Self-Service Manager for Communications Architecture Overview

Version 5.0.0
Date Published: 3.31.05

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.

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 Customer Self-Service and Siebel Tools

Siebel's Self-Service for Communications includes every application that communications service providers need to enable a complete online customer-self service experience at their website. The suite includes software applications for:

- e-Billing and Payment
- Service and Order Management
- Point-of-Sale
- Reporting and Analytics
- Rate Plan Advice

Siebel's Self-Service applications for the telecommunications industry combine Siebel's unrivaled Customer Self-Service and e-Billing software suite with its extensive industry domain expertise. The packaged, out-of-the-box applications are tailored to solve communications service providers' distinct business problems and to meet communications industry-specific process requirements.

Siebel's Self-Service for Communications includes:

Communications Billing Manager

Communication Billing Manager is a complete e-billing application for communications service providers that gives business and consumer customers valuable and convenient access to their communications bills along with the ability to easily make online payments.

Communications Self-Service Manager

Communications Self-Service Manager enables customers of communications service providers to manage every aspect of their service relationship online. From a single convenient interface, customers can easily activate and manage subscriptions, change rate plans and features, and modify subscriber profile settings. Business customers are able to complete these activities for individual employees, as well as company departments and divisions, across their entire organization.

Communication Analytics Manager

Communication Analytics Manager is a reporting solution for business customers that empowers both individual employees and business managers to analyze and understand their communications costs and usage by investigating and identifying trends and patterns across multiple views of their own unique organization.

Rate Plan Advisor

Rate Plan Advisor is a web-based application that recommends the ideal rate plan for wireless subscribers in real-time. Individual consumers as well as large businesses can analyze their actual historical voice/mobile/data usage, find the best-fit rate plans, and compare the features offered by those plans. With its intuitive wizard user interface, Rate Plan Advisor quickly guides end-customers or customer service representatives through the entire analysis process. In addition, a service provider’s customer care and marketing groups can also use Rate Plan Advisor to identify pre-churn subscribers, simulate new rate plans, and run predictive analytics.

About This Guide

The Siebel Self Service Manager Software Developers Kit allows developers to write custom code against Sibel applications. This guide is intended for Sibel system integrator partners, senior developers with an Sibel client company, and Sibel Professional Services representatives who need to understand the overall architecture Self Service Manager Architecture.

The Self Service Manager Architecture 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 Self Service Manager product documentation and are familiar with Self Service Manager functionality
- Knowledge of how to develop J2EE web applications using JSP, Struts, Tiles and XML

Related Documentation

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

Print Document	Description
Self Service Manager Developers Guide	How to extend, develop and otherwise work with the Self Service Manager product.
Billing Manager and Self Service Manager Installation Guides	How to install Self Service Manager and Billing Manager in a distributed environment.
Billing Manager Presentation Design Guide	How to use Composer to define the rules for mapping data to templates for viewing statements.
Billing Manager Administration Guide	How to set up and run a live *Billing Manager* application in a J2EE environment.
Billing Manager Data Definition Guide	How to use DefTool to define the rules for data extraction in a DDF file.
Billing Manager Developers Guide	How to extend, develop and otherwise work with the *Billing Manager* product.

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

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

2 Self Service for Communications Architecture

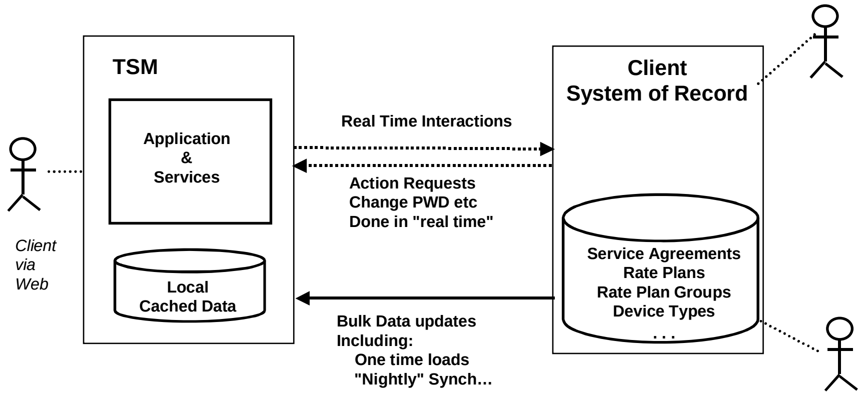
Self Service Manager Features

Self Service Manager, as a whole, interconnects Siebel applications with client systems for the purpose of web enabling a variety of client services. In general Self Service Manager allows a subscriber to view his/her account information via the web and change and/or modify those settings in real-time. Such a real-time interaction is propagated to the clients system, where the actual operation is performed. The client system then returns a response, request completed, failed etc and the client, again via the web interface, sees the change immediately.

Self Service Manager contains a local data store that must be initially loaded and then updated over time. The Self Service Manager data store is used to manage data such as service agreements, rate plans, device type data etc. The Self Service Managerf data store is required to provide acceptable web performance.

Since outside actors, such as client customer service representatives, may change the client system of record the Self Service Managerf bulk update process is used to maintain a consistent state within both systems. For example, imagine the client calls a CSR and makes a change to a voice mail password. The resulting changed service agreement would then be re-synchronized within the Self Service Managerf database via the nightly data push from the client system. The diagram below shows these concepts.

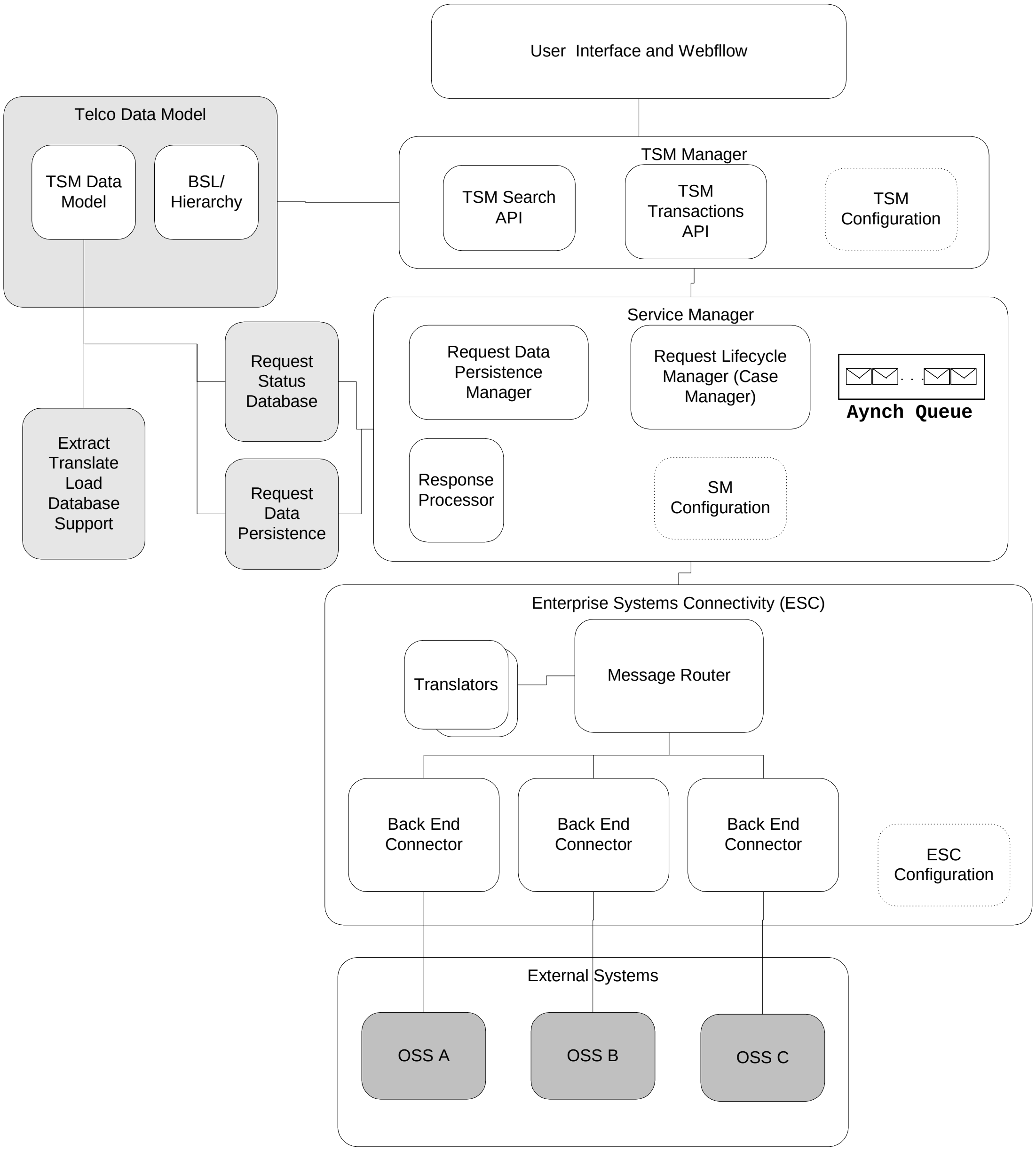
Conceptual Overview



The following table provides a complete list of the use cases that explain in more detail the functionality of Self Service Managerf 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 data in a table Filter data in a table Select printer friendly version of a page
Manage Account	Users select rate plans and features. Adding, deleting and changing features individually or in bulk. Users can also change port numbers, change phone numbers and/or serial number associated with a phone or set of phones.	Change Rate Plan (individual & bulk) Add/Delete Features (individual & bulk) Change Mobile Telephone Number Change Device Serial Number(individual & bulk) Change Port Number from one subscriber to another (individual & bulk)
Manage Service	Users may manage phone service, activating, deactivating, suspending or resuming service.	Activate Phone (individual) Suspend Service (individual) Resume Service (individual) Deactivate Number(individual)
Order Status	Users may review the status of a current order or requested change.	Order Status
Change Subscriber Profile	Users may change various profile characteristics such as name or password.	Change Voicemail Password Change Address Change Mobile User Name

Self Service Manager Architecture



Self Service Manager Architecture Overview

Overview

Self Service Manager provides end users with facilities to process service/order requests in accordance with existing business rules while maintaining self-service privileges and scope. The business rules associated with products and services are maintained and stored within Communications Provider systems. The Siebel Self Service for Communications Suite governs the actions a user can perform by delegating roles and privileges. The scope of a user then dictates what accounts and/or handsets on which they can perform these actions upon.

These Siebel rules dictate what data a user has access to and how they should be able act on the data. These Siebel enabled rules are used in conjunction with rules stored within Provider systems (like rate plan and product catalogs) to supply the total self-service capabilities. Siebel provides both inbound and outbound message APIs in order to be able to synchronize data needed to complete self-service transactions.

Architecture Components

The Self Service Manager architecture is based on a set of core concepts and sub-systems that allow developers and external operational support systems (OSS) to interact with the Self Service Manager core in order to provide service management. Conceptually Self Service Manager is based on:

- Service Agreements – the definition of services provided for a given device, such as a wireless phone, as selected by a subscriber. Service Agreements are defined when such a device is purchased, and normally includes a set of features based on those specified for a given rate plan. Rate plans define the starting point feature set for service agreement and include definitions of core and optional features.
- Local Database – Self Service Manager relies on a local database for access to current data about subscribers, service agreements and service agreement components. The local database must be pre-loaded and synchronized with the provider system of record in order to provide the required run-time services. Self Service Manager provides extensive data loading support and run-time support for managing the local database.
- Operational Support Systems (OSS) – back-end systems which implement the actual functionality required to modify a service agreement or contract. OSS systems typically are entities (e.g. billing systems or CRM systems) that provide interfaces to enable fulfillment of the various transactions.
- Self Service Manager Transactions – the operations that can be performed on service agreements. An example operation might be Change Voice Mail Password or Enable Caller ID. Self Service Manager Transactions can either be *individual*, e.g. apply only to a single service agreement, or *bulk*, e.g. applying to many service agreements.

- Self Service Manager Searches – queries against a service manager database for current features, rate plans, and service agreements. Additionally the Self Service Manager Search APIs support propagating updates to the Self Service Manager local database.
- Self Service Manager Connectors or proxies – implementations, typically written in Java, based on a well defined interface which connect the Self Service Manager Service Manager to a back-end OSS which performs operations in real-time.
- Self Service Manager Translations – well defined data transforms which take Self Service Manager Transactions and transform them into a format suitable for a given OSS. Typically, Self Service Manager Translations are written using XSLT Transforms.

Self Service Manager Local Database

The Self Service Manager Local Database is a Communications Services specific extension of the Billing Manager database and provides a local copy of subscriber and service agreements and supporting data. The Self Service Manager Local Database is designed for efficient access to Self Service Manager data for the purposes of storage, search and update. Key features include:

- Extract, Translate and Load (ETL) process designed to flexibly and efficiently pre-load and update the Self Service Manager database.
- Efficient data storage and search capacities supporting all major aspects of Self Service Manager.
- Tight integration with provider system of record via interfaces capturing key data lifecycle events and passing these events to custom handlers.

At run-time the Self Service Manager Local data database provides the required fast access to subscriber and service agreement data, without sacrificing synchronization with a system of record. Custom handlers can be developed which are inserted into the database update lifecycle providing synchronization and custom interaction points with Self Service Manager. By combining custom handlers, and Self Service Manager application performance can be maintained without loss of data synchronization. Note that the specifics of developing a custom handler are detailed in the Self Service Manager Developers' Guide. Handlers and developed in Java, and integrated into the Self Service Manager run-time via the spring framework. For more information on the Spring Framework see http://www.springframework.org.

Service Manager(SM)

The Service Manager is a domain-independent component that provides design-time and runtime environment for the execution of domain-specific service transactions. In the communications domain such a transaction might be change voicemail password, etc. Key features include:

- Design-time configuration of service transactions
- Run-time execution of the configured service transaction

- Connectors to OSS and BSS
- Request Lifecycle Management Interactions. Applications can access request and response data at various points during the transaction lifecycle.
- Synchronous and Asynchronous modes of communication to the OSS/BSS
- Response Management, including updating the Billing Manager local database

At design-time, service transactions are *configured* into the Service Manager with attributes such as OSS/BSS adapter with which to communicate with, mode (async, sync etc), which response processor to execute on return, and so on.

At run-time, the SM provides an *execution environment* for these configured services. It identifies an incoming service transaction, records it in a service status database, provides it to an underlying OSS/BSS, processes the response from OSS/BSS, updates the service status database and finally, then invokes the appropriate response processor to perform any custom processing. In the asynchronous case, Service Manager is also responsible for matching inbound asynchronous responses to requests.

Self Service Manager

Self Service Manager is a communications company domain-specific component that provides run time support for execution of communications-specific service transactions. Self Service Manager extends the Services Manager (SM) component to provide communications related transactions for the Self Service Manager application. Key features beyond those provided by SM include:

- Core set of communications specific transactions which can be executed against a given Service Agreement
- Core set of communications specific search functionality supporting both searching for features etc of rate plans, as well as specific service agreements for existing subscribers.

Key features and use cases for Self Service Manager can be found listed in section 1.1 *Self Service Manager Features*.

Enterprise Systems Connectivity

Enterprise Systems Connectivity (ESC) is an OSS/BSS connector framework. The end-to-end implementation of self service management includes communication with OSS/BSS systems such as Billing, CRM etc. ESC provides a framework for these integration tasks and includes support for synchronous and asynchronous communication with OSS/BSS, XML and non-XML based messages, and API calls.

Key features include:

- Extensible Request/Response architecture supporting easy integration of back-end OSS systems and custom transaction definition
- XML-based configuration for adding, removing and updating new ESC connector implementations

- Support for individual and bulk transactions
- Support for Synchronous and Asynchronous[1] communications.

General Process and Flow

The basic self-service Self Service Manager transaction process flow is simple. Typically via a GUI interface, a user chooses a transaction to execute. The GUI application then submits a request to Self Service Manager. Self Service Manager determines which underlying provider should perform the request, and submits the request to a backend system for fulfillment. After the backend has completed Self Service Manager processes the response and returns it to the requestor, optionally calling custom lifecycle handlers. The complexity lies in the details – the individual process flows, heterogeneous back-office systems used for fulfillment and synchronization of Self Service Manager with providers systems of record.

The Self Service Manager local database supplements the Billing Manager data model adding elements such as Plans, Features, Service Agreements and Devices. These elements are required to enable the full breadth of Self Service Manager transactions.

Search and Filter Functionality

Self Service Manager provides a combination of search, filter and update functionality, allowing end users to quickly and easily locate service agreements and service agreement elements. Searches enabled through Self Service Manager will combine the scope of a service agreement that a user has access to (via Hierarchy) with the locally cached data within the Self Service Manager data store. Some characteristics of search and filtering are:

User will be able to search for service agreement by attributes such as:

- Hierarchy location
- Phone Number
- Service Agreement
- Device Type
- Rate Plan Group
- Rate Plan Name
- Subscriber Name.

[1] Supported in version 5.0.1.

Processing Self Service Manager Requests

The processing of any Self Service Manager request involves both outbound (API) and inbound (SPI) interfaces. The API interfaces are responsible for transmitting the user initiated Self Service Manager requests to backend systems while the SPI interfaces are responsible for receiving responses and updates from the OSS provider systems.

Characteristics of SPI implementations:

- Service transactions to backend systems for fulfillment.
- Support service transaction is of a particular type with a set of parameters that are needed to fulfill a given request.
- Transaction parameters for an outbound API definition.
- Common data elements across transactions.
- Support Synchronous Message-Based interaction with backend OSS systems for fulfillment.
- Support bulk transactions by servicing multiple single transactions serially, i.e. passing each individual transaction component, of a bulk transaction, to the underlying provider one by one.

Characteristics of Synchronous Message-based messaging:

- The backend system exposes a remote message based interface (HTTP/XML, Web Services, SOAP etc) to which Self Service Manager can send a service transaction.
- The response from the backend system is real-time, e.g. processed by the back-end system and then returned to the client directory in a synchronous fashion.

Self Service Manager supports transforming messages before sending them to the backend system.

Transaction Status Tracking

Within Self Service Manager a transaction is defined as the sequence of events required to perform a communications specific action relating to a service agreement. Examples of transactions might be enable caller id for a given device. Self Service Manager developers and end users have the ability to view the current status of transactions. Transaction history is maintained and configurable. Transaction history can be searched, filtered, and sorted in order to quickly locate the desired transactions.

Self Service Manager Transactions move through a well-defined lifecycle:

- When a transaction is submitted, it is marked as “Pending” and a Self Service Manager-assigned unique transaction id is generated for the transaction.
- When Self Service Manager receives a response from the backend system, the service is status is updated to “Success”, “Partial Success”, or “Failed” based on the response from the backend system. In the event of failure data provided by the back-end system is preserved and stored with the transaction.

- Based on the response from the OSS/BSS layer (typically a backend egacy system), a “Response Handler” is invoked and performs the appropriate behavior. The Response Handling mechanism provides an extension point for the Lifecycle of a service request.

The backend system defines the semantics of a “Success” or “Failed” transaction. Self Service Manager merely records the status in the service transaction database. The backend system is responsible for indicating information about the success or failure of each transaction. The lifecycle management (SM) is responsible for saving this status information for later retrieval.

Data Model

The master copy of Self Service Manager required data elements is stored in the service provider’s backend system of record. Self Service Manager caches a copy of these data elements for its own use. Initially, when the Self Service Manager application is first deployed, the Self Service Manager database is initialized and then can be updated via jobs as required. Local storage of these data elements is important for several reasons:

- Many of the data elements are presented online as part of the Self Service Manager user experience and it us unreasonable to retrieve this information from external systems for web presentment in a timely manner.
- Data elements need to be available locally to make efficient complex searches.

Initial load into the Self Service Manager data store & Synchronization

For each deployment of Self Service Manager, there will be an initial preparation of the Self Service Manager local database. The data will periodically be synchronized with the legacy system to maintain consistency. Self Service Manager provides a specialized Extract, Translate and Load (ETL) process, specifically designed to support ELT, with the following characteristics:

- An XML-based well-defined data definition supporting bulk loads and updates
- Can be scheduled to run as required to provide periodic batch updates and bulk data loads
- Can add, update, delete and modify cached data.

The service provider is responsible for providing the initial upload feed to load the Self Service Manager system using the data definitions provided.

Synchronization Support

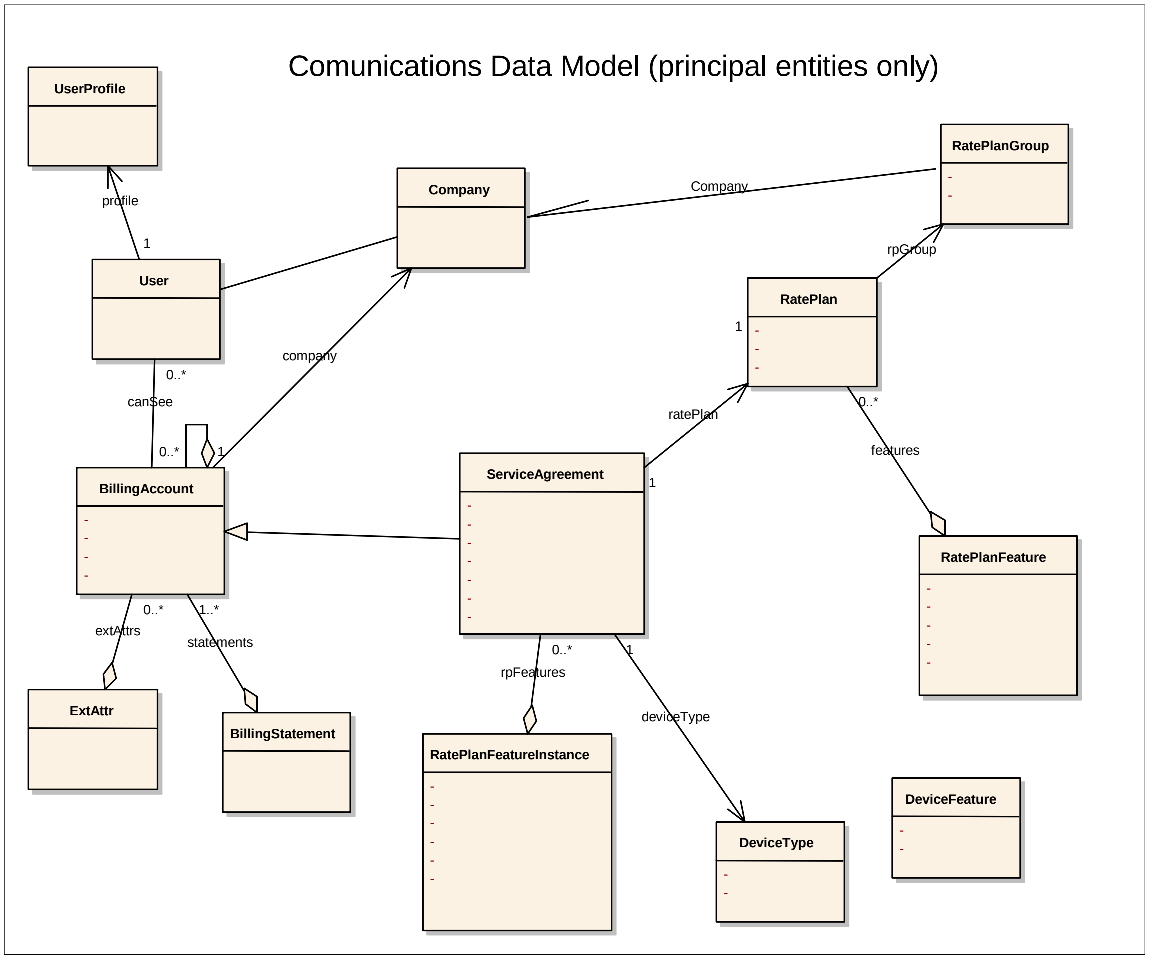
Self Service Manager contacts an external system to fulfill a request in real-time synchronous mode and receives an immediate response. If fulfilling the service request causes an entity in the system of record to change and this entity is cached within the Self Service Manager data model, Self Service Manager updates this entity locally in its cache during the service request lifecycle.

Data Model

The Self Service Manager data model is an extension to the Billing Manager data model adding elements such as Plans, Features, Service Agreement, and Devices. These additions to the data model are required to enable Self Service Manager service transactions.

The Self Service Manager Data Model consists of:

- Rate Plans — a named grouping of required and optional features. Includes fields such as plan name, description, cost, and features etc. Rate plans define a core and optional features which can be assigned to a given subscriber, at which point the rate plan becomes a “Service Agreement”.
- Rate Plan Feature — a named feature of a rate plan, such as “caller id”, “Weekend Minutes Free” etc. Includes fields such as name, description, price, units, etc.
- Rate Plan Group — a named group of rate plans. Rate plans may belong to one and only one group. Includes fields such as name and description.
- Device Type — a named device such as “Nokia5800”, “`Kyocera SE47”, etc.
- Service Agreement — Adds fields such as name, description, subscriber name, Device Serial Number (DSN) for a given device, device type, and a set of rate plan feature instances representing the characteristics of this Service Agreement. Service Agreements are effectively “instances” of rate plans and represent a given set of features at the point in time the rate plan was assigned. Service Agreements contain a set of Rate Plan Feature Instances representing the current feature set. Note that service agreement instances are subclasses of Billing Manager Billing account.
- Rate Plan feature instance — A feature particular to a given Service agreement. Includes fields such as name, description, price, included (always part of the given agreement, ie NOT optional), units, activated, etc.



The Self Service Manager Developers Guide details the specific developer use cases that must be addressed when deploying an implementation of Self Service Manager. For more information, and a complete list of the Self Service Manager developers use cases, see the *Self Service Manager Developers Guide*.

Index