSLT to transform it into another XML file or into a flat file. The default `arTransform.xsl` transforms the original XML file into the same format as the one generated by `arFlat_template.txt`. Using XSLT is the recommended way to do the customization.

The A/R job using the `arXML_template.xml` file does the following:

- Loops through the list of check and credit card payments to print out their details.
- Calculates the totals for check debits, check credits, credit card debits and credit card credits (reversals).

To generate different file formats, change `arTransform.xsl`. Or, customize the `arXML_template.xml` file directly.

Customizing the arXML_template.xml File and Using XSLT to Generate an XML Flat AR File

The arXML_template.xml file generates the same information as arFlat_template.txt, but in XML format. After generating the XML file, you can use XSLT to transfer it into another XML file or into a flat file. The default arTransform.xsl transforms the XML file into the same format as the one generated by arFlat_template.txt. XSLT is the recommended way to make the customization.

This template does the following:

- Loops through the list of check and credit card payments to print out their details.
- Calculates the totals for check debits, check credits, credit card debits and credit card credits (reversals).

To generate different file formats, change arTransform.xsl. If required, then you can also customize the arXML_template.xml file.

To rename the generated files

- To rename the files generated by these utilities you must write an implementation of IARPaymentIntegrator. The following code demonstrates how to rename the XSLT output file to another name:

```
import java.io.*;
public class MyARIntegrator extends ARPaymentIntegrator
{protected void getTransformedARFileName(ARPaymentIntegratorParams
payIntegratorParam, ) throws Exception{
return "newARName.txt"; }}
```

Reimplement IARPaymentIntegrator

You might want to reimplement the default SampleARPaymentIntegrator if you want to add any of the following features.

To reimplement the default SampleARPaymentIntegrator

- 1 Rename the default AR files.
- 2 Change the SQL query to add more "?" variables and to set values for those variables in the IARPaymentIntegrator implementation.
- 3 Add any additional steps, such as putting more objects into Template context before it is parsed.
- 4 Change the result of the template parsing. For example, because of limitations of Template engine, sometimes unwanted empty new lines are added. Remove those lines.
- 5 Modify the check or credit card objects before they are updated in the database. By default, only the arFlag field is updated from N to Y. Another alternative is to update the check or credit card object in the template, and all your updates will be updated in the database.

To add any of the preceding features, you must extend from SampleARPaymentIntegrator and configure the pmtARIntegrator job to use your implementation.

You can overwrite following methods for your customization:

- **getARFileName()**. Overwrite to change the name of the AR flat file generated from arFlat_template.txt.
- **getMap()**. Overwrite

Select Only Check or Credit Card Payments

A biller might support only one of check or credit card payments. In this case, you must configure the pmtARIntegrator job to leave the Credit card query name in XML query file field blank. To optionally remove any reference to the unavailable payment type, customize the template files (arFlat_template.txt or arXML_template.xml).

Compiling and Packaging a Custom IARIntegrator

If you reimplement IARIntegrator or you have some custom Java classes to call from the AR template, then you must recompile and package your changes.

In most cases, you put your custom code into payment_custom.jar. Unfortunately, the IARIntegrator and its related classes are packaged as part of ejb-Payment-ar.jar, not payment_custom.jar, so a different procedure is required.

To compile, put ejb-payment-ar.jar along with payment-1.0-SNAPSHOT.jar, payment_custom.jar and payment_client.jar in your class path to reimplement IARIntegrator.

To package, drop all your AR custom classes into the ejb-payment-ar.jar.

A/R Filenames

The generated A/R files have default names of ar_yyyyMMddHHmmssSSS.*template_file_ext*. In the files, *template_file_ext* is the file extension of the template file. The XSLT transformed file has a default name of ar_trans_yyyyMMddHHmmssSSS.*extension*. The extension is defined in the pmtARIntegrator job configuration. You can rename these files to a more meaningful name.

To rename the files, write an implementation of IARPaymentIntegrator. The following code demonstrates how to rename the XSLT output file to another name:

```
package com.edocs.common.ps.ar;
import com.edocs.common.payment.tasks.ar.*;
public class MyARIntegrator extends ARPaymentIntegrator
{/**Override this method to give a new name*/protected void
getTransformedARFileName(ARPaymentIntegratorParams payIntegratorParam, ) throws
Exception
{return "newARName.txt";}}
```

Single Payment Type

A biller might have only ACH and not credit card payments, or conversely. In this case, you can customize the template files (arFlat_template.txt or arXML_template.xml) to remove any references to the unavailable payment type.

Or, when configuring the pmtARIntegrator job enter an empty value for the Check query name in XML query file or Credit card query name in XML query file parameter.

Customizing the Payment Amount Format

You can customize the payment amount format for the following features:

- Credit card registration fee
- Minimum and maximum payment amounts
- Two decimal pattern
- Whether to allow payments greater than the amount due
- Whether to enable the preNote feature

To configure the payment amount format

- 1 Open the payment.xma.xml file, found in the `EDX_HOME\xma\config\modules\payment` directory.
- 2 Modify the parameters in the paymentConfigurationBean section as needed:

```
<bean id="paymentConfigurationBean" class
="com.edocs.common.payment.config.PaymentConfigurationBean" scope="singleton">
  <property name="creditCardRegisterFee">
    <value>1.0</value>
  </property>
  <property name="paymentAmountGreaterThanAmountDue">
    <value>Yes</value>
  </property>
  <property name="minimumPayAmount">
    <value>1.0</value>
  </property>
  <property name="maximumPayAmount">
    <value>99999999.99</value>
  </property>
  <property name="twoDecimalPattern">
    <value>(([0-9]*[G]*)+)[0-9]*[D]?[0-9]{0,2}</value>
  </property>
  <property name="enablePreNote">
    <value>Yes</value>
  </property>
</bean>
```

Packaging Oracle Billing Insight Payment Custom Code

You can package your custom code, both plug-in code and custom A/R jobs and templates, by adding it to payment_custom.jar. The Oracle Billing Insight Payment EAR files will access this JAR, and find the custom code. The Oracle Billing Insight Payment EAR files merge into the Command Center EAR file as part of installation, so your custom code will also be seen by the Command Center.

To make this JAR file accessible by all of the Oracle Billing Insight Payment EJB, JAR and WAR files, place it in the classpath of the MANIFEST file of each JAR and WAR file. For details of how the MANIFEST file works, refer to the J2EE or EJB specifications or the SDK: Customizing and Deploying Applications document that comes with the Command Center SDK. When the EJB JAR or WAR files are loaded, this JAR will be loaded and can be accessed by the EJB JAR files or WAR files.

CAUTION: Never put your custom EJB code into `payment_custom.jar`. Put your EJB code in your own JAR files.

To write a new plug-in for `IAchCheckSubmitPlugIn`

- 1 Write and then compile your implementation class. You might want to use `payment-1.0-SNAPSHOT.jar` and `payment_client.jar` from Oracle Billing Insight Payment as part of your class path.
- 2 Create a JAR file called `payment_custom.jar`, or use the `payment_custom.jar` from any of the Oracle Billing Insight Payment EAR files. Place your implementation class into that JAR file using the JAR command.
- 3 Replace all the `Payment_custom.jar` files under the `lib` directory of all the deployed Oracle Billing Insight Payment EAR files with the new `payment_custom.jar`, using JAR command.
- 4 Deploy the new Oracle Billing Insight Payment EAR files on your application server.
- 5 Go to Payment Settings in the Command Center, and configure the payment gateways to use the new class by replacing the default one, `com.edocs.common.payment.cassette.ach.AchCheckSubmitPlugIn`, with your new plug-in.
For more information about configuring payment gateways, see *Administration Guide for Oracle Billing Insight*.
- 6 Run the `pmtCheckSubmit` job, which will load the new class from `payment_custom.jar`, because you added it to the classpath of the MANIFEST file of `ejb-payment-chksubmit.jar`.

About the APIs for Implementing Custom Payment Cassette Cartridges

You can implement custom payment cassette cartridges with Oracle Billing Insight.

Oracle Billing Insight provides a set of well-defined interfaces for payment gateway integration, and it enables you to develop or customize the cassette according to your payment gateway requirements. You must set up the gateway to communicate with Oracle Billing Insight. You do not have to change anything internal to Oracle Billing Insight.

Figure 20 shows an overview of the interaction between the payment cassette cartridge, Oracle Billing Insight, and the payment gateway.

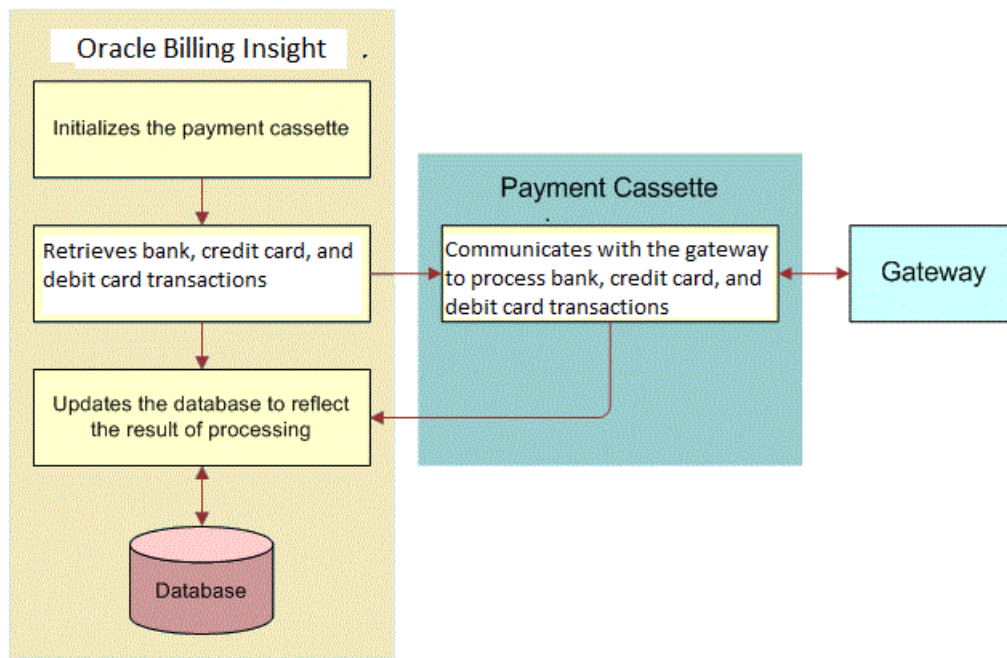


Figure 20. Overview of the Payment Cassette Interaction

All APIs related to the payment cassette and cassette configuration are defined in the `com.edocs.common.api.payment` package. Figure 21 shows the contents of the `com.edocs.common.api.payment` package.

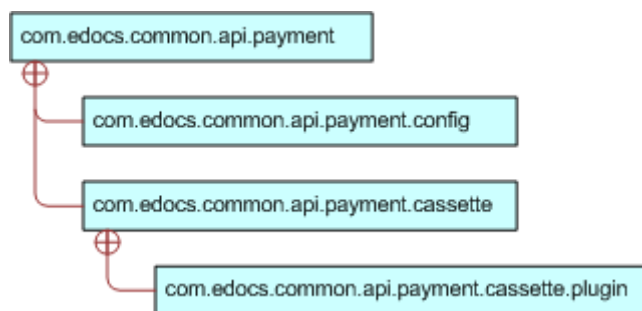


Figure 21. Contents of the `com.edocs.common.api.payment` Package

Figure 22 shows the class structure of the Payment Cassette.

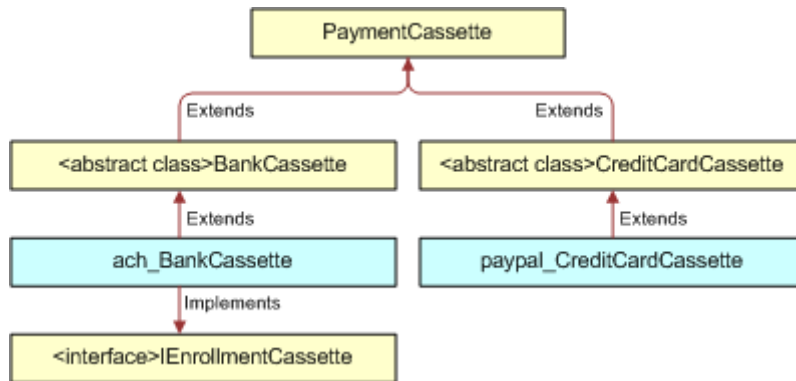


Figure 22. Class Structure of the Payment Cassette

Figure 23 shows the class structure of the Payment Cassette Config.

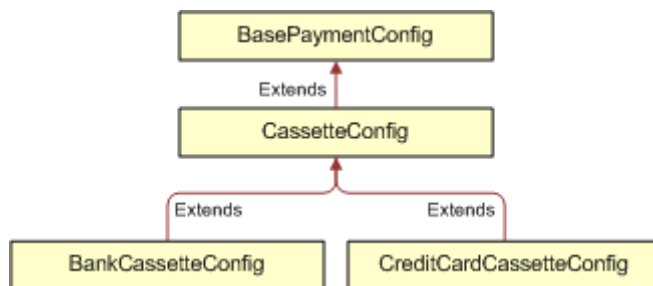


Figure 23. Class Structure of the Payment Cassette Config

Figure 24 shows the class structure of the Payment Cassette extended parameter.

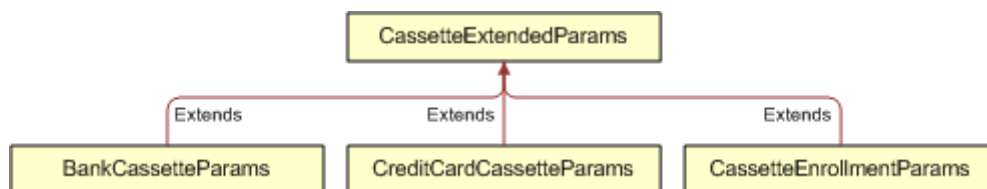


Figure 24. Class Structure of the Payment Cassette Extended Parameter

Implementing Custom Payment Cassette Cartridges

Follow these steps to implement a custom payment cassette cartridge using the Command Center for Oracle Billing Insight.

To implement a custom payment cassette

- 1 Extend your cassette configuration class to add customized gateway attributes:

- **Checks.** Use the BankCassetteConfig class.
- **Credit cards.** Use the CreditCardCassetteConfig class.

If the common configuration parameters defined in the CreditCardCassetteConfig or BankCassetteConfig classes cannot meet your requirements, then implement `CassetteConfig.setDefaultAttributeList()` to define a list of parameters of type `com.edocs.common.payment.cassette.config.Attribute` to configure the cassette. These parameters display on the Payment Settings page in the Command Center. The following example shows an extended CreditCardCassetteConfig class:

```
public class customCassetteConfig extends CreditCardCassetteConfig
{
    public void setDefaultAttributeList()
    {
        Attribute at = new Attribute
        ("Custom files 1", //the display name shown in Payment Settings
        "Custom key1", //the key value of attribute
        Attribute.Text); //the type of attribute
        at.setRequire(true); //flag indicate that whether the attribute is required,
        default is false
        at.addDefaultVAlue("default value"); //set default value
        at.setNeedEncrypt(true); //decide whether the value need to be encrypted when
        stored in db, default is false.
        defaultAttributeList.add(at); // Add Attribute to default attribute list
        at = new Attribute("Custom files2", //the display name shown in Payment Settings
        "Custom key2", //the key value of attribute
        Attribute.Text); //the type of attribute
        at.addDefaultVAlue("default value2");

        defaultAttributeList.add(at); // Add Attribute to default attribute list
    }
}
```

For more information about updating payment settings in the Command Center, see *Administration Guide for Oracle Billing Insight*.

- 2 Extend your payment cassette class to write a customized process flow:
 - **Checks.** Use the BankCassette class.
 - **Credit cards cards.** Use the CreditCardCassette class.
- 3 Set the custom config class in the constructor method of the custom payment cassette class so that the custom config class can be recognized when the payment cassette is created. For example:

```
public class CustomPaymentCassette extends CreditCardCassette
{
    public CustomPaymentCassette()
    {
        this.setCassetteConfig(new CustomCassetteConfig()); //set cassette config
    }
    public ICreditCardTransaction submit(CreditCardCassetteParams
    ccardCassetteParams) throws CassetteException;
    { //TODO implemented by customer }
}
```

```
public List<ICreditCardTransaction> batchSubmit(CreditCardCassetteParams
ccardCassetteParams) throws CassetteException;
{ //TODO implemented by customer}}
```

- 4 If you want to initialize the payment cassette, and you want to get gateway configuration parameters in the Oracle Billing Insight database, you can use the following method in `IPaymentConfigManager`:

```
public CassetteConfig getCassetteConfig(String payeeId, PaymentType
paymentType);
```

- 5 Implement the methods defined in `CreditCardCassette` or `BankCassette`. The parameter classes, such as `CreditCardCassetteParams` and `BankCassetteParams`, provide methods to get check transactions or credit card transactions for processing. The parameter classes also provide Getter and Setter methods to get additional parameters from Command Center jobs or an external system.
- 6 The payment gateway returns a response, which you use to update the status of the `ICheckTransaction` or `ICreditCardTransaction` object as follows:
- If a payment transaction processes successfully, then set the status to Settled-Paid by calling one of the following objects:
 - ❑ `ICreditCardTransaction.setStatus(PaymentStatus.settled)`
 - ❑ `ICheckTransaction.setStatus(PaymentStatus.paid)`
 - If a system or network error occurs, then set the status to Failed by calling one of the following objects:
 - ❑ `ICreditCardTransaction.setStaas(PaymentStatus.failed)`
 - ❑ `ICheckTransaction.setStatus(PaymentStatus.failed)`
 - If the credit card transaction fails authorization, then set the status to Failed-Authorize by calling the `ICreditCardTransaction.setStatus(PaymentStatus.failed_authorize)` object.

For the status types supported by Oracle Billing Insight, refer to the code in the `com.edocs.common.api.payment.PaymentStatus` object.

Refer to the code in the `ICreditCardtransaction` or `ICheckTransaction` objects for setting status and transaction information. You can set the status or other transaction information from the response to the `ICreditCardTransaction` and `ICheckTransaction` objects for your gateway specifications.

The `CheckSubmit` and `CreditCardSubmit` jobs update the Oracle Billing Insight database to reflect the result of processing by the payment cassette.

- 7 You must process the list of `ICreditCardtransaction` or `ICheckTransaction` objects from the `CheckSubmit` or `CreditCardSubmit` job:
- a Set the payment status with the gateway response.
 - b Return the processed transaction list back to the `CheckSubmit` or `CreditCardSubmit` job for the Oracle Billing Insight database update.
- 8 If your payment cassette supports the payment account enrollment process, then implement the `IEnrollmentCassette` interface to enable the enrollment feature.

- 9 Populate the `payment_gateway_configure` table to inform Oracle Billing Insight about your payment cassette implementation class. For example, if your cartridge class name is `com.edocs.common.payment.cassette.creditcard.MyCreditCardCassette`, and you want to name it `customCCardCassette`, then use the following command:

```
Insert into payment_gateway_configure(GATEWAY, PAYMENT_TYPE, CARTRIDGE_CLASS)
values ('customCCardCassette', 'ccard',
       'com.edocs.common.payment.cassette.creditcard.MyCreditCardCassette')
```

Configuring Convenience Fee Payments

Oracle Billing Insight supports charging convenience fees to one-time payments (anonymous, instant, or scheduled) but not on recurring payments. You must create a custom class for calculating convenience fees if the default implementation does not meet your requirements.

The convenience fee amount can be based on:

- A fixed amount.
- A percentage of the payment amount.
- A custom calculation based on rules for your organization.

Two fee transaction models are supported:

- **Single Transaction.** In the single transaction model, you configure one merchant account, which is owned by the merchant. Oracle Billing Insight invoices or automatically debits the merchant monthly for the total fee processed, and the main payment and convenience fee are deposited into the merchant's bank account as one total amount.

The cardholder sees one transaction on his or her statement representing the total amount including the convenience fee. The main payment amount and the fee appear as a single entry on the payment activity report.

- **Dual Transaction.** In the dual transaction model, you configure separate merchant accounts. One account is owned by the merchant and the other is owned by the hosting service provider. The main payment amount is deposited into merchant's bank account and the convenience fee is deposited into service provider account.

Two transactions are processed, and once both transactions have processed successfully, the cardholder sees two transactions on his or her statement. The fee transaction processes first. The main payment amount and the fee appear as a single entry on the payment activity report, with the total amount and fee in separate columns.

Multiple payee accounts are also supported for other business needs. For example, the main payment could go to different accounts based on business logic, or you could direct business and consumer payments to different accounts. You can configure the transaction model and convenience fee type at the payment method level using the custom `IGatewayAccountPlugin` plug-in.

You can use the `EDX_PMT_CONV_FEE_TYPES` table to store the convenience types supported by your organization. Identify the convenience type you want to use for each payment method in the `EDX_PMT_PAYMENT_METHOD` table.

Oracle Billing Insight does not support convenience fees for ACH payments. In the account type database table, Oracle Billing Insight preconfigures the convenience fee type for ACH accounts as None.

To configure convenience fee payments

- 1 Create a custom class for calculating convenience fees in accordance with your organization's business rules if the default implementation does not meet your requirements. The default implementation calculates the convenience fee for each type of payment method based on the data in EDX_PMT_PAYMENT_METHOD table only.
- 2 In the Command Center, specify the following values in the payment gateway configuration to enable convenience fee support for credit cards in the Self-Service and the Agent applications. For details on how to update your payment gateway configuration, see *Administration Guide for Oracle Billing Insight*.

Convenience Fee Parameters for the Credit Card Payment Gateway	Value	Description
Enable Convenience Fee Payments for End User	Y	Enables convenience fee support in the Self-Service application.
Enable Convenience Fee Payments for CSR	Y	Enables convenience fee support in the Customer Service Representative application.
Convenience Fee Configuration Class	The name of your custom class	The name of your custom class for calculating convenience fees. Leave the default value if you are using the default implementation.
Include Convenience Fee in Refunds	Y	Indicates whether to include the fee amount within refund transactions.

- 3 Configure the biller bank accounts in the Command Center.
- 4 If you are implementing more than two billing accounts for use with convenience fees, customize the IGatewayAccountPlugin with the logic that implements your chosen payment model and any business logic based on convenience fee type.

The default implementation calculates the convenience fee for each type of payment method based on the data in the EDX_PMT_PAYMENT_METHOD table only.

In the Command Center, specify the custom plug-in name in the Gateway Account Plugin Implementation parameter on the Global Configuration page.

Handling Multiple Payee ACH Accounts

By default, Oracle Billing Insight only allows one payee (biller) ACH account for a DDN, which is limited by payment settings. However, some billers can have multiple ACH accounts and their users will usually choose to pay to one of the ACH accounts when scheduling a payment. The way that the user chooses the ACH account to pay with can be based on business rules added to the JSP. The rest of this topic describes a solution to this problem.

The assumptions for this solution are:

- All ACH accounts are at the same bank, which means they have the same immediate origination and immediate destination but different company name and company ID.
- The business logic elements required to route the payment transaction to one ACH account compared with another is available or can be made available in the Web application and in the execution context of an Oracle Billing Insight Payment plug-in.

Oracle Billing Insight also assumes there are multiple ACH accounts and there is one DDN for this biller. This DDN is the real DDN.

NOTE: To implement multiple payees for use with convenience fees, see [“Configuring Convenience Fee Payments” on page 277](#).

To handle multiple payee ACH accounts

- 1 Create a real DDN to use to configure payment settings for one of the ACH accounts.
 - 2 Create $N - 1$ virtual DDNs, where each of their payment settings is configured to one of the $N - 1$ ACH accounts, respectively. Make sure the immediate origination and immediate destination are the same for all DDNs, but their company name and company ID are different.
- NOTE:** No load jobs run against these virtual DDNs. They are used solely for payment purposes.
- 3 Customize the UI with business logic to determine which DDN (effectively, ACH account) the payment transaction is to be entered against and set the payee ID of the payment to that DDN.
 - 4 Configure a single pmtCheckSubmit job under the real DDN and configure it to pull payments from the all the $N - 1$ virtual DDNs in addition to the real DDN. The payments from the same DDN will be under same batch.
 - 5 Because return files include returns from all DDNs and the pmtCheckUpdate job can process these returns, create one pmtCheckUpdate job under the real DDN to process all the returned transactions (even though the returns could belong to other virtual DDNs).
 - 6 A single recurring payment job configured with the real DDN is required. A Recurring Payment plug-in is required to execute the same logic as in scheduled payment. That is, apply the business rules to determine which DDN (effectively, ACH account) the recurring payment must be applied against. Override the preSchedulePayment() method of the plug-in for this purpose.
 - 7 In the Payment pmtPaymentReminder Job settings, configure six payment reminders, one for each DDN.

- 8 Run the pmtARIntegrator Job. The AR_Query.xml file is an XML definition of the database query that queries the Oracle Billing Insight payment tables to build the default A/R file. Customize the default query to include the virtual DDNs. Because the query uses the DDN reference numbers, you must pass that information into the query using one of the following methods:
 - Directly hard code the DDN reference numbers in the query, though this is risky in the sense that if the DDN is recreated, then your query will fail.
 - Extend the SampleARIntegrator and overwrite the getMap() method and use com.edocs.payment.util.DDNUtil to find out the DDN reference number of a DDN, then set it as a "?" parameter used by the query. In this solution, the DDN names are hard coded but not the DDN reference numbers.
 - Pass in the names of virtual DDNs as a flexible job configuration parameter from the job UI. The getMap() method can then parse the parameter to get the list of virtual DDNs. This method is recommended.
- 9 If you are using the ACH prenote, then you must create pmtSubmitEnroll, pmtConfirmEnroll, and pmtNotifyEnroll jobs for each virtual DDN, which means you will get N prenote ACH files. The pmtSubmitEnroll job cannot aggregate prenotes from different DDNs into one.

Using Payment APIs

Use the following APIs to customize Payment. These are part of the com.edocs.common.api.services.payment package:

- BillIDepot
- CustomRecurringPaymentPlugin
- ICreditCardPlugin
- DummyUserAccountAccessor
- IPayment
- IPaymentAccountService
- IPaymentService
- IRecurringPaymentService
- Payment
- PaymentAccountService
- PaymentConfigurationBean
- PaymentService
- RecurringPaymentService
- IConvenienceFeeConfig
- IGatewayAccountPlugin
- ITopUpPaymentPlugin

Configuring International Bank Routing

Oracle Billing Insight supports ACH gateways with US routing number standards as the default. To provide your customers with the option to make payments using international bank accounts, configure the check gateway specifications for the particular country.

To configure international bank routing

1 Implement the following custom classes:

- **Check cassette class.** For the country's specific check gateway standard, including properties for communicating with the gateway, replace the default implementation in `\com\edocs\common\payment\cassette\bank\ach\ach_BankCassette.class`.
- **ACH check class.** For extending the standard Check class, including properties coming from different check gateway standards. This class is used to generate files sent to the check gateway for authentication or check transaction purposes. Replace the default implementation in `\com\edocs\common\payment\cassette\bank\ach\core\AchCheckTransaction.class`.
- **Returned check class.** For processing the returned check file for the gateway. Replace the default implementation in `\com\edocs\common\payment\cassette\bank\ach\core\AchReturnedCheck.class`.

Replace the default files with your custom class files in the `payment_custom.jar` file in the following directories. For Windows, change the slashes and root as necessary.

- `EDX_HOME\J2EEApps\commandcenter\weblogic\command-center-weblogic-7.1.ear\lib\payment_custom.jar`
- `EDX_HOME\J2EEApps\sel fservice\weblogic\sel fservice-weblogic-7.1.ear\lib\payment_custom.jar`
- `EDX_HOME\J2EEApps\csr\weblogic\csr-app-7.1.ear\lib\payment_custom.jar`
- `EDX_HOME\payment\lib\payment_custom.jar`

2 Implement a custom RoutingNumber.class. This class is used to validate routing numbers when creating a new check account. This file is located in the following directory. For Windows, change the slashes and root as necessary.

`EDX_HOME/J2EEApps/sel fservice/weblogic/sel fservice-weblogic-7.1.ear/sel fservice-web-1.0-SNAPSHOT.war/WEB-INF/lib/app-common-1.0-SNAPSHOT.jar/com/edocs/appl ication/common/payment/util/RoutingNumber.class`

3 In the validation-payment.xml file, edit the validation rules for the routingNumber field property. This file is located in the following directory:

`EDX_HOME/J2EEApps/sel fservice/weblogic/sel fservice-weblogic-7.1.ear/sel fservice-web-1.0-SNAPSHOT.war/WEB-INF/classes/validators.xml`

4 Customize your templates files for generating the files sent to the gateway or for parsing the returned file from the gateway.

Customizing Anonymous Payment Validation

You can customize the validation of anonymous payments for your payment validation requirements.

The Payment service invokes the anonymousPaymentValidator implementation to verify an anonymous payment. Figure 25 shows the anonymous payment validation.

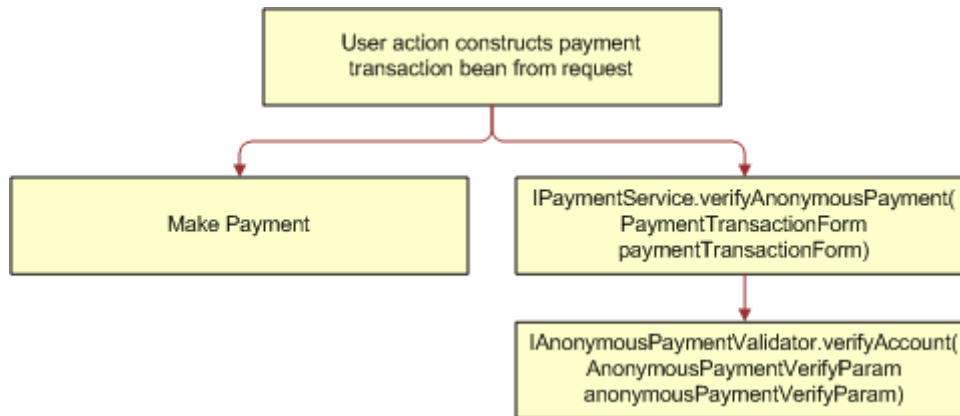


Figure 25. Anonymous Payment Validation

You customize the anonymousPaymentValidator implementation by configuring the payment service XMA file and implementing the anonymous payment API.

To customize anonymous payment validation

- 1 Open the paymentService.xma.xml file in the following directory. In the path, *EDX_HOME* is the directory where you installed Oracle Billing Insight.

- **UNIX.** *EDX_HOME*\xma\config\modules\services
- **Windows.** *EDX_HOME*/xma/config/modules/services

- 2 Replace the anonymousPaymentValidator Javabeen content.

If you have a Telco implementation, then replace the following content:

```
<bean id="anonymousPaymentValidator"
class="com.edocs.common.services.payment.DefaultAnonymousPaymentValidator"
scope="singleton"></bean>
```

If you have a Utility implementation, then replace the following Javabeen content:

```
<bean id="anonymousPaymentValidator"
class="com.edocs.common.services.payment.DefaultUtilityAnonymousPaymentValidator"
scope="singleton"> </bean>
```

- 3 Implement the `com.edocs.common.api.services.IAnonymousPaymentValidator` interface. This interface provides the method `verifyAccount(AnonymousPaymentVerifyParam anonymousPaymentVerifyParam)`. By default, the method checks whether the account and service ID are enrolled.

You can use the following code to validate an anonymous payment:

```
IPaymentService pmtService = EBillingServiceFactory.getPaymentService();
anonymousPmtVerifyFlag = pmtService.verifyAnonymousPayment(pmtFormbean);
```

Updating the Back-End Billing System With a Top-Up Payment

Users can make a top-up payment against a prepay billing account. When a top-up payment is made, the account balance updates with the newly paid amount. You must notify the back-end service billing system that the service has been paid and credits need to be updated to allow the customer to continue using the service. Oracle Billing Insight provides the `ITopUpPaymentPlugin` plug-in for updating a back office system. You must implement the plug-in based on your own business rules.

To update a back-end billing system with a top-up payment using ITopUpPaymentPlugin

- 1 Open the `services.xma.xml` file, located in the following directory. In the path, `EDX_HOME` is the directory where you installed Oracle Billing Insight.

- **UNIX.** `EDX_HOME/xma/confi g/modul es/servi ces`
- **Windows.** `EDX_HOME/xma/confi g/modul es/servi ces`

- 2 Add your implementation class for `ITopUpPaymentPlugin` as a JavaBean definition, such as:

```
<bean id="customerTopUpPaymentPlugin" class="com.edocs.common.payment.plugin.
CustomerTopUpPaymentPlugin" scope="singleton"></bean>
```

- 3 Replace the `IPaymentServiceTarget` Javabeen content with the following code:

```
<bean id="IPaymentServiceTarget"
class="com.edocs.common.services.payment.PaymentService" scope="prototype">
  <property name="accountValidator">
    <ref bean="anonymousPaymentValidator"/>
  </property>
  <property name="topUpPaymentPlugin">
    <ref bean="customerTopUpPaymentPlugin"/>
  </property>
</bean>
```

- 4 Implement the `com.edocs.common.api.payment.ITopUpPaymentPlugin` interface. This interface provides the method `postTopUpPayment(IPaymentTransaction pmtTransaction)`.
- 5 Add the implemented class into the `payment_custom.jar` file, located in the following directories. For Windows, change the slashes and root as necessary.
 - `EDX_HOME\J2EEApps\commandcenter\weblogi c\command-center-weblogi c-7.1.ear\lib`

- *EDX_HOME\J2EEApps\sel fservi ce\web\ogi c\sel fservi ce-web\ogi c-7. 1. ear\I i b*
- *EDX_HOME\J2EEApps\csr\web\ogi c\csr-app-7. 1. ear\I i b*
- *EDX_HOME\payment\I i b*

10 Customizing the Payment Consolidator Integration Module

This chapter covers information about how to customize the payment consolidator integration module for your implementation. It includes the following topics:

- [Overview of Processing with a Payment Consolidator on page 285](#)
- [Process of Implementing a Payment Consolidator on page 290](#)
- [Creating Customized Payment Consolidation Templates on page 290](#)
- [Setting Payment Consolidation Configuration Parameters on page 308](#)
- [Customizing the Bill Detail URL on page 309](#)
- [Implementing Payment Consolidator Plug-Ins on page 310](#)
- [Payment Consolidator Java APIs on page 313](#)
- [Default File Formats for Payment Consolidation on page 314](#)
- [Data Element Mappings - Example A on page 331](#)
- [Data Element Mappings - Example B on page 342](#)
- [Payment Consolidation Database Tables on page 351](#)

Overview of Processing with a Payment Consolidator

Oracle Billing Insight provides a file interface, including business logic and a data model, that you can use to interact with any payment consolidator. Each payment consolidator has different standards for integrating its software. Open API standards let you configure Oracle Billing Insight to provide and accept data files from any consolidator.

At a consolidator's site, users can enroll, view bill summary information and pay bills, update account details, and unenroll. To view bill details, a user can click a link to Oracle Billing Insight and register there to view bill details.

The consolidator passes enrollment information to the biller for validation and authentication. Once the biller authorizes the accounts to view and pay bills through the consolidator, the biller sends this information to Oracle Billing Insight, which sends bill summary information for all active and authorized accounts to the consolidator.

Data Flow with a Payment Consolidator

The creation and exchange of data between Oracle Billing Insight and a payment consolidator follow this general flow:

- 1 Consumer users register with the payment consolidator, and the consolidator sends the enrollment data to the billing system.
- 2 The biller validates the registration requests and responds to the consolidator, which sends a response acknowledgement to the biller and to Oracle Billing Insight.
- 3 In the Command Center, the PCAccountEnrollment job is scheduled to run. The job uses an XML template which you have customized for use with the particular consolidator to read the consolidator's enrollment response acknowledgement file and map it to the corresponding Java objects in Oracle Billing Insight. The job then processes this input and stores the enrollment information in the relevant database tables in Oracle Billing Insight.

The Account Enrollment job also produces its own enrollment response file which has the details on the accounts that were successfully processed. The format of the Oracle Billing Insight enrollment response file is also controlled using an output template that you customize. The output template is also an XML file which has instructions on constructing and formatting the output file.

- 4 Oracle Billing Insight receives bills from the billing system.
- 5 The PCBillSummary job is scheduled to run in the Command Center, and generates a file with the bill summary information for the enrolled users. The output content and file format is dictated by the template associated with the job.
- 6 The payment consolidator processes the Bill Summary file and returns an acknowledgement file to the biller.
- 7 The PCBillsSummaryAcknowledgement is scheduled to process the Bill Summary Acknowledgement file sent by the consolidator using the relevant consolidator template. This job also stores information about which records were accepted or rejected by the consolidator into Oracle Billing Insight database.

Figure 26 shows an overview of the payment consolidator integration architecture.

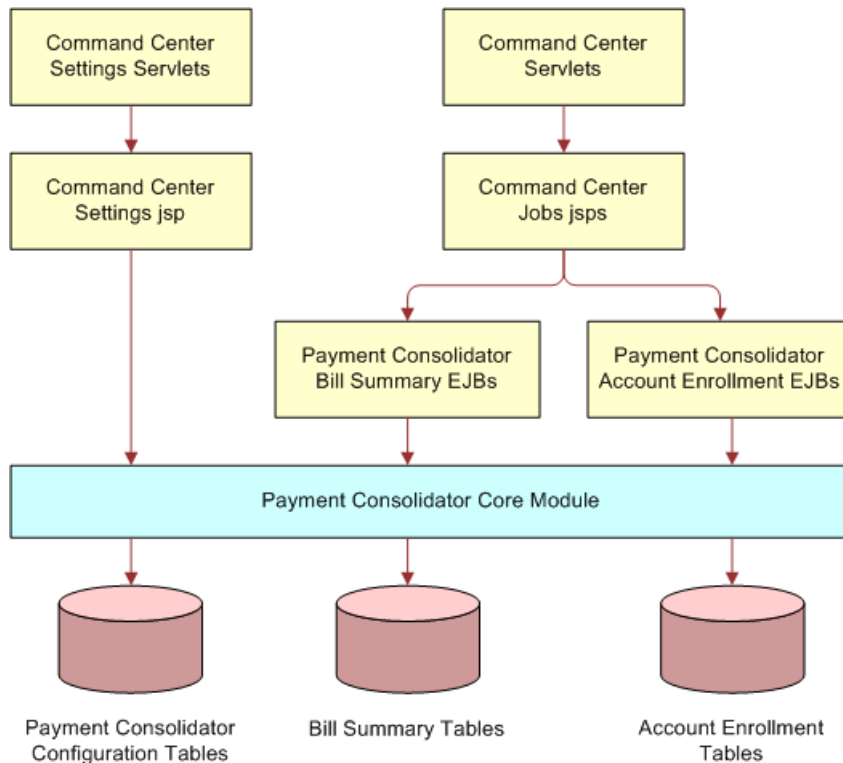


Figure 26. Payment Consolidator Integration Architecture

Payment Consolidation Jobs

The following Command Center production jobs process the data files exchanged with your payment consolidator using templates customized for use in processing your consolidator's files. You must configure and schedule these jobs to run in accordance with your billing cycle:

- **PCAccountEnrollment Job.** The Account Enrollment job processes the Enrollment Data File provided by a payment consolidator to bring account information into Oracle Billing Insight for consumer customers who successfully enrolled or unenrolled to use a payment consolidator, or who changed their account information. The PCAccountEnrollment job can produce an enrollment response file with details on the accounts that were processed successfully.
- **PCBillSummary Job.** The Bill Summary job generates a file with bill summary information for consumer users who have successfully enrolled to use a payment consolidator with a particular biller. This file is provided to the payment consolidator.
- **PCBillSummaryAcknowledgement Job.** The Bill Summary Acknowledgement job processes the bill summary acknowledgement (confirmation) file received from a payment consolidator, updating the summary records accepted and rejected by the consolidator into the Oracle Billing Insight database.

About Payment Consolidator Job Notifications

If one or more errors occur when a payment consolidator job runs in the Command Center, then Oracle Billing Insight sends an email notification to all administrators (or the administrators configured to receive notifications if an alert profile is configured for payment notification jobs). The email contains information about the errors and a link to the corresponding log file. For details about payment consolidator log files, see [“About Payment Consolidator Job Logs” on page 288](#). For details about the content of the email notifications sent by payment consolidator jobs, see [Chapter 6, “Using and Customizing Email and SMS Notifications.”](#)

You can configure alerts to narrow the list of administrators who receive email notifications for individual payment consolidator jobs. For details on configuring alert profiles for payment consolidator jobs, see *Administration Guide for Oracle Billing Insight*.

About Payment Consolidator Job Logs

The payment consolidator jobs generate the following log files that indicate whether the records were processed successfully:

- **PCAccountEnrollment Job.** The AccountEnrollment and EnrollmentResponse tasks generate separate log files in this job. If account information is rejected by the biller or consolidator, or if errors are generated by either the AccountEnrollment or EnrollmentResponse task, then the error information is recorded in a log.
- **PCBillSummary Job.** This job generates one log file.
- **PCBillSummaryAcknowledgement Job.** This job generates one log file.

The log files also capture information about individual records that were rejected. [Table 78](#) describes the content written to the log files based on the payment consolidator job results.

Table 78. Payment Consolidator Log Content

Job Result	Description	Log File Content
Successful	No records failed while processing.	The following message appears in the log file: All records were processed successfully.
Partial Failure	Some of the records failed to process or were rejected by the external system (the payment consolidator).	The following information about failed records appears in the log file: The following accounts are not processed successfully: AccountNumber, ConsolidatorName, BillerName, BillingSystemId, StatusCode, and StatusMessage

Table 78. Payment Consolidator Log Content

Job Result	Description	Log File Content
All Rejected	If the status code in the file header is not a successful code, then all records were rejected.	The following message appears in the log file: All records are rejected by biller/consolidator.
No Statement	The bill summary was not generated because there is no billing information in Oracle Billing Insight.	The following message appears in the log file: No Statements available for the criteria specified.

For details on configuring the log file location for the payment consolidator jobs, see [“Implementing Payment Consolidator Plug-Ins” on page 310](#).

Payment Consolidation Input and Output Files

A summary of the input and output files processed by the jobs are listed in [Table 79](#).

Table 79. Payment Consolidation Job Input and Output Files

Command Center Job	Input File	Output File
PCAccountEnrollment	The account enrollment data file received from the payment consolidator. This file contains information about users who have enrolled, unenrolled, or changed their information with the consolidator.	(Optional) The enrollment response file provided to the consolidator. This file provides information about accounts in the enrollment input file that were successfully processed by Oracle Billing Insight.
PCBillSummary	(Optional) If you do not want to generate the bill summary information for all accounts, then you can create an input CSV file listing the biller and account IDs and specify the file name in the job configuration.	The bill summary file provided to the payment consolidator. This file is viewed by an enrolled user at the consolidator’s site and or as an PDF attachment in an email notification.
PCBillSummary Acknowledgement	The bill summary acknowledgement file received from the payment consolidator.	A CSV file that lists all the billing accounts and related billing system IDs that were rejected by the payment consolidator.

For additional details about what these jobs do and how to configure and run the jobs, see *Administration Guide for Oracle Billing Insight*.

The Payment Consolidation Template Engine

The Payment Consolidation Integration module provides a generic template processing mechanism based on Java reflection. The templates specify rules for reading data elements from the input file into members of certain Java objects or write values of Java object members to certain data elements in the output file.

Preconfigured templates containing business logic are provided for use with the Oracle Billing Insight jobs to parse the input data files and generate output files. You must customize the preconfigured templates for the data file requirements of your particular payment consolidator.

Process of Implementing a Payment Consolidator

Each payment consolidator has its own file formats, data requirements, and standards. You must implement each payment consolidator individually for use with Oracle Billing Insight.

To implement a payment consolidator, perform the following tasks:

- 1 [“Creating Customized Payment Consolidation Templates” on page 290](#)
- 2 (Optional) [“Setting Payment Consolidation Configuration Parameters” on page 308](#)
- 3 (Optional) [“Customizing the Bill Detail URL” on page 309](#)
- 4 (Optional) [“Implementing Payment Consolidator Plug-Ins” on page 310](#)
- 5 Configure the biller, payment consolidator, and the payment consolidation jobs in the Command Center. For details, see *Administration Guide for Oracle Billing Insight*.

Creating Customized Payment Consolidation Templates

A set of custom template files is required for each type of input and output file processed or generated by the payment consolidation jobs in the Command Center. The payment consolidator jobs use the templates to parse the data input files received from a particular payment consolidator (either batch or real-time) and to structure the output files provided to the consolidator.

You must customize the sample templates provided with Oracle Billing Insight to accommodate your consolidator’s requirements. The two sets of preconfigured sample files closely describe the structure of common payment consolidator formats.

Each type of input and output file that a payment consolidator uses requires a set of custom XML templates that includes:

- **A file structure XML template.** The file structured XML template defines the structure of the individual records in the input or output file and maps to each of the individual record XML template files designed for each record.

- **An XML template file for each record type in the file.** Custom template XML files are required for each type of record, including the file header, file tail, detail record, and batch header and tail records if supported by the consolidator. Record XML template files are used to parse input file records or generate output file records.

This task is step in [“Process of Implementing a Payment Consolidator” on page 290](#).

To create customized payment consolidator templates

- 1 Review the sample XML templates provided with Oracle Billing Insight to determine which ones most closely match the input and output data files required by your payment consolidator. For information about the sample template files provided with Oracle Billing Insight, see [“Sample Payment Consolidation Template Files” on page 291](#). If your consolidator supports batch processing, then use the batch input file templates.

The default input and output data file formats that the API is preconfigured to use are described in [“Default File Formats for Payment Consolidation” on page 314](#). For samples of how a payment consolidator’s data file elements can map to Oracle Billing Insight, see the following topics:

- [“Data Element Mappings - Example A” on page 331](#)
 - [“Data Element Mappings - Example B” on page 342](#)
- 2 Using the closest matching set of files (or copies), edit the content for your consolidator’s specifications. See to the following topics as needed for information about the content and format of template XML files:
 - [“The File Structure XML Template” on page 293](#)
 - [“The Record XML Templates” on page 295](#)
 - [“Template Attributes” on page 299](#)
 - [“Template Variables” on page 300](#)
 - [“Adding a New Java Object Instance or Static Class” on page 305](#)
 - [“Adding Cartridge Property Keys” on page 306](#)
 - 3 Place the templates in the template directory for the payment consolidator that you configured for the consolidator in the Command Center.

Sample Payment Consolidation Template Files

The preconfigured sample template XML files provided with Oracle Billing Insight contain a common extraction of data that a payment consolidator can require. These files provide a starting point for you to create customized files that meet your payment consolidator’s data and format requirements. Oracle Billing Insight provides two sets of sample templates for customization.

Sample Templates - Set 1

The following set of sample XML templates files is in the `EDX_HOME\conf\paymentconsolidator\templates\sampl e1` directory.

This set of sample templates contain examples for a consolidator that does not support batch processing. It comes with a sample cartridge property file called `CartridgeProperty_sample1.properties`.

Account Enrollment (Input) File

The following sample files provide an example of templates that could be used by the `PCAccountEnrollment` job to parse the account enrollment input file provided by the payment consolidator:

- `AccountEnrollment_2060_sample1.xml`
- `AccountEnrollment_2060_fileHeader_template.xml`
- `AccountEnrollment_2060_entryDetail_template.xml`
- `AccountEnrollment_2060_fileTailer_template.xml`

Bill Summary (Output) File

The following sample files provide an example of templates that could be used by the `PCBillSummary` job to generate the bill summary output file:

- `BillSummary_sample1.xml`
- `BillSummary_fileHeader_template.xml`
- `BillSummary_entryDetail_template.xml`
- `BillSummary_fileTailer_template.xml`

Bill Summary Acknowledgement (Input) File

The following sample files provide an example of templates that could be used by the `PCBillSummaryAcknowledgement` job to parse the bill summary acknowledgement file provided by the payment consolidator:

- `BillSummary_ack_sample1.xml`
- `BillSummary_ack_fileHeader_template.xml`
- `BillSummary_ack_entryDetail_template.xml`
- `BillSummary_ack_fileTailer_template.xml`

Sample Templates - Set 2

This set of sample XML templates files is in the `EDX_HOME\config\paymentconsolidator\templates\sample2` directory.

This set of sample templates contain examples for a consolidator that supports batch processing. It comes with a sample cartridge property file called `CartridgeProperty_sample2.properties`.

Account Enrollment (Input) File

The following sample files provide an example of templates that could be used by the PCAccountEnrollment job to parse the account enrollment input file provided by the payment consolidator:

- AccountEnrollment_sample2.xml
- AccountEnrollment_fileHeader_template.xml
- AccountEnrollment_batchHeader_template.xml
- AccountEnrollment_entryDetail_template.xml
- AccountEnrollment_batchTailer_template.xml
- AccountEnrollment_fileTailer_template.xml

Bill Summary (Output) File

The following sample files provide an example of templates that could be used by the PCBillSummary job to generate the bill summary output file:

- BillSummary_sample2.xml
- BillSummary_fileHeader_template.xml
- BillSummary_batchHeader_template.xml
- BillSummary_entryDetail_template.xml
- BillSummary_batchControl_template.xml
- BillSummary_fileControl_template.xml

Bill Summary Acknowledgement (Input) File

The following sample files provide an example of templates that could be used by the PCBillSummaryAcknowledgement job to parse the bill summary acknowledgement file provided by the payment consolidator:

- BillSummary_ack_sample2.xml
- BillSummary_ack_fileHeader_template.xml
- BillSummary_ack_batchHeader_template.xml
- BillSummary_ack_entryDetail_template.xml
- BillSummary_ack_batchControl_template.xml
- BillSummary_ack_fileControl_template.xml

The File Structure XML Template

The structure XML template file maps to the set of record XML template files that a particular type of input or output file requires. Each type of input and output file requires a file structure XML template file as part of the template set required by the associated payment consolidation job.

The sample AccountEnrollment_2060_sample1.xml file, found in the *EDX_HOME\config\paymentconsolidator\templates\sample1* directory, provides a simple example of a file structure XML template:

```
<?xml version="1.0"?>
<Enrollment>
$cartridgeProperty.setProperty("com.edocs.common.pc.detailrecord_class",
"com.edocs.common.pc.core.EnrollmentDetailRecord")$
$<INCLUDE>$
$templateDir$AccountEnrollment_fileHeader_template.xml
$</INCLUDE>$
$<INCLUDE>$
$templateDir$AccountEnrollment_entryDetail_template.xml
$</INCLUDE>$<INCLUDE>$
$templateDir$AccountEnrollment_fileTailor_template.xml
$</INCLUDE>$
</Enrollment>
```

If the payment consolidator supports batch processing, then the file structure XML file must also include a batch header and batch tailer template XML files enclosing the detail record XML file. For an example of a file structure XML template that includes batch records, see the AccountEnrollment_sample2.xml file in the *EDX_HOME\config\paymentconsolidator\templates\sample2* directory.

Components of the File Structure XML Template File

The file structure XML template file requires the following components:

- **A statement using the INCLUDE operator to map each record template XML file.** The `<INCLUDE>` operator identifies the individual template files required to process the particular record types for the consolidator's input file. For example, the sample file structure template AccountEnrollment_2060_sample1.xml includes the following record templates:
 - The `$templateDir$AccountEnrollment_fileHeader_template.xml` template for parsing the FileHeader record.
 - The `$templateDir$AccountEnrollment_entryDetail_template.xml` template for parsing the Detail record.
 - The `$templateDir$AccountEnrollment_fileTailer_template.xml` template for parsing the FileTailer record.
 - **The XML Delimiter.** Use the dollar sign (\$) as the delimiter to distinguish between the file structure definition XML file and each record template XML file, as well as the cartridge property file.
- NOTE:** In the record XML template, the delimiter is a percentage sign (%).
- **The Cartridge Property Statement.** An XML statement for specifying the detail record implementation is required in the file structure XML (except for the bill summary file), for example:

```
$cartridgeProperty.setProperty("com.edocs.common.pc.detailrecord_class",  
"com.edocs.common.pc.core.EnrollmentDetailRecord")$
```

The name of the `java.util.Properties` instance is `cartridgeProperty`, which is preset in the template context. Its method `setProperty` is invoked to add the value `com.edocs.common.pc.core.EnrollmentDetailRecord` to the key `com.edocs.common.pc.detailrecord_class`. The value of this key is used to parse the detail record, and represents the full path of the detail record implementation class. The `cartridgeProperty` contains a set of key-value pairs that are used to parse an input file or generate an output file. The preconfigured cartridge properties are defined in a properties file located in the same directory as the sample files.

Oracle Billing Insight provides a default detail record implementation for the enrollment detail and bill summary records:

- Enrollment detail record class: `com.edocs.common.pc.core.EnrollmentDetailRecord`
- Bill Summary detail record class: `com.edocs.common.pc.core.BillSummaryRecord`

The default implementation class is designed with a complete collection of the fields required by most consolidators for the account enrollment detail record and the bill summary record. It is recommended to use the default implementation. Flexible fields are also reserved for your customization.

If you use a customized implementation of a detail record, then you must replace the key value of `"com.edocs.common.pc.detailrecord_class"` in file structure XML. For example, if the customized class for enrollment is `com.businesshome.detailrecord.enrollment`, then the required statement in the file structure XML file is:

```
$cartridgeProperty.setProperty("com.edocs.common.pc.detailrecord_class", "  
com.businesshome.detailrecord.enrollment")$
```

The customized class must implement interface `"com.edocs.common.api.pc.IDetailRecord"` and extend the parent class `com.edocs.common.pc.core.BaseRecord`.

The Record XML Templates

The record XML templates contain instructions for the payment consolidation jobs to read the data in the input file and map it to the Java objects in Oracle Billing Insight. Similarly, individual record XML templates are used by the payment consolidation jobs to generate output files with the data required by the consolidator.

For each type of input and output file required by your consolidator, you must create a separate record XML template file for each part of the file record:

- **File header record.** Marks the beginning of the file and provides the file creation and identification information.
- **File tail record.** Marks the end of the file and always contains statistical information such as how many types of records are in the file and the total number of each.

- **Detail record.** Detail records appear between the file header and tail records and contain information on each account, depending on the purpose of the file. It could be account enrollment, account deactivation, account update or bill summary.
- **(If supported) Batch header and tail records.** If the payment consolidator supports batch processing, then the detail records in the input file will be grouped by batch record. Batch header is stored at the beginning of a batch and the tail record goes at the end of a batch and enclose detail records.

Figure 27 shows the common file layout, consisting of a file header, file tail, and detail records.

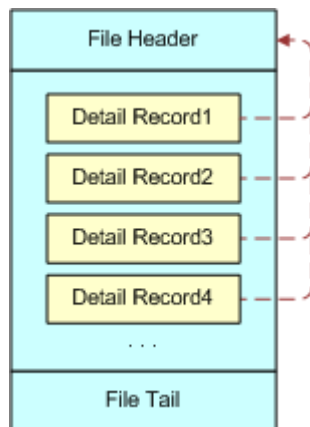


Figure 27. Common File Layout for Payment Consolidation

Figure 28 shows the common file layout for a payment consolidator that uses batch processing.

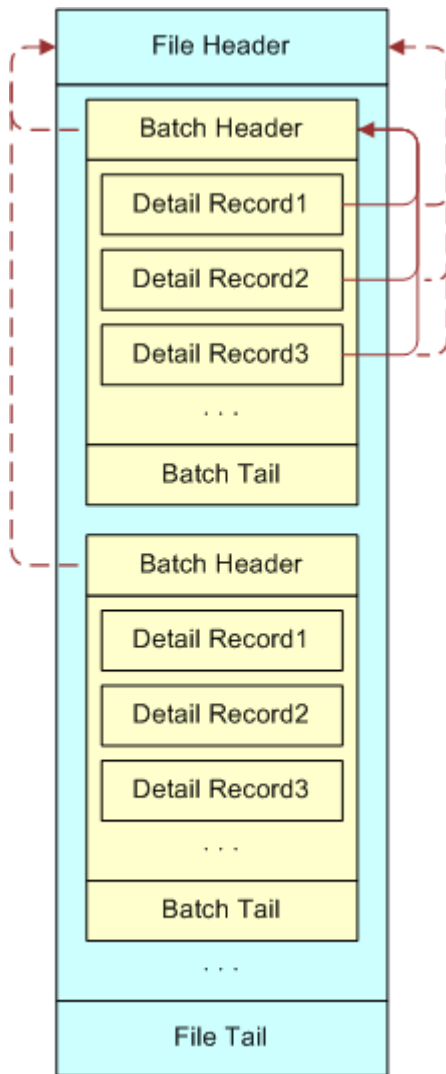


Figure 28. Common File Layout for Payment Consolidation With Batch Processing

Components of the Record XML Template Files

The record XML template files require the following components:

- **A root element name.** Use the record prefix plus your consolidator's record type code as the root element name, as shown in the following table.

Record Type	Record Prefix	Example of a Consolidator's Record Type Code	Root Element Name
File Header	PC_FileHeader_	0000	PC_FileHeader_0000
Batch Header	PC_BatchHeader_	1111	PC_BatchHeader_1111
Detail	PC_Detail_	2050	PC_Detail_2050
Batch Trailer	PC_BatchTrailer_	2222	PC_BatchTrailer_2222
File Trailer	PC_FileTrailer_	9999	PC_FileTrailer_9999

- **The <record> element.** Enclose all data element XML tags between <record> and </record>, for example:

```
<PC_Detail_2050>
<record><recordType pos="1" len="1" required = "Y"
target='%detailRecord.setRecordType(?)%'></recordType>
<billerID pos="2" len="16" fmt="AN" required = "N"
target='%detailRecord.setBillerExternalID(?)%'></billerID></record>
</PC_Detail_2050>
```

- **A data element for the record type.** The record XML template must include a subelement called recordType within the <record> element.
- **Delimiters.** The template engine only executes template statements that are enclosed in a pair of delimiters. In the record template XML file the delimiter is a percentage sign (%).

NOTE: In the file structure XML template, the delimiter is a dollar sign (\$).

- **Record element tags.** For input files, each template XML file must specify:
 - Where the data text is located in the input file using the pos and len attributes.
 - (Optional) How to transform the data by specifying fmt and fract attributes.
 - Whether the data field is mandatory by specifying the required attributes.
 - Which class member to use for storing the data field using the target attributes.

When parsing an input data file such as the account enrollment file, the job reads all data elements of a line in the input file into certain class members according to the format specified by the template XML tags. The generic format of a template XML tag is:

```
<billerID pos="104" len="32" fmt="AN" required = "N"
target=' %detailRecord.setBillerExternalID(?)%' ></billerID>
```

This tag retrieves the text from position 104 to 136, puts it into the member of class called detailRecord by invoking the method detailRecord.setBillerExternalID(?) defined by the target to set the biller external ID. The template executes the statement specified by the element attribute target enclosed between percentage signs (%).

For an output file, the rules in the record template XML file describing how to generate the output file fields are also defined by the XML elements, for example:

```
<billerID pos="49" len="32" fmt="AN"
required="Y">%detailRecord.getBillerExternalID()%</billerID>
```

This tag writes the billerID field to the output file. The start position in line is 49 and the length of the field is 32. If the length of value is less than 32, then padding characters are used to fill the field. The value is retrieved by invoking detailRecord.getBillerExternalID(). For more information about using attributes, see ["Template Attributes" on page 299](#). For information about variables you can use in the template XML files, see ["Template Variables" on page 300](#).

Template Attributes

Attributes in the template XML files describe the format of each type of record, such as position, length and so on. Each element in a template file corresponds with one of the payment consolidator's data elements. For example, the element <recordType .. /> represents the record type and the element <ConsolidatorID .. /> represents the consolidator ID. The name of element can have any value but it should be meaningful so that it can be easily understood.

Table 80 shows the attribute names you can use in the template XML files.

Table 80. Attributes Used in the Payment Consolidator Templates

Attribute Name	Value	Description
pos	int	The data element start position in the data file.
len	int	The fixed data element value length.
fmt	N: number AN: String date: dateFormat timestamp: timestampFormat time: timeFormat	The type of date and time element and the format. Some payment consolidators require a maximum of 12 numbers before the decimal and two numbers after the decimal, without a decimal point: \$\$\$\$\$\$\$\$\$\$cc.
fract	A digital number.	The number of digits after decimal point if the fmt attribute is N.

Table 80. Attributes Used in the Payment Consolidator Templates

Attribute Name	Value	Description
required	Y - Yes or N - No	Indicates whether the field is mandatory. If the attribute is Y, then when parsed or generating a file, the job checks whether the value is not null. If it is null, then this record fails to parse. If the field value could be all spaces, then you must set the required attribute to N. Otherwise, mandatory field checking will fail as all spaces are treated as a null value.
target	The invoked statement enclosed between the percentage sign (%), such as: %detailRecord.getConsolidatorID()%	Used only in template files for the input data file. When parsing this attribute, the template engine invokes the corresponding method or gets the value of the property for certain Java objects (you can get the Java object from the template context by name).

Template Variables

The template engine hosts a pool of objects in its context in the form of a hash table. You can refer to the variables in that context by their names, between a pair of percentage sign (%) delimiters. The delimiter character distinguishes between common text messages and special placeholders, which are processed by the template engine to replace the values of the Java objects. For example, there's an IFileRecord object with the name fileHeaderRecord in the template context, and you can refer to the object using %fileRecord%.

To escape %, use %%. For example, %%40 means %40.

Oracle Billing Insight provides some preconfigured variables you can use in the XML template files.

Some variables can be used in the record template files only. See [Table 82 on page 301](#) for a list.

You can use the variables described in [Table 81](#) in either file structured or record XML template files.

Table 81. Variables for All XML Template Files

Name	Type	Description
cartridgeProperty	java.util.Properties	Contains all of the properties configured in the cartridge properties file. The cartridge properties file maps different values with the same key for a payment consolidator. You can define your own properties which can be used directly in template files by invoking the setProperty method of object cartridgeProperty, for example: \$cartridgeProperty.setProperty("newKey", "newValue")\$
template	com.edocs.payment.util.template	The template object used to parse the template file.
templateDir	java.lang.String	The directory where the template files are located for a payment consolidator. You set this value in the Command Center when setting up the payment consolidator.
dateUtil	com.edocs.common.pc.util.DateUtil	The utility class for processing data of Date type.
stringUtil	com.edocs.payment.util.StringUtil	The utility class for processing data of String type. This variable makes calling the static methods of StringUtil easier. Use StringUtil.concat("a","b","c")% instead of %com.edocs.payment.util.StringUtil.concat("a","b","c")%
decimalUtil	com.edocs.payment.util.DecimalUtil	The utility class for processing data of Decimal type.

[Table 82](#) shows the variables that you can use in record XML templates only.

Table 82. Variables for Record XML Templates Only

Record Template	Variable Name	Type	Description
FileHeader	fileHeaderRecord	com.edocs.common.pc.core.FileRecord	Sets or gets file header record information.
BatchHeader	batchHeaderRecord	com.edocs.common.pc.core.BatchRecord	Sets or gets batch header record information.
Detail	detailRecord	com.edocs.common.pc.core.DetailRecord	Sets or gets detail record information.

Table 82. Variables for Record XML Templates Only

Record Template	Variable Name	Type	Description
Detail	flexibleFields	Java.util.HashMap	A property member of the detail record object which is used to set or get extended field information.
BatchTailer	batchTailerRecord	com.edocs.common.pc.core.BatchRecord	Sets or gets batch tailer record information.
FileTailer	fileTailerRecord	com.edocs.common.pc.core.FileRecord	Sets or gets file tailer record information.

Logic Operators

The Payment Consolidator Integration module supports the simple logic operators shown in [Table 83](#). You must enclose operators between percentage signs (%).

Table 83. Logic Operators Available in the Payment Consolidator Integration Module

Operator	Function	Usage	Parameters	Example
FOREACH	Performs a loop operation.	<pre>%<FOREACH var in list>% ... %</FOREACH>%</pre>	<p>Parameters you can use with this operator are:</p> <ul style="list-style-type: none"> ■ var. The temporary variable for storing the Java object. The parameter can be any value, but must be the same as name of variable in the following operation. ■ list. The name of the Java object, which is preset into template context before invoking the FOREACH operator. 	<pre>%<FOREACH fileRecObj in fileRecordList>% The send id is: %fileRecObj. getSenderID()% %</FOREACH>% The send ID of the FileRecord instance in the fileReocrdLi st is the output.</pre>
IF	Performs a conditional operation.	<pre>%<IF var >% ... %</IF>%</pre>	<p>The var parameter is the name of a variable that is preset into template context before invoking the IF operator.</p>	<pre>%<IF outputFlag>% The value of flag is true %</IF>% If the value of outputFlag is true, then the following statement is written to the file: The value of flag is true.</pre>

Table 83. Logic Operators Available in the Payment Consolidator Integration Module

Operator	Function	Usage	Parameters	Example
INCLUDE	Includes other template files in the current template file.	<pre>%<I NCLUDE>% ... %</I NCLUDE>%</pre>	None	<pre>%<I NCLUDE>% D: \templ ate\templ at eB. xml "%</I NCLUDE>% The context of the templateB.xml file is included in the parent template.</pre>
< * >	Adds comments to the template file.	<pre>%< * >% ... %</ * >%</pre>	None	<pre>%< * >% This is only the comments here. %</ * >% The following statement is written to the template file: This is only the comments here.</pre>

Accessing an Object's Public Fields and Methods

You can access an object's public fields and methods. The valid reference is:

```
%name. fi el d%, %name. method(param1, param2, ...)%
```

Each parameter of the method can be one of the following:

```
name, name. fi el d, or name. method(param1, param2, ...)
```

The number of parameters is unlimited and an arbitrary level of method nesting is allowed. *Nesting* means that a method's return value is used as a parameter when calling another method. For example, if there are two objects in contexts, buf (a StringBuffer), and str (a String), then the following references are valid:

```
%buf%, %buf. append(str)%, %buf. append(str. toStri ng())%.
```

You can access a static field or method directly without instantiating an object, for example:

```
%j ava. l ang. l nteger. MI N_ VALU E%
```

Or

```
%j ava. l ang. l nteger. parse l nt(" 12. 34")%
```

You must preset all variables by calling putToContext on the template class. The Payment Consolidator Integration module also provides some preconfigured variables that you can use directly. For a list, see ["Template Variables" on page 300](#).

In addition to preconfigured variables, you can put the Java object into context by writing:

```
%template.putToContext(String, Object)%
```

For example:

```
%template.putToContext("buf", new java.lang.StringBuffer())%
```

This puts a new `StringBuffer` object called `buf` into the template context. You can then refer to this object by its name: `%buf.append("abc")`.

Adding a New Java Object Instance or Static Class

In output file templates, you can add any Java objects into the template context for future use by invoking the `putToContext(String, Object)` method of the template class.

For input template files, you can add new Java objects to the template context in the file structure XML template only.

To add a new Java object instance or static class

- For input template files, add the new Java object as follows in the file structure definition XML file:

```
$template.putToContext(String, Object)$
```

For output template files, add the new Java object as follows:

- In the file structure definition XML file:

```
$template.putToContext(String, Object)$
```

- In the record template XML file:

```
%template.putToContext(String, Object)%
```

For example, for a new class called `com.business.core.PmtConsolidatorTest` in the file structure XML, use:

```
$template.putToContext("pmtConsolidator", new  
com.business.core.PmtConsolidatorTest(parameter1, parameter2))$
```

To add a static class, use:

```
$template.putToContext("pmtConsolidatorStatic",  
com.business.core.PmtConsolidatorStatic)$
```

You can use these objects by name in both the file structure XML and the record template files. (Add the object into the template context first.)

The Payment Consolidator Integration engine has the following limitations:

- Math operations are not allowed. You must call a Java method for math operations.
- Concatenate method calls, such as `%variable.method().method() %`, are not allowed. You must write your own Java method to do method concatenation.

Adding Cartridge Property Keys

Oracle Billing Insight provides some preconfigured cartridge property keys in the cartridgeProperty object. The name of the Java object that is preset in the template context is cartridgeProperty. Oracle Billing Insight uses the properties along with the templates to parse the input data files and generate the response files.

You can add any additional cartridge properties required by your payment consolidator for input data files. If you add the properties in the cartridgeProperty object for the file structure XML template, then it can be accessed by all templates. If you add it to a particular record XML template, then it is only used within that template.

Table 84 shows the cartridge property keys preconfigured the cartridgeProperty object.

Table 84. Preconfigured Cartridge Property Keys

Property Key Name	Description
com.edocs.common.pc.actiontype.enroll	The action type code to enroll or create a new account.
com.edocs.common.pc.actiontype.change	The action type code to change information on an existing account.
com.edocs.common.pc.actiontype.unenroll	The action type code to deregister an existing account. If it is not supported by the payment consolidator, then leave this key blank.
com.edocs.common.pc.statuscode.filerecord.success	The successful status code defined by the payment consolidator in the file header record.
com.edocs.common.pc.statuscode.detailrecord.success	The successful status code defined by the payment consolidator in the detail record.
com.edocs.common.pc.statuscode.biller_not_exist	The status code defined by payment consolidator indicating that the biller does not exist in the payment consolidator.
com.edocs.common.pc.statusmessage.biller_not_exist	The status message defined by payment consolidator indicating that the biller does not exist in the payment consolidator.
com.edocs.common.pc.statuscode.consolidator_not_exist	The status code indicating that the consolidator does not exist in the payment consolidator.
com.edocs.common.pc.statusmessage.consolidator_not_exist	The status message indicating that the consolidator does not exist in the payment consolidator.
com.edocs.common.pc.statuscode.billingsystemid_not_exist	The status code indicating that the billing system ID does not exist in the payment consolidator.
com.edocs.common.pc.statusmessage.billingsystemid_not_exist	The status message indicating that the billing system ID does not exist in the payment consolidator.

Table 84. Preconfigured Cartridge Property Keys

Property Key Name	Description
com.edocs.common.pc.statuscode.account_not_exist	The status code indicating that the active account does not exist in the payment consolidator.
com.edocs.common.pc.statusmessage.account_not_exist	The status message indicating that the active account does not exist in Payment Consolidator.
com.edocs.common.pc.statuscode.account_not_activated	The status code indicating that the account is not activated in the payment consolidator.
com.edocs.common.pc.statusmessage.account_not_activated	The status message indicating that the account is not activated in the payment consolidator.
com.edocs.common.pc.statuscode.account_already_exist	The status code indicating that the account already exists in the payment consolidator.
com.edocs.common.pc.statusmessage.account_already_exist	The status message indicating that the account already exists in the payment consolidator.
com.edocs.common.pc.statuscode.mandatory_filed_missing	The status code indicating that a mandatory field is missing in the payment consolidator.
com.edocs.common.pc.statusmessage.mandatory_filed_missing	The status message indicating that a mandatory field is missing in the payment consolidator.
com.edocs.common.pc.initial_status.biller_enrollment	The initial biller enrollment status indicating whether the record is enrolled successfully with the biller. The value can be Y - Yes or N - No.
com.edocs.common.pc.initial_status.consolidator_enrollment	The initial consolidator enrollment status indicating whether the record is enrolled successfully with the consolidator. The value can be Y - Yes or N - No.
com.edocs.common.pc.end_of_file_symbol	The end-of-line symbol while generating the output file. The default value is \r\n.
com.edocs.common.pc.billing_account_creator	If the newly enrolled account doesn't exist in Oracle Billing Insight, then it creates a new billing account. The value of this key is the operator ID specifying who created the billing account and is stored in the CREATEDBY column in the database.

To add a new property to the cartridgeProperty object

- 1 Use one of the following:

In the file structure definition XML file:

```
$cartridgeProperty.setProperty(String KeyName, String KeyVal ue)$
```

In the record XML template file:

```
%cartridgeProperty.setProperty(String KeyName, String KeyVal ue)%
```

To reference the property values in all template files, use the following command:

```
$cartridgeProperty.getProperty(String KeyName)$
```

Setting Payment Consolidation Configuration Parameters

You can optionally specify the following parameters in the paymentConsolidator.xma.xml file when integrating a payment consolidator with Oracle Billing Insight:

- **Log File Path.** The directory where the log files generated by the payment consolidator jobs are stored. The default log file location is the `EDX_HOME\Data\appl i cati onName\j obName` directory (or the `EDX_HOME/Data/appl i cati onName/j obName` directory in Windows).
- **Batch Size.** The total number of records within a batch when the PCBillSummary job generates the bill summary file.
- **Trunk Size.** The total number of accounts that PCBillSummary, PCBillSummaryAcknowledgement, and PCAccountEnrollment jobs can process at a time. This parameter impacts job performance. If the trunk size is too small, then it increases the database access and file write times. If the trunk size is too large, then it can use too much memory. Tune this parameter value for the target environment.
- **Statement Period.** You can change the statement period parameter name used in the bill detail URL for the Bill Summary. For details on customizing the bill detail URL, see [“Setting Payment Consolidation Configuration Parameters” on page 308](#).

This task is step in [“Process of Implementing a Payment Consolidator” on page 290](#).

To configure payment consolidation parameters

- 1 Open the paymentConsolidator.xma.xml file found in the `EDX_HOME\xma\confi g\modul es\paymentConsol i dator` directory (or the `EDX_HOME/xma/confi g/modul es/paymentConsol i dator` directory in Windows).
- 2 Edit the XML for the properties you want to set:
 - To specify a log file path for each of the payment consolidator tasks, edit the configBean section for the logFilePath property.
 - To specify batch size for the PCBillSummary job, edit the bi l l SummaryCartri dge l mp bean section and specify a value for the batchSize property.
 - To specify trunk size for the PCBillSummary, PCBillSummaryAcknowledgement, or PCAccountEnrollment jobs, edit the bi l l SummaryCartri dge l mp, bi l l SummaryAckCartri dge l mp, and enrollResponseCartri dge l mp bean sections and specify a value for the trunkSize property.

Customizing the Bill Detail URL

The PCBillSummary job generates a URL that links to the bill detail. This link is provided to consumer users after logging in to the payment consolidator's site and viewing the statement summary or when viewing the bill summary PDF attachment. The user must log into Oracle Billing Insight to view the bill detail. The default bill detail URL is

```
https://hostname:selfserviceAppPort/selfservice/
login.action?redirectTo=statementSummary&statementPeriodStart=XXX
```

You can:

- Customize the entire bill detail URL.
- Let users view the bill detail without logging into Oracle Billing Insight.
- Change the statement start period.

This task is step in ["Process of Implementing a Payment Consolidator" on page 290](#)

To customize the entire bill detail URL

- To change the entire URL, implement the `com.edocs.common.api.pc.ICartridgePlugin` plug-in for the bill summary file. In the method `preProcessDetailRecord(IDetailRecord record)`, invoke the method `IBillSummaryRecord.setBillURL()` to compose and overwrite the URL.

CAUTION: The following portion of the URL is configured when setting up a biller in the Command Center. Do not edit this section directly: `https://hostname:selfserviceAppPort/selfservice/login.action?redirectTo=statementSummary`.

To let customers view the bill detail without logging in to Oracle Billing Insight

- Use the two flexible fields, `token1` and `token2`, provided with the bill summary record. These fields are the unique IDs for each bill summary and are stored in the `EDX_PC_BILLSUMMARY_LOG` file for later user authentication.

The fields can be accessed using `com.edocs.common.api.pc.IBillSummaryRecord`. You can set it at the implementation of

`com.edocs.common.api.pc.ICartridgePlugin.preProcessDetailRecord(IDetailRecord record)`. The field values are null by default.

To customize the statement start period

- 1 Open the `paymentConsolidator.xma.xml` file found in the `EDX_HOME\xma\config\modules\paymentConsolidator` directory (or the `EDX_HOME\xma\config\modules\paymentConsolidator` directory in Windows).
- 2 Edit the XML in the `billSummaryCartridgeImp` bean section for the `statementPeriodParamName` property.

Implementing Payment Consolidator Plug-Ins

The Oracle Billing Insight provides the following plug-ins for extending Payment Consolidator functionality:

- IBillingSystemIDPlugin
- ICartridgePlugin

For each plug-in, Oracle Billing Insight provides a default implementation.

This task is step in [“Process of Implementing a Payment Consolidator”](#) on page 290.

Implementing the IBillingSystemIDPlugin

Only one method is defined in IBillingSystemIDPlugin:

```
public String getBillingSystemID(IDetailRecord record) throws ProcessException
```

This method is called while processing the detail record. It is used to get the billing system ID from Oracle Billing Insight, which is used in combination with the account number to uniquely identify an account. Each billing system has been assigned a unique identifier called the billing system ID in Oracle Billing Insight. When enrolling a new account or changing an account number, this method is invoked to get the billing system ID according to the detail record information, and is then combined with the account number to check whether the account exists in Oracle Billing Insight. If the account does not exist, then it creates a new account with the billing system ID and account number.

Figure 29 shows the data flow for IBillingSystemIDPlugin.

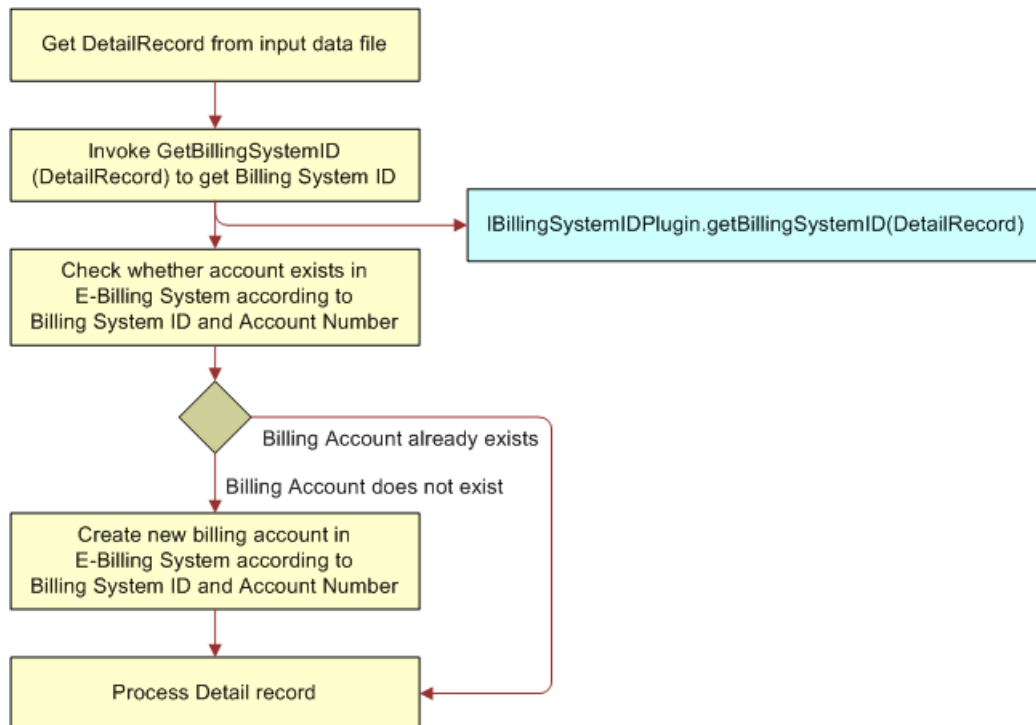


Figure 29. Data Flow for the IBillingSystemIDPlugin

The default implementation is `com.edocs.common.pc.core.DefaultBillingSystemIDPlugin`. You might have to implement the plug-in using your business rules to get the real billing system ID. The default plug-in works well only when there is single billing system ID in Oracle Billing Insight.

To implement the IBillingSystemIDPlugin plug-in

- Set the Billing System ID Plug-In parameter in the biller's configuration in the Command Center. For details on configuring a biller, see *Administration Guide for Oracle Billing Insight*.

Implementing the ICartridgePlugin Plug-In

The ICartridgePlugin plug-in defines a group of methods that are invoked before a payment consolidation job can populate a record in the Oracle Billing Insight database or write to the response file in order to overwrite the default value of a record's field and change the behavior.

Figure 30 shows the workflow for the ICartridgePlugin.

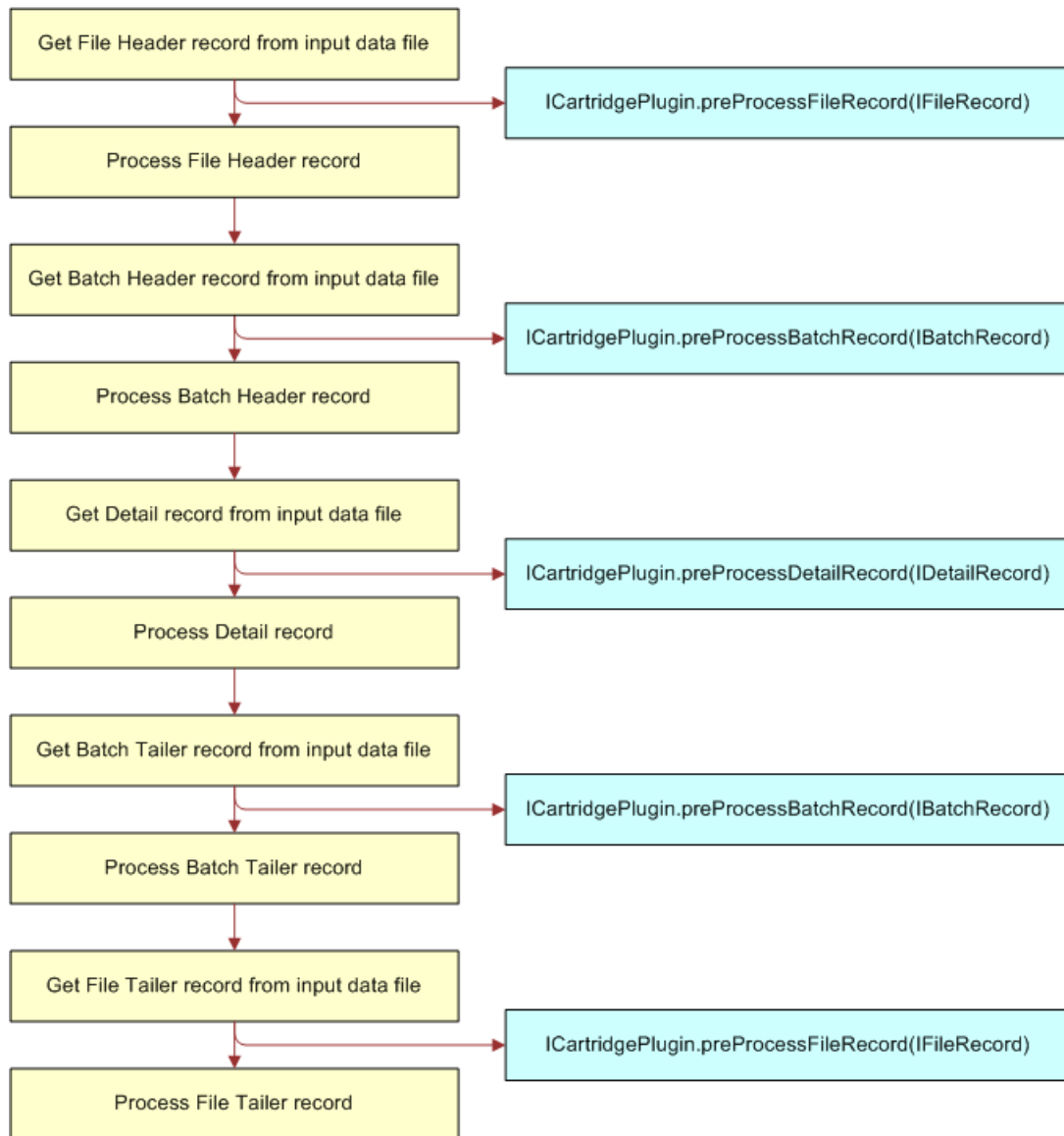


Figure 30. Data Flow for the ICartridgePlugin

The default implementation is `com.edocs.common.pc.core.cartridge.DummyCartridgePlugin`, and is used for all payment consolidation file types. It is a dummy implementation and does nothing with the record. You can customize the default implementation for each file type.

You can implement the following methods in `ICartridgePlugin`:

- **public IFileRecord preProcessFileRecord(IFileRecord record)**. Use to add customized operations before processing the file header or file trailer record, such as for a customized field.

The ReceiverID field in the file header is required and is used to generate the enrollment response file. If the ReceiverID field is missing, then the enrollment response file generation will fail. For some payment consolidators, there is no ReceiverID field doesn't exist in the file header. You must implement preProcessFileRecord in ICartridgePlugin to set the value of the receiver ID for the fileHeaderRecord object.
- **public IBatchRecord preProcessBatchRecord(IBatchRecord record)**. Use to add customized operations before processing the batch header or trailer records, such as for a customized field.
- **public IDetailRecord preProcessDetailRecord(IDetailRecord record)**. Use to add customized operations before processing the detail record, such as for a customized field. You can also use this method to communicate with an external system for verifying the detail record, such as invoking billing system to verify whether the detail record is valid on the biller's side.

To implement the ICartridgePlugin plug-in

- Set the Cartridge Plug-in parameter for the particular file type in the payment consolidator's configuration in the Command Center. For details on configuring a payment consolidator, see *Administration Guide for Oracle Billing Insight*.

Payment Consolidator Java APIs

Oracle Billing Insight provides the interfaces described in [Table 85](#) to present each type of file record.

Table 85. APIs for Payment Consolidator File Records

API	Description
com.edocs.common.api.pc. IFileRecord	This API represents the file record and contains the header and tail information. The default implementation is: com.edocs.common.pc.core.FileRecord
com.edocs.common.api.pc. IBatchRecord	This API represents the batch record and contains the header and tail information. For each batch record, it has a reference to IFileRecord. The default implementation is: com.edocs.common.pc.core.BatchRecord
com.edocs.common.api.pc. IDetailRecord	This API represents the detail record and is the base interface of IBillSummaryRecord and IEnrollmentRecord. For each detail record, it has a reference to IFileRecord and IBatchRecord. The default implementation is: com.edocs.common.pc.core.DetailRecord

Table 85. APIs for Payment Consolidator File Records

API	Description
com.edocs.common.api.pc.IBillSummaryRecord	<p>This API represents an electronic bill summary for each account and is used to generate the bill summary file or confirm the consolidator's bill summary acknowledgement. It is derived from com.edocs.common.api.pc.IDetailRecord.</p> <p>The default implementation is:</p> <p>com.edocs.common.pc.core.BillSummaryRecord</p>
com.edocs.common.api.pc.IEnrollmentRecord	<p>This API represents a biller account and is used to activate, deactivate, and update a biller account. It is derived from com.edocs.common.api.pc.IDetailRecord.</p> <p>The default implementation is:</p> <p>com.edocs.common.pc.core.EnrollmentRecord</p>

Default File Formats for Payment Consolidation

This topic describes the default input and output file data and formats that Oracle Billing Insight is preconfigured to use. These files represent an extraction of data common to many payment consolidators. You must customize and configure your implementation to work with the specific data required by your payment consolidator.

The Payment Consolidation Module is preconfigured for the following input and output files as described in this section:

- Account Enrollment (Input)
- Enrollment Acknowledgement (Output)
- Bill Summary (Output)
- Bill Summary Acknowledgement (Input)

Account Enrollment Input File Format

This topic describes the default format of the account enrollment input file that Oracle Billing Insight is preconfigured to use.

Table 86 shows the header data elements in default format of the account enrollment input file.

Table 86. Header Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
Header Record Type	4	Num	No	The header record type, which is the same as the input file.
Sender ID	32	Varchar	Yes	The unique ID for the entity sending the request.
Receiver ID	32	Varchar	Yes	The unique ID for the entity receiving the request.
Sender Name	32	Varchar	No	The name of the entity sending the request.
File ID	4	Varchar	Yes	The file identifier.
File Creation Date	8	Date	Yes	The file creation date, in YYYYMMDD format.
File Creation Time	12	Time	Yes	The file creation time, in HHMMSSmmmmmm format (uses microseconds).
Time Zone Offset	3	Varchar	No	The time zone offset, such as +03, -05, and so on.
Status Code	5	Varchar	No	The error code associated with the header.
Status Message	80	Varchar	No	The error message associated with the header.
Custom Field 1	32	Varchar	No	For future use.
Custom Field 2	32	Varchar	No	For future use.

Table 87 shows the batch header data elements in the default format of the account enrollment input file.

Table 87. Batch Header Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
Batch Record Type	4	Num	No	Part of the batch header.
Biller ID	32	Varchar	No	Part of the batch header.
Biller Name	32	Varchar	No	Part of the batch header.
Batch ID	7	Num	No	Part of the batch header.
Custom Field 1	32	Varchar	No	Part of the batch header.
Custom Field 2	32	Varchar	No	Part of the batch header.

Table 88 shows the body detail record data elements in the default format of the account enrollment input file.

Table 88. Body Detail Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
Detail Record Type	4	Num	No	A number indicating this is the detail record type.
Action Type	10	Varchar	Yes	The type of action. The values can be Register - Enrolling a user, Change - Changing account details, or Deregister - Deregistering a user.
Action Description	23	Varchar	No	A description of the action type.
Sponsor ID	32	Varchar	No	A unique ID for the sponsor, sent by the consolidator.
Consolidator User ID	32	Varchar	No	A unique ID for the service user, sent by the consolidator.
Biller User ID	32	Varchar	No	A unique ID for the service user, sent by the biller.
Consolidator ID	32	Varchar	Yes	The name of the consolidator.
Consolidator Transaction ID	32	Varchar	No	A unique transaction ID provided by the consolidator.
Biller Transaction ID	32	Varchar	No	A unique transaction ID provided by the biller.

Table 88. Body Detail Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
Biller ID	32	Varchar	Yes	A unique ID for the biller, implementation-specific.
Biller Name	32	Varchar	No	The biller's name.
Biller Address Line 1	32	Varchar	No	The biller's remittance address line 1.
Biller Address Line 2	32	Varchar	No	The biller's remittance address line 2.
Biller City	32	Varchar	No	The biller's city.
Biller State	2	Varchar	No	The biller's state.
Biller ZipCode	5	Varchar	No	The biller's five-digit postal code.
Biller Zip + 4	4	Varchar	No	The biller's four-digit postal code extension.
Billing System ID	32	Varchar	No	The billing system ID used in combination with account number to uniquely identify an account.
Account Number	32	Varchar	Yes	The user's account number.
New Account Number	32	Varchar	No	The new user account number, populated only for requests to change the account number.
Bill Processing Center ID	9	Varchar	No	The ID of the bill processing center that processes bills for this account. This maps to the Biller ID column in Oracle Billing Insight used to identify the billing system.
User Type	1	Char	No	The type of user: I- Individual or B- Business Retail.
Business Name	40	Varchar	No	The name of the business as known to the biller, populated only if the user type is B- Business Retail.
User Last Name	32	Varchar	No	The user's last name.
User First Name	32	Varchar	No	The user's first name.
User Middle Name	32	Varchar	No	The user's middle name.
User Email Address	100	Varchar	No	The user's email ID.
User Address Line 1	40	Varchar	No	Line one of the address where service is provided.

Table 88. Body Detail Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
User Address Line 2	40	Varchar	No	Line two of the address where service is provided.
User Address Line 3	40	Varchar	No	Line three of the address where service is provided.
User City	32	Varchar	No	The user's city.
User State	2	Varchar	No	The user's state.
User Zip 5	5	Varchar	No	The user's five-digit postal code.
User Zip 4	4	Varchar	No	The user's four-digit postal code extension.
User Zip 2	2	Varchar	No	The user's two-digit postal code extension.
User Country Code	3	Varchar	No	The user's country code.
User Postal Code	11	Varchar	No	The user's postal code.
User Province Name	30	Varchar	No	The user's province.
User Day Phone Number	32	Varchar	No	The user's day phone number.
User Evening Phone Number	32	Varchar	No	The user's evening phone number.
Bill Payment Activated	1	Char	No	Indicates whether the bill payment service is activated, typically Y or N.
Service Verification Field 1	32	Varchar	No	First field used by the biller to verify service activation.
Service Verification Field 2	32	Varchar	No	Second field used by the biller to verify service activation.
Service Verification Field 3	32	Varchar	No	Third field used by the biller to verify service activation.
Service Verification Field 4	32	Varchar	No	Fourth field used by the biller to verify service activation.
Service Verification Field 5	32	Varchar	No	Fifth field used by the biller to verify service activation.
Service Verification Field 6	32	Varchar	No	Sixth field used by the biller to verify service activation.

Table 88. Body Detail Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
Alternate Biller Account Number	32	Varchar	Yes	The user's internal account number as maintained by the biller. This can be the same as the User Account Number.
New Alternate Biller Account Number	32	Varchar	No	The new internal account number provided in case the existing internal account number is changed. Used only when the Action Type is Change.
Server Transaction Timestamp	20	Varchar	No	The time stamp assigned to the original service activation request.
Auto Activation Flag	1	Char	No	Identifies whether the subscriber's account was activated through a sponsor-initiated marketing program that did not require the explicit approval of the subscriber. The value is A if the account was auto-activated or S if the account was self-activated by subscriber.
Paper Question Status	1	Char	No	The state of the attempt to get the subscriber to agree to paper suppression.
Paper Status Code	1	Char	No	Indicates the state of paper bill or E-bill delivery.
Paper Suppression Date	20	Date Time	No	The date when paper suppression starts.
Incentive Message ID	4	Varchar	No	The identifier of the incentive message currently active for the account. If the biller does not support incentive messages, then this field contains zeroes.
Bill Remittance Account Number	32	Varchar	No	The account number to which any remittance transactions from the biller are made. If this is blank, then remittance is made to the account number
Consolidator Enrollment Status	1	Char	No	The status of the consolidator's enrollment. The default value is blank.
Biller Enrollment Status	1	Char	No	The status of the biller's enrollment. The default value is blank.

Table 88. Body Detail Data Elements for the Default Account Enrollment Input File

Data Element	Length	Type	Required by the API	Description
Deactivation Source	1	Char	No	Identifies whether the account was deactivated as a result of subscriber action or because the trial period ended.
E-Bill Start Date	20	Date Time	No	The date when next the next E-Bill can be expected
Response Code	1	Char	No	The response to the service activation request sent to the biller by the consolidator. Typical values are A and R.
Rejected Text	255	Varchar	No	If a service activation request is rejected, then this field has the rejection reasons.
Status Code	5	Varchar	No	The Success or Failure status code.
Status Message	80	Varchar	No	The description of the failure codes.
Custom Field 1	255	Varchar	No	For future use.
Custom Field 2	255	Varchar	No	For future use.
Custom Field 3	255	Varchar	No	For future use.
Custom Field 4	255	Varchar	No	For future use.
Custom Field 5	1024	Varchar	No	For future use.
Custom Field 6	1	Char	No	For future use.
Custom Field 7	1	Char	No	For future use.
Custom Field 8	20	Date Time	No	For future use.
Custom Field 9	20	Date Time	No	For future use.

Enrollment Acknowledgement Output File Format

This topic describes the default format of the enrollment acknowledgement output file that Oracle Billing Insight is preconfigured to create.

Table 89 shows the header data elements in the default format of the enrollment acknowledgement output file.

Table 89. File Header Data Elements for the Enrollment Output File

Data Element	Length	Type	Required by the API	Description
Header Record Type	4	Num	No	The header record type, which is the same as the input file.
Sender ID	32	Varchar	Yes	The unique ID for the entity sending the request.
Receiver ID	32	Varchar	Yes	The unique ID for the entity receiving the request.
Sender Name	32	Varchar	No	The name of the entity sending the request.
Status Code	5	Varchar	No	The error code associated with the header.
Status Message	80	Varchar	No	The error message associated with the header.
Custom Field 1	32	Varchar	No	For future use.
Custom Field 2	32	Varchar	No	For future use.

Table 90 shows the batch header data elements in the default format of the enrollment acknowledgement output file.

Table 90. Batch Header Data Elements for the Enrollment Output File

Data Element	Length	Type	Required by the API	Description
Batch Record Type	4	Num	No	Part of the batch header.
Biller ID	32	Varchar	No	Part of the batch header.
Biller Name	32	Varchar	No	Part of the batch header.
Batch ID	7	Num	No	Part of the batch header.
Custom Field 1	32	Varchar	No	Part of the batch header.
Custom Field 2	32	Varchar	No	Part of the batch header.

Table 91 shows the body detail data elements in the default format of the enrollment acknowledgement output file.

Table 91. Body Detail Data Elements for the Enrollment Output File

Data Element	Length	Type	Required by the API	Description
Detail Record Type	4	Num	No	A number indicating this is the detail record type.
Action Type	10	Varchar	No	The type of action: Activate - Enrolling a user. The value can be Change - Changing account details, or Inactivate - Deregistering a user.
Sponsor ID	32	Varchar	No	A unique ID for the sponsor, sent by the consolidator.
Service User ID	32	Varchar	No	A unique ID for the service user, sent by the consolidator.
Consolidator ID	32	Varchar	Yes	The name of the consolidator.
Biller ID	32	Varchar	Yes	A unique ID for the biller, implementation-specific.
Account Number	32	Varchar	Yes	The user account number.
Bill Processing Center ID	9	Varchar	No	An identifier for the bill processing center that processes bills for this account.
Status Code	7	Varchar	No	The Success or Failure status code.
Status Message	80	Varchar	No	The error type and error message, concatenated.
Custom Field 1	255	Varchar	No	For future use.
Custom Field 2	255	Varchar	No	For future use.
Custom Field 3	255	Varchar	No	For future use.
Custom Field 4	255	Varchar	No	For future use.
Custom Field 5	1024	Varchar	No	For future use.
Custom Field 6	1	Char	No	For future use.
Custom Field 7	1	Char	No	For future use.
Custom Field 8	20	Date Time	No	For future use.
Custom Field 9	20	Date Time	No	For future use.

Bill Summary Output File Format

This topic describes the default format of the bill summary output file that Oracle Billing Insight is preconfigured to create.

Table 92 shows the header data elements data elements in the default format of the bill summary output file.

Table 92. File Header Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Header Record Type	4	Num	No	A number indicating this is the header record type.
Sender ID	32	Varchar	Yes	The unique ID for the entity sending the request.
Receiver ID	32	Varchar	Yes	The unique ID for the entity receiving the request.
Sender Name	32	Varchar	Yes	The name of the entity sending the request.
File ID	32	Varchar	Yes	The file identifier.
File Creation Date	8	Date	Yes	The file creation date, in YYYYMMDD format.
File Creation Time	12	Time	Yes	The file creation time, in HHMMSSmmmmmm format.
Time Zone Offset	3	Varchar	No	The time zone offset, such as +03, -05, and so forth.
Custom Field 1	32	Varchar	No	For future use.
Custom Field 2	32	Varchar	No	For future use.
End of Record	2	None	Yes for some consolidators	For some consolidators, this field must contain the hexadecimal value 0D0A.

Table 93 shows the data elements in the batch header record for the Bill Summary file.

Table 93. Batch Header Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Batch Record Type	4	Num	No	Part of the batch header.
Biller ID	32	Varchar	No	Part of the batch header.

Table 93. Batch Header Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Biller Name	32	Varchar	No	Part of the batch header.
Batch ID	7	Num	No	Part of the batch header.
Custom Field 1	32	Varchar	No	Part of the batch header.
Custom Field 2	32	Varchar	No	Part of the batch header.

Table 94 shows the body detail data elements in the default format of the bill summary output file.

Table 94. Body Detail Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Detail Record Type	4	Num	No	A number indicating this is the detail record type.
Action Type	10	Varchar	No	The type of action, for example: <ul style="list-style-type: none"> ■ ADD. When the consolidator sends a record to activate bill payment service. ■ CHG. If the biller later sends a change regarding the service holder's current information, such as a change to the service address.
Consolidator ID	32	Varchar	Yes	The ID of the consolidator, available in the Consolidator User Enrollment table.
Bill Processing Center Number	9	Varchar	No	The ID of the bill processing center that processes bills for this account, available in the Consolidator User Enrollment table.
Biller ID	32	Varchar	Yes	A unique ID for the biller, implementation-specific, available in the Consolidator User Enrollment table.
Billing System ID	32	Varchar	No	The ID of the billing system used in combination with account number to uniquely identify an account.
Biller Account Number	32	Varchar	Yes	The user account number.

Table 94. Body Detail Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Alternate Biller Account Number	32	Varchar	Yes	The internal biller account number, available in the Consolidator User Enrollment table.
Biller Transaction ID	32	Varchar	No	The biller transaction ID, sent in the enrollment input file.
Biller User ID	32	Varchar	No	The biller user ID, sent in the enrollment input file.
Biller E-Bill ID	36	Varchar	Yes	The unique ID for the bill, assigned by the biller. It is usually the invoice number on the bill. Otherwise, it is populated with amount due and bill due date.
User First Name	32	Varchar	No	The user's first name.
User Last Name	32	Varchar	No	The user's last name.
Payment Received	20	Numeric	Yes	The net payment received and credited to the account since the last period.
Amount Due	20	Numeric	Yes	The amount that must be displayed to the subscriber for payment. It can be the minimum amount due, this period's payment due, or the total account balance.
Balance Amount	20	Numeric	No	The account balance at the time of the closing cycle date.
Minimum Amount Due	20	Numeric	No	The minimum amount due.
Previous Balance Amount	20	Numeric	No	The previous balance amount.
Biller Reference Text	20	Varchar	No	Text the biller sends with the electronic bill. Invoice-like information to be sent with particular bills that must be returned to the biller for posting purposes.
Statement Date	20	Datetime	Yes	The timestamp when bill was created, in YYYYMMDDHHMMSSmmmmmm format.

Table 94. Body Detail Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Next Statement Date	20	Datetime	No	The timestamp when the bill was created, in YYYYMMDDHHMMSSmmmmmm format.
Bill Cycle End Date	20	Datetime	No	The timestamp when the bill period was closed.
Payment Due Text Flag	1	Char	No	A flag indicating to the consolidator whether the biller wants the sponsor to display Text Due Date instead of the Electronic Bill Due Date Timestamp. If the sponsor does not support Text Due Date, then it uses the Electronic Bill Due Date Timestamp.
Bill Due Date	20	Datetime	No	Timestamp when the bill is due.
Text Due Date	20	Datetime	No	When the Payment Due Text Flag value is Y, the values in this field are displayed to the user, spaces included.
Bill Cycle Start Date	20	Datetime	No	Timestamp when this bill was opened.
Electronic Billing URL	1024	Varchar	No	The URL for viewing bill details in Oracle Billing Insight.
Bill URL Expiry Date	20	Datetime	No	The time when the URL expires.
Teaser Link URL	1024	Varchar	No	The URL for the teaser ad that the subscriber can follow to receive additional information from the biller. This URL is used in conjunction with either the Teaser Ad URL or the Teaser Ad Text. The subscriber can click on either the image or text to be transferred to the biller site specified by this URL.
Teaser Ad URL	1024	Varchar	No	The URL where the teaser ad image is stored. This is used to display the image on the user interface.

Table 94. Body Detail Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Teaser Ad Text	80	Varchar	No	A text link that can be used instead of the image link. Using this text might improve performance over downloading a teaser ad image.
E-Bill Category URL	1024	Varchar	No	The URL associated with the ebill's category.
E-Bill Category Text	80	Varchar	No	The text associated with the ebill's category.
E-Bill Magnet Type	1	Char	No	<p>Determines whether a customer magnet is required for this electronic bill. The values can be:</p> <ul style="list-style-type: none"> ■ M. Magnet has its own URL. ■ Y. Magnet uses the Electronic Bill URL. (The Magnet URL must be spaces). ■ N. No, a customer magnet is not required (The Magnet URL must be spaces). This is the default. <p>NOTE: The customer magnet feature is not supported by many sponsors. The feature is being deprecated.</p>
E-Bill Magnet URL	1024	Varchar	No	The URL (teaser ad or promotional offer) to where a subscriber is taken before making a payment.
E-Bill Category	2	Varchar	No	The e-bill category. Values can be spaces for a normal e-bill (the default), or 02 for an e-bill supplied by a biller to replace a previous ebill.
Status Code	5	Varchar	No	The status code.
Status Message	80	Varchar	No	The status message.
Custom Field 1	255	Varchar	No	For future use.
Custom Field 2	255	Varchar	No	For future use.
Custom Field 3	255	Varchar	No	For future use.
Custom Field 4	255	Varchar	No	For future use.
Custom Field 5	1024	Varchar	No	For future use.

Table 94. Body Detail Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Custom Field 6	1	Char	No	For future use.
Custom Field 7	1	Char	No	For future use.
Custom Field 8	20	String	No	For future use.
Custom Field 9	20	String	No	For future use.
End of Record	2	None	No	For some consolidators, this field must contain the hexadecimal value 0D0A.

Table 95 shows the batch control data elements in the default format of the bill summary output file.

Table 95. Batch Control Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Batch Control Record Type Code	4	Varchar	No	The batch control record type code.
Total Record Count	8	Num	No	The total number of records.
Batch ID	7	Num	No	The ID identifying the batch.

Table 96 shows the file control (trailer) data elements in the default format of the bill summary output file.

Table 96. File Control Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Trailer Record Type	4	Num	No	The value must be 9999 for some consolidators.
Record Type 1	4	Varchar	No	Record Type 1
Number of Records 1	6	Num	No	Number of Records 1
Record Type 2	4	Varchar	No	Record Type 2
Number of Records 2	6	Num	No	Number of Records 2
Record Type 3	4	Varchar	No	Record Type 3
Number of Records 3	6	Num	No	Number of Records 3

Table 96. File Control Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Record Type 4	4	Varchar	No	Record Type 4
Number of Records 4	6	Num	No	Number of Records 4
Record Type 5	4	Varchar	No	Record Type 5
Number of Records 5	6	Num	No	Number of Records 5
Record Type 6	4	Varchar	No	Record Type 6
Number of Records 6	6	Num	No	Number of Records 6
Record Type 7	4	Varchar	No	Record Type 7
Number of Records 7	6	Num	No	Number of Records 7
Record Type 8	4	Varchar	No	Record Type 8
Number of Records 8	6	Num	No	Number of Records 8
Record Type 9	4	Varchar	No	Record Type
Number of Records 9	6	Num	No	Number of Records 9
Record Type 10	4	Varchar	No	Record Type 10
Number of Records 10	6	Num	No	Number of Records 10
Record Type 11	4	Varchar	No	Record Type 11
Number of Records 11	6	Num	No	Number of Records 11
Record Type 12	4	Varchar	No	Record Type 12
Number of Records 12	6	Num	No	Number of Records 12
Record Type 13	4	Varchar	No	Record Type 13
Number of Records 13	6	Num	No	Number of Records 13
Record Type 14	4	Varchar	No	Record Type 14
Number of Records 14	6	Num	No	Number of Records 14
Record Type 15	4	Varchar	No	Record Type 15
Number of Records 15	6	Num	No	Number of Records 15
Record Type 16	4	Varchar	No	Record Type 16
Number of Records 16	6	Num	No	Number of Records 16
Record Type 17	4	Varchar	No	Record Type 17
Number of Records 17	6	Num	No	Number of Records 17
Custom Field 1	85	Varchar	No	Custom field 1

Table 96. File Control Data Elements for the Bill Summary Output File

Data Element	Length	Type	Required by the API	Description
Total Batch Count	6	Varchar	No	Total batch count.
Total Record Count	8	Num	No	Total number of records.
File ID	32	Num	No	The file identifier.
End of Record	2	Varchar	No	For some consolidators, this field must contain the hexadecimal value 0D0A.

Bill Summary Acknowledgement Input File Format

This topic describes the default format of the bill summary acknowledgement file that Oracle Billing Insight is preconfigured to use.

The data elements in the file header record for the consolidator response file are the same as those shown in [Table 86 on page 315](#).

The data elements in the batch header record for the consolidator response file are the same as those shown in [Table 87 on page 316](#).

[Table 97](#) shows the body detail data elements in the default format of the consolidator response input file.

Table 97. Body Detail Data Elements for the Bill Summary Acknowledgement Output File

Data Element	Length	Type	Required by the API	Description
Detail Record Type	4	Num	No	A number indicating this is the detail record type.
Action Type	10	Varchar	No	The type of action, for example: <ul style="list-style-type: none"> ■ ADD. When the consolidator sends a record to activate bill payment service. ■ CHG. If the biller later sends a change regarding the service holder's current information, such as a change to the service address.
Consolidator ID	32	Varchar	Yes	The name of the consolidator.
Biller ID	32	Varchar	Yes	The unique ID for the biller, implementation-specific.

Table 97. Body Detail Data Elements for the Bill Summary Acknowledgement Output File

Data Element	Length	Type	Required by the API	Description
Account Number	32	Varchar	No	The user account number.
Alternate Biller Account Number	32	Varchar	No	The biller internal account number.
Bill Processing Center ID	9	Varchar	No	The ID for the bill processing center that processes bills for this account.
Biller E-Bill ID	36	Varchar	Yes	The unique ID for the bill, assigned by biller.
Status Code	5	Varchar	No	This field is defined by the consolidator and sent in the output file.
Status Message	80	Varchar	No	This field is defined by the consolidator and sent in the output file.
Custom Field 1	255	Varchar	No	For future use.
Custom Field 2	255	Varchar	No	For future use.
Custom Field 3	255	Varchar	No	For future use.
Custom Field 4	255	Varchar	No	For future use.
Custom Field 5	1024	Varchar	No	For future use.
Custom Field 6	1	Char	No	For future use.
Custom Field 7	1	Char	No	For future use.
Custom Field 8	20	Datetime	No	For future use.
Custom Field 9	20	Datetime	No	For future use.
End of Record	2	None	No	For some consolidators, this field must contain the hexadecimal value 0D0A.

Data Element Mappings - Example A

This topic shows an example of how the following example file elements can map between the data files for consolidator A and Oracle Billing Insight:

- SIS2050ACK
- SIS2070
- SIS2070ACK

Mapping of the SIS2050ACK Data Elements to Oracle Billing Insight

NOTE: A batch header is not applicable in this example.

Table 98 shows how the file header elements in the SIS2050ACK file map to data elements in Oracle Billing Insight.

Table 98. Mapping of SIS2050ACK File Header Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Header Record Type
Sender ID	Sender ID
Receiver ID	Receiver ID
Entity Name	Sender Name
File ID	File ID
File Creation Date	File Creation Date
File Creation Time	File Creation Time
None	Time Zone Offset
Filler	Custom Field 1
Filler	Custom Field 2
Error Code	Status Code
Error Message	Status Message
End of Record	End of Record

Table 99 shows how the body detail elements in the SIS2050ACK file map to elements in Oracle Billing Insight.

Table 99. Mapping of SIS2050ACK Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Detail Record Type
Record Action	Action Type
Sponsor ID	Sponsor ID
Subscriber ID	Consolidator User ID
Bill Publisher ID	Consolidator ID
Biller ID	Biller ID
Biller Account Number	Account Number
Service Holder Business Name	Business Name
Service Holder Last Name	User's Last Name

Table 99. Mapping of SIS2050ACK Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Service Holder First Name	User's First Name
Service Holder Middle Name	User's Middle Name
Service Address Line 1	User Address Line 1
Service Address Line 2	User Address Line 2
Service Address Line 3	User Address Line 3
Service City	User City
Service State	User State
Service Zip 5	User Zip 5
Service Zip 4	User Zip 4
Service Zip 2	User Zip 2
Service Country Code	User Country Code
Service Postal Code	User Postal Code
Service Province Name	User Province Name
Service Day Phone Number	User Day Phone Number
Service Evening Phone Number	User Evening Phone Number
Bill Processing Center Number	Bill Processing Center ID
Alternate Biller Account Number	Alternate Biller Account Number
New Alternate Biller Account Number	New Alternate Biller Account Number
Server Transaction Timestamp	Server Transaction Timestamp
Bill Remittance Account Number	Bill Remittance Account Number
Response Code	Response Code
Rejection Text	Rejection Text
Paper Question Status	Paper Question Status
Paper Status Code	Paper Status
Incentive Message ID	Incentive Message ID
Filler	Custom Field 1
Error Code	Status Code
Error Message	Status Message
None	Biller Name
None	Biller User ID
None	Consolidator Transaction ID

Table 99. Mapping of SIS2050ACK Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
None	Biller Transaction ID
None	Biller Address Line 1
None	Biller Address Line 2
None	Biller City
None	Biller State
None	Biller Zipcode
None	Biller Zip + 4
None	New Account Number
None	User Type
None	User Email Address
None	Bill Payment Activated
None	Service Verification Field 1
None	Service Verification Field 2
None	Service Verification Field 3
None	Service Verification Field 4
None	Service Verification Field 5
None	Service Verification Field 6
None	Auto Activation Flag
None	Paper Suppression Date
None	Custom Field 2
None	Consolidator Enrollment Status
None	Biller Enrollment Status
None	Deactivation Source
None	Custom Field 3
None	Custom Field 4
None	Custom Field 5
None	Custom Field 6
None	Custom Field 7
None	Custom Field 8
None	Custom Field 9
None	Record Type

Table 99. Mapping of SIS2050ACK Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
None	Biller ID
None	Biller Name
None	Batch ID
None	Custom Field 1
None	Custom Field 2
End of Record	End of Record

Mapping of the SIS2070 File Elements to the Oracle Billing Insight Bill Summary File

Table 100 shows the how the header elements in the SIS2070 file map to elements in Oracle Billing Insight.

Table 100. Mapping of SIS2070 File Header Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Header Record Type
Sender ID	Sender ID
Receiver ID	Receiver ID
Sender Name	Sender Name
File ID	File ID
File Creation Date	File Creation Date
File Creation Time	File Creation Time
None	Time Zone Offset
None	Custom Field 1
None	Custom Field 2
End of Record	End of Record

Table 101 shows how the body detail elements in the SIS2070 file map to elements in Oracle Billing Insight.

Table 101. Mapping of the SIS2070 Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Detail Record Type
Record Action	Action Type
Bill Publisher ID	Consolidator ID

Table 101. Mapping of the SIS2070 Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Bill Processing Center Number	Bill Processing Center Number
Biller ID	Biller ID
Biller Account Number	Account Number
Alternate Biller Account Number	Alternate Biller Account Number
Biller E-Bill ID	Biller E-Bill ID
Electronic Bill Activity Amount	Payment Received
Electronic Bill Amount Due	Amount Due
Electronic Bill Balance Amount	Balance Amount
Electronic Bill Minimum Amount Due	Minimum Amount Due
Electronic Bill Previous Balance Amount	Previous Balance Amount
Biller Reference Text	Biller Reference Text
Electronic Bill Timestamp	Bill Statement Date
Electronic Bill Closed Timestamp	Bill Cycle End Date
Payment Due Text Flag	Payment Due Text Flag
Electronic Bill Due Date Timestamp	Bill Due Date
Text Due Date	Text Due Date
Electronic Billing Open Statement Timestamp	Bill Cycle Start Date
Electronic Billing URL	Electronic Billing URL
Teaser Link URL	Teaser Link URL
Teaser Ad URL	Teaser Ad URL
Teaser Ad Text	Teaser Ad Text
E-Bill Category URL	E-Bill Category URL
E-Bill Category Text	E-Bill Category Text
FI Transaction Stamp	None
Electronic Bill Magnet Type	Bill Magnet Type
Magnet URL	Bill Magnet URL
Filler	Custom Field 1
E-bill Category	E-Bill Category
Filler	Custom Field 2
Error Code	Status Code

Table 101. Mapping of the SIS2070 Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Error Message	Status Message
None	Biller Transaction ID
None	Biller User ID
None	User First Name
None	User Last Name
None	Next Statement Date
None	Bill URL Expiry Date
None	Custom Field 3
None	Custom Field 4
None	Custom Field 5
None	Custom Field 6
None	Custom Field 7
None	Custom Field 8
None	Custom Field 9
None	Record Type
None	Biller ID
None	Biller Name
None	Batch ID
None	Custom Field 1
None	Custom Field 2
End of Record	None

Table 102 shows how the file control elements in the SIS2070 file map to elements in Oracle Billing Insight.

Table 102. Mapping of the SIS2070 File Control Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Trailer Record Type
Record Type 1	Record Type 1
Number of Records 1	Number of Records 1
Record Type 2	Record Type 2
Number of Records 2	Number of Records 2

Table 102. Mapping of the SIS2070 File Control Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type 3	Record Type 3
Number of Records 3	Number of Records 3
Record Type 4	Record Type 4
Number of Records 4	Number of Records 4
Record Type 5	Record Type 5
Number of Records 5	Number of Records 5
Record Type 6	Record Type 6
Number of Records 6	Number of Records 6
Record Type 7	Record Type 7
Number of Records 7	Number of Records 7
Record Type 8	Record Type 8
Number of Records 8	Number of Records 8
Record Type 9	Record Type 9
Number of Records 9	Number of Records 9
Record Type 10	Record Type 10
Number of Records 10	Number of Records 10
Record Type 11	Record Type 11
Number of Records 11	Number of Records 11
Record Type 12	Record Type 12
Number of Records 12	Number of Records 12
Record Type 13	Record Type 13
Number of Records 13	Number of Records 13
Record Type 14	Record Type 14
Number of Records 14	Number of Records 14
Record Type 15	Record Type 15
Number of Records 15	Number of Records 15
Record Type 16	Record Type 16
Number of Records 16	Number of Records 16
Record Type 17	Record Type 17
Number of Records 17	Number of Records 17
Custom Field 1	Custom Field 1

Table 102. Mapping of the SIS2070 File Control Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
None	Custom Field 1
None	Total Batch Count
None	Total Record Count
None	File ID
Total Record Count	End of Record

About Amount Fields

For some consolidators, amount fields consist of four parts:

- **The amount sign.** This value is a plus sign (+) for a positive balance or a minus sign (-) for a negative balance.
- **The amount.** The dollar amount in 9(11)V9(4) format. The amount is exactly 15 digits in length, padded with zeros if necessary. There is an implied decimal point before the right-most four numbers of the amount.
- **A debit or credit indicator.** The value code is always C.
- **The currency code.** The value is always USD (United States Dollar).

Check with your consolidator. If the biller wants to send something in an amount field, then all of these parts must be populated. If the biller does not want to send something in an amount field, then all four parts must be blank.

Mapping of the SIS2070ACK File Elements to Oracle Billing Insight

NOTE: The SIS2070ACK file contains no batch header, batch control, or file control records.

Table 103 shows how the header record elements in the SIS2070ACK file map to elements in Oracle Billing Insight.

Table 103. Mapping of SIS2070ACK Header Record Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Header Record Type
Sender ID	Sender ID
Receiver ID	Receiver ID
Sender Name	Sender Name
File ID	File ID
File Creation Date	File Creation Date
File Creation Time	File Creation Time
None	Time Zone Offset

Table 103. Mapping of SIS2070ACK Header Record Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Error Code	Status Code
Error Message	Status Message
None	Custom Field 1
None	Custom Field 2
End of Record	End of Record

Table 104 shows how the body detail elements in the SIS2070ACK file map to elements in Oracle

Table 104. Mapping of the SIS2070ACK Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Record Type	Detail Record Type
Record Action	Action Type
Bill Publisher ID	Consolidator ID
Bill Processing Center Number	Bill Processing Center Number
Biller ID	Biller ID
Biller Account Number	Biller Account Number
Alternate Biller Account Number	Alternate Biller Account Number
Biller E-Bill ID	Biller E-Bill ID
Electronic Bill Activity Amount	Payment Received
Electronic Bill Amount Due	Amount Due
Electronic Bill Balance Amount	Balance Amount
Electronic Bill Minimum Amount Due	Minimum Amount Due
Electronic Bill Previous Balance Amount	Previous Balance Amount
Biller Reference Text	Biller Reference Text
Electronic Bill Timestamp	Bill Statement Date
Electronic Bill Closed Timestamp	Bill Cycle End Date
Payment Due Text Flag	Payment Due Text Flag
Electronic Bill Due Date Timestamp	Bill Due Date
Text Due Date	Text Due Date
Electronic Billing Open Statement Timestamp	Bill Cycle Start Date
Electronic Billing URL	Electronic Billing URL
Teaser Link URL	Teaser Link URL
Teaser Ad URL	Teaser Ad URL
Teaser Ad Text	Teaser Ad Text
E-Bill Category URL	E-Bill Category URL
E-Bill Category Text	E-Bill Category Text
FI Transaction Stamp	None
Electronic Bill Magnet Type	Bill Magnet Type
Magnet URL	Bill Magnet URL

Table 104. Mapping of the SIS2070ACK Body Detail Elements to Oracle Billing Insight

File Element of Consolidator A	Oracle Billing Insight Element
Filler	Custom Field 1
E-bill Category	E-Bill Category
Filler	Custom Field 2
Error Code	Error Code
Error Message	Error Message
None	Custom Field 3
None	Custom Field 4
None	Custom Field 5
None	Custom Field 6
None	Custom Field 7
None	Custom Field 8
None	Custom Field 9
None	Record Type
None	Biller ID
None	Biller Name
None	Batch ID
None	Custom Field 1
None	Custom Field 2
End of Record	End of Record

Billing Insight.

Data Element Mappings - Example B

This topic shows an example of how the following file elements can map between the data files for Consolidator B and Oracle Billing Insight:

- A consolidator's enrollment response file
- Oracle Billing Insight bill summary file
- A consolidator's details confirmation file

Table 105 shows how the file header elements of Consolidator B's enrollment response file map to elements in Oracle Billing Insight.

Table 105. Mapping of Consolidator B's Enrollment Response File Header Elements to Oracle Billing Insight

File Element of Consolidator B	Oracle Billing Insight Element
Record Type code	Header Record Type
ENV	None
Originator ID	Sender ID
Originator Name	Sender Name
File ID	File ID
File Creation Date	File Creation Date
File Creation Time	File Creation Time
File Type	None
UTC	Time Zone Offset
UTC Timezone	Time Zone Offset
Custom Field 1	Custom Field 1
Custom Field 2	Custom Field 2
None	Receiver ID
None	Status Code
None	Status Message

Table 106 shows how the batch header elements of Consolidator B's enrollment response file map to elements in Oracle Billing Insight.

Table 106. Mapping of Consolidator B's Enrollment Response File Batch Header Elements to Oracle Billing Insight

File Element of Consolidator B	Oracle Billing Insight Element
Record Type code	Batch Record Type
Biller ID	Biller ID
Biller Name	Biller Name
Batch ID	Batch ID
Custom Field 1	Custom Field 1
Custom Field 2	Custom Field 2

Table 107 shows how body detail elements of Consolidator B's enrollment response file map to elements in Oracle Billing Insight.

Table 107. Mapping of Consolidator B's Enrollment Response File Body Detail Elements to Oracle Billing Insight

File Element of Consolidator B	Oracle Billing Insight Element
Record Type code	Detail Record Type
Transaction Code	Action Type
EBill Channel Type ID	Sponsor ID
Transaction Expression	Action Description
Originator Enrollment ID	Consolidator Transaction ID
Processor Enrollment ID	Biller Transaction ID
Originator Payor ID	Consolidator User ID
Processor Payor ID	Biller User ID
eBill Start Date	E-Bill Start Date
Paper Suppression Date	Paper Suppression Date
Biller ID	Biller ID
Custom Field 1	Account Number
Custom Field 2	Custom Field 1
Custom Field 3	Custom Field 2
Custom Field 4	Custom Field 3
Custom Field 5	Custom Field 4
Custom Field 6	Custom Field 5
Status Code	Status Code
Status Message	Status Message
None	Consolidator ID,
None	New Account Number
None	User Type
None	Business Name
None	Biller Name
None	Biller Address 1
None	Biller Address 2
None	City
None	State

Table 107. Mapping of Consolidator B's Enrollment Response File Body Detail Elements to Oracle Billing Insight

File Element of Consolidator B	Oracle Billing Insight Element
None	ZipCode
None	Zip+4
None	User Last Name
None	User First Name
None	User Middle Name
None	User Email Address
None	Address Line 1
None	Address Line 2
None	Address Line 3
None	City
None	State
None	Zip 5
None	Zip 4
None	Zip 2
None	Country Code
None	Postal Code
None	Province Name
None	Day Phone Number,
None	Evening Phone Number
None	Bill Processing Center ID
None	Bill Payment Activated
None	Service Verification Field 1
None	Service Verification Field 2
None	Service Verification Field 3
None	Service Verification Field 4
None	Service Verification Field 5
None	Service Verification Field 6
None	Alternate Biller Account Number
None	New Alternate Biller Account Number
None	Server Transaction Timestamp

Table 107. Mapping of Consolidator B's Enrollment Response File Body Detail Elements to Oracle Billing Insight

File Element of Consolidator B	Oracle Billing Insight Element
None	Auto Activation Flag
None	Bill Remittance Account Number
None	Response Code
None	Rejection Text
None	Paper Question Status
None	Paper Status
None	Custom Field 6
None	Custom Field 7
None	Custom Field 8
None	Custom Field 9
Record Type Code	None
Total Transaction Count	None
Batch ID	None
Record Type Code	None
Total Batch Count	None
Total Transaction Count	None
File ID	None

Mapping of the Oracle Billing Insight Bill Summary File Elements

Table 108 shows how the file header elements in the Oracle Billing Insight file map to elements for Consolidator B.

Table 108. Mapping of the Bill Summary File Header Elements to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
Record Type code	Header Record Type
ENV	None
Originator ID	Sender ID
Originator Name	Sender Name
File ID	File ID
File Creation Date	File Creation Date
File Creation Time	File Creation Time
File Type	None

Table 108. Mapping of the Bill Summary File Header Elements to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
UTC	Time Zone Offset
UTC Timezone	Time Zone Offset
Custom Field 1	Custom Field 1
Custom Field 2	Custom Field 2
None	Receiver ID

The batch header elements in the Bill Summary file map to Consolidator B's file details as shown in [Table 107 on page 344](#).

[Table 109](#) shows how the body detail elements in the Oracle Billing Insight Bill Summary File map to Consolidator B's data elements.

Table 109. Mapping of the Body Detail Elements in the Bill Summary File to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
Record Type	Detail Record Type
Transaction Code	Action Type
Transaction Expression	None
Processor Enrollment ID	Biller Transaction ID
Processor Payor ID	Biller User ID
Biller ID	Biller ID
Payor Account Number	Biller Account Number
Processor E- Bill Detail ID	Biller E- Bill ID
Invoice Number	Biller E- Bill ID
Payor First Name	User First Name
Payor Last Name	User Last Name
Total Amount Due	Amount Due
Account Balance	Balance Amount
Next Statement Date	Next Statement Date
Minimum Due	Minimum Amount Due
Previous Balance	Previous Balance Amount
Statement Date	Statement Date
Due Date	Bill Due Date
Bill URL Type	None

Table 109. Mapping of the Body Detail Elements in the Bill Summary File to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
Bill URL Expire Date	Bill URL Expiry Date
NOC 1	None
NOC 2	None
Bill URL	Electronic Billing URL
Custom1	Custom Field 1
Custom2	Custom Field 2
Custom3	Custom Field 3
Custom4	Custom Field 4
None	Bill Processing Center Number
None	Alternate Biller Account Number
None	Payment Received
None	Biller Reference Text
None	Bill Cycle End Date
None	Payment Due Text Flag
None	Text Due Date
None	Bill Cycle Start Date
None	Teaser Link URL
None	Teaser Ad URL
None	Teaser Ad Text
None	E-Bill Category URL
None	E-Bill Category Text
None	E-Bill Magnet Type
None	E-Bill Magnet URL
None	E-Bill Category
None	Custom Field 5
None	Custom Field 6
None	Custom Field 7

Table 110 shows how the batch control elements in the Oracle Billing Insight Bill Summary file map to Consolidator B's data elements.

Table 110. Mapping of the Batch Control Elements in the Bill Summary File to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
Record Type code	Batch Control Record Type Code
Total Transaction Count	Total Record Count
Batch ID	Batch ID
Custom Field 1	Custom Field 1
Custom Field 2	Custom Field 2

Table 111 shows how the file control elements in the Oracle Billing Insight Bill Summary File map to Consolidator B's elements.

Table 111. Mapping of the File Control Elements in the Bill Summary File Map to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
Record Type Code	Trailer Record Type
Total Batch Count	Total Batch Count
Total Transaction Count	Total Record Count
File ID	File ID
None	Record Type 1
None	Number of Records 1
None	Record Type 2
None	Number of Records 2
None	Record Type 3
None	Number of Records 3
None	Record Type 4
None	Number of Records 4
None	Record Type 5
None	Number of Records 5
None	Record Type 6
None	Number of Records 6
None	Record Type 7
None	Number of Records 7

Table 111. Mapping of the File Control Elements in the Bill Summary File Map to Consolidator B's Elements

File Element of Consolidator B	Oracle Billing Insight Element
None	Record Type 8
None	Number of Records 8
None	Record Type 9
None	Number of Records 9
None	Record Type 10
None	Number of Records 10
None	Record Type 11
None	Number of Records 11
None	Record Type 12
None	Number of Records 12
None	Record Type 13
None	Number of Records 13
None	Record Type 14
None	Number of Records 14
None	Record Type 15
None	Number of Records 15
None	Record Type 16
None	Number of Records 16
None	Record Type 17
None	Number of Records 17
None	Custom Field 1

Mapping of the Details Confirmation File to Oracle Billing Insight

The file header elements in the details confirmation file map to elements in Oracle Billing Insight as shown in [Table 106 on page 343](#).

The batch header elements in the details confirmation file map to elements in Oracle Billing Insight as shown in [Table 107 on page 344](#).

Table 112 shows how the body detail elements of Consolidator B's details confirmation file map to elements in Oracle Billing Insight.

Table 112. Mapping of the Body Detail Elements of Consolidator B's Details Confirmation File Map to Oracle Billing Insight

File Element of Consolidator B	Oracle Billing Insight Element
Record Type	Detail Record Type
Processor E-Bill Detail ID	Biller E-Bill ID
Status Code	Status Code
Status Message	Status Message
Custom1	Custom Field 1
Custom2	Custom Field 2
Custom3	Custom Field 3
Custom4	Custom Field 4
None	Custom Field 5
None	Custom Field 6
None	Custom Field 7
None	Consolidator ID
None	Bill Processing Center Number
None	Biller ID
None	Biller Account Number
None	Alternate Biller Account Number
None	E-Bill Category

Payment Consolidation Database Tables

The database tables shown in Table 113 Oracle Billing Insight database tables store data used exclusively for use with the payment consolidation feature.

Table 113. Payment Consolidation Database Tables

Payment Consolidation Database Table	Description
EDX_PC_ACCOUNT_ENROLL	This table stores information on the enrollment status of an account with a consolidator.
EDX_PC_ACCOUNT_ENROLL_HISTORY	This table stores information on the changes to the enrollment status of an account over time.

Table 113. Payment Consolidation Database Tables

Payment Consolidation Database Table	Description
EDX_PC_ACCT_ENROLL_DETAIL	This table stores detail information on an account.
EDX_PC_ACCT_ENROLL_DETAIL_HIST	This table stores information on every file and record that has been processed by the enrollment jobs.
EDX_PC_ENROLL_FILE	This table stores file record information for the enrollment input file.
EDX_PC_ENROLL_BATCH	This table stores batch record information for the enrollment input file.
EDX_PC_BILLER	This table stores records that were rejected by the external system or failed internal processing.
EDX_PC_BILLER	This table stores biller ID and names, processing center, and the URLs to bill and marketing links.
EDX_PC_CONSOLIDATOR	This table stores the consolidator ID and names.
EDX_PC_BILLSUMMARY_LOG	This table stores bill summary generation and acknowledgement status information.
EDX_PC_FILETYPE_MAPPING	This table stores the file-type mapping information configured for the payment consolidator.

11 Customizing the Assisted Service Application

This chapter covers customizing the Assisted Service application for your implementation of Oracle Billing Insight. It includes the following topics:

- [About the Assisted Service Application Functionality on page 353](#)
- [Assisted Service Application Files on page 353](#)
- [About Impersonation Auditing on page 354](#)

About the Assisted Service Application Functionality

The Assisted Service application delivered with Oracle Billing Insight provides an interface where agents, or customer service representatives, can assist end users with all functions that are available to the end user. For details, see *Assisted Service Application Guide for Oracle Billing Insight* and *Self-Service Application Guide for Oracle Billing Insight*.

Assisted Service Application Files

The Assisted Service application WAR file contains the tiles (*.JSP) for the application. Under the `war/src/main/webapp` directory are a variety of packages containing tiles that address key agent view functions such as impersonating and finding an agent's customer (`access-cust`), enrolling the agent and searching for a customer's agent (`manage-csr`), enrolling the customer (`manage-cust`), and searching and managing organizations (`manage-org`).

Under the `\web-actions\src\main\java\com\edocs\application\csr` (compiled source) directory are action, form, and tag classes which comprise the model and controller of the Assisted Service application. The common package contains a variety of agent helper classes for logging in, enrolling, authentication, and configuration.

See the `war/src/main/webapp/WEB-INF/classes` directory for the struts configuration JavaBeans and forwarding actions for this Assisted Service application. The tiles configuration also resides in this file.

See the `web-actions\src\main\config\csr.xma.xml` file for how to configure access to the customer application from the Assisted Service application and the list of agent roles that are enabled.

The Oracle Billing Insight API Javadoc contains additional information about customizing the Assisted Service application. For information about accessing the Javadoc, see ["Accessing Oracle Billing Insight Data Dictionary and Javadoc" on page 25](#).

About Impersonation Auditing

Oracle Billing Insight audits impersonation actions and regular billing-related activities that are performed during impersonation sessions in the EDX_UMF_USER_AUDIT database table.

The EDX_UMF_USER_AUDIT table includes the following information for impersonation activity:

- **USER_ID.** The ID of the user or agent user who performed the action or had actions performed on his behalf by the user identified in the IMP_CSR_ID field during an impersonation session.
- **TARGET_USER_ID.** The ID of the user whose information was processed by another user during impersonation.
- **IMP_CSR_ID.** Indicates that the action was performed during an impersonation session and identifies the agent user who impersonated the user identified in the USER_ID field. The IMP_CSR_ID field can be an ID from an external customer service application.
- **TRUST_USER.** The name of the trust user who has an agent role and is required to get authentication for access to Oracle Billing Insight, if the impersonation session is initiated by an external customer service system and the external user does not exist in Oracle Billing Insight.

Table 114 shows sample data for various types of impersonation actions that Oracle Billing Insight audits

Table 114. Examples of Impersonation Actions Audited

Impersonation Action	User ID	TARGET_USER_ID	IMP_CSR_ID	TRUST_USER	Description
Impersonate - Login	ftown	None	Agent1	None	User Agent1 impersonates user ftown and Agent1 exists in Oracle Billing Insight.
Impersonate - Login	ftown	None	Agent1	trustedUser1	User Agent1 impersonates user ftown using trustedUser1 and Agent1 does not exist in Oracle Billing Insight.
Update a user	ftown	None	Agent1	None	User Agent1 impersonates user ftown to update ftown's information and Agent1 exists in Oracle Billing Insight.
Update User	ftown	None	Agent1	trustedUser1	User Agent1 impersonates user ftown using trustedUser1 and Agent1 does not exist in Oracle Billing Insight.

Table 114. Examples of Impersonation Actions Audited

Impersonation Action	User ID	TARGET_USER_ID	IMP_CSR_ID	TRUST_USER	Description
Update User	ftown	lgreen	Agent1	None	User Agent1 impersonates user ftown to update information for user lgreen. User Agent1 exists in Oracle Billing Insight.
Update User	ftown	lgreen	Agent1	trustedUser1	User Agent1 impersonates user ftown using trustedUser1 to update information for user lgreen. User Agent1 does not exist in Oracle Billing Insight.
Update User	ftown	lgreen	None	None	User ftown updates information for user lgreen.
Update User	ftown	None	None	None	User ftown updates his own information.
Update notification settings	ftown	None	Agent1	None	User Agent1 impersonates user ftown to update ftown's notification settings. Agent1 exists in Oracle Billing Insight.
Impersonation - Logout	ftown	None	Agent1	None	User Agent1 impersonates user ftown and logs out. Agent1 exists in Oracle Billing Insight.

12 Reporting on Audit Data

This chapter describes reporting on audit data in Oracle Billing Insight. It includes the following topics:

- [Reporting on User Audit Data on page 357](#)
- [Reporting on System Administrator Audit Data on page 362](#)

Reporting on User Audit Data

Oracle Billing Insight audits some enrollment user actions performed in the Self-Service application.

Oracle Billing Insight audits the following actions that occur when creating users:

- A business administrator creates another business user's account.
- A consumer user creates his or her own account.
- The default agent administrator creates another agent administrator account.
- An agent administrator creates an agent user's account.
- An agent administrator creates another business administrator's account.
- An agent user (administrator or agent) impersonates a business administrator creating another business user's account.

Oracle Billing Insight audits the following actions that occur when enrolling users:

- A user enrolls.
- An agent user (administrator or agent) enrolls.

Oracle Billing Insight audits the following actions that occur when updating user profiles:

- A user updates his or her own user access information.
- A user updates his or her own notification settings.
- A business user (administrator or manager) updates another business user's user access information.
- An agent user (administrator or agent) updates his or her own user access information.
- An agent administrator updates another agent user's user access information.
- An agent (administrator or agent) updates another business user's user access information.
- An agent (administrator or agent) updates the following by impersonation:
 - His or her own notification settings.
 - His or her own user access information.

- A business user (administrator or manager) updates another business user's user access information.
- A user or agent user enrolls to complete the reactivation process after clicking the URL in an email notification.
- A user or agent user resets his or her forgotten password.
- A migrated user or agent user creates a new HIPPA-compliant password after clicking the URL in the email notification.
- A user or agent user updates his or her own expired password.

Oracle Billing Insight audits the following actions that occur when deleting users:

- A business administrator user deletes another business user's account.
- An agent user (administrator or agent) deletes a user's account.
- An agent administrator deletes another agent user's account.
- An agent user (administrator or agent) impersonates a business administrator user deleting another business user's account.

Oracle Billing Insight audits the following actions that occur when logging in and out:

- A user logs in.
- An agent user (administrator or agent) logs in.
- An agent user (administrator or agent) impersonates a business or consumer user logging in or out.
- A business, consumer, or agent user fails to log in.
- An agent administrator reactivates a locked-out account.

You can report on the audit data for each user role, including the user who performed the action, the date and time, IP address, and various attributes. For details about payment audit data, see ["About Payment Auditing" on page 240](#). For information about database auditing, see ["Identifying the Installed Version of Oracle Billing Insight" on page 26](#).

You can create customized reports on the audited user enrollment data. Oracle Billing Insight stores audit data for user enrollment activities in the EDX_UMF_USER_AUDIT table. [Table 115](#) describes the EDX_UMF_USER_AUDIT table:

Table 115. EDX_UMF_USER_AUDIT Table

Column Name	Description
ID	A unique ID assigned to the each occurrence of a user enrollment event, generated automatically by sequence.
USER_ID	The ID of the user who performed the action, either on his or her own account or on another user's account.
ACTION	The ID indicating the type of user action. (Action type IDs are defined in the EDX_UMF_USER_ACTION_TYPE table.)

Table 115. EDX_UMF_USER_AUDIT Table

Column Name	Description
ACTION_DATE	The date of the user action.
TARGET_USER_ID	The ID of the target user. If the user performed the action on his or her own account (the target user is the same as the USER_ID), then the value in this column is null.
USER_ROLE	The role of the user who performed the action.
NOTES	The reason for locking an account: Incorrect Login, Reset Password, Security Question, or Account Expired. (User account reactivation only.)
ATTRIBUTES	The changed attribute.
IP_ADDRESS	The IP address of the user who accessed the Self-Service application.

Table 116 Describes the user action type table, EDX_UMF_USER_ACTION_TYPE:

Table 116. Definition of the EDX_UMF_USER_ACTION_TYPE Table

Column Name	Description
ID	ID associated with the user action type.
TYPE	User action type.
RESOURCE_KEY	Key for the language resource bundle.

Table 117 shows the ID associated with each type of user action. These associations are stored in the EDX_UMF_USER_ACTION_TYPE Table:

Table 117. ID of User Action Types Stored in the EDX_UMF_USER_ACTION_TYPE Table

ID	User Action Type
1	Reactivate
2	Impersonate-login
3	Impersonate-logout
4	Login
5	Logout
6	Update user profile
7	Update notifications
9	Reset password
10	Create user
13	Enroll user

Table 117. ID of User Action Types Stored in the EDX_UMF_USER_ACTION_TYPE Table

ID	User Action Type
16	Delete user
17	Login failure

Table 118 shows the user action type recorded in the EDX_UMF_USER_AUDIT table for each user enrollment activity in Oracle Billing Insight.

Table 118. User Action Types Used for Each User Enrollment Activity

User Action Type ID and Description	Associated User Enrollment Activities	Valid Attributes
1 - Reactivate	An agent administrator reactivates a locked-out account.	Password
2-Impersonate-login	An agent user (administrator or agent) impersonates a business or consumer user logging in.	Null
3-Impersonate-logout	An agent user (administrator or agent) impersonates a business or consumer user logging out.	Null
4-Login	A business, consumer, or agent user logs in.	Null
5-Logout	A business, consumer, or agent user logs in.	Null
6 - Update user profile	A business, consumer, or agent user updates his or her own user access information.	One of the following: <ul style="list-style-type: none"> ■ One or more of the following: First name, last name, and email address ■ Password ■ Security question and answer
6 - Update user profile	A business user (administrator or manager) updates another business user's access information.	One of the following: <ul style="list-style-type: none"> ■ One or more of the following: First name, last name, and email address ■ Role

Table 118. User Action Types Used for Each User Enrollment Activity

User Action Type ID and Description	Associated User Enrollment Activities	Valid Attributes
6 - Update user profile	An agent administrator updates another agent user's access information.	One of the following: <ul style="list-style-type: none"> ■ One or more of the following: First name, last name, and email address ■ Role
6 - Update user profile	An agent user (administrator or CSR) updates a business or consumer user's access information by impersonation.	One of the following: <ul style="list-style-type: none"> ■ One or more of the following: First name, last name, and email address ■ Role
6 - Update user profile	A migrated business, consumer, or agent user creates a new HIPPA-compliant password.	Null
6 - Update user profile	A business, consumer, or agent user updates his or her own expired password.	Password
7 - Update notifications	<ul style="list-style-type: none"> ■ A business or consumer user updates his or her own notification settings. ■ An agent user (administrator or agent) updates a business or consumer user's notifications by impersonation. 	Notifications
9 - Reset password	A business, consumer, or agent user resets his or her forgotten password.	Password
10 - Create user	<ul style="list-style-type: none"> ■ A business administrator creates another business user's account. ■ A consumer user creates his or her own account. ■ The default agent administrator creates another agent administrator account. ■ An agent administrator creates another agent user's account. ■ An agent administrator creates a business administrator's account. ■ An agent administrator or user creates a business user's account while impersonating a business administrator. 	Null

Table 118. User Action Types Used for Each User Enrollment Activity

User Action Type ID and Description	Associated User Enrollment Activities	Valid Attributes
13 - Enroll user	<ul style="list-style-type: none"> ■ A business or consumer user enrolls. ■ An agent user (administrator or agent) enrolls. ■ A migrated business, consumer, or agent user creates a new HIPPA-compliant password. 	Null
	<ul style="list-style-type: none"> ■ A business, consumer, or agent user enrolls to complete the reactivation process after clicking the URL in an email notification. 	Security Question and Answer
16 - Delete user	<ul style="list-style-type: none"> ■ A business administrator user deletes another business user's account. ■ An agent user (administrator or agent) deletes a business or consumer user's account by impersonation. ■ An agent administrator deletes another agent user's account. 	Null
17 - Login failure	A business, consumer, or agent user login fails.	Null

Reporting on System Administrator Audit Data

Oracle Billing Insight audits the following system administrator actions performed in the Command Center application:

- Creating new jobs
- Updating jobs
- Removing jobs
- Scheduling jobs
- Running jobs
- Creating a new administrator
- Enrolling a new administrator
- Updating an administrator's information
- Adding, updating, and deleting payment settings
- Logging into and out of the Command Center

You can create customized reports on the audited administrator data. Oracle Billing Insight stores the audit data for these system administrator activities in the administrator activity table, ADMIN_ACTIVITY.

Table 119 describes the ADMIN_ACTIVITY table.

Table 119. Definition of the ADMIN_ACTIVITY Table

Column Name	Description
ACTIVITY_ID	Unique ID assigned to the each occurrence of a Command Center activity, generated automatically by sequence.
ACTIVITY_CODE	The name of the activity performed by the system administrator. See Table 120 on page 363 for a list of valid activity codes.
PRODUCT_CODE	The product name for the Command Center application. The default value is ESTATEMENT.
DDN_REFERENCE	The DDN reference number.
LOGIN_ID	The ID of the system administrator who logged into the Command Center to perform the activity.
ACTIVITY_START_TIME	The time the activity started.
ACTIVITY_END_TIME	The time the activity finished.
FLEX_FIELD1	For logging into and out of the Command Center: Whether the activity was successful (Yes or No). For creating, updating, removing, running, or scheduling a job: The name of the job.
FLEX_FIELD2	The IP address where the administrator logged in and performed the Command Center activity.
DATE_CREATED	Date when the audit record was created.
CREATED_BY	The ID of the system administrator who performed the activity.
UPDATE_DATE	This field is not currently used.
UPDATED_BY	This field is not currently used.

Table 120 shows the valid activity codes that can be stored in the ACTIVITY_CODE column in the ADMIN_ACTIVITY:

Table 120. Valid Activity Codes

Valid Activity Code	Description
LOGIN	An administrator logs in.
LOGOUT	An administrator logs out.

Table 120. Valid Activity Codes

Valid Activity Code	Description
CREATE JOB	An administrator creates a new job.
UPDATE JOB	An administrator updates job information.
DELETE JOB	An administrator deletes a job.
CREATE USER	The default administrator creates a new administrator user.
ENROLL USER	A newly created administrator user enrolls his or her own information.
UPDATE USER	An administrator updates his or her own information.
CREATE PAYMENT SETTINGS	An administrator creates payment settings.
UPDATE PAYMENT SETTINGS	An administrator updates payment settings.
DELETE PAYMENT SETTINGS	An administrator deletes payment settings.

13 Load Input File Specifications and Data Mapping

This chapter describes the load input file specifications and data mapping tasks. It includes the following topics:

- [Preprocessor Tasks on page 365](#)
- [Data File Loading Tasks on page 366](#)
- [About Data Load Processing on page 366](#)
- [File Record and Dimension Table Mapping on page 367](#)
- [File Record and Fact Table Mapping on page 367](#)
- [File Formats for Dimension-Level Information on page 368](#)
- [File Formats for Statement-Level Information on page 375](#)
- [File Formats for Account-Level Information on page 380](#)
- [File Formats for Service-Level Information on page 383](#)
- [File Formats for Service Detail-Level Information on page 387](#)
- [File Formats for Unbilled Detail-Level Information on page 391](#)
- [File Formats for Prepaid Information on page 393](#)
- [File Formats for Batch Provisioning Data on page 397](#)

NOTE: Before running core load tasks, you must process all billing data files using a customized preprocessor.

For information on running load processes and other jobs, see *Administration Guide for Oracle Billing Insight*.

Preprocessor Tasks

Oracle Billing Insight requires that all input bill data files conform to the file format specified in this section. The flat file format is pipe delimited. Each row in the file has specific record type associated with it indicating the type of the record. The preprocessor performs the following tasks:

- Converts data file from an outside billing system file format to the one Oracle Billing Insight uses.
- Converts all dimension value literal strings into dimension value business keys.

You can use many dimension values in a data file. For example, the data file might contain a record with the following text: service 781-359-1000 Peak 2000 minutes. In this record, Peak could be interpreted as one of Tariff dimension value, which might have a business key PEAK_CALL. For the load process to recognize this record indicating that the service with number 781-359-1000 made a total of 2000 call minutes during peak hours, Oracle Billing Insight expects the file to contain PEAK_CALL (the business KEY string rather than the word Peak, the literal string that came from the billing system).

Data File Loading Tasks

The following tasks must occur when loading a data file:

- Preload any new dimension values for dimension tables.
- Run the preprocessor for each file you are loading.

This approach enables bill files in different languages to be stored with the correct business key value, allowing for data to be aggregated at a later time.

NOTE: Files must be Unicode compliant so that Oracle Billing Insight can process data in multiple languages.

About Data Load Processing

The data load processes two types of data files:

- Files to populate any new dimension information, including the following 3-character record types (REC_TYPE): 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 260, 270, 280.
- Files to populate fact data into all the fact tables and some dimension tables, including the following 4-character record types (REC_TYPE): 0000, 1000, 1100, 1200, 1300, 2000, 2100, 3000, 3100, 3200, 3300, 3400, 3500, 4000, 4900, 5000, 5100, 9000, 9200, 9500.

File Record and Dimension Table Mapping

Table 121 shows the mapping between load record types and Oracle Billing Insight database tables.

Table 121. File Record and DIM Table Mapping

Record Type	Table Name	Prepopulated	Industry Application
100	EDX_RPT_PAYMENT_TYPE_DIM	Can be populated from a data file	Telco and Utility
110	EDX_RPT_ADJUSTMENT_TYPE_DIM	Can be populated from a data file	Telco and Utility
120	EDX_RPT_CHARGE_TYPE_DIM	Can be populated from a data file	Telco and Utility
130	EDX_RPT_SUB_CHARGE_TYPE_DIM	Can be populated from a data file	Telco and Utility
140	EDX_RPT_PLAN_TYPE_DIM	Can be populated from a data file	Telco
150	EDX_RPT_PRODUCT_DIM	Can be populated from a data file	Telco
150	EDX_RPT_SUB_PRODUCT_DIM	Can be populated from a data file	Telco
160	EDX_RPT_USAGE_TYPE_DIM	Can be populated from a data file	Telco and Utility
170	EDX_RPT_TARIFF_DIM	Can be populated from a data file	Telco
180	EDX_RPT_REGION_DIM	Can be populated from a data file	Telco
190	EDX_RPT_CARRIER_DIM	Can be populated from a data file	Telco
200	EDX_RPT_AREA_CD_DIM	Can be populated from a data file	Telco
210	EDX_RPT_SERVICE_TYPE_DIM	Can be populated from a data file	Telco and Utility
220	EDX_RPT_UNIT_DIM	Can be populated from a data file	Telco and Utility
230	EDX_RPT_CHARGE_NOTES_DIM	Can be populated from a data file	Telco and Utility
240	EDX_RPT_DIRECTION_DIM	Can be populated from a data file	Telco
260	EDX_RPT_COUNTRY_DIM	Can be populated from a data file	Telco and Utility
270	EDX_RPT_CURRENCY_TYPE_DIM	Can be populated from a data file	Telco and Utility
280	EDX_RPT_TIME_ZONE_DIM	Can be populated from a data file	Telco and Utility

File Record and Fact Table Mapping

Table 122 shows the mapping between load record types and fact tables.

Table 122. File Record and Fact Table Mapping

Record Type	Table Name	Prepopulated	Industry Application
0000	None	Fact Data	Telco and Utility
1000	EDX_RPT_STATEMENT_FACT	Fact Data	Telco and Utility
1100	EDX_RPT_STATEMENT_PAYMENT_FACT	Fact Data	Telco and Utility

Table 122. File Record and Fact Table Mapping

Record Type	Table Name	Prepopulated	Industry Application
1200	EDX_RPT_STATEMENT_ADJUST_FACT	Fact Data	Telco and Utility
1300	EDX_RPT_ADDRESS_DIM	Populates from data file	Telco and Utility
2000	EDX_RPT_ADDRESS_DIM	Populates from data file	Telco and Utility
2000	EDX_RPT_ACCOUNT_FACT	Fact Data	Telco and Utility
2100	EDX_RPT_ACCOUNT_CHARGE_FACT	Fact Data	Telco and Utility
3000	EDX_RPT_SERVICE_FACT	Fact Data	Telco and Utility
3100	EDX_RPT_SERVICE_CHARGE_FACT	Fact Data	Telco and Utility
3200	EDX_RPT_SERVICE_PRODUCT_FACT	Fact Data	Telco
3300	EDX_RPT_SERVICE_USAGE_FACT	Fact Data	Telco and Utility
3400	EDX_RPT_SERVICE_TARIFF_FACT	Fact Data	Telco
3500	EDX_RPT_SERVICE_MISC_FACT	Fact Data	Utility
4000	EDX_RPT_SERVICE_DETAIL_FACT	Fact Data	Telco
4900	EDX_RPT_UNBILLED_DETAIL_FACT	Fact Data	Telco
5000	EDX_RPT_PREMISE_DIM	Populates from data file	Utility
5100	EDX_RPT_CONSUM_FACT	Fact Data	Utility
9000	EDX_RPT_PREPAY_ACCOUNT_FACT	Fact Data	Telco
9200	EDX_RPT_PREPAY_PRODUCT_FACT	Fact Data	Telco
9500	EDX_RPT_PREPAY_DETAIL_FACT	Fact Data	Telco

File Formats for Dimension-Level Information

This topic shows the file formats for dimension-level information.

Table 123 shows the file format for payment type information.

Table 123. Payment Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
100	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
100	Payment Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_PAYMENT_TYPE_DIM	PAYMENT_TYPE_CD	Telco and Utility
100	Payment Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_PAYMENT_TYPE_DIM	PAYMENT_TYPE_NAME	Telco and Utility

Table 124 shows the file format for the adjustment type information.

Table 124. Adjustment Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
110	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
110	Adjustment Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_ADJUSTMENT_TYPE_DIM	ADJUSTMENT_TYPE_CD	Telco and Utility
110	Adjustment Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_ADJUSTMENT_TYPE_DIM	ADJUSTMENT_TYPE_NAME	Telco and Utility

Table 125 shows the file format for charge type information.

Table 125. Charge Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
120	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
120	Charge Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_CHARGE_TYPE_DIM	CHARGE_TYPE_CD	Telco and Utility
120	Charge Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_CHARGE_TYPE_DIM	CHARGE_TYPE_NAME	Telco and Utility

Table 126 shows the file format for subcharge type information.

Table 126. Subcharge Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
130	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
130	Sub Charge Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_SUB_CHARGE_TYPE_DIM	SUB_CHARGE_TYPE_CD	Telco and Utility
130	Sub Charge Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_SUB_CHARGE_TYPE_DIM	SUB_CHARGE_TYPE_NAME	Telco and Utility

Table 127 shows the file format for plan type information.

Table 127. Plan Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
140	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
140	Plan Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_PLAN_TYPE_DIM	PLAN_TYPE_CD	Telco
140	Plan Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_PLAN_TYPE_DIM	PLAN_TYPE_NAME	Telco

Table 128 shows the file format for product and subproduct type information.

Table 128. Product and Subproduct Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
150	Rec Type	1	4	VARCHAR2	No	REC_TYPE	None	None	Telco
150	Product Code	2	50	VARCHAR2	No	COL1	EDX_RPT_PRODUCT_DIM	PRODUCT_CD	Telco
150	Product Name	3	100	VARCHAR2	No	COL2	EDX_RPT_PRODUCT_DIM	PRODUCT_NAME	Telco
150	Subproduct Code	4	50	VARCHAR2	No	COL3	EDX_RPT_SUB_PRODUCT_DIM	SUB_PRODUCT_CD	Telco
150	Subproduct Name	5	100	VARCHAR2	No	COL4	EDX_RPT_SUB_PRODUCT_DIM	SUB_PRODUCT_NAME	Telco
150	Subproduct Usage	6	16,2	NUMBER	No	COL5	EDX_RPT_SUB_PRODUCT_DIM	SUB_PRODUCT_USAGE	Telco
150	Subproduct Unit (Not used by reporting)	7	20	VARCHAR2	No	COL6	EDX_RPT_SUB_PRODUCT_DIM	SUB_PRODUCT_UNIT	Telco
150	Product Note1	8	2000	VARCHAR2	No	COL7	EDX_RPT_PRODUCT_DIM	PRODUCT_NOTE1	Telco
150	Product Note2	9	2000	VARCHAR2	No	COL8	EDX_RPT_PRODUCT_DIM	PRODUCT_NOTE2	Telco
150	Product Note3	10	2000	VARCHAR2	No	COL9	EDX_RPT_PRODUCT_DIM	PRODUCT_NOTE3	Telco
150	Allowance Indicator (Not used by reporting)	11	1	VARCHAR2	No	COL10	EDX_RPT_SUB_PRODUCT_DIM	ALLOWANCE_ID	Telco
150	Add on Indicator	12	1	VARCHAR2	No	COL11	EDX_RPT_PRODUCT_DIM	ADD_ON_FLAG	Telco

Table 128. Product and Subproduct Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
150	Product Description	13	500	VARCHAR2	No	COL12	EDX_RPT_PRODUCT_DIM	PRODUCT_DESC	Telco
150	Subproduct Description	14	500	VARCHAR2	No	COL13	EDX_RPT_SUB_PRODUCT_DIM	SUB_PRODUCT_DESC	Telco

Table 129 shows the file format for service usage type information. For the Usage Type Category Key, select the Key value from the preconfigured list in the EDX_RPT_USAGE_CATEGORY_DIM table.

Table 129. Service Usage Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
160	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
160	Usage Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_USAGE_TYPE_DIM	USAGE_TYPE_CD	Telco and Utility
160	Usage Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_USAGE_TYPE_DIM	USAGE_TYPE_NAME	Telco and Utility
160	Usage Type Category Description	4	50	VARCHAR2	No	COL3	EDX_RPT_USAGE_TYPE_DIM	USAGE_TYPE_DESC	Telco and Utility
160	Usage Type Category Key	5	8	NUMBER	No	COL4	EDX_RPT_USAGE_TYPE_DIM	USAGE_TYPE_CATE_KEY	Telco

Table 130 shows the file format for tariff type information.

Table 130. Tariff Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
170	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
170	Tariff Code	2	50	VARCHAR2	No	COL1	EDX_RPT_TARIFF_DIM	TARIFF_CD	Telco
170	Tariff Name	3	100	VARCHAR2	No	COL2	EDX_RPT_TARIFF_DIM	TARIFF_NAME	Telco

Table 131 shows the file format for other type (dimension) related information.

Table 131. Other Type (Dimension) Related File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
180	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
180	Region Code	2	50	VARCHAR2	No	COL1	EDX_RPT_REGION_DIM	REGION_CD	Telco
180	Region Name	3	100	VARCHAR2	No	COL2	EDX_RPT_REGION_DIM	REGION_NAME	Telco
190	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
190	Carrier Code	2	50	VARCHAR2	No	COL1	EDX_RPT_CARRIER_DIM	CARRIER_CD	Telco
190	Carrier Name	3	100	VARCHAR2	No	COL2	EDX_RPT_CARRIER_DIM	CARRIER_NAME	Telco
190	Carrier SMS Gateway	4	255	VARCHAR2	No	COL3	EDX_RPT_CARRIER_DIM	CARRIER_SMS_GATEWAY Note: The service number must be enclosed in parentheses, as in O(Number)@Company-sms.de	Telco
190	Resource bundle Key	5	255	VARCHAR2	No	COL4	EDX_RPT_CARRIER_DIM	RESOURCE_BUNDLE_ID	Telco
200	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
200	Calling/ Called City and State	2	100	VARCHAR2	No	COL1	EDX_RPT_AREA_CD_DIM	AREA_CD	Telco
200	Calling/ Called Country	3	100	VARCHAR2	No	COL2	EDX_RPT_AREA_CD_DIM	COUNTRY_CD	Telco

Table 132 shows the file format for service type information.

Table 132. Service Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
210	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
210	Service Type Code	2	50	VARCHAR2	No	COL1	EDX_RPT_SERVICE_TYPE_DIM	SERVICE_TYPE_CD	Telco and Utility
210	Service Type Name	3	100	VARCHAR2	No	COL2	EDX_RPT_SERVICE_TYPE_DIM	SERVICE_TYPE_NAME	Telco and Utility

Table 133 shows the file format for unit type information.

Table 133. Unit Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
220	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
220	Unit Code	2	50	VARCHAR2	No	COL1	EDX_RPT_UNIT_DIM	UNIT_CD	Telco and Utility
220	Unit Name	3	100	VARCHAR2	No	COL2	EDX_RPT_UNIT_DIM	UNIT_NAME	Telco and Utility

Table 134 shows the file format for charge notes information.

Table 134. Charge Notes File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
230	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Utility
230	Notes Code	2	50	VARCHAR2	No	COL1	EDX_RPT_CHARGE_NOTES_DIM	NOTES_CD	Utility
230	Notes Name	3	100	VARCHAR2	No	COL2	EDX_RPT_CHARGE_NOTES_DIM	NOTES_NAME	Utility

Table 135 shows the file format for direction type information.

Table 135. Direction Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
240	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
240	Direction Code	2	50	VARCHAR2	No	COL1	EDX_RPT_DIRECTION_DIM	DIRECTION_CD	Telco
240	Direction Name	3	100	VARCHAR2	No	COL2	EDX_RPT_DIRECTION_DIM	DIRECTION_NAME	Telco

Table 136 shows the file format for country information.

Table 136. Country File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
260	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
260	Country Code	2	50	VARCHAR2	No	COL1	EDX_RPT_COUNTRY_DIM	COUNTRY_CD	Telco and Utility
260	Country Name	3	100	VARCHAR2	No	COL2	EDX_RPT_COUNTRY_DIM	COUNTRY_NAME	Telco and Utility
260	Country Desc	4	255	VARCHAR2	No	COL3	EDX_RPT_COUNTRY_DIM	COUNTRY_DESC	Telco and Utility

Table 137 shows the file format for currency information.

Table 137. Currency File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
270	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
270	Currency Code	2	50	VARCHAR2	No	COL1	EDX_RPT_CURRENCY_TYPE_DIM	CURRENCY_TYPE_CD	Telco and Utility
270	Currency Name	3	100	VARCHAR2	No	COL2	EDX_RPT_CURRENCY_TYPE_DIM	CURRENCY_TYPE_NAME	Telco and Utility
270	Currency Desc	4	255	VARCHAR2	No	COL3	EDX_RPT_CURRENCY_TYPE_DIM	CURRENCY_TYPE_DESC	Telco and Utility
270	Currency Type Symbol	15	30	VARCHAR2	No	COL14	EDX_RPT_CURRENCY_TYPE_DIM	CURRENCY_TYPE_SYMBOL	Telco and Utility
								Note: If this column is null, the currency symbol cannot display correctly for multiple languages. Instead, the currency code will appear.	
270	Currency Is Default	16	1	VARCHAR2	No	COL15	EDX_RPT_CURRENCY_TYPE_DIM	CURRENCY_IS_DEFAULT	Telco and Utility

Table 138 shows the file format for time zone information.

Table 138. Time Zone File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
280	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
280	Time Zone Code	2	50	VARCHAR2	No	COL1	EDX_RPT_TIME_ZONE_DIM	TIME_ZONE_CD	Telco and Utility
280	Time Zone Name	3	100	VARCHAR2	No	COL2	EDX_RPT_TIME_ZONE_DIM	TIME_ZONE_NAME	Telco and Utility
280	Time Zone Desc	4	255	VARCHAR2	No	COL3	EDX_RPT_TIME_ZONE_DIM	TIME_ZONE_DESC	Telco and Utility

File Formats for Statement-Level Information

This topic shows the file formats for statement-level information.

Table 139 shows the file format for control records for billed file information.

Table 139. Billed Control Record File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
0000	Rec Type	1	3	VARCHAR2	Yes	None	None	None	Telco and Utility
0000	HEADER TYPE	2	20	VARCHAR2	Yes	None	None	None	Telco and Utility
0000	BILLING SYSTEM	3	20	VARCHAR2	Yes	None	None	None	Telco and Utility
0000	Line of Business	4	20	VARCHAR2	Yes	None	None	WIRELESS for telecommunications and UTILITY for the utility industry	Telco and Utility
0000	Group ID/ Key	5	7	NUMBER	Yes	COL1	None	None	Telco and Utility
0000	Master Bill File Name	6	255	VARCHAR2	No	COL2	None	None	Telco and Utility

Table 140 shows the file format for control record for unbilled file information.

Table 140. Unbilled Control Record File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
0000	Rec Type	1	4	VARCHAR2	Yes	None	None	None	Telco
0000	HEADER TYPE	2	20	VARCHAR2	Yes	None	None	None	Telco
0000	BILL SYSTEM	3	20	VARCHAR2	Yes	None	None	None	Telco
0000	Line of Business	4	20	VARCHAR2	Yes	None	WIRELESS	None	Telco
0000	File Type	5	20	VARCHAR2	Yes	COL1	UNBILLED	None	Telco

Table 141 shows the file format for control record for prepaid file information.

Table 141. Prepaid Control Record File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
0000	Rec Type	1	4	VARCHAR2	Yes	None	None	None	Telco
0000	HEADER TYPE	2	20	VARCHAR2	Yes	None	None	None	Telco
0000	BILL SYSTEM	3	20	VARCHAR2	Yes	None	None	None	Telco
0000	Line of Business	4	20	VARCHAR2	Yes	None	WIRELESS	None	Telco
0000	File Type	5	20	VARCHAR2	Yes	COL1	PREPAY	None	Telco

Table 142 shows the file format for summary-level detail information.

Table 142. Summary-Level Detail Record File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1000	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
1000	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_STATEMENT_FACT	STATEMENT_NUMBER	Telco and Utility
1000	Company ID	3	20	VARCHAR2	No	COL2	EDX_RPT_COMPANY_DIM	COMPANY_CD	Telco and Utility

Table 142. Summary-Level Detail Record File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1000	Company Name	4	255	VARCHAR2	No	COL3	EDX_RPT_COMPANY_DIM	COMPANY_NAME	Telco and Utility
1000	Statement Date	5	8	DATE	No	COL4	EDX_RPT_STATEMENT_FACT	STATEMENT_DATE	Telco and Utility
1000	Billing Cycle Start Date	6	8	DATE	Yes	COL5	EDX_RPT_STATEMENT_FACT	BILL_CYCLE_START_DATE	Telco and Utility
1000	Billing Cycle End Date	7	8	DATE	Yes	COL6	EDX_RPT_STATEMENT_FACT	BILL_CYCLE_END_DATE	Telco and Utility
1000	Previous Balance	8	16,2	NUMBER	No	COL7	EDX_RPT_STATEMENT_FACT	PREVIOUS_BALANCE	Telco and Utility
1000	Total Payment Posted	9	16,2	NUMBER	No	COL8	EDX_RPT_STATEMENT_FACT	TOTAL_PAYMENT_POSTED	Telco and Utility
1000	Total Adjustments	10	16,2	NUMBER	No	COL9	EDX_RPT_STATEMENT_FACT	TOTAL_ADJUSTMENTS	Telco
1000	Balance Forward Due	11	16,2	NUMBER	No	COL10	EDX_RPT_STATEMENT_FACT	BALANCE_FORWARD_DUE	Telco and Utility
1000	Total Current Charge Due	12	16,2	NUMBER	No	COL11	EDX_RPT_STATEMENT_FACT	TOTAL_CURRENT_CHARGE_DUE	Telco and Utility
1000	Total Amount Due	13	16,2	NUMBER	No	COL12	EDX_RPT_STATEMENT_FACT	TOTAL_AMOUNT_DUE	Telco and Utility
1000	Monthly Service Charges	14	16,2	NUMBER	No	COL13	EDX_RPT_STATEMENT_FACT	MONTHLY_CHARGE_AMT	Telco
1000	Usage Charges	15	16,2	NUMBER	No	COL14	EDX_RPT_STATEMENT_FACT	USAGE_CHARGE_AMT	Telco
1000	Credits	16	16,2	NUMBER	No	COL15	EDX_RPT_STATEMENT_FACT	CREDIT_ADJUST_AMT	Telco and Utility
1000	Other Charges	17	16,2	NUMBER	No	COL16	EDX_RPT_STATEMENT_FACT	OTHER_CHARGE_AMT	Telco
1000	Taxes and Fees	18	16,2	NUMBER	No	COL17	EDX_RPT_STATEMENT_FACT	TAXES_SURCHARGES_FEE	Telco
1000	Flex Field_1	19	16,2	NUMBER	No	COL18	EDX_RPT_STATEMENT_FACT	FLEX_FIELD1	Telco and Utility
1000	Flex Field_2	20	16,2	NUMBER	No	COL19	EDX_RPT_STATEMENT_FACT	FLEX_FIELD2	Telco and Utility
1000	Flex Field_3	21	16,2	NUMBER	No	COL20	EDX_RPT_STATEMENT_FACT	FLEX_FIELD3	Telco and Utility
1000	Flex Field_4	22	16,2	NUMBER	No	COL21	EDX_RPT_STATEMENT_FACT	FLEX_FIELD4	Telco and Utility
1000	Flex Field_5	23	16,2	NUMBER	No	COL22	EDX_RPT_STATEMENT_FACT	FLEX_FIELD5	Telco and Utility
1000	Minimum Amount Due	24	16,2	NUMBER	No	COL23	EDX_RPT_STATEMENT_FACT	MINIMUM_DUE_AMT	Telco
1000	Statement Due Date	25	8	DATE	No	COL24	EDX_RPT_STATEMENT_FACT	STATEMENT_DUE_DATE	Telco and Utility
1000	Statement Currency	26	50	VARCHAR2	No	COL25	EDX_RPT_STATEMENT_FACT	CURRENCY_TYPE_CD	Telco and Utility

Table 142. Summary-Level Detail Record File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1000	Statement Country	27	50	VARCHAR2	No	COL26	EDX_RPT_STATEMENT_FACT	COUNTRY_CD	Telco
1000	Statement Time Zone	28	50	VARCHAR2	No	COL27	EDX_RPT_STATEMENT_FACT	TIME_ZONE_CD	Telco
1000	Note1	29	2010	VARCHAR2	No	COL28	EDX_RPT_STATEMENT_FACT	NOTE1	Telco and Utility
1000	Note2	30	2010	VARCHAR2	No	COL29	EDX_RPT_STATEMENT_FACT	NOTE2	Telco and Utility
1000	Note3	31	2010	VARCHAR2	No	COL30	EDX_RPT_STATEMENT_FACT	NOTE3	Telco and Utility
1000	Note4	32	2010	VARCHAR2	No	COL31	EDX_RPT_STATEMENT_FACT	NOTE4	Telco and Utility
1000	Note5	33	2010	VARCHAR2	No	COL32	EDX_RPT_STATEMENT_FACT	NOTE5	Telco and Utility
1000	MEDIA_TYPE	34	50	VARCHAR2	No	COL33	EDX_RPT_STATEMENT_FACT	MEDIA_TYPE	Telco
1000	Corporation Account No	35	255	VARCHAR2	No	COL34	EDX_RPT_COMPANY_DIM	FLEX_FIELD1	Telco
1000	Corporation Tax ID	36	255	VARCHAR2	No	COL35	EDX_RPT_COMPANY_DIM	FLEX_FIELD2	Telco
1000	Flex Field_6	37	255	VARCHAR2	No	COL36	EDX_RPT_STATEMENT_FACT	FLEX_FIELD6	Telco and Utility
1000	Flex Field_7	38	255	VARCHAR2	No	COL37	EDX_RPT_STATEMENT_FACT	FLEX_FIELD7	Telco and Utility
1000	Flex Field_8	39	255	VARCHAR2	No	COL38	EDX_RPT_STATEMENT_FACT	FLEX_FIELD8	Telco and Utility
1000	Company Display Name	42	255	VARCHAR2	No	COL41	EDX_RPT_COMPANY_DIM	FLEX_FIELD3	Telco

Table 143 shows the file format for statement payment fact information.

Table 143. Statement Payment Fact File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1100	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
1100	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_STATEMENT_PAYMENTS_FACT	STATEMENT_KEY	Telco and Utility
1100	Payment Type Code	3	50	VARCHAR2	Yes	COL2	EDX_RPT_STATEMENT_PAYMENTS_FACT	PAYMENT_TYPE_KEY	Telco and Utility

Table 143. Statement Payment Fact File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1100	Payment Amount	4	16,2	NUMBER	Yes	COL3	EDX_RPT_STATEMENT_PAYMENT_FACT	PAYMENT_AMOUNT	Telco and Utility
1100	Payment Date	5	8	DATE	No	COL4	EDX_RPT_STATEMENT_PAYMENT_FACT	PAYMENT_DATE	Telco and Utility
1100	Payment Note	6	255	VARCHAR2	No	COL5	EDX_RPT_STATEMENT_PAYMENT_FACT	PAYMENT_NOTE	Telco and Utility

Table 144 shows the file format for statement adjustment fact information.

Table 144. Statement Adjustment Fact File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1200	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
1200	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_STATEMENT_ADJUST_FACT	STATEMENT_KEY	Telco and Utility
1200	Adjustment Type Code	3	50	VARCHAR2	Yes	COL2	EDX_RPT_STATEMENT_ADJUST_FACT	ADJUSTMENT_TYPE_KEY	Telco and Utility
1200	Adjustment Amount	4	16,2	NUMBER	Yes	COL3	EDX_RPT_STATEMENT_ADJUST_FACT	ADJUSTMENT_AMOUNT	Telco and Utility
1200	Adjustment Date	5	8	DATE	No	COL4	EDX_RPT_STATEMENT_ADJUST_FACT	ADJUSTMENT_DATE	Telco and Utility
1200	Service Number	6	20	VARCHAR2	No	COL5	EDX_RPT_STATEMENT_ADJUST_FACT	SERVICE_NUMBER	Telco and Utility
1200	Adjustment Note	7	255	VARCHAR2	No	COL6	EDX_RPT_STATEMENT_ADJUST_FACT	ADJUSTMENT_NOTE	Telco and Utility

Table 145 shows the file format for statement mail to address type information.

Table 145. Statement Mail to Address File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
1300	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
1300	Statement Number	2	20	VARCHAR2	Yes	COL1	None	None	Telco and Utility
1300	Statement Address Line 1	3	50	VARCHAR2	No	COL2	EDX_RPT_ADDRESS_DIM	ADDRESS1	Telco and Utility
1300	Statement Address Line 2	4	50	VARCHAR2	No	COL3	EDX_RPT_ADDRESS_DIM	ADDRESS2	Telco and Utility
1300	Statement Address Line 3	5	50	VARCHAR2	No	COL4	EDX_RPT_ADDRESS_DIM	ADDRESS3	Telco and Utility
1300	Statement City	6	50	VARCHAR2	No	COL5	EDX_RPT_ADDRESS_DIM	CITY	Telco and Utility
1300	Statement State	7	50	VARCHAR2	No	COL6	EDX_RPT_ADDRESS_DIM	STATE	Telco and Utility
1300	Statement Postal Code	8	20	VARCHAR2	No	COL7	EDX_RPT_ADDRESS_DIM	ZIP_CD	Telco and Utility
1300	Statement Country Code	9	20	VARCHAR2	No	COL8	EDX_RPT_ADDRESS_DIM	COUNTRY	Telco and Utility

File Formats for Account-Level Information

This topic shows the account level file formats.

Table 146 shows the file format for account fact charge (charge summary for account) information.

Table 146. Account Fact Charges (Charge Summary for Account) File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
2000	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
2000	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_ACCOUNT_FACT	STATEMENT_KEY	Telco and Utility
2000	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_ACCOUNT_DIM	ACCOUNT_NUM	Telco and Utility

Table 146. Account Fact Charges (Charge Summary for Account) File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
2000	Account Owner Name	4	255	VARCHAR2	Yes	COL3	EDX_RPT_ACCOUNT_DIM	CONTACT_NAME	Telco and Utility
2000	Address1	5	200	VARCHAR2	No	COL4	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	ADDRESS1, ADDRESS1	Telco and Utility
2000	Address2	6	200	VARCHAR2	No	COL5	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	ADDRESS2, ADDRESS2	Telco and Utility
2000	City	7	100	VARCHAR2	No	COL6	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	CITY, CITY	Telco and Utility
2000	State	8	100	VARCHAR2	No	COL7	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	STATE, STATE	Telco and Utility
2000	Country	9	100	VARCHAR2	No	COL8	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	COUNTRY, COUNTRY	Telco and Utility
2000	Zip	10	20	VARCHAR2	No	COL9	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	ZIP, ZIP	Telco and Utility
2000	Monthly Service Charges	11	16,2	NUMBER	Yes	COL 10	EDX_RPT_ACCOUNT_FACT	MONTHLY_CHARGE_AMT	Telco and Utility
2000	Usage Charges	12	16,2	NUMBER	Yes	COL 11	EDX_RPT_ACCOUNT_FACT	USAGE_CHARGE_AMT	Telco and Utility
2000	Credits	13	16,2	NUMBER	Yes	COL 12	EDX_RPT_ACCOUNT_FACT	CREDIT_ADJUST_AMT	Telco and Utility
2000	Other Charges	14	16,2	NUMBER	Yes	COL 13	EDX_RPT_ACCOUNT_FACT	OTHER_CHARGE_AMT	Telco and Utility
2000	Taxes and Fees	15	16,2	NUMBER	Yes	COL 14	EDX_RPT_ACCOUNT_FACT	TAXES_SURCHARGES_FEE	Telco and Utility
2000	Total Charge Amount	16	16,2	NUMBER	Yes	COL15	EDX_RPT_ACCOUNT_FACT	TOTAL_CHARGE_AMT	Telco and Utility
2000	Charge Flag	17	1	VARCHAR2	Yes	COL16	EDX_RPT_ACCOUNT_FACT	CHARGE_FLAG	Telco and Utility
2000	Account Type	18	20	VARCHAR2	No	COL17	EDX_RPT_ACCOUNT_DIM	ACCOUNT_TYPE_KEY	Telco and Utility
2000	Flex Field_1	19	16,2	NUMBER	No	COL18	EDX_RPT_ACCOUNT_FACT	FLEX_FIELD1	Telco and Utility
2000	Flex Field_2	20	16,2	NUMBER	No	COL19	EDX_RPT_ACCOUNT_FACT	FLEX_FIELD2	Telco and Utility
2000	Flex Field_3	21	16,2	NUMBER	No	COL20	EDX_RPT_ACCOUNT_FACT	FLEX_FIELD3	Telco and Utility

Table 146. Account Fact Charges (Charge Summary for Account) File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
2000	Flex Field_4	22	16,2	NUMBER	No	COL21	EDX_RPT_ACCOUNT_FACT	FLEX_FIELD4	Telco and Utility
2000	Flex Field_5	23	16,2	NUMBER	No	COL22	EDX_RPT_ACCOUNT_FACT	FLEX_FIELD5	Telco and Utility
2000	Address3	24	200	VARCHAR2	No	COL23	EDX_RPT_ACCOUNT_DIM, EDX_RPT_ADDRESS_DIM	ADDRESS3, ADDRESS3	Telco and Utility
2000	Auto Debit Payment Date	25	8	DATE	No	COL24	EDX_RPT_ACCOUNT_FACT	AUTO_PAYMENT_DR_DT	Utility
2000	Cycle End Day	26	2	NUMBER	No	COL25	EDX_RPT_ACCOUNT_DIM	BILL_CYCLE_ENDDAY	Telco

Table 147 shows the file format for account level charges at the charge type level information.

Table 147. Account Level Charges at Charge Type Level File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
2100	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
2100	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_ACCOUNT_CHARGE_FACT	STATEMENT_KEY	Telco and Utility
2100	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_ACCOUNT_CHARGE_FACT	ACCOUNT_KEY	Telco and Utility
2100	Charge Type Code	4	50	VARCHAR2	Yes	COL3	EDX_RPT_ACCOUNT_CHARGE_FACT	CHARGE_TYPE_KEY	Telco and Utility
2100	Monthly Service Charges	5	16,2	NUMBER	Yes	COL4	EDX_RPT_ACCOUNT_CHARGE_FACT	MONTHLY_CHARGE_AMT	Telco and Utility
2100	Usage Charges	6	16,2	NUMBER	Yes	COL5	EDX_RPT_ACCOUNT_CHARGE_FACT	USAGE_CHARGE_AMT	Telco and Utility
2100	Credits	7	16,2	NUMBER	Yes	COL6	EDX_RPT_ACCOUNT_CHARGE_FACT	CREDIT_ADJUST_AMT	Telco and Utility
2100	Other Charges	8	16,2	NUMBER	Yes	COL7	EDX_RPT_ACCOUNT_CHARGE_FACT	OTHER_CHARGE_AMT	Telco and Utility
2100	Taxes and Fees	9	16,2	NUMBER	Yes	COL8	EDX_RPT_ACCOUNT_CHARGE_FACT	TAXES_SURCHARGES_FEE	Telco and Utility
2100	Total Charge Amount	10	16,2	NUMBER	Yes	COL9	EDX_RPT_ACCOUNT_CHARGE_FACT	TOTAL_CHARGE_AMT	Telco and Utility
2100	Charge Note	11	255	VARCHAR2	No	COL10	EDX_RPT_ACCOUNT_CHARGE_FACT	CHARGE_NOTE	Telco and Utility

File Formats for Service-Level Information

This topic shows the file formats for service-level data.

Table 148 shows the file format for service level total information.

Table 148. Service Level Total File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3000	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
3000	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_FACT	STATEMENT_KEY	Telco and Utility
3000	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_DIM	ACCOUNT_KEY	Telco and Utility
3000	Service Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_DIM	SERVICE_NUM	Telco and Utility
3000	Service Owner Name	5	100	VARCHAR2	No	COL4	None	None	Telco and Utility
3000	Monthly Service Charges	6	16,2	NUMBER	Yes	COL5	EDX_RPT_SERVICE_FACT	MONTHLY_CHARGE_AMT	Telco and Utility
3000	Usage Charges	7	16,2	NUMBER	Yes	COL6	EDX_RPT_SERVICE_FACT	USAGE_CHARGE_AMT	Telco and Utility
3000	Adjustments	8	16,2	NUMBER	Yes	COL7	EDX_RPT_SERVICE_FACT	CREDIT_ADJUST_AMT	Telco and Utility
3000	Other Charges	9	16,2	NUMBER	Yes	COL8	EDX_RPT_SERVICE_FACT	OTHER_CHARGE_AMT	Telco and Utility
3000	Taxes and Fees	10	16,2	NUMBER	Yes	COL9	EDX_RPT_SERVICE_FACT	TAXES_SURCHARGES_FEE	Telco and Utility
3000	Total Charge Amount	11	16,2	NUMBER	Yes	COL10	EDX_RPT_SERVICE_FACT	TOTAL_CHARGE_AMT	Telco and Utility
3000	Charge Note	12	255	VARCHAR2	No	COL11	EDX_RPT_SERVICE_FACT	CHARGE_NOTE	Telco and Utility
3000	Flex Field_1	13	16,2	NUMBER	No	COL12	EDX_RPT_SERVICE_FACT	FLEX_FIELD1	Telco and Utility
3000	Flex Field_2	14	16,2	NUMBER	No	COL13	EDX_RPT_SERVICE_FACT	FLEX_FIELD2	Telco and Utility
3000	Flex Field_3	15	16,2	NUMBER	No	COL14	EDX_RPT_SERVICE_FACT	FLEX_FIELD3	Telco and Utility
3000	Flex Field_4	16	16,2	NUMBER	No	COL15	EDX_RPT_SERVICE_FACT	FLEX_FIELD4	Telco and Utility
3000	Flex Field_5	17	16,2	NUMBER	No	COL16	EDX_RPT_SERVICE_FACT	FLEX_FIELD5	Telco and Utility
3000	Service Type	18	50	VARCHAR2	No	COL17	EDX_RPT_SERVICE_DIM	SERVICE_TYPE_KEY	Utility
3000	Service Start Date	19	8	DATE	No	COL18	EDX_RPT_SERVICE_FACT	START_DT	Utility

Table 148. Service Level Total File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3000	Service End Date	20	8	DATE	No	COL19	EDX_RPT_SERVICE_FACT	END_DT	Utility
3000	Premise ID	21	20	VARCHAR2	No	COL20	EDX_RPT_SERVICE_FACT	PREMISE_KEY	Utility
3000	Zoning	22	21	VARCHAR2	No	COL21	EDX_RPT_SERVICE_DIM	ZONING	Utility

Table 149 shows the file format for the service level on charge type information.

Table 149. Service Level on Charge Type Level File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3100	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
3100	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_CHARGE_FACT	STATEMENT_KEY	Telco and Utility
3100	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_CHARGE_FACT	ACCOUNT_KEY	Telco and Utility
3100	Service Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_CHARGE_FACT	SERVICE_KEY	Telco and Utility
3100	Charge Type Code	5	50	VARCHAR2	Yes	COL4	EDX_RPT_SERVICE_CHARGE_FACT	CHARGE_TYPE_KEY	Telco and Utility
3100	Sub Charge Type Code	6	50	VARCHAR2	No	COL5	EDX_RPT_SERVICE_CHARGE_FACT	SUB_CHARGE_TYPE_KEY	Telco and Utility
3100	Monthly Charge Amount	7	10	NUMBER	Yes	COL6	EDX_RPT_SERVICE_CHARGE_FACT	CHARGE_AMT	Telco and Utility
3100	Charge Note	8	255	VARCHAR2	No	COL7	EDX_RPT_SERVICE_CHARGE_FACT	CHARGE_NOTE	Telco and Utility
3100	Service Start Date	9	8	DATE	No	COL8	EDX_RPT_SERVICE_CHARGE_FACT	START_DT	Utility
3100	Service End Date	10	8	DATE	No	COL9	EDX_RPT_SERVICE_CHARGE_FACT	END_DT	Utility
3100	Charge Note Code	11	50	VARCHAR2	No	COL10	EDX_RPT_SERVICE_CHARGE_FACT	NOTES_KEY	Utility

Table 150 shows the file format for the service charge on product and plan information.

Table 150. Service Charge on Product and Plan File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3200	Rec Type	1	4	VARCHAR2	No	REC_TYPE	None	None	Telco
3200	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_PRODUCT_FACT	STATEMENT_KEY	Telco
3200	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_PRODUCT_FACT	ACCOUNT_KEY	Telco
3200	Service Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_PRODUCT_FACT	SERVICE_KEY	Telco
3200	Plan Type Code	5	50	VARCHAR2	Yes	COL4	EDX_RPT_SERVICE_PRODUCT_FACT	PLAN_TYPE_KEY	Telco
3200	Product Code	6	50	VARCHAR2	Yes	COL5	EDX_RPT_SERVICE_PRODUCT_FACT	PRODUCT_PARENT_KEY	Telco
3200	Monthly Charge Amount	7	16,2	NUMBER	Yes	COL6	EDX_RPT_SERVICE_PRODUCT_FACT	CHARGE_AMT	Telco
3200	Product Note	8	255	VARCHAR2	No	COL7	EDX_RPT_SERVICE_PRODUCT_FACT	PRODUCT_NOTE	Telco
3200	Product Subcode	9	50	VARCHAR2	Yes	COL8	EDX_RPT_SERVICE_PRODUCT_FACT	PRODUCT_CHILD_KEY	Telco

Table 151 shows the file format for service charge on usage type information.

Table 151. Service Charge on Usage Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3300	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco and Utility
3300	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_USAGE_FACT	STATEMENT_KEY	Telco and Utility
3300	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_USAGE_FACT	ACCOUNT_KEY	Telco and Utility
3300	Service Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_USAGE_FACT	SERVICE_KEY	Telco and Utility
3300	Usage Type Code	5	50	VARCHAR2	Yes	COL4	EDX_RPT_SERVICE_USAGE_FACT	USAGE_TYPE_KEY	Telco and Utility
3300	Total Usage	6	16,2	NUMBER	Yes	COL5	EDX_RPT_SERVICE_USAGE_FACT	TOTAL_USAGE	Telco and Utility
3300	Usage Unit Code	7	50	VARCHAR2	Yes	COL6	EDX_RPT_SERVICE_USAGE_FACT	UNIT_KEY	Telco and Utility
3300	Amount	8	16,2	NUMBER	Yes	COL7	EDX_RPT_SERVICE_USAGE_FACT	TOTAL_CHARGE_AMT	Telco and Utility

Table 151. Service Charge on Usage Type File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3300	Usage Note	9	255	VARCHAR2	No	COL8	EDX_RPT_SERVICE_USAGE_FACT	USAGE_NOTE	Telco and Utility
3300	Gross Amount	10	16,2	NUMBER	No	COL9	EDX_RPT_SERVICE_USAGE_FACT	GROSS_AMT	Telco and Utility
3300	Discount Amount	11	16,2	NUMBER	No	COL10	EDX_RPT_SERVICE_USAGE_FACT	DISCOUNT_AMT	Telco and Utility
3300	Tax Amount	12	16,2	NUMBER	No	COL11	EDX_RPT_SERVICE_USAGE_FACT	TAX_AMT	Telco and Utility
3300	Total Call Count	13	16	NUMBER	No	COL12	EDX_RPT_SERVICE_USAGE_FACT	TOTAL_CALL_CNT	Telco and Utility

Table 152 shows the file format for the service charges on various tariff information.

Table 152. Service Charges on Various Tariffs File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3400	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
3400	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_TARIFF_FACT	STATEMENT_KEY	Telco
3400	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_TARIFF_FACT	ACCOUNT_KEY	Telco
3400	Service Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_TARIFF_FACT	SERVICE_KEY	Telco
3400	Usage Type Code	5	50	VARCHAR2	Yes	COL4	EDX_RPT_SERVICE_TARIFF_FACT	USAGE_TYPE_KEY	Telco
3400	Tariff Code	6	50	VARCHAR2	Yes	COL5	EDX_RPT_SERVICE_TARIFF_FACT	TARIFF_KEY	Telco
3400	Allowance	7	20	VARCHAR2	Yes	COL6	EDX_RPT_SERVICE_TARIFF_FACT	ALLOWANCE	Telco
3400	Total Usages	8	16,2	NUMBER	Yes	COL7	EDX_RPT_SERVICE_TARIFF_FACT	TOTAL_USAGE	Telco
3400	Usage Unit Code	9	50	VARCHAR2	Yes	COL8	EDX_RPT_SERVICE_TARIFF_FACT	UNIT_KEY	Telco
3400	Billable	10	16,2	NUMBER	Yes	COL9	EDX_RPT_SERVICE_TARIFF_FACT	BILLABLE	Telco
3400	Amount	11	16,2	NUMBER	Yes	COL10	EDX_RPT_SERVICE_TARIFF_FACT	CHARGE_AMT	Telco
3400	Gross Amount	12	16,2	NUMBER	No	COL11	EDX_RPT_SERVICE_TARIFF_FACT	GROSS_AMT	Telco and Utility
3400	Discount Amount	13	16,2	NUMBER	No	COL12	EDX_RPT_SERVICE_TARIFF_FACT	DISCOUNT_AMT	Telco and Utility
3400	Tax Amount	14	16,2	NUMBER	No	COL13	EDX_RPT_SERVICE_TARIFF_FACT	TAX_AMT	Telco and Utility

Table 153 shows the services miscellaneous charges type information.

Table 153. Services Miscellaneous Charges File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
3500	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Utility
3500	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_MISC_FACT	STATEMENT_KEY	Utility
3500	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_MISC_FACT	ACCOUNT_KEY	Utility
3500	Service Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_MISC_FACT	SERVICE_KEY	Utility
3500	Charge Description	5	255	VARCHAR2	No	COL4	EDX_RPT_SERVICE_MISC_FACT	MISC_COMMENT	Utility
3500	Total Amount	6	16,2	NUMBER	Yes	COL5	EDX_RPT_SERVICE_MISC_FACT	MISC_AMOUNT	Utility

File Formats for Service Detail-Level Information

This topic shows the file formats for service detail-level data.

Table 154 shows the file format for service call usage detail (voice, data, message) information.

Table 154. Service Call Usage Detail (Voice, Data, Message) File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
4000	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Telco
4000	Usage Name	2	50	VARCHAR2	Yes	COL1	EDX_RPT_SERVICE_DETAIL_FACT	USAGE_TYPE_KEY	Telco
4000	Statement Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_SERVICE_DETAIL_FACT	STATEMENT_KEY	Telco
4000	Account Number	4	20	VARCHAR2	Yes	COL3	EDX_RPT_SERVICE_DETAIL_FACT	ACCOUNT_KEY	Telco
4000	Service Number	5	20	VARCHAR2	Yes	COL4	EDX_RPT_SERVICE_DETAIL_FACT	SERVICE_KEY	Telco
4000	Called Date	6	8	DATE	Yes	COL5	EDX_RPT_SERVICE_DETAIL_FACT	DATE_KEY	Telco
4000	Called Time	7	10	VARCHAR2	Yes	COL6	EDX_RPT_SERVICE_DETAIL_FACT	DURATION	Telco
4000	Called Number	8	20	VARCHAR2	Yes	COL7	EDX_RPT_SERVICE_DETAIL_FACT	CALLED_NUM	Telco
4000	Tariff Code	9	50	VARCHAR2	Yes	COL8	EDX_RPT_SERVICE_DETAIL_FACT	TARIFF_KEY	Telco

Table 154. Service Call Usage Detail (Voice, Data, Message) File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
4000	Call Type Code	10	50	VARCHAR2	No	COL9	EDX_RPT_SERVICE_DETAIL_FACT	CALL_TYPE_KEY	Telco
4000	Direction Code	11	20	VARCHAR2	Yes	COL10	EDX_RPT_SERVICE_DETAIL_FACT	DIRECTION_KEY	Telco
4000	Service Type Code	12	50	VARCHAR2	No	COL11	EDX_RPT_SERVICE_DETAIL_FACT	SERVICE_TYPE_KEY	Telco
4000	Total Usages	13	16,2	NUMBER	Yes	COL12	EDX_RPT_SERVICE_DETAIL_FACT	TOTAL_USAGE	Telco
4000	Usages Unit Code	14	50	VARCHAR2	Yes	COL13	EDX_RPT_SERVICE_DETAIL_FACT	UNIT_KEY	Telco
4000	Other Charge	15	16,2	NUMBER	Yes	COL14	EDX_RPT_SERVICE_DETAIL_FACT	OTHER_CHARGE_AMT	Telco
4000	Total Charge	16	16,2	NUMBER	Yes	COL15	EDX_RPT_SERVICE_DETAIL_FACT	TOTAL_CHARGE_AMT	Telco
4000	Calling City and State	17	100	VARCHAR2	Yes	COL16	EDX_RPT_SERVICE_DETAIL_FACT	CALLING_AREA_CD_KEY	Telco
4000	Calling Country Code	18	100	VARCHAR2	Yes	COL17	EDX_RPT_SERVICE_DETAIL_FACT	CALLING_AREA_CD_KEY	Telco
4000	Reference Number	19	100	VARCHAR2	No	COL18	EDX_RPT_SERVICE_DETAIL_FACT	REFERENCE_NUM	Telco
4000	Carrier Code	20	50	VARCHAR2	No	COL19	EDX_RPT_SERVICE_DETAIL_FACT	CARRIER_KEY	Telco
4000	Region Code	21	50	VARCHAR2	No	COL20	EDX_RPT_SERVICE_DETAIL_FACT	REGION_KEY	Telco
4000	Note1	22	2000	VARCHAR2	No	COL21	EDX_RPT_SERVICE_DETAIL_FACT	NOTE1	Telco
4000	Note2	23	2000	VARCHAR2	No	COL22	EDX_RPT_SERVICE_DETAIL_FACT	NOTE2	Telco
4000	Note3	24	2000	VARCHAR2	No	COL23	EDX_RPT_SERVICE_DETAIL_FACT	NOTE3	Telco
4000	Note4	25	2000	VARCHAR2	No	COL24	EDX_RPT_SERVICE_DETAIL_FACT	NOTE4	Telco
4000	Note5	26	2000	VARCHAR2	No	COL25	EDX_RPT_SERVICE_DETAIL_FACT	NOTE5	Telco
4000	Called City and State	27	100	VARCHAR2	Yes	COL26	EDX_RPT_SERVICE_DETAIL_FACT	CALLED_AREA_CD_KEY	Telco
4000	Called Country Code	28	100	VARCHAR2	Yes	COL27	EDX_RPT_SERVICE_DETAIL_FACT	CALLED_AREA_CD_KEY	Telco
4000	Gross Amount	29	16,2	NUMBER	No	COL28	EDX_RPT_SERVICE_DETAIL_FACT	GROSS_AMT	Telco and Utility
4000	Discount Amount	30	16,2	NUMBER	No	COL29	EDX_RPT_SERVICE_DETAIL_FACT	DISCOUNT_AMT	Telco and Utility
4000	Tax Amount	31	16,2	NUMBER	No	COL30	EDX_RPT_SERVICE_DETAIL_FACT	TAX_AMT	Telco and Utility
4000	FlexField1	32	255	VARCHAR2	No	COL31	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD1	Telco

Table 154. Service Call Usage Detail (Voice, Data, Message) File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
4000	FlexField2	33	255	VARCHAR2	No	COL32	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD2	Telco
4000	FlexField3	34	255	VARCHAR2	No	COL33	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD3	Telco
4000	FlexField4	35	255	VARCHAR2	No	COL34	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD4	Telco
4000	FlexField5	36	255	VARCHAR2	No	COL35	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD5	Telco
4000	FlexField6	37	16	NUMBER	No	COL36	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD6	Telco
4000	FlexField7	38	16	NUMBER	No	COL37	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD7	Telco
4000	FlexField8	39	16	NUMBER	No	COL38	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD8	Telco
4000	FlexField9	40	8	DATE	No	COL39	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD9	Telco
4000	FlexField10	41	8	DATE	No	COL40	EDX_RPT_SERVICE_DETAIL_FACT	FLEX_FIELD10	Telco

Table 155 shows the file format for the premise data information.

Table 155. Premise Data File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
5000	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Utility
5000	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_PREMISE_DIM	STATEMENT_KEY	Utility
5000	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_PREMISE_DIM	ACCOUNT_KEY	Utility
5000	Premise ID	4	20	VARCHAR2	Yes	COL3	EDX_RPT_PREMISE_DIM	PREMISE_KEY	Utility
5000	Address Line 1	5	200	VARCHAR2	Yes	COL4	EDX_RPT_PREMISE_DIM	ADDRESS1	Utility
5000	Address Line 2	6	200	VARCHAR2	No	COL5	EDX_RPT_PREMISE_DIM	ADDRESS2	Utility
5000	Address Line 3	7	200	VARCHAR2	No	COL6	EDX_RPT_PREMISE_DIM	ADDRESS3	Utility
5000	City	8	100	VARCHAR2	No	COL7	EDX_RPT_PREMISE_DIM	CITY	Utility
5000	State	9	100	VARCHAR2	No	COL8	EDX_RPT_PREMISE_DIM	STATE	Utility

Table 155. Premise Data File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
5000	Postal Code	10	20	VARCHAR2	No	COL9	EDX_RPT_PREMISE_DIM	ZIP_CD	Utility
5000	Country	11	100	VARCHAR2	No	COL10	EDX_RPT_PREMISE_DIM	COUNTRY	Utility

Table 156 shows the file format for the consumption data information.

Table 156. Consumption Data File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
5100	Rec Type	1	3	VARCHAR2	No	REC_TYPE	None	None	Utility
5100	Statement Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_CONSUM_FACT	STATEMENT_KEY	Utility
5100	Account Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_CONSUM_FACT	ACCOUNT_KEY	Utility
5100	Service Agreement ID	4	20	VARCHAR2	Yes	COL3	EDX_RPT_CONSUM_FACT	SERVICE_KEY	Utility
5100	Premise ID	5	20	VARCHAR2	Yes	COL4	EDX_RPT_CONSUM_FACT	PREMISE_KEY	Utility
5100	Meter Number	6	20	VARCHAR2	Yes	COL5	EDX_RPT_CONSUM_FACT	METER_NUMBER	Utility
5100	Service Unit Code	7	50	VARCHAR2	No	COL6	EDX_RPT_CONSUM_FACT	UNIT_KEY	Utility
5100	Start Meter Reading Amount	8	16,2	NUMBER	No	COL7	EDX_RPT_CONSUM_FACT	START_AMT_CONSUM	Utility
5100	Start Meter Reading Date	9	8	DATE	No	COL8	EDX_RPT_CONSUM_FACT	START_READING_DT	Utility
5100	Start Meter Reading Time	10	8	DATE	No	COL9	EDX_RPT_CONSUM_FACT	START_READING_TM	Utility
5100	Start Reading Estimated Flag	11	10	VARCHAR2	No	COL10	EDX_RPT_CONSUM_FACT	START_READING_EST	Utility
5100	End Meter Reading Amount	12	16,2	NUMBER	No	COL11	EDX_RPT_CONSUM_FACT	END_AMT_CONSUM	Utility
5100	End Meter Reading Date	13	8	DATE	No	COL12	EDX_RPT_CONSUM_FACT	END_READING_DT	Utility
5100	End Meter Reading Time	14	8	DATE	No	COL13	EDX_RPT_CONSUM_FACT	END_READING_TM	Utility

Table 156. Consumption Data File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
5100	End Reading Estimated Flag	15	10	VARCHAR2	No	COL14	EDX_RPT_CONSUM_FACT	END_READING_EST	Utility
5100	Multiplier	16	10	NUMBER	No	COL15	EDX_RPT_CONSUM_FACT	MULTIPLIER	Utility
5100	Measure Quantity	17	16,2	NUMBER	No	COL16	EDX_RPT_CONSUM_FACT	MEASURE_QTY	Utility
5100	Time of Use	18	16,2	NUMBER	No	COL17	EDX_RPT_CONSUM_FACT	TIMEOFUSE	Utility
5100	Billed Quantity	19	16,2	NUMBER	No	COL18	EDX_RPT_CONSUM_FACT	BILLED_QTY	Utility
5100	Service Usage Type Code	20	50	VARCHAR2	No	COL19	EDX_RPT_CONSUM_FACT	USAGE_TYPE_KEY	Utility

File Formats for Unbilled Detail-Level Information

This topic shows the file formats for unbilled detail-level data.

Table 157 shows the file format for unbilled detail information.

Table 157. Unbilled Detail File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
4900	Rec Type	1	4	VARCHAR2	Yes	REC_TYPE	None	None	Telco
4900	Account Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_UNBILLED_DETAIL_FACT	ACCOUNT_KEY	Telco
4900	Service Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_UNBILLED_DETAIL_FACT	SERVICE_KEY	Telco
4900	Called Date	4	8	DATE	Yes	COL3	EDX_RPT_UNBILLED_DETAIL_FACT	DATE_KEY	Telco
4900	Duration	5	10	VARCHAR2	Yes	COL4	EDX_RPT_UNBILLED_DETAIL_FACT	TIME_KEY	Telco
4900	Called Number	6	20	VARCHAR2	Yes	COL5	EDX_RPT_UNBILLED_DETAIL_FACT	CALLED_NUM	Telco

Table 157. Unbilled Detail File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
4900	Usage Type Code	7	50	VARCHAR2	Yes	COL6	EDX_RPT_UNBILLED_DETAIL_FACT	USAGE_TYPE_KEY	Telco
4900	Tariff Code	8	50	VARCHAR2	Yes	COL7	EDX_RPT_UNBILLED_DETAIL_FACT	TARIFF_KEY	Telco
4900	Usage Unit Code	9	50	VARCHAR2	Yes	COL8	EDX_RPT_UNBILLED_DETAIL_FACT	UNIT_KEY	Telco
4900	Call Type Code	10	50	VARCHAR2	No	COL9	EDX_RPT_UNBILLED_DETAIL_FACT	CALL_TYPE_KEY	Telco
4900	Direction Code	11	20	VARCHAR2	Yes	COL10	EDX_RPT_UNBILLED_DETAIL_FACT	DIRECTION_KEY	Telco
4900	Calling City, State Code	12	100	VARCHAR2	Yes	COL11	EDX_RPT_UNBILLED_DETAIL_FACT	CALLING_AREA_CD_KEY	Telco
4900	Calling Country Code	13	100	VARCHAR2	Yes	COL12	EDX_RPT_UNBILLED_DETAIL_FACT	CALLING_AREA_CD_KEY	Telco
4900	Called City, State Code	14	100	VARCHAR2	Yes	COL13	EDX_RPT_UNBILLED_DETAIL_FACT	CALLED_AREA_CD_KEY	Telco
4900	Called Country Code	15	100	VARCHAR2	Yes	COL14	EDX_RPT_UNBILLED_DETAIL_FACT	CALLED_AREA_CD_KEY	Telco
4900	Total Usages	16	16,2	NUMBER	Yes	COL15	EDX_RPT_UNBILLED_DETAIL_FACT	TOTAL_USAGE	Telco
4900	Total Charge	17	16,2	NUMBER	Yes	COL16	EDX_RPT_UNBILLED_DETAIL_FACT	TOTAL_CHARGE_AMT	Telco
4900	Other Charge	16	16,2	NUMBER	No	COL17	EDX_RPT_UNBILLED_DETAIL_FACT	OTHER_CHARGE_AMT	Telco
4900	FlexField1	17	255	VARCHAR2	No	COL18	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD1	Telco
4900	FlexField2	18	255	VARCHAR2	No	COL19	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD2	Telco
4900	FlexField3	19	255	VARCHAR2	No	COL20	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD3	Telco
4900	FlexField4	20	255	VARCHAR2	No	COL21	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD4	Telco
4900	FlexField5	21	255	VARCHAR2	No	COL22	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD5	Telco
4900	FlexField6	22	16,2	NUMBER	No	COL23	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD6	Telco

Table 157. Unbilled Detail File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
4900	FlexField7	23	16,2	NUMBER	No	COL24	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD7	Telco
4900	FlexField8	24	16,2	NUMBER	No	COL25	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD8	Telco
4900	FlexField9	25	8	DATE	No	COL26	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD9	Telco
4900	FlexField10	26	8	DATE	No	COL27	EDX_RPT_UNBILLED_DETAIL_FACT	FLEX_FIELD10	Telco
4900	Allowance	27	20	VARCHAR2	No	COL28	EDX_RPT_UNBILLED_DETAIL_FACT	ALLOWANCE	Telco
4900	Gross Amount	28	16,2	NUMBER	No	COL29	EDX_RPT_UNBILLED_DETAIL_FACT	GROSS_AMT	Telco and Utility
4900	Discount Amount	29	16,2	NUMBER	No	COL30	EDX_RPT_UNBILLED_DETAIL_FACT	DISCOUNT_AMT	Telco and Utility
4900	Tax Amount	30	16,2	NUMBER	No	COL31	EDX_RPT_UNBILLED_DETAIL_FACT	TAX_AMT	Telco and Utility

File Formats for Prepaid Information

This topic shows the file formats for prepaid data.

[Table 158](#) shows the file format for prepaid account information.

Table 158. Prepaid Account File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
9000	Rec Type	1	4	VARCHAR2	Yes	None	None	None	Telco
9000	Account Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_PREPAY_ACCOUNT_FACT	ACCOUNT_KEY	Telco
9000	Month Charge Amt	3	16,2	NUMBER	Yes	COL2	EDX_RPT_PREPAY_ACCOUNT_FACT	MONTHLY_CHARGE_AMT	Telco
9000	Usage Amt	4	16,2	NUMBER	No	COL3	EDX_RPT_PREPAY_ACCOUNT_FACT	USAGE_CHARGE_AMT	Telco
9000	Adjustment Amt	5	16,2	NUMBER	No	COL4	EDX_RPT_PREPAY_ACCOUNT_FACT	ADJUST_AMT	Telco
9000	Other Charges Amt	6	16,2	NUMBER	No	COL5	EDX_RPT_PREPAY_ACCOUNT_FACT	OTHER_CHARGE_AMT	Telco
9000	Taxes and Fees	7	16,2	NUMBER	No	COL6	EDX_RPT_PREPAY_ACCOUNT_FACT	TAXES_SURCHARGES_FEE	Telco
9000	Total Charge Amt	8	16,2	NUMBER	Yes	COL7	EDX_RPT_PREPAY_ACCOUNT_FACT	TOTAL_CHARGE_AMT	Telco

Table 158. Prepaid Account File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
9000	Top Off Date	9	8	DATE	Yes	COL8	EDX_RPT_PREPAY_ACCOUNT_FACT	TOP_OFF_DATE	Telco
9000	Expiry Date	10	8	DATE	Yes	COL9	EDX_RPT_PREPAY_ACCOUNT_FACT	EXPIRY_DATE	Telco
9000	FlexField1	17	225	VARCHAR2	No	COL10	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD1	Telco
9000	FlexField2	18	225	VARCHAR2	No	COL11	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD2	Telco
9000	FlexField3	19	225	VARCHAR2	No	COL12	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD3	Telco
9000	FlexField4	20	225	VARCHAR2	No	COL13	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD4	Telco
9000	FlexField5	21	225	VARCHAR2	No	COL14	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD5	Telco
9000	FlexField6	22	16,2	NUMBER	No	COL15	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD6	Telco
9000	FlexField7	23	16,2	NUMBER	No	COL16	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD7	Telco
9000	FlexField8	24	16,2	NUMBER	No	COL17	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD8	Telco
9000	FlexField9	25	8	DATE	No	COL18	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD9	Telco
9000	FlexField10	26	8	DATE	No	COL19	EDX_RPT_PREPAY_ACCOUNT_FACT	FLEX_FIELD10	Telco
9000	Remaining Balance	27	16,2	NUMBER	No	COL20	EDX_RPT_PREPAY_ACCOUNT_FACT	REMAIN_BAL_AMT	Telco
9000	Remaining Balance as of	28	8	DATE	No	COL21	EDX_RPT_PREPAY_ACCOUNT_FACT	REMAIN_BAL_AS_OF	Telco

Table 159 shows the file format for prepaid product information. This table stores charges for a product and plan.

Table 159. Prepaid Product File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
9200	Rec Type	1	4	VARCHAR2	Yes	REC_TYPE	None	None	Telco
9200	Account Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_PREPAY_PRODUCT_FACT	ACCOUNT_KEY	Telco

Table 159. Prepaid Product File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
9200	Service Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_PREPAY_PRODUCT_FACT	SERVICE_KEY	Telco
9200	Plan Type Code	4	50	VARCHAR2	Yes	COL3	EDX_RPT_PREPAY_PRODUCT_FACT	PLAN_TYPE_KEY	Telco
9200	Product Code	5	50	VARCHAR2	Yes	COL4	EDX_RPT_PREPAY_PRODUCT_FACT	PRODUCT_PARENT_KEY	Telco
9200	Monthly Charge Amount	6	16,2	NUMBER	Yes	COL5	EDX_RPT_PREPAY_PRODUCT_FACT	CHARGE_AMT	Telco
9200	Product Sub Code	7	50	VARCHAR2	Yes	COL6	EDX_RPT_PREPAY_PRODUCT_FACT	PRODUCT_CHILD_KEY	Telco
9200	Transaction Date	8	8	DATE	Yes	COL7	EDX_RPT_PREPAY_PRODUCT_FACT	TRANS_DT	Telco

Table 160 shows the file format for prepay detail information. This file contains details for each transaction performed by a user.

Table 160. Prepaid Detail File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
9500	Rec Type	1	4	VARCHAR2	Yes	REC_TYPE	None	None	Telco
9500	Account Number	2	20	VARCHAR2	Yes	COL1	EDX_RPT_PREPAY_DETAIL_FACT	ACCOUNT_KEY	Telco
9500	Service Number	3	20	VARCHAR2	Yes	COL2	EDX_RPT_PREPAY_DETAIL_FACT	SERVICE_KEY	Telco
9500	Transaction Date	4	8	DATE	Yes	COL3	EDX_RPT_PREPAY_DETAIL_FACT	TRANS_DT	Telco
9500	Called Number	5	20	VARCHAR2	Yes	COL4	EDX_RPT_PREPAY_DETAIL_FACT	CALLED_NUM	Telco
9500	Usage Type Code	6	50	VARCHAR2	Yes	COL5	EDX_RPT_PREPAY_DETAIL_FACT	USAGE_TYPE_KEY	Telco
9500	Tariff Code	7	50	VARCHAR2	Yes	COL6	EDX_RPT_PREPAY_DETAIL_FACT	TARIFF_KEY	Telco
9500	Usage Unit Code	8	50	VARCHAR2	Yes	COL7	EDX_RPT_PREPAY_DETAIL_FACT	UNIT_KEY	Telco

Table 160. Prepaid Detail File Format

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
9500	Call Type Code	9	50	VARCHAR2	No	COL8	EDX_RPT_PREPAY_DETAIL_FACT	CALL_TYPE_KEY	Telco
9500	Direction Code	10	20	VARCHAR2	Yes	COL9	EDX_RPT_PREPAY_DETAIL_FACT	DIRECTION_KEY	Telco
9500	Calling City, State Code	11	100	VARCHAR2	Yes	COL10	EDX_RPT_PREPAY_DETAIL_FACT	CALLING_AREA_CD_KEY	Telco
9500	Calling Country Code	12	100	VARCHAR2	Yes	COL11	EDX_RPT_PREPAY_DETAIL_FACT	CALLING_AREA_CD_KEY	Telco
9500	Called City, State Code	13	100	VARCHAR2	Yes	COL12	EDX_RPT_PREPAY_DETAIL_FACT	CALLED_AREA_CD_KEY	Telco
9500	Called Country Code	14	100	VARCHAR2	Yes	COL13	EDX_RPT_PREPAY_DETAIL_FACT	CALLED_AREA_CD_KEY	Telco
9500	Total Usages	15	16,2	NUMBER	Yes	COL14	EDX_RPT_PREPAY_DETAIL_FACT	TOTAL_USAGE	Telco
9500	Total Charge	16	16,2	NUMBER	Yes	COL15	EDX_RPT_PREPAY_DETAIL_FACT	TOTAL_CHARGE_AMT	Telco
9500	Other Charge	17	16,2	NUMBER	No	COL16	EDX_RPT_PREPAY_DETAIL_FACT	OTHER_CHARGE_AMT	Telco
9500	FlexField1	18	255	VARCHAR2	No	COL17	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD1	Telco
9500	FlexField2	19	255	VARCHAR2	No	COL18	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD2	Telco
9500	FlexField3	20	255	VARCHAR2	No	COL19	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD3	Telco
9500	FlexField4	21	255	VARCHAR2	No	COL20	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD4	Telco
9500	FlexField5	22	255	VARCHAR2	No	COL21	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD5	Telco
9500	FlexField6	23	16,2	NUMBER	No	COL22	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD6	Telco
9500	FlexField7	24	16,2	NUMBER	No	COL23	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD7	Telco
9500	FlexField8	25	16,2	NUMBER	No	COL24	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD8	Telco
9500	FlexField9	26	8	DATE	No	COL25	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD9	Telco
9500	FlexField10	27	8	DATE	No	COL26	EDX_RPT_PREPAY_DETAIL_FACT	FLEX_FIELD10	Telco
9500	Gross Amount	28	16,2	NUMBER	No	COL27	EDX_RPT_PREPAY_DETAIL_FACT	GROSS_AMT	Telco
9500	Discount Amount	29	16,2	NUMBER	No	COL28	EDX_RPT_PREPAY_DETAIL_FACT	DISCOUNT_AMT	Telco
9500	Tax Amount	30	16,2	NUMBER	No	COL29	EDX_RPT_PREPAY_DETAIL_FACT	DISCOUNT_AMT	Telco

File Formats for Batch Provisioning Data

This topic shows the file and record formats for batch provisioning data. The input file must be pipe-delimited. The file name must use the pattern PROV_BILLING_YYYYMMDDHHMISS.DAT, where YYYYMMDDHHMISS is the time when the file is generated.

The ProvisioningData batch job processes one file at a time, oldest file first. File processing order is critical, as an object created in the first file must be processed before it can be updated in the second, otherwise the second file will fail.

Each provisioning input file consists of the following:

- A file header (record type 00)
- One or more of the following record types, depending on the file type:
 - **Company.** Record type 10.
 - **Account.** Record type 20.
 - **Service.** Record type 30.

Each record has start date, end date, and delete date, which indicate when the object was created, canceled, and deleted. Each of these dates must be a value in the past.

The bill cycle end date is valid only when the file is extracted from the billing file. Also, when the file is extracted from the billing file, the billing cycle end date is not null in file header, and the start date and end date of record will both be null.

The date format is MM/DD/YYYY. The primary contact username and email must both have either a value or no value.

Follow these guidelines when creating batch provisioning input files:

- Create a pipe-delimited file with the content and the format described in *Implementation Guide for Oracle Billing Insight*.
- Use the following naming format for the provisioning input file you generate: PROV_BILLING_YYYYMMDDHHMISS.DAT
- Add objects records to the input file in the following sequence:
 - a Companies
 - b Accounts
 - c Service Agreements
- Add a row in the input file for each object with a minimum of the following information:
 - ID
 - Start date attributes
 - End date attributes
- The start and end dates indicate when the object was created and expired. Use the dates to indicate the following actions:

- **Add, or create.** Make the start date not null.
 - **Delete.** Make the delete date not null.
 - **Expire.** Make the end date not null (to expire the object).
 - **Update.** Make both the start and end dates null. Note that provisioning objects are not versioned.
 - You can use multiple input files, however, it is critical to date and load them in chronological order. The ProvisioningData job processes the file with the oldest modified date and time first, so date the files accordingly to sequence the proper load order. If you do not date the files properly, subsequent input files will fail due to missing objects.
- The ProvisioningData job processes the input file one row at a time.

Table 161 shows the file format for batch provisioning file header.

Table 161. File Header Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
00	Rec Type	1	2	VARCHAR2	Yes	None	None	None	Telco and Utility
00	BILLING SYSTEM	2	20	VARCHAR2	Yes	None	None	None	Telco and Utility
00	Billing Cycle End Date	3	10	DATE	No	None	None	None	Telco and Utility

Table 162 shows the file format for the company record in the batch provisioning file.

Table 162. Company Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
10	Rec Type	1	2	VARCHAR2	Yes	None	None	None	Telco and Utility
10	Start Date	2	10	DATE	No	None	EDX_OMF_COMPANY	START_DATE	Telco and Utility
10	End Date	3	10	DATE	No	None	EDX_OMF_COMPANY	END_DATE	Telco and Utility
10	Delete Date	4	10	DATE	No	None	EDX_OMF_COMPANY	DATE_DELETED	Telco and Utility
10	Company ID	5	255	VARCHAR2	Yes	None	EDX_OMF_COMPANY	COMPANY_CD	Telco and Utility

Table 162. Company Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
10	Company Name	6	255	VARCHAR2	Yes	None	EDX_OMF_COMPANY	DISPLAYNAME	Telco and Utility
10	Display Name	7	255	VARCHAR2	No	None	EDX_OMF_COMPANY	DISPLAYNAME	Telco and Utility
10	Corp Account No	8	255	VARCHAR2	No	None	EDX_OMF_COMPANY	CORPACCNO	Telco and Utility
10	Corp Tax ID	9	255	VARCHAR2	No	None	EDX_OMF_COMPANY	CORPTAXID	Telco and Utility
10	Street	10	255	VARCHAR2	No	None	EDX_OMF_COMPANY	STREET	Telco and Utility
10	City	11	255	VARCHAR2	No	None	EDX_OMF_COMPANY	CITY	Telco and Utility
10	State	12	255	VARCHAR2	No	None	EDX_OMF_COMPANY	STATE	Telco and Utility
10	Country	13	255	VARCHAR2	No	None	EDX_OMF_COMPANY	COUNTRY	Telco and Utility
10	Zip Code	14	20	VARCHAR2	No	None	EDX_OMF_COMPANY	ZIPCODE	Telco and Utility
10	Primary Contact Username	15	255	VARCHAR2	No	None	EDX_BSL_UMF_USER EDX_BSL_AUTH_SECPROFILE	USERID	Telco and Utility
10	Primary Contact Email	16	255	VARCHAR2	No	None	EDX_BSL_UMF_USER	EMAIL1	Telco and Utility
10	Primary Contact First Name	17	255	VARCHAR2	No	None	EDX_BSL_UMF_USER	FIRSTNAME	Telco and Utility
10	Primary Contact Last Name	18	255	VARCHAR2	No	None	EDX_BSL_UMF_USER	LASTNAME	Telco and Utility
10	Flex Field_1	19	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_1	Telco and Utility
10	Flex Field_2	20	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_2	Telco and Utility
10	Flex Field_3	21	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_3	Telco and Utility
10	Flex Field_4	22	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_4	Telco and Utility
10	Flex Field_5	23	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_5	Telco and Utility
10	Flex Field_6	24	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_6	Telco and Utility
10	Flex Field_7	25	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_7	Telco and Utility
10	Flex Field_8	26	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_8	Telco and Utility

Table 162. Company Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
10	Flex Field_9	27	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_9	Telco and Utility
10	Flex Field_10	28	255	VARCHAR2	No	None	EDX_OMF_COMPANY	FLEX_FIELD_10	Telco and Utility

Table 163 shows the file format for the account record in the batch provisioning file.

Table 163. Account Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
20	Rec Type	1	2	VARCHAR2	Yes	None	None	None	Telco and Utility
20	Start Date	2	10	DATE	No	None	EDX_OMF_ACCOUNT	STARTDATE	Telco and Utility
20	End Date	3	10	DATE	No	None	EDX_OMF_ACCOUNT	ENDDATE	Telco and Utility
20	Delete Date	4	10	DATE	No	None	EDX_OMF_ACCOUNT	DELETEDAT	Telco and Utility
20	Company ID	5	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	BILLEE_ID	Telco and Utility
20	Account Number	6	255	VARCHAR2	Yes	None	EDX_OMF_ACCOUNT	ACCOUNT_NUMBER	Telco and Utility
20	Account Name	7	100	VARCHAR2	No	None	EDX_OMF_ACCOUNT	NAME	Telco and Utility
20	Account Owner Name	8	100	VARCHAR2	Yes	None	EDX_OMF_ACCOUNT	CONTACTNAME	Telco and Utility
20	Account Type	9	64	VARCHAR2	No	None	EDX_OMF_ACCOUNT	ACCOUNT_TYPE_KEY	Telco and Utility
20	Bill Cycle End Day	10	2	NUMBER	No	None	EDX_OMF_ACCOUNT	BILL_CYCLE_ENDDAY	Telco and Utility
20	Bill Type	11	2	VARCHAR2	No	None	EDX_OMF_ACCOUNT	BILL_TYPE	Telco and Utility
20	Paper On Flag	12	1	CHAR2	No	None	EDX_OMF_ACCOUNT	PAPER_ON_FLAG	Telco and Utility
20	Billable Flag	13	1	CHAR2	No	None	EDX_OMF_ACCOUNT	BILLABLE_FLAG	Telco and Utility
20	Address Type	14	50	VARCHAR2	No	None	EDX_OMF_ADDRESS	ADDRESS_TYPE_KEY	Telco and Utility
20	Address1	15	200	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	ADDRESS1	Telco and Utility
20	Address2	16	200	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	ADDRESS2	Telco and Utility

Table 163. Account Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
20	Address3	17	200	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	ADDRESS3	Telco and Utility
20	City	18	255	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	CITY	Telco and Utility
20	State	19	255	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	STATE	Telco and Utility
20	Country	20	255	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	COUNTRY	Telco and Utility
20	Zip Code	21	20	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	ZIP_CD	Telco and Utility
20	Contact Name	22	255	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	CONTACT_NAME	Telco and Utility
20	Home Number	23	128	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	HOME_PHONE	Telco and Utility
20	Work Number	24	128	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	WORK_PHONE	Telco and Utility
20	Mobile Number	25	128	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	MOBILE_PHONE	Telco and Utility
20	External Reference	26	128	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	EXTERNAL_REF	Telco and Utility
20	Email Address	27	128	VARCHAR2	No	None	EDX_OMF_CONTACT_INFO	EMAIL_ADDRESS	Telco and Utility
20	Flex Field_1	28	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD1	Telco and Utility
20	Flex Field_2	29	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD2	Telco and Utility
20	Flex Field_3	30	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD3	Telco and Utility
20	Flex Field_4	31	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD4	Telco and Utility
20	Flex Field_5	32	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD5	Telco and Utility
20	Flex Field_6	33	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD6	Telco and Utility
20	Flex Field_7	34	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD7	Telco and Utility
20	Flex Field_8	35	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD8	Telco and Utility
20	Flex Field_9	36	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD9	Telco and Utility
20	Flex Field_10	37	255	VARCHAR2	No	None	EDX_OMF_ACCOUNT	FLEX_FIELD10	Telco and Utility

Table 164 shows the file format for the service record in the batch provisioning file.

Table 164. Service Record Format for the Batch Provisioning File

Record Type	Field	Pos	Max Len	Data Type	Req?	Staging Table (STG_CDR) Column Name	DB_TABLE_NAME	DB_COLUMN_NAME	Industry Application
30	Rec Type	1	2	VARCHAR2	Yes	None	None	None	Telco and Utility
30	Start Date	2	10	DATE	No	None	EDX_OMF_SERVICE AGREEMENT	START_DATE	Telco and Utility
30	End Date	3	10	DATE	No	None	EDX_OMF_SERVICE AGREEMENT	END_DATE	Telco and Utility
30	Delete Date	4	10	DATE	No	None	EDX_OMF_SERVICE AGREEMENT	DELETE_DATE	Telco and Utility
30	Account Number	5	255	VARCHAR2	Yes	None	EDX_OMF_SERVICE AGREEMENT	ACCOUNT_NUMBERRY	Telco and Utility
30	Service Number	6	255	VARCHAR2	Yes	None	EDX_OMF_SERVICE AGREEMENT	SERVICE_NUMBER	Telco and Utility
30	Service Type	7	40	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	SERVICE_TYPE_KEY	Telco and Utility
30	Product Code	8	50	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	PRODUCT_KEY	Telco and Utility
30	Subscriber Name	9	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	SUBSCRIBER_NAME	Telco and Utility
30	Zoning	10	1	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	ZONING	Telco and Utility
30	Description	11	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	DESCRIPTION	Telco and Utility
30	Flex Field_1	12	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD1	Telco and Utility
30	Flex Field_2	13	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD2	Telco and Utility
30	Flex Field_3	14	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD3	Telco and Utility
30	Flex Field_4	15	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD4	Telco and Utility
30	Flex Field_5	16	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD5	Telco and Utility
30	Flex Field_6	17	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD6	Telco and Utility
30	Flex Field_7	18	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD7	Telco and Utility
30	Flex Field_8	19	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD8	Telco and Utility
30	Flex Field_9	20	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD9	Telco and Utility
30	Flex Field_10	21	255	VARCHAR2	No	None	EDX_OMF_SERVICE AGREEMENT	FLEX_FIELD10	Telco and Utility

Internationalization Support Settings

Oracle Billing Insight supports multiple languages.

Unicode data storage requires a Unicode Database solution, which involves creating a Unicode-based database using UTF-8 as the encoding not only for CHAR and VARCHAR2 character datatypes but also for all SQL names and literals. To implement the Unicode Database solution, the Oracle Billing Insight database character set is configured as AL32UTF8, the Oracle name for UTF-8.

The NLS_LENGTH_SEMANTICS parameter in the init.ora (parameter) file of the target Oracle database is set to CHAR instead of the default BYTE to enable global character semantic support.

Character semantics changed the way multibyte characters were treated in Oracle Database. Instead of doubling or tripling column or variable precision, setting NLS_LENGTH_SEMANTICS = CHAR causes Oracle Database to treat storage of the string 'Today' the same as the Japanese string '. With this setting, glyphs (characters) are the measure for column and variable precision rather than the bytes required to store the characters.

Oracle Billing Insight uses the init.ora file parameter settings shown in [Table 165](#) for the Oracle Billing Insight database instance.

Table 165. Internationalization Support Settings

Parameter	Value
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
NLS_CURRENCY	\$
NLS_ISO_CURRENCY	AMERICA
NLS_NUMERIC_CHARACTERS	.
NLS_CALENDAR	GREGORIAN
NLS_DATE_FORMAT	YYYY-MM-DD
NLS_DATE_LANGUAGE	AMERICAN
NLS_CHARACTERSET	AL32UTF8
NLS_SORT	BINARY
NLS_TIME_FORMAT	HH.MI.SSXFF AM
NLS_TIMESTAMP_FORMAT	DD-MON-RR HH.MI.SSXFF AM
NLS_TIME_TZ_FORMAT	HH.MI.SSXFF AM TZR
NLS_TIMESTAMP_TZ_FORMAT	DD-MON-RR HH.MI.SSXFF AM TZR
NLS_DUAL_CURRENCY	\$
NLS_NCHAR_CHARACTERSET	AL16UTF16
NLS_COMP	BINARY

Table 165. Internationalization Support Settings

Parameter	Value
NLS_LENGTH_SEMANTICS	CHAR
NLS_NCHAR_CONV_EXCP	FALSE
NLS_LANGUAGE	AMERICAN

Index

A

- about application EAR files** 19
- about Assisted Service application functionality** 353
- about custom JSP pages and action classes** 31
- about customizing reports** 31
- about Oracle Billing Insight** 13
 - architecture 17
 - installation directories 18
- about payment plug-ins** 245
- about repacking EAR files** 29
- about resource authorization** 51
- about the Assisted Service application** 15
- about the Command Center application** 15
- about the Oracle Billing Insight database** 17
- about the Self-Service applicatoin** 14
- about the Web Services application** 16
- about Velocity templates** 31
- accessing an object's public fields and methods for payment consolidation** 304
- accessing the Data Dictionary and Javadoc** 25
- account enrollment input file format** 314
- account lockout** 48
- account verification with Paymentech** 216
- ACH**
 - change codes 214
 - creating a plug-in 252
 - customizing 258
 - effective date 215
 - individual ID 256
 - return codes 214
 - return files 262
 - settlement date 216
 - templates 258
- action classes**
 - custom 31
- adding a custom message provider** 140
- adding a new java object instance or static class for payment consolidation**
 - java object instance or static class, adding for payment consolidation 305
- adding a new permission set** 56
- adding a new user role** 55
- adding cartridge property keys for payment consolidation** 306
- adding custom language resource bundle files** 38
- Address Verification Service** 221
- Amex authorization transactions with Paymentech** 216
- anonymous payment validation, customization** 282
- APIs for payment consolidation** 313
- ApplicationLabels_en_US.properties file** 33
- ApplicationMessages_en_US.properties file** 33
- applications**
 - Assisted Service 13, 15
 - Assisted Service EAR file 21
 - authorization 56
 - Command Center 14, 15
 - Command Center EAR file 23
 - EAR files 19
 - languages in user interface 33
 - resource authorization 51
 - Self-Service 13, 14
 - Self-Service EAR file 19
 - Web Services 14, 16
 - Web Services EAR file 22
- architecture of Oracle Billing Insight** 17
- assets** 28
- Assisted Service application** 13
 - customizing 353
 - deactivating and reactivating the bootstrap user 41
 - EAR file 21
 - files 353
 - functionality 353
 - language in the user interface 33
- attributes for payment consolidation templates** 299, 300
- audit data, reporting** 357, 362
- auditing database activity** 17
- auditing impersonation** 354
- authentication and authorization, customizing** 49
- Automated Clearing House (ACH)** 213
- automatic resource file updates, changing the frequency** 38
- azcfg.properties file** 51
- azpolicy.xml file** 52, 55, 56

B

- bank routing**
 - international 281
- bar chart properties** 199
- base objects, non-permitted changes** 28
- batch reports**
 - configuring 149
- batches**
 - threshold values, customizing 195
- best practices for customizing Oracle Billing Insight** 27
- bill detail URL**
 - customizing for payment consolidation 309
- bill summary**
 - acknowledgement input file format 330
 - output file format 323
- billing periods, defining** 65
- bootstrap user ID**
 - deactivating and reactivating the Assisted Service and Command Center applications 41
- branding, customizing the UI** 31
- Brazilian Portuguese** 32
- bundles, language resource (property) files** 32

C

- canceling payments for consumer users in batch**
 - input file format 244
- cartridge**
 - property keys, adding for payment consolidation 306
- cascading style sheet (CSS)** 29
- changing the database, non-permitted alterations** 28
- changing the frequency of automatic resource file updates** 38
- changing the local storage location for the HTML report formatting files** 207
- charts**
 - configuring properties 198
 - customizing 197
 - Statement Summary 206
- check payments**
 - transaction cycle 211
- codes, ACH** 213
- color** 29
- Command Center application** 14
 - deactivating and reactivating the bootstrap user 41
 - EAR file 23
- CommonFieldValidator_en_US.properties**

- file** 33
- configurable chart properties** 198
- configuration parameters for payment consolidation** 308
- configuring batch reporting** 149
- configuring convenience fee payments** 277
- configuring international bank routing** 281
- configuring messaging properties** 137
- configuring the logging level for log4j files** 24
- configuring the maximum attempt times** 48
- configuring the secure bill-ready notification password** 142
- convenience fee payments, configuring** 277
- creating a credit card plug-in** 252
- creating a payment reminder plug-in** 253
- creating a PDF template for reporting** 178
- creating a recurring payment plug-in** 254
- creating an ACH plug-in** 252
- creating customized payment consolidation templates** 290
- credit card**
 - creating a plug-in 252
 - overview 216
 - statuses 217
 - transaction overview 218
 - user options 220
- CRM**
 - updating disputes 64
- CSS**
 - customizing the UI default 29
- CurrencyText_en_US.properties file** 33
- custom message provider**
 - adding 140
- custom resource bundle files for languages**
 - adding 38
- customizations to the database, non-permitted** 28
- customizing account reactivation** 48
- customizing anonymous payment validation** 282
- customizing charts** 197
- customizing default display patterns used in notifications** 135
- customizing email notifications** 69
- customizing Oracle Billing Insight**
 - best practices 27
- customizing reports** 31
- customizing Spring Security** 49
- customizing the administrator user password validation rule** 43
- customizing the agent user password update requirements** 43
- customizing the Assisted Service**

- application 285, 353
- customizing the bill detail URL 309
- customizing the content of notifications 134
- customizing the default UI CSS 29
- customizing the end user and agent user password validation rule 44
- customizing the payment amount format 271
- customizing the Statement Summary chart 206
- customizing the UI for multiple branding 31
- customizing the user interface for multiple branding 31
- customizing the validation code generator 47
- customizing threshold values for batch reporting 195
- customizing user authentication and authorization with Spring Security 49
- customizing user enrollment 46
- customizing user lockout 48
- CyberSource 221

D

- Data Dictionary 25
- data element mappings
 - example A 331
 - example B 342
- data file loading tasks 366
- Data Visualization Tools 197
- database 17
 - non-permitted changes 28
- database tables for payment consolidation 351
- DB-logging 23
- deactivating the bootstrap user 41
- debug, payment 244
- debugging Oracle Billing Insight 23
- default payment consolidation file formats 314
- defining billing periods 65
- defining custom language resource bundle files 38
- DefUsrAcctRelationship input file format 59
- deleting consumer users in batch
 - input file format 61
- directories 18
- disabling SMS notifications 142
- Discover authorization transactions with Paymentech 216
- display patterns, customizing default for notifications 135

- displaying SQL statements 24
- disputes
 - updating an external CRM 64
- Dutch 32
- dynamic SQL 186

E

- EAR files
 - applications 19
 - repackaging 29
- email notifications
 - customizing content 134
 - customizing default display patterns 135
 - localizing 32
 - processing 69
 - template content for business users 74
 - template content for consumer users 101
- English, U.S.
 - language in applications 32
 - language resource (property) files 33
- enrollment
 - acknowledgement output file format 320
 - customizing 46
 - email template 257
 - validation API 47
- examples of data element mappings 331, 342

F

- fact table mapping and file records (load process) 367
- file formats (load process)
 - account-level information 380
 - dimension-level information 368
 - service detail-level information 387, 391, 393
 - service-level information 383
 - statement-level information 374
- file formats (payment consolidation) 314
- file processing for data load 366
- file record and dimension table mapping (load process) 367
- file record and fact table mapping (load process) 367
- file structure XML template 293
- fonts 29

G

- German 32

H

- holidays

- and ACH effective date 216
- Global Template variable 260
- HTML report formatting files, changing the local storage location** 207

I

- IBillingSystemIDPlugin** 310
- ICartridgePlugin** 311
- identifying the installed version of Oracle Billing Insight** 26
- impersonation**
 - auditing 354
- implementing payment consolidator plug-ins** 310
- input and output files**
 - formats for payment consolidation 314
 - payment consolidation 289
- input file formats**
 - for canceling payments for consumer users in batch 244
 - for deleting consumer users in batch 61
 - for migrating consumer users in batch 59
 - for the DefUsrAcctRelationship job 59
 - load process 366, 368, 374, 380, 383, 387, 391, 393
- installation directories** 18
- installed version of Oracle Billing Insight, identifying** 26
- international bank routing** 281
- internationalization**
 - support settings 397
 - support settings for load process 403
 - Velocity templates 187
- Italian** 32

J

- Java APIs for payment consolidation** 313
- Javadoc** 25
- Java-option-logging** 23
- JavaScripts** 31
- jobs**
 - DefUsrAcctRelationship 59
 - payment consolidation 287
 - PCAccountEnrollment 287
 - PCBillSummary 287
 - PCBillSummaryAcknowledgement 287
- JSP pages**
 - custom 31

L

- language in the user interface** 38
- language in user interface**
 - changing the default 33

- list of available 32
- localizing for additional languages 33
- resource (property) file set 33

- line chart properties** 204

load process

- about 366
- data file loading tasks 366
- file record and dimension table mapping 367
- file record and fact table mapping 367
- input file format for account-level information 380
- input file format for dimension-level information 368
- input file format for service detail-level information 387, 391, 393
- input file format for service-level information 383
- input file format for statement-level information 374
- input file formats 366
- internationalization support settings 397, 403
- preprocessor tasks 365

- localizing the user interface** 32

- logout** 48

- Log4j** 23

- logging** 23

- logic operators for payment consolidation** 303

M

mapping

- data elements example A 331
- data elements example B 342

- Mastercard authorization transactions with Paymentech** 216

maximum attempt times

- configuring 48

message provider

- custom 140

migrating consumer users in batch

- input file format 59

- multiple branding of the UI** 31

N

NOC

- ACH change codes 214
- ACH returns 257
- codes 214
- returns 257
- transactions 215

- non-permitted database changes** 28

notifications

- configuring the password for bill-ready 142
- email template content for business users 74
- email template content for consumer users 101
- SMS template content 130
- NotificationsEmail_en_US.properties file** 33
- NotificationsSMS_en_US.properties file** 33
- O**
- ODFI**
 - ACH return codes 214
 - and ACH payment transaction cycle 211
 - and received check processing 212
- OLAP schema** 17
- OLTP schema** 17
- one cent authorization in Paymentech** 216
- overview of Oracle Billing Insight** 13
- overview of processing with a payment consolidator** 285
- P**
- passwords**
 - bill-ready notification 142
 - customizing the administrator user validation rule 43
 - customizing the end user and agent user validation rule 44
- payment**
 - configuring convenience fee support 277
 - customizing the amount format 271
 - debug 244
- payment consolidation**
 - accessing an object's public fields and methods 304
 - customizing 290
 - customizing the bill detail URL 309
 - database tables 351
 - default file formats 314
 - implementation process 290
 - implementing plug-ins 310
 - input and output files 289
 - Java APIs 313
 - jobs 287
 - overview 285
 - sample template files 291
 - setting configuration parameters 308
 - template engine 290
- Paymentech** 221
 - account authorization transactions 216
- PayPal Payflow Pro** 220
- PCAccountEnrollment job** 287
- PCBillSummary job** 287
- PCBillSummaryAcknowledgement job** 287
- PDF template for reporting** 178
- Period_en_US.properties file** 33
- permission sets** 51, 52
 - adding 56
- pie chart properties** 202
- plug-ins**
 - ACH 252
 - credit cards 252
 - payment 245
 - payment consolidation 310
 - payment reminders 253
 - recurring payments 254
- pmtCheckSubmit**
 - and ACH effective date 215
 - date 216
 - plug-in 252
- pmtCheckUpdate**
 - and ACH change codes 214
 - and ACH return codes 214
- pmtCreditCardSubmit**
 - and the payment transaction cycle 220
- pmtRecurPayment**
 - email template 258
- pmtSubmitEnroll** 258
- Portuguese (Brazilian)** 32
- preprocessor tasks** 365
- principal user roles** 51
- process of implementing a payment consolidator** 290
- property files for language localization** 33
- public fields and methods, adding for payment consolidation** 304
- Q**
- query**
 - and IReportConfig 147
 - and single payment type 270
 - and transformer 146
 - and user context variable 179
 - attributes for 154
 - case study 267
 - changing SampleARPaymentIntegrator 269
 - checkQuery and creditCardQuery 266
 - credit card 270
 - dynamic SQL 186
 - element 154
 - element and paging 185
 - element of Report XML 153
 - elements in jdbcDirect 153
 - in multiple payee ACH accounts 280
 - in presentment engine reports 28
 - in report context 146
 - SQL and report XML 163

values for 158

R

- reactivate account lockout**
 - configuring 48
- reactivating the bootstrap user** 41
- record XML template** 295
- recurring payments**
 - email template 258
 - plug-in 254
- reminders** 253
- repackaging EAR files** 29
- report list properties file** 149
- reporting on system administrator audit data** 362
- reporting on user audit data** 357
- reports** 28, 143
 - creating a PDF template 178
 - customizing reports 31
- resource authorization** 51
- resource file updates**
 - changing the frequency 38
- return codes** 213

S

- sample payment consolidation template files** 291
- SampleARPaymentIntegrator** 269
- security**
 - Spring 49
- security.xma.xml file** 56
- Self-Service application** 13, 14
 - EAR file 19
 - language in the user interface 33
- setting payment consolidation configuration parameters** 308
- SIS2050ACK file** 331
- SIS2070 file** 331
- SIS2070ACK file** 331
- SMS notifications**
 - customizing content 134
 - customizing default display patterns 135
 - disabling 142
 - template content 130
- SMTP** 141
- Spanish (Worldwide)** 32
- Spring** 28
- Spring Security, customizing** 49
- Spring Security, customizing user authentication and authorization** 49
- SQL**
 - displaying 24
 - dynamic 186

hibernate 25

- SQL statements, displaying** 24
- stack bar chart properties** 199
- Statement Summary chart, customizing** 206
- support settings** 403
- system administrator audit data, reporting** 362

T

- table mapping (dimension) and file records (load process)** 367
- template content**
 - SMS notifications 130
- templates** 28
 - and action classes 193
 - and attribute for column element 167
 - and attribute for transformer element 165
 - and attributes for printList element 174
 - and dynamic localization 190
 - and EAR structure 20, 22
 - and ITransformer 147
 - and localization 189
 - and predefined context variables 179
 - and ReportContext objects in 182
 - and reporting engine 144, 145, 148
 - and reporting XML 149
 - and resource bundle definition 188
 - attributes for downloadList element 173
 - attributes for payment consolidation templates 299
 - attributes for template element 168
 - changing 27
 - customizing for payment consolidation 290
 - customizing report 192
 - download element of report XML 174
 - element of report XML 151, 152, 167
 - email composer 70
 - enabling sorting 184
 - file structure XML for payment consolidation 293
 - generating custom link report 175
 - included in report package 177
 - modify reports 192
 - paging 185
 - record XML for payment consolidation 295
 - report XML 195
 - reporting 176
 - reporting engine 187
 - sample files for payment consolidation 291
 - testing report XML 192
 - to generate the UI 155
 - variables for payment consolidation templates 300

- Velocity 31, 143, 146, 147
- Velocity and attributes for prompts element 157
- Velocity and attributes for query element 154
- Velocity and changing files 27
- Velocity and drilldown and breadcrumb link 191
- Velocity and predefined context variables 179
- Velocity and reporting API 183
- Velocity and reporting engine 144
- Velocity and SQL query 186
- Velocity engine in parsing 182
- Velocity for internationalization and localization 187
- Velocity, changing 195
- XML operationGroup element report 170
- translation** 32, 33
- Turkish** 32
- U**
- U.S. English** 32, 33
- UNIX**
 - Oracle WebLogic payment debug flag 244
- updating disputes in an external CRM** 64
- user audit data, reporting** 357
- user authentication and authorization, customizing** 49
- user enrollment, customizing** 46
- user interface**
 - changing the default language 33
 - customizing 28
 - customizing for multiple branding 31
 - files 28
 - language resource (property) files 33
 - languages available 32
 - localizing 32
 - localizing for additional languages 33
- user lockout**
 - customizing 48
- user passwords**
 - customizing the administrator password validation rule 43
 - customizing the agent password update requirements 43
 - customizing the end user and agent password validation rule 44
- user roles** 51
 - adding 55
- user.xma.xml file**
 - using to configure maximum attempt

- times 48
- using Chase Paymentech as a payment processor** 221
- using CyberSource as a payment processor** 221
- using payment debug** 244
- using PayPal Payflow Pro as a payment gateway** 220
- using the enrollment validation API** 47
- using the report list properties file** 149
- using the reporting engine** 143

V

- validation code generator, customizing** 47
- variables for payment consolidation templates** 300

Velocity

- and changing files 27
- and ReportContext class 147
- and reporting engine 143, 144
- changing templates 195
- new version 145
- template 146
- template and attributes for query element 154
- template and drilldown and breadcrumb link 191
- template and reporting API 183
- template and SQL query 186
- template engine in parsing 182
- templates 31
- templates and attributes for prompts element 157
- templates and predefined context variables 179
- templates and reporting 176
- templates for internationalization and localization 187
- version of Oracle Billing Insight, identifying** 26
- viewing log4j log files** 23
- Visa authorization transactions with Paymentech** 216

W

- Web documents** 29
- Web Services application** 14
 - EAR file 22

X

- XML** 266

Z
zero dollar authorization in

Paymentech 216