Communications Billing Analytics - Business Edition Architecture Overview

V5.0.3
Date Published: 5.10.05

Siebel Systems, Inc., 2207 Bridgepointe Parkway, San Mateo, CA 94404
Copyright © 2005 Siebel Systems, Inc.
All rights reserved.
Printed in the United States of America

No part of this publication may be stored in a retrieval system, transmitted, or reproduced in any way, including but not limited to photocopy, photographic, magnetic, or other record, without the prior agreement and written permission of Siebel Systems, Inc.

Siebel, the Siebel logo, UAN, Universal Application Network, Siebel CRM OnDemand, TrickleSync, Universal Agent, and other Siebel names referenced herein are trademarks of Siebel Systems, Inc., and may be registered in certain jurisdictions.

Other product names, designations, logos, and symbols may be trademarks or registered trademarks of their respective owners.

PRODUCT MODULES AND OPTIONS. This guide contains descriptions of modules that are optional and for which you may not have purchased a license. Siebel's Sample Database also includes data related to these optional modules. As a result, your software implementation may differ from descriptions in this guide. To find out more about the modules your organization has purchased, see your corporate purchasing agent or your Siebel sales representative.

U.S. GOVERNMENT RESTRICTED RIGHTS. Programs, Ancillary Programs and Documentation, delivered subject to the Department of Defense Federal Acquisition Regulation Supplement, are “commercial computer software” as set forth in DFARS 227.7202, Commercial Computer Software and Commercial Computer Software Documentation, and as such, any use, duplication and disclosure of the Programs, Ancillary Programs and Documentation shall be subject to the restrictions contained in the applicable Siebel license agreement. All other use, duplication and disclosure of the Programs, Ancillary Programs and Documentation by the U.S. Government shall be subject to the applicable Siebel license agreement and the restrictions contained in subsection (c) of FAR 52.227-19, Commercial Computer Software - Restricted Rights (June 1987), or FAR 52.227-14, Rights in Data—General, including Alternate III (June 1987), as applicable. Contractor/licensor is Siebel Systems, Inc., 2207 Bridgepointe Parkway, San Mateo, CA 94404.

Proprietary Information

Siebel Systems, Inc. considers information included in this documentation and in Siebel eBusiness Applications Online Help to be Confidential Information. Your access to and use of this Confidential Information are subject to the terms and conditions of: (1) the applicable Siebel Systems software license agreement, which has been executed and with which you agree to comply; and (2) the proprietary and restricted rights notices included in this documentation.

Contents

1 Preface

About Communications Billing Analytics

Communications Billing Analytics empowers both business managers and individual employees to analyze and understand their communication costs and usage by investigating and identifying trends and patterns across multiple views of their own unique organization.

The Communications Billing Analytics application is built based on Siebel’s eXtensible Modular Architecture (XMA™) concepts. XMA is a J2EE platform developed for rapid deployment of industry specific Customer Self-Service and e-Billing applications. XMA offers multi-channel presentation, configurable business logic, real-time and batch integration, several data repositories, and system management tools.

The main features of Communications Billing Analytics are:

Billing Reports – Provides visibility into the account level charges summary and contract level call details within an invoice. Multiple accounts can be consolidated through a user-defined hierarchy to report on invoice groups. Reports provide drilldown capability from high-level summaries to specific call details. The drilldown through the billing hierarchy of accounts, contracts, and call details is defined by the structures inherent in the billing data feed.

Organizational Spending Reports – Provides cost allocation management through cost center summary and contract level call detail reporting. Reports provide drilldown capability from high-level summaries to specific call details.

Custom Reports – Most standard billing and organizational spending reports may be customized to provide pseudo-ad hoc reporting capabilities. A user specifies filter parameters from a dropdown list or by entering a search string corresponding to each column in the report. Additional parameters such as bill periods, relative dates, and value ranges may also be specified depending on the report. Once the custom version of the report is run, a user may save the report parameters for future use.

Interactive or Batch Reporting – Requests for reports may be serviced interactively, in real-time, or off-line as a batch process. Batch processing can be requested for reports that may take some time to generate; the user can be notified when the report is ready.

Printer-Friendly and Download – The default output for the reporting engine is HTML for viewing in an internet browser. Many reports support additional output formats including a printer-friendly HTML format designed to be compatible with most printers or formatted into a CSV file for download to the user’s computer. Additional download formats may be configured based on customer requirement.

Business and Billing Hierarchy Management – Hierarchies allow customers to organize their billing data in ways that make sense for their organization. Billing hierarchies are automatically created by Communications Billing Analytics, based on the customer’s billing structure. Additional organization (or business) hierarchies, based on a combination of business objects such as cost center, location, or department, can be created to help organize data along appropriate lines. Customers can control access to billing data by assigning users to specific locations in a hierarchy – users can only see data at or below their location in the hierarchy. This feature can be used to grant department managers the ability to review reports on accounts associated with users in their department, while preventing them from seeing data associated with other departments. Communications Billing Analytics provides support for creating and maintaining organizational hierarchies.

About This Guide

This guide is intended for Siebel system integrator partners, senior developers with an Siebel client company, and Siebel Professional Services representatives who need to understand the overall Communications Billing Analytics Architecture.

The document assumes you have an in-depth understanding of and practical experience with:

- Java 2 Enterprise Edition (J2EE), Enterprise JavaBeans (EJBs), servlets, and JSPs
- Database concepts

This guide also assumes you have:

- Read the Communications Billing Analytics product documentation and are familiar with Communications Billing Analytics functionality
- Knowledge of how to develop J2EE web applications using JSP, Struts, Tiles, Velocity, and XML

Related Documentation

This guide is part of the Telco documentation set including Communications Self Service Manager and Communications Billing Manager. For more information about implementing your Communications Billing Analytics application, see one of the following guides:

Document	Description
Communications Billing Analytisc Application Guide	Provides an overview of Communications Billing Analytics functionality.
Communications Billing Analytics Installation Guide	How to install Communications Billing Analytics.
Self Service Manager Administration Guide	How to set up and run a live Self Service Manager application in a J2EE environment.
Communications Billing Analytics Software Development Kit	How to extend, develop and otherwise work with the Communications Billing Analytics product.
Packaging and Deployment Guide	How to package and deploy customized Siebel applications.

Obtaining Siebel Software and Documentation

You can download Siebel software and documentation directly from Customer Central at https://support.edocs.com. After you log in, click the Downloads button on the left. When the next page appears, a table displays all of the available downloads. To search for specific items, select the Version and/or Category and click the Search Downloads button. If you download software, Siebel Technical Support automatically sends you (the registered owner) an email with your license key information.

If you received an Siebel product installation CD, load it on your system and navigate from its root directory to the folder where the software installer resides for your operating system. You can run the installer from that location, or you can copy it to your file system and run it from there. The product documentation included with your CD is in the Documentation folder located in the root directory. The license key information for the products on the CD is included with the package materials shipped with the CD.

If You Need Help

Technical Support is available to customers who have an active maintenance and support contract with Siebel. Technical Support engineers can help you install, configure, and maintain your Siebel application.

Information to provide

Before contacting Siebel Technical Support, try resolving the problem yourself using the information provided in this guide. If you cannot resolve the issue on your own, be sure to gather the following information and have it handy when you contact technical support. This enables your Siebel support engineer to more quickly assess your problem and get you back up and running more quickly.

Please be prepared to provide Technical Support the following information:

Contact information:

- Your name and role in your organization.
- Your company's name
- Your phone number and best times to call you
- Your e-mail address

Product and platform:

- In which Siebel product did the problem occur?
- What version of the product do you have?
- What is your operating system version? RDBMS? Other platform information?

Specific details about your problem:

- Did your system crash or hang?
- What system activity was taking place when the problem occurred?
- Did the system generate a screen error message? If so, please send us that message. (Type the error text or press the Print Screen button and paste the screen into your email.)
- Did the system write information to a log? If so, please send us that file. For more information, see the *Troubleshooting Guide*.
- How did the system respond to the error?
- What steps have you taken to attempt to resolve the problem?
- What other information would we need to have (supporting data files, steps we'd need to take) to replicate the problem or error?
- **Problem severity:**
- Clearly communicate the impact of the case (Severity I, II, III, IV) as well as the Priority (Urgent, High, Medium, Low, No Rush).
- Specify whether the problem occurred in a production or test environment.

Contacting Siebel Technical Support

You can contact Technical Support online, by email, or by telephone.

Siebel provides global Technical Support services from the following Support Centers:

US Support Center
Natick, MA
Mon-Fri 8:30am – 8:00pm US EST
Telephone: 508-652-8400

Europe Support Center
London, United Kingdom
Mon-Fri 9:00am – 5:00 GMT
Telephone: +44 20 8956 2673

Asia Pac Rim Support Center
Melbourne, Australia
Mon-Fri 9:00am – 5:00pm AU
Telephone: +61 3 9909 7301

Customer Central
https://support.edocs.com

Email Support
mailto:support@edocs.com

Escalation process

Siebel managerial escalation ensures that critical problems are properly managed through resolution including aligning proper resources and providing notification and frequent status reports to the client.

Siebel escalation process has two tiers:

1. **Technical Escalation -** Siebel technical escalation chain ensures access to the right technical resources to determine the best course of action.
2. **Managerial Escalation -** All severity 1 cases are immediately brought to the attention of the Technical Support Manager, who can align the necessary resources for resolution. Our escalation process ensures that critical problems are properly managed to resolution, and that clients as well as Siebel executive management receive notification and frequent status reports.

By separating their tasks, the technical resources remain 100% focused on resolving the problem while the Support Manager handles communication and status.

To escalate your case, ask the Technical Support Engineer to:

Raise the severity level classification

1. Put you in contact with the Technical Support Escalation Manager
2. Request that the Director of Technical Support arrange a conference call with the Vice President of Services
3. Contact VP of Services directly if you are still in need of more immediate assistance.

2 Communications Billing Analytics Architecture

Communications Billing Analytics Features

The following table provides a list of the use cases that highlight the functionality of Communications Billing Analytics based applications:

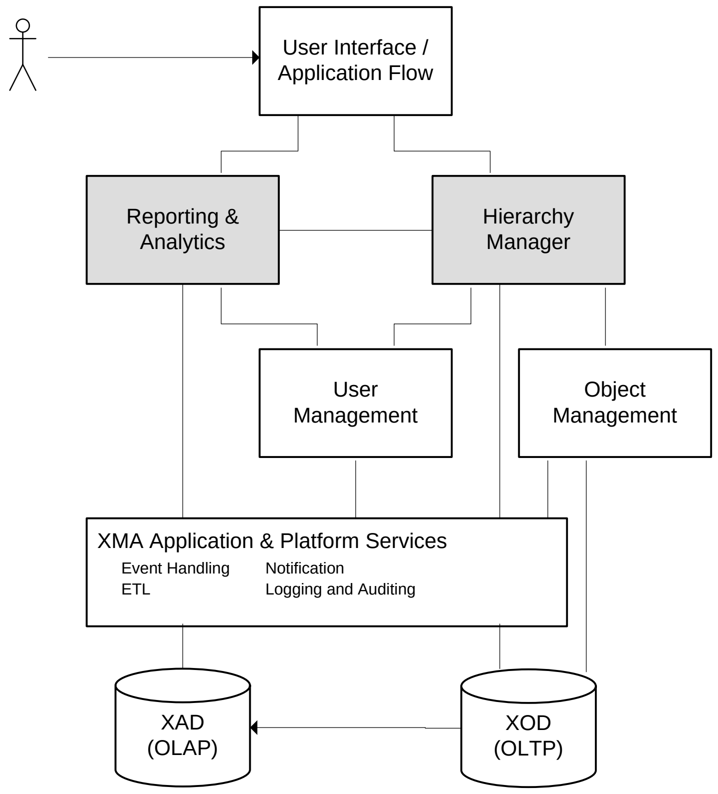
Requirement Category	**Description**	**Use Cases**
General	Users can cancel, page, sort, filter and otherwise manage pages and page contents.	Cancel a request Sort report data Change report query parameters Select printer friendly version of a report Download report data
Reporting	Users can view any of a set of predefined reports, as well as create and save custom reports. Reports may be displayed on demand or batch processed	Display report list Display billing report Display *Top X* report Create and save a custom report Run a report in batch mode View a list of available batch reports
Hierarchy	Users can navigate and manage their billing and organization hierarchies according to their assigned privileges	Navigate/Search hierarchy Create/Edit/Delete hierarchy Set position in hierarchy Import/Download hierarchy
Database	System synchronizes hierarchy data in XOD (eXtensible Operations Database) with the XAD (eXtensible Analytics Datamart) when changes occur, such as: billing data is loaded, existing hierarchy is changed, new hierarchy is created	Synchronize hierarchy data in XOD with XAD

For a list of the out-of-the-box reports, see the Application Guide.

Communications Billing Analytics Architecture

Communications Billing Analytics Architecture Overview

The figure below illustrates the major functional components in the Communications Billing Analytics product.



Communications Billing Analytics provides end users with the ability to access reports analyzing billing data and organizational spending. Report data is subject to hierarchy-based access control (HBAC). Billing and organizational hierarchies reflecting business structures – such as cost center or location – allow for multiple views of the data while at the same time limiting the data that users can access. Users may create complex business structures to report on usage patterns and cost trends throughout their organization. Users can view reports for the hierarchy nodes to which they have been assigned. Administrators with appropriate privileges can manage their organizations hierarchies.

Architecture Components

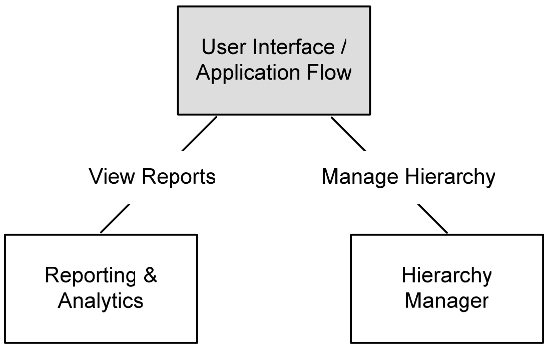
The Communications Billing Analytics architecture is based on a set of core sub-systems that allow developers and external operational support systems (OSS) to interact with Communications Billing Analytics . Conceptually, the architecture consists of the following sub-systems:

- *User Interface / Application flow* – Users work with Communications Billing Analytics through the UI. The user interface is built on the Struts MVC model and includes a set of JSP pages, Velocity templates, and Java classes that implement the page flow and render the user experience. The user interface provides users with access to analytic reports and the hierarchy manager.
- *Reporting and Analytics* – This sub-system is responsible for generating reports, including extracting data from the eXtensible Analytics Datamart (XAD) and transforming that data prior to its rendering by the user interface. The Reporting & Analytics module accesses user information from the User Management system to determine a user's access rights for reporting. It also accesses hierarchy information in order to determine in which areas of which hierarchies the user has privileges. Reporting and Analytics allows users to create customized reports – these custom reports are associated with the user that created the report, via the User Management system.
- *Hierarchy Manager* – This sub-system is used to manage billing and organizational hierarchies. Billing hierarchies are derived from the customer's billing data, while organizational hierarchies are created by the customer to reflect some meaningful organization structure based on, for example, cost center, department, or location. A user with administrative privileges can create and modify hierarchies, as well as assign users to positions in the hierarchy. Each node in the hierarchy is linked to a business object in the database. Hierarchy Manager works with the User Management subsystem to accomplish these goals.
- *Object Management* – this sub-system is used by Communications Billing Analytics to model the business objects that hierarchy nodes are linked to. Out-of-the-box, Communications Billing Analytics includes definitions for the following types of business objects: folder, account, company, service, service agreement, charge type, and service charge type. Each of these object types includes a set of attributes that can form the basis of a search. Instances of these objects are persisted to the XOD database. Additional object types can be modeled by implementing the appropriate Java interfaces. These custom objects can then be instantiated and used as nodes in a hierarchy.
- *User Management* – this sub-system is responsible for maintaining user, account, and session information and for providing that information to the components that require it. The Reporting and Analytics module requires information about the user's reporting context and privileges in order to render a report that contains only data to which the user has access rights. In addition, Reporting and Analytics associates saved custom reports with the users that created them. Hierarchy Manager retrieves user privileges from the User Management component and associates users with positions in the hierarchies.

- *XMA Application and Platform Services* – the XMA platform provides a key set of services that can be utilized by any XMA-based application. Among the services that Communications Billing Analytics uses are: Event handling, ETL, Notification, and Logging and Auditing.
- *XAD and XOD databases* – the XAD (eXtensible Analytics Datamart) is the repository for all the analytic data used in reporting. The XOD (eXtensible Operations Database) contains transactional data (payments, user management, accounts, etc.). Hierarchy data is represented in both databases. Changes made through Hierarchy Manager are written to the XOD and then synchronized to the XAD. Hierarchy data that is uploaded into the XAD as part of an ETL process is synchronized to the XAD.

In the remainder of this document, these subsystems are described in more detail.

User Interface and Application Flow

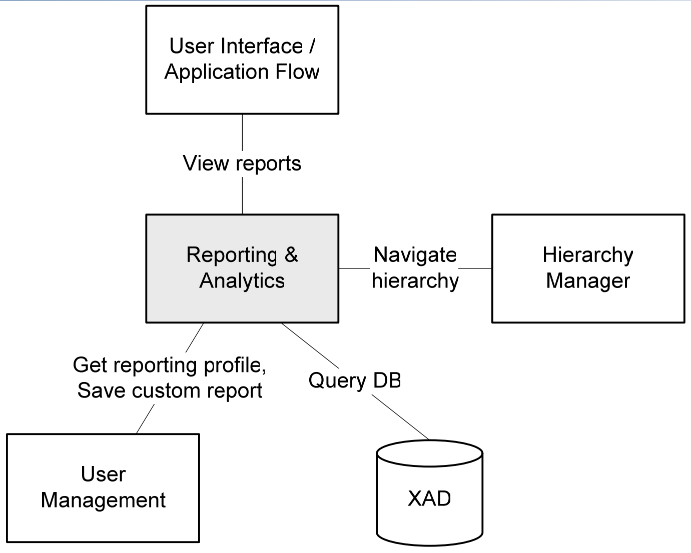


The user interface and application flow system is Communications Billing Analytics front-end. It provides the GUI that allows users to view and interact with reports and to manage the hierarchy. The UI has been designed to be highly customizable. The Struts MVC framework is used for application flow. Struts Form beans perform basic validation on users' submitted requests, while Action classes implement business logic validation and interact with the back end data model (this may involve, for example, requesting the reporting module to generate a named report). Communications Billing Analytics provides core form bean and action classes which should suit most customer's needs; however, extension of these classes can be performed.

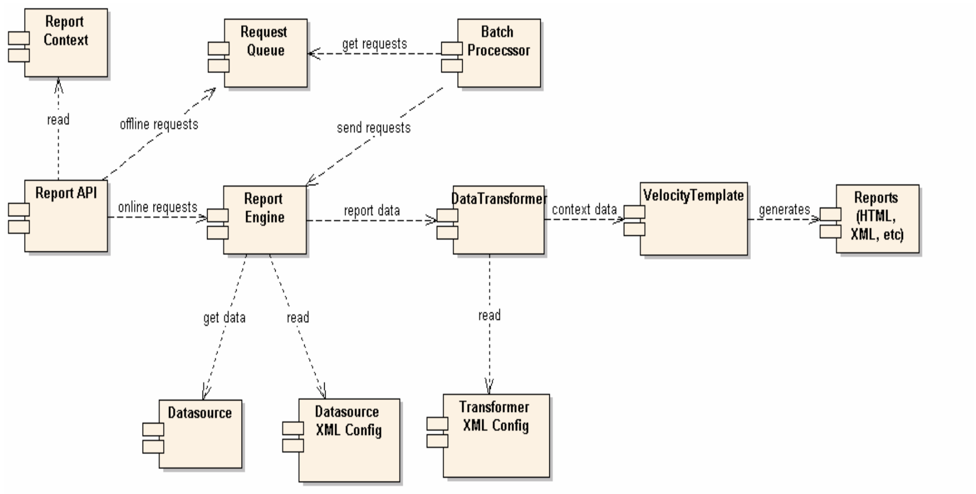
The html page that the user sees is rendered using the Tiles framework: the page is constructed compositionally from a set of tiles, where each tile is typically implemented as a customizable JSP page. Cascading style sheets govern the overall look and feel of the application.

Reporting and Analytics

Reporting and Analytics depends on the components shown below.



The reporting module itself is comprised of a number of collaborating packages, as shown in the diagram below. APIs allow customers to integrate with and customize the behavior of the reporting module.



- Report API – a public API that allows report clients to initiate a report request.
- Report Context – holds report context information including the SQL query criteria, user authentication and authorization information, and hierarchy context.
- Request Queue and Batch Processor – used for batch-based reporting processing to improve system's scalability.
- Report Engine – the central controller of the reporting model. It is responsible for coordinating the work of generating and displaying the report.
- Datasource – encapsulates the data that will appear in the report, including the source from which the data is drawn and the SQL query that is used to extract the data from the source. The Datasource API provides methods for manipulating the encapsulated data.

- Datasource Config(uration) – an XML element that is part of the definition of a report. The datasource configuration information includes a URI that points to the source of the reported data (typically, this is the eXtensible Analytics Datamart (XAD), a relational database optimized for reporting). It also defines the SQL query to be executed against the datasource and information about the return type for the columns in the resulting data set.

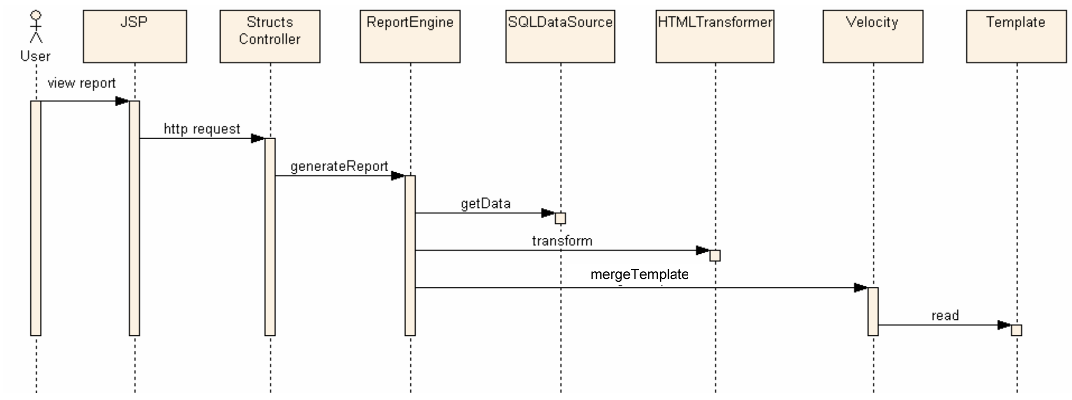
  Multiple data source configurations, referring to multiple datasources, can be specified as part of the definition of a report, allowing data from multiple sources to be combined into a single report.

- Data Transformer – transforms the data according to the Transformer Configuration specification in order to prepare the data for rendering in the user interface. The transformed data is cached for local computation and sorting without needing to initiate a new request to the datasource.
- Transformer Config(uration) – XML configuration information that specifies how the Data Transformer should transform the resulting data set that was extracted by the Datasource.
- Velocity template – each report (actually, each Data Transformer) is associated with a Velocity template that will be invoked to format the requested report. Typically, the template renders the data in a table, although data can also be displayed using other HTML elements such as drop-down lists, check boxes, etc.

Each Communications Billing Analytics report is defined by an XML report element. The report element specifies two important pieces of information: the Datasource Config and the Transformer Config. Developers can create custom reports by supplying a new XML definition for the desired report. Additionally, the reporting components include published APIs that allow customization of report processing, external submission of report requests, etc. See the Developers' Guide for more information on extending the Reporting & Analytics component.

Once a report is generated, it is cached in the user's session, for efficient manipulation (paging, sorting, etc.). Reports are rendered using a combination of JSP files and Velocity templates, which can be customized to meet a customer's display requirements.

The basic process flow for servicing a report request is illustrated below.



1. User submits a request for a report by making a selection in the user interface. Typically, this involves clicking a link or selecting report parameters in a form and performing a post action. (Alternatively, reports may be requested via the APIs.)

2. The Struts Controller – the Action Class – calls the Report Engine to generate the specified, named report. At this point, any query parameters that serve as input to the report are made available in the Struts Form bean.

3. The Report Engine looks up the report definition and then invokes the datasource to retrieve the SQL data.

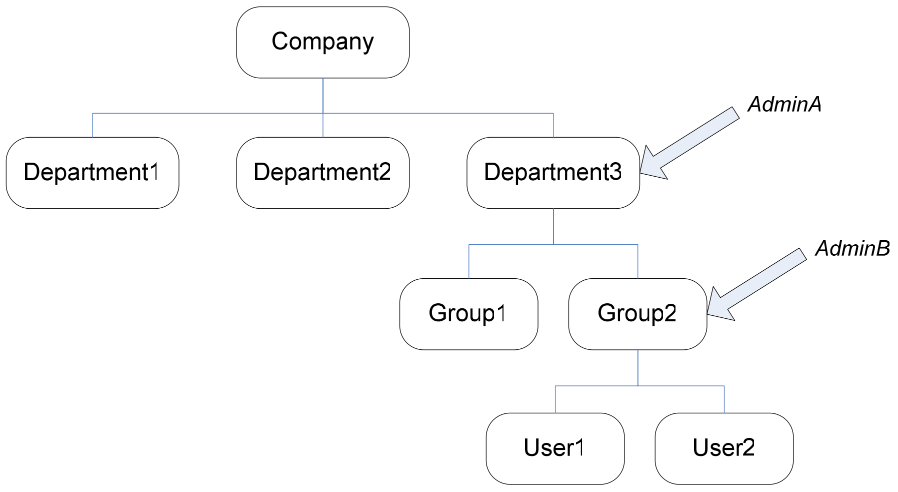
4. The result set is transformed according to the report's transformer definition. At this point, the resulting report object is cached in the user's session.

5. The Velocity template engine is called to render the report. The report is added to the Velocity context, and the report's associated template is then parsed. The resulting HTML is displayed to the user, typically in a tile in the User Interface.

The Reporting component provides a *callback* mechanism between the Struts Action Class and the Report Engine. This callback mechanism allows a developer to insert custom processing hooks into the report process flow. For example, special processing can be performed pre- and post- getting the report and pre- and post- sorting the report. For more information, see the Developer's Guide use cases related to the callback mechanism.

Hierarchy Manager

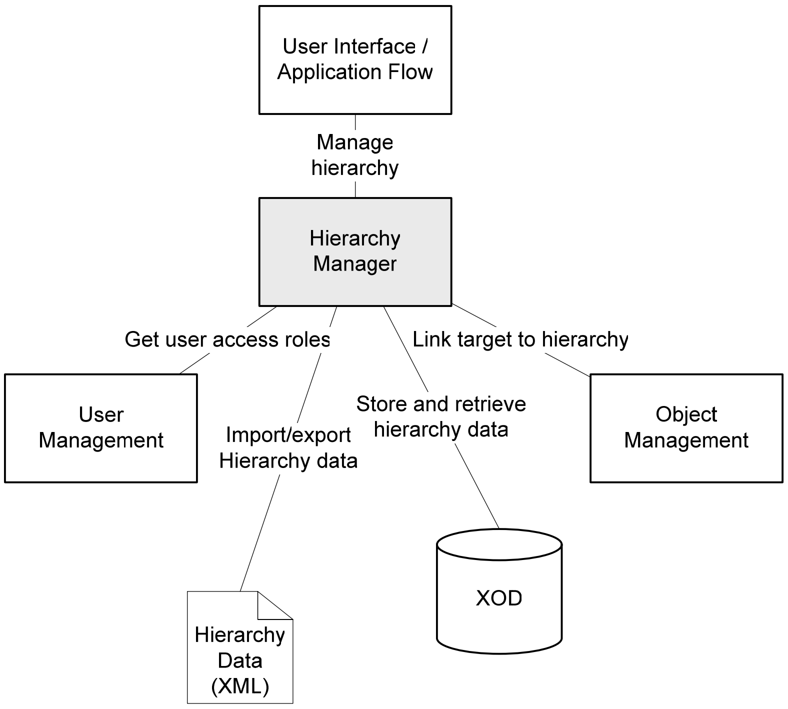
Hierarchy Manager provides a generic class structure that can be used to describe hierarchical relationships between different types of business objects. The business objects can be of any kind of concrete class as long as they obey interfaces defined by the hierarchy model. For purposes of illustration, a sample hierarchy is shown below.



The hierarchy structure utilizes a set of generic classes (Node, Link, Attribute) to represent the relationships between nodes. In the above example, each node is an instance of a Node object, and each connection is represented by a Link object. In addition, each node is associated with, or linked to, a business object that is ultimately represented by an entry in a table in the database. The hierarchy in the example above combines several types of linked business objects: company, department, group (a custom object type) and user. These business objects are implemented using the object management system.

Administrative users, such as "AdminA" and "AdminB" in the example, can be assigned to positions in the hierarchy, giving them administrative privileges over that position and below.

The Hierarchy Manager depends on the components shown below.



All hierarchy changes initiated through the Hierarchy Manager UI are written to the XOD database. As an alternative to manual creation of hierarchies, hierarchy data can be imported / exported by an administrator via XML files.

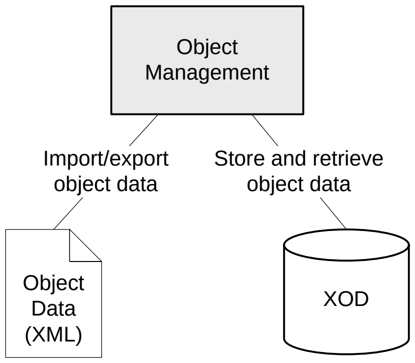
APIs are provided to create and manipulate hierarchies, import and export hierarchy data, and link additional business object types. Features of the hierarchy manager APIs include:

- Lazy loading to improve performance – when a hierarchy node is accessed, the related objects (associated users, attributes, and child nodes) are not loaded if lazy loading is turned on (the default). Lazy loading reduces unwanted data loading of the hierarchical tree, which gives application developers control on when to load what data into memory. Lazy loading is completely transparent to modules outside hierarchy.

- Flexible transaction management – transaction proxy interceptors are used to define transaction boundaries. The interceptor is aware of the outer transaction, e.g., if a method is invoked to create a hierarchy node within the context of a bigger transaction, the current method will not be committed until the outermost transaction is committed. This allows several hierarchy related operations to be grouped together into a single transaction.

- Concurrency control – additional version fields are added to each hierarchy object database table, allowing a version number to be kept for each record. This allows dirty writes to be prevented; multiple threads can operate on the same record without overriding each other's changes.

Object Management



The object management framework (omf) provides support for business object creation and manipulation, registration, lookup, and persistence, as well as ETL upload and import/export. Business object instances are stored in the XOD database. Some of these instances are created automatically during the billing data load process, while others are created via import from an XML file (there is no UI to allow an administrator to create business objects directly). These business objects can be linked into a hierarchy if the business object class implements the proper interface.

User Management

The user management system can be viewed as being comprised of four related XMA modules that implement user, account, session, and security functions.

- User management framework (umf) – handles user information, including user authentication, validation, and persistence of profile information. Users can be associated with different types of profiles, such as login, reporting, notification, and security, and custom profiles can be developed using the APIs. User information is persisted by default to the XOD database, although the module can be customized to integrate with a customer's user management system.

- Account management framework (amf) – handles account information, including account access, validation, lockout, and persistence. Accounts can be associated with nodes in the hierarchy using Hierarchy Manager. Accounts are associated with users managed through the UMF module.
- Session management framework (smf) – handles session information, including session access (login, logout, timeout), validation, and persistence. APIs allow manipulation of session attributes. When a user requests a report, the resulting report object is cached in the user's session.
- Security – provides robust authentication, authorization, and role management services, including Role-Based Access Control (RBAC) and single sign-on (SSO).

XAD, XOD, and Synchronization of Hierarchy Data

The XOD is an operational database that contains transactional data (payment, account information, user information, etc.). The XAD is a database designed to support analytics. All the data used by the Report Engine to generate reports comes from the XAD database. The XAD includes flexible database fields to allow customers to integrate their own custom data into the reports (see the Data Dictionary document for more information on these flexible fields). Synchronization of related data between the two databases is required.

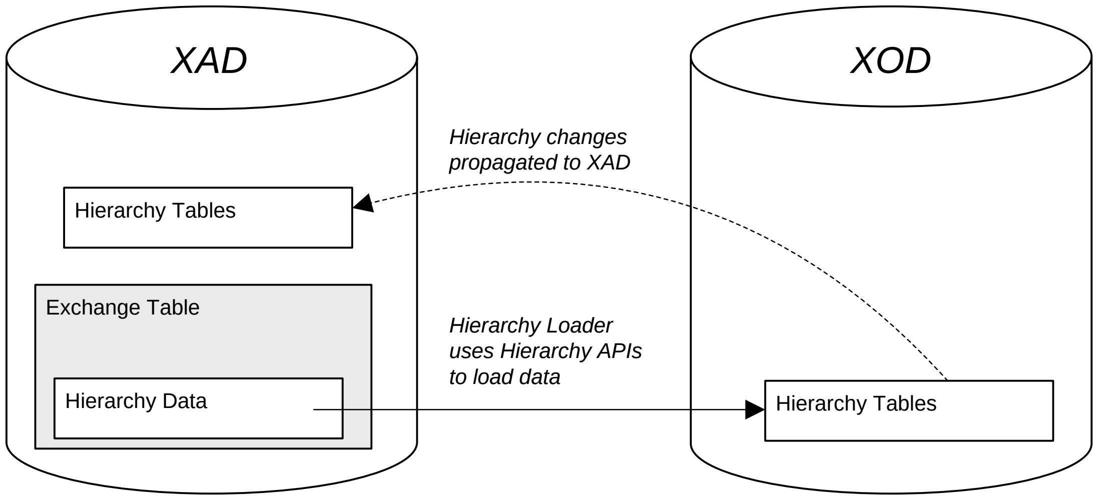
Hierarchy data is found in both databases. Changes to the hierarchy must be synchronized. Changes to the hierarchy can occur either as the result of administrator activity (submission of changes through the user interface) or during data upload batch jobs.

Changes made to the hierarchy through the user interface (hierarchy *transactions*) are written by Hierarchy Manager to the XOD database. Hierarchy manager triggers certain types of events when hierarchy changes happen. The event handler processes these events, which result in the update of hierarchy data in the XAD database. The changes made in XOD and XAD are bounded by a single transaction to guarantee data integrity.

This means that hierarchy changes made using the Hierarchy Manager's APIs result in updates being made to both the XOD and XAD databases, whether these changes are initiated interactively by an administrative user through the Hierarchy Manager's user interface, by an API call, or some data upload.

During a billing upload ETL process, billing data is loaded into the XAD database. Some of this data (account, service, company, hierarchy) needs to be loaded into the XOD database. The XAD load process records those changes that need to be synchronized in an exchange table, then an ETL controller job calls different module loaders to load module specific data into the XOD. For example, hierarchy data is loaded by invoking the Hierarchy ETL Loader. This loader reads hierarchy data from the exchange table and populates hierarchy tables into the XOD using the Hierarchy Manager's APIs. As changes are made to the hierarchy tables in the XOD, events will cause these changes to be pushed back into the XAD.

The process of propagating hierarchy data during an ETL load is illustrated below. During the ETL process, hierarchy data is loaded into the XAD's data exchange table. This data is then propagated over to the XOD, during which it is in turn pushed back to the XAD's hierarchy tables.


XAD
XOD
Hierarchy changes propagated to XAD
Hierarchy Tables
Exchange Table
Hierarchy Data
Hierarchy Loader uses Hierarchy APIs to load data
Hierarchy Tables