Comverse ONE™

3.5 RT TR2.0

Bulk Provisioning Guide

COMONE-3.5-RT-TR1.0-BPRUG-2011-10-04

Notice

The material furnished in this document is believed to be accurate and reliable. However, no responsibility is assumed by Comverse, Inc. for the use of this material. Comverse, Inc. reserves the right to make changes to the material at any time and without notice. This document is intended for information and operational purposes only. No part of this document shall constitute any contractual commitment by Comverse, Inc.

Corporate Headquarters
200 Quannapowitt Parkway
Wakefield, MA 01880 USA
Tel: (781) 246-9000
Fax: (781) 224-8143
www.comverse.com

Revision History

The following table lists the document changes since the initial publication:

Date	Chapter	Description
10/15/2010	All	Published the 3.5.50 Version.
04/29/2011		Published the 3.5 RT TR1.0 Version. See "New Features for This Release," on page 3.
10/04/2011		Published the 3.5 RT TR2.0 Version. See "New Features for This Release," on page 3.

Contents

Figures

Tables

Notational Conventions

NOTE Useful information appears in this format.

SEE Provides direction to important information

IMPORTANT Important information appears in this format.

CAUTION Indicates possible risk of damage to data, software, or hardware.

WARNING! Indicates serious risk of damage to data, software, or hardware.

Table 1 Notational Conventions

Notation	Explanation of Convention
References to printed documents	*Helvetica italic* **Example:** See *Database Reference Volume 2*.
<KEYS>	UPPERCASE HELVETICA, in angle brackets **Example:** Press <CTRL><Q><SHIFT><P> to create an em dash.
`User-entered text`	**`Courier bold`** **Example:** Enter **`Total Charges`** in the field.
`Placeholders for user-determined text`	*`Courier italic`*, in angle brackets **Example:** Enter your <*`password`*>.
`Code samples, TABLE_NAMES, field_names, file and directory names, file contents, user names, passwords, UNIX ENVIRONMENT_VARIABLES`	`Courier`
Placeholders for system-generated text	*Helvetica italic* **Example:** Messages appear in this form: *timestamp messageID* >> *text.*
Buttons, Icon Names, and Menu items	**Helvetica bold** **Example:** Choose **Reports** from the main menu.

Special Markers

The Comverse ONE Billing and Active Customer Management solution has the three derivatives shown in Table 2, "Labels in Markers." For user convenience, any content that is specifically included in a derivative is highlighted with special markers so that it can readily be distinguished.

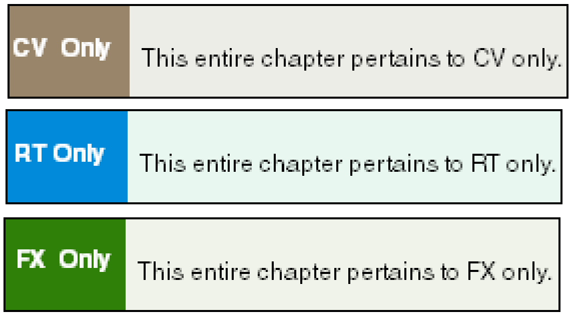
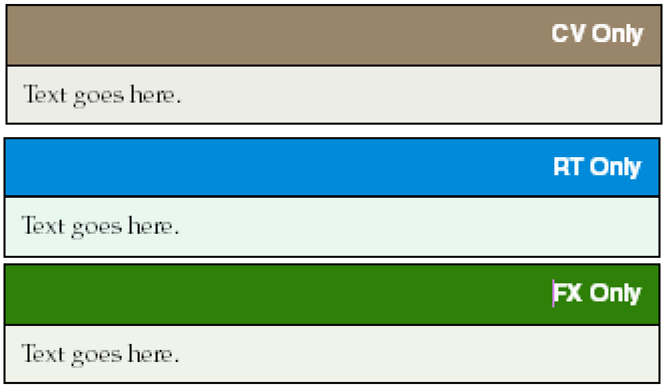
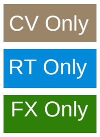
Table 2 Labels in Markers

Derivative	Label Shown in Markers
Comverse ONE CV Converged Billing derivative	**CV Only**
Comverse ONE RT Real-Time Charging derivative	**RT Only**
Comverse ONE FX Postpaid Billing derivative	**FX Only**

Each derivative has a set of three color-coded markers, as shown in Table 3, "Types of Markers." The markers are used individually or in combination to highlight derivative-specific content by:

- Entire chapters
- Selected portions of chapters
- Tables, either entire or partial

Table 3 Types of Markers

Marker	Example	Description
Alert	 **CV Only** This entire chapter pertains to CV only. **RT Only** This entire chapter pertains to RT only. **FX Only** This entire chapter pertains to FX only.	■ Placed at the beginning of an entire chapter that pertains only to a specific derivative. ■ Placed just before a table that partially or entirely pertains only to a specific derivative.
Block	 **CV Only** Text goes here. **RT Only** Text goes here. **FX Only** Text goes here.	A shaded box that encloses sections of documentation that pertain only to a specific derivative.
Flag	 CV Only RT Only FX Only	■ Designates a shaded table row whose contents pertain only to a specific derivative. ■ In a bulleted list, designates an item that pertains only to a specific derivative.

Comverse ONE Documentation List

NOTE this is a comprehensive list. As such, it may include documentation for products which you have not licensed.

The documents described below reference the Comverse ONE solution products. All documentation available with the Comverse ONE solution is described in the following pages, organized by the following categories:

- Infrastructure Domain
- Rating, Charging, and Promotions Domain
- Billing and Financials Domain (Converged only)
- Customer and Order Management Domain (Converged only)
- Mediation and Roaming Solutions Domain
- Self-Service Solutions Domain

NOTE Read the relevant Solution Description first to get an overview of *your* Comverse ONE solution. It gives an overview of the functionality in each product domain and also includes cross-references to the user documentation that provides more detailed information about the functionality.
There are three such documents and they are listed under the Infrastructure Domain heading below.

- Converged *Billing & Active Customer Management Solution Description*
- Postpaid *Billing & Active Customer Management Solution Description*
- Real-Time *Billing & Active Customer Management Solution Description*

Infrastructure Domain

Download every document in the Infrastructure domain if you purchase the Comverse ONE solution. Documentation for this domain includes the following (in alphabetical order):

- *Alarms Reference*
  Contains tables of alarm IDs, descriptions, likely causes, and recommended resolutions for systems and components.
- *Back Office Administration GUI Guide*
  Provides information about the BackOffice subsystems for Inventory Administration, Address Management and Bulk Operations.
- Converged *Billing & Active Customer Management Solution Description*
  General overview of the Comverse ONE Converged Offer and the functionality available in each domain.

- *Database Failure Recovery - Monetary Loss Scenarios*
  This document outlines the potential monetary loss scenarios in case of a Comverse ONE database failure that may have caused data loss. The document provides awareness of such loss potential. While guidance may be provided for possible recovery of such losses, this document is not intended to be a monetary loss recovery guide, because such recovery may not be possible due to permanent data loss in a database failure, file system failure or total disaster scenario.
- *Database Failure Recovery - Database Synchronization Guide*
  This document outlines and implements the application-level synchronization procedures in case of a Comverse ONE database failure that has caused data integrity and/or synchronization issues among the different database instances.
- *Database Reference*
  Describes all database tables and fields in detail.
- *Disaster Recovery Operations Guide* (Optional Module)
  The Disaster Recovery Operations Guide serves as both a technical overview of the optional Disaster Recovery solution and as a guide which details the operational procedures for failover, switchover and switchback provided by the solution.
- *Glossary*
  Provides a list of terms used specifically for the Comverse ONE solution
- *Investigation Units and Financial GUIs Guide*
  Describes the GUI-based tools used for investigating and troubleshooting various financials related processes: payments, bill invoices, refunds, and incomplete data work entries
- *Operation Reference*
  Describes the processes in the Comverse ONE solution.
- *Platform Operations Guide*
  Describes the back-end operations and maintenance functionality of the core Comverse ONE solution components. Includes AIX/HACMP platform and cluster operations, Linux/Veritas platform and cluster operations, backup/recovery, shared storage and fiber switch operations, and tape backup operations.
- Postpaid *Billing & Active Customer Management Description*
  General overview of the Comverse ONE Postpaid Offer and the functionality available in each domain.
- *Product Catalog Overview*
  Provides a high-level description of the Comverse ONE solution Product Catalog, which is the primary mechanism for creating, configuring, managing, and propagating Product Catalog versions.
- *Product Catalog User Guide*
  Instructions on using the Product Catalog application to define and manage all aspects of Service provisioning.
- Real-Time *Billing & Active Customer Management Description*
  General overview of the Comverse ONE Real-Time Offer and the functionality available in each domain.
- *Schedulable Entity Reference Manual*
  Documents all the jobs, monitors, and workflows, for each component.
- *Security Platform Operations Guide*
  Technical overview of the security platform and information on how to provision and administer the platform.
- *Security Server API Guide*
  Provides an overview of the interfaces exposed by the Java-based Security SDK API, which client applications can leverage to access various security services, such as authentication, authorization, auditing, key management, and credentials management. Also provides

information on the Security Web Services API, which provides interfaces to a subset of Security Server commands (Identity Management commands).

- *Signaling Gateway Unit Guide*
  Describes the hardware, installation, configuration, and maintenance of the Signaling Gateway Unit (SGU) used to connect Comverse real-time systems to the SS7 signaling network using either traditional SS7 protocols or Sigtran (SS7 over IP).
- *System Measurements Guide*
  The Comverse ONE Solution automatically collects statistical data from the Service Logic Unit (SLU) and the Service Gateway Unit (SGU). This includes service statistics on the SLF layer and platform data on the IPF layer.
  This guide describes the format and location of this measurement information and provides a description of the meaning of the data.The measurement data can be used to create reports. It can also be imported into other applications (such as Excel) to be viewed.
- *System Parameters Guide*
  Describes the various system parameters used in Comverse ONE.
- *System Validation Check Reference*
  Details all the system validation checks performed by the Comverse ONE Unified Platform on its components.
- *Unified API Guide*
  General overview of the Unified API, a brief description of its architecture, and information about:
  - Framework classes and the functionality they provide
  - Two standard interfaces provided with the Unified API (client SDK and web services)
  - A subset of Unified API business methods most commonly used
- *Unified Platform Guide*
  Technical overview of the Unified Platform and information on the procedures to manage core systems operations in the Comverse ONE solution.

Rating, Charging, and Promotions Domain

Documentation for this domain includes the following (in alphabetical order):

- *Bulk Provisioning Guide*
  - The *CC Batch* utility enables bulk creation of recharge vouchers and subscribers.
  - The *Bulk Provisioning* Utility enables bulk creation of anonymous accounts to support the pre-activation of pre-paid SIM cards.
- *Charging Interfaces Guide*
  Describes the four interfaces that enable external services to support real-time authorization, rating, and charging for transactional usage: (1) the Event Charging Interface, a simple TCP/IP-based interface, (2) Open Services Access (OSA), (3) a Diameter-based interface version enhanced to take advantage of features of the Comverse ONE solution, and (4) a Diameter-based interface packet-switched version.
- *Customer Care Client Provisioning Guide* — Real-Time
  Detailed task-oriented instructions for using Customer Care Client.
- *Diameter Gateway Unit Guide*
  Describes the hardware, installation, configuration and maintenance of the Diameter Gateway Unit (DGU) used to connect Comverse real-time systems to external services, using the diameter protocol over IP.
- *IVR Call Flows Reference*
  These call flows detail the logic flow of specific scenarios. Multiple access numbers can map

to the same call flow. Different resellers have the option to publish different numbers but share the same logic.

- *Network Interfaces and Notifications Guide*
  Describes the operation, features, and provisioning of notifications, CAMEL-enabled services, and USSD-enabled services.
- *Network Self-Care Guide*
  Describes the configuration, structure, and features.
- *Operational Reports and Data Warehouse Utility Guide*
  Describes the real-time Operational Reports Interface (ORI) and the Data Warehouse Extract Utility.
- *Rating Technical Reference*
  Describes the Unified Rating Engine, which is the subsystem responsible for gathering incoming CDRs and processing them for billing.
- *Recurring–Non-Recurring Charges Server Guide*
  Describes all processes commonly available through the Recurring —Non-Recurring Charges Server.
- *Voucher and Recharge Guide*
  Describes the process by which subscribers add funds to accounts using recharge vouchers through IVR, interaction with Customer Service, and other methods. Provides details of the Recharge Control Table, which allows resellers to provision the effects of recharges so that bonuses, discounts, and other changes to offers can result from a successful recharge. Also describes the Card Generator software used to create batches of recharge vouchers.

Billing and Financials Domain (*Converged* only)

Documentation for this domain includes the following (in alphabetical order):

- *Advanced Invoice Numbering Guide*
  Describes how to configure and use Advanced Invoice Numbering.
- *Billing Reports and File Layouts User Guide*
  Describes control reports and other file formats.
- *Billing Technical Reference*
  High-level descriptions of billing architecture, administration, bill generation and formatting, and system parameters
- *Collections Guide*
  Contains information on configuring Collections database tables, running the Collections module, and using the Collections interface.
- *Invoice Designer Strings and Filters Reference*
  Describes the static strings, dynamic strings, and filters in the Invoice Designer.
- *Invoice Designer Technical Reference*
  Describes how to configure and run Invoice Designer.
- *Invoice Designer User Guide*
  Describes the Invoice Designer and how to perform the tasks needed to create an invoice template.
- *Journals Guide*
  Describes the theory, configuration, and running of Journals processes.
- *Miscellaneous Configurable Entities*
  Instructions for configuring late fees, adjustments, and several other database entities used in postpaid and converged billing.

- *Process Workflow Orchestration Guide*
  Describes the command-line entries and the default queries for running billing-related processes via the Unified Platform.
- *Taxation Guide*
  Describes the configuration, operation, structure, and features of Taxation.

Customer and Order Management Domain (*Converged* only)

Documentation for this domain includes the following (in alphabetical order):

- *Application Integrator Operator Guide*

  Describes the commands that operate the Application Integrator at creation and runtime.
- *Application Integrator System Administrator Guide*

  Outlines installation, sizing, operation, and administration of the Application Integrator and logging. Describes configuration of the user environment and commands for creation and operation of the Application Integrator.
- *Application Integrator User Guide*

  Describes creating integration specifications, creating instances of the Application Integrator, and commands for operation of the Application Integrator. Provides a complete user guide for the iMaker compiler.
- *Application Integrator File Adapter User Guide*

  Describes the configuration process and rules for the file adapter.
- *Customer Center User Guide*

  Detailed task-oriented instructions for using Customer Center.
- *Inventory Guide*

  Describes the configuration, operation, structure, and features of Inventory.
- *Inventory Replenishment Guide*

  Describes the operation, structure, and features of Inventory Replenishment.
- *Orders Services Guide*

  Describes the structure and features of Orders Services.
- *Request Handling and Tracking and Service Fulfillment User Guide*

  Describes the configuration, operation, structure and features of Request Handling and Tracking and Service Fulfillment.
- *Workflow Developers Guide*

  Helps new users understand the rules-based business process management system so users can create solutions and integrate Workpoint within those solutions.
- *Workflow User Guide*

  Describes the configuration, operation, structure, and features of Workpoint.

Customer Relationship Management

- *Campaign Management Data Mapping Reference*
  Describes how the data in DataMart is mapped to information in the Comverse ONE Customer database, the Comverse ONE ODS, and the Comverse ONE Sales and Service database.
- *Campaign Management DataMart Implementation Guide*
  Contains in-depth technical information on how to configure and populate the data mart used by all Campaign Management applications.
- *Campaign Management Outbound Marketing Manager Reference*
  Describes how to use the Campaign Management Outbound Marketing Manager features and guides you through the program's basic functionality.
- *Campaign Management Quick Implementation Guide*
  Helps novice users get started with implementing Campaign Management. It contains an overview of the product architecture, information on data mart design and creation, an explanation of how extraction works, and procedures for creating web pages, reports, lists, and campaigns.
- *Campaign Management Topic Implementation Guide*
  Provides information for implementers and professional services personnel who are creating applications that will run on an Campaign Management EpiCenter. Summarizes the Campaign Management functionality, architecture, and administration and contains in-depth technical information for configuring the Campaign Management topics required for Campaign Management and analysis.
- *Campaign Management User Guide*
  Provides you with basic information about the Campaign Management applications.
- *Case Management User and Administration Guide*
  Contains detailed information about GUI screens and form fields that appear in the Case Management application. Also provides information on performing general procedures in the GUI and administrative tasks.
- *Customer Center User Guide*
  Detailed task-oriented instructions for using Customer Center.
- *Sales and Service Admin Console User Guide*
  Provides supervisors, managers, and executives with the information to use the Case Management and Sales Force Automation Admin Console application.
- *Sales and Service Application Reference*
  Contains technical reference information relevant to implementers involved in implementing and customizing CRM applications at customer sites. This book provides the reference context for the procedural information available in the Implementation Guide.
- *Sales and Service Architecture Reference*
  Provides technical information relevant to individuals involved in implementing the Open Architecture and the applications built on the architecture
- *Sales and Service Data Dictionary Reference*
  Includes a listing and description of the tables and columns used to store CRM operational business data. It also includes a description of the naming conventions for the tables. The target audience includes database administrators, application developers, and implementers.
- *Sales and Service Dialog Designer User Guide*
  Describes the Sales & Service Dialog Designer, a web-based graphical application for defining and editing dialogs. Includes procedures for using it.
- *Sales and Service IBR Designer User Guide*
  Describes how to use the IBR Designer to create Intelligent Business Rules, which can be used to implement rule-based behavior within your CRM applications.

- *Sales and Service Implementation Guide*
  Provides procedural information relevant to individuals involved in implementing and customizing the core and the Sales and Service applications built on the core.
- *Sales and Service Integration Guide*
  Provides overview and configuration information for the set of tools used to exchange data with a variety of back-end data sources, including generic SQL sources, Java and EJB-based sources, Web services, and other database types.
- *Sales and Service Workflow Designer*
  Explains how to use Workflow Designer, a web-based graphical tool for defining and editing workflows
- *Sales Force Automation User and Administration Guide*
  Contains detailed information about GUI screens and form fields that appear in the Sales Force Automation application. Also provides information on performing general procedures in the GUI and administrative tasks.

Mediation and Roaming Solutions Domain

Documentation for this domain is subdivided into Mediation/Roaming and Revenue Settlements.

Mediation and Roaming

Mediation and Roaming documentation includes the following (in alphabetical order):

- *API Guide*
  Provides the concepts and functions for the Collection Application Programming Interface (CAPI), Mediation API, and Socket-Based Transmission API.
- *Data Manager GUI Reference*
  Contains detailed information about GUI screens and form fields that appear in the Data Manager interface
- *GRID Mapping Language Developer Guide*
  Describes the mediation feature components, semantics, and general syntax of the GRID Mapping Language (GML).
- *Installation Guide for HP*
  Describes how to install and configure the application, components, and some third-party applications associated with the HP platform.
- *Installation Guide for HP Itanium*
  Describes how to install and configure the application, components, and some third-party applications associated with the HP Itanium platform.
- *Installation Guide for HP PA-RISC*
  Describes how to install and configure the application, components, and some third-party applications associated with the HP PA-RISC platform.
- *Installation Guide for IBM*
  Describes how to install and configure the application, components, and some third-party applications associated with the IBM platform.
- *Installation Guide for SUN*
  Describes how to install and configure the application, components, and some third-party applications associated with the SUN platform.
- *Mediation and Roaming User Guide*
  Provides information on how to use the GUI interface, including information on using the Data System Manager application pages.
- *Roaming Database Reference*
  Provides reference information on the Roaming database.

- *Roaming Setup Guide*
  Describes how to configure the Roaming Setup application pages. It also provides information on working with TAP, RAP, and CIBER statistics.
- *Scripts Guide*
  Provides information on script files, which contain additional instructions on functions for data collection and transmission.
- *System Manager GUI Reference*
  Contains detailed information about GUI screens and form fields that appear in the System Manager interface
- *Variable-Length GRID Guide*
  Provides information on how to configure the control files for variable-length GRID.

Revenue Settlements

Revenue Settlements documentation includes the following (in alphabetical order):

- *Comverse Revenue Settlements Billing System Adapter Guide*
  Describes the configuration, operation, and installation for the Billing System adapter.
- *Comverse Revenue Settlements Data Model Guide*
  Overview of data model entities (such as partners, accounts, revenue sharing, and rate schedules) and how to configure them in the database.
- *Comverse Revenue Settlements Database Reference*
  Detailed descriptions of fields and tables in the database.
- *Comverse Revenue Settlements Technical Reference*
  Instructions for installing and operating Revenue Settlements. Also contains processing descriptions.
- *Comverse Revenue Settlements User Guide*
  Instructions for using the Revenue Settlements GUI.

Self-Service Solutions Domain

The Comverse ONE Self-Service Solutions domain consists of the core products plus the optional separately licensed premium products. The core products consist of the following:

- Self-Service Solutions Platform
- Self-Service Solutions Applications

Self-Service Solutions Platform Documentation

The Self-Service Solutions Platform has a comprehensive set of documentation covering the installation, configuration, and use of our products. The documentation set is divided into the following categories:

- **Manuals**: These manuals cover installing and using the platform.
- **Reference**: These reference documents contain information about APIs, databases, configuration files, and so on. These documents are delivered in HTML.

Self-Service Solutions Platform Manuals

Self-Service Solutions Platform manuals include the following (in alphabetical order):

- *Self-Service Platform Administration Guide*
  Provides operations and maintenance instructions for Web applications using the Self-Service Solutions Platform.

- *Self-Service Platform Catalog Loader Reference*
  Provides information about the Catalog Loader, including a functional description as well as installation, configuration, and use instructions.
- *Self-Service Platform Communications Billing and Usage Reference*
  Provides detailed descriptions of the data models and structure of the Self-Service Solutions Platform Communications Billing and Usage (CBU) database.
- *Self-Service Platform Connectors Development Guide*
  Provides instructions for developing and customizing Connectors of the Self-Service Solutions Platform.
- *Self-Service Platform Core Module Development Guide*
  Provides instructions for configuring and developing features of the core module of the Self-Service Solutions Platform.
- *Self-Service Platform Customer Interaction Datastore Reference*
  Provides detailed descriptions of the data models and the structure of the Self-Service Solutions Platform Customer Interaction Datastore (CID).
- *Self-Service Platform Database Modules Development Guide*
  Provides instructions for configuring, customizing, and developing features of the database module of the Self-Service Solutions Platform.
- *Self-Service Platform Installation Guide*
  Provides installation and configuration instructions for the Self-Service Solutions Platform.
- *Self-Service Platform Services Guide*
  Provides instructions for configuring, customizing, and developing features that use the services provided by the Self-Service Solutions Platform.
- *Self-Service Platform Processors Development Guide*
  Provides instructions for developing and customizing Processors of the Self-Service Solutions Platform.
- *Self-Service Platform Reports Development Guide*
  Provides instructions for developing and customizing Reports of the Self-Service Solutions Platform.
- *Self-Service Platform Web Applications Development Guide*
  Provides instructions for configuring, developing, and deploying Web applications that use the Self-Service Solutions Platform.
- *Self-Service Solutions Overview Guide*
  Provides a high-level architectural and functional description of the Comverse ONE Self-Service Solutions. It also includes a detailed description of the concepts and development process to create and deploy Self-Service Solutions.

Self-Service Solutions Platform Reference

Self-Service Solutions Platform reference documentation includes the following (in alphabetical order):

- *Base Logic Manager Reference*
  Describes usage syntax and configuration files for the Base Logic Manager (BLM) APIs. These APIs are the core services of the Self-Service Solutions Platform.
- *CID2CBU Object Mapping Reference*
  Describes the default mapping of Customer Interaction Datastore (CID) and Communications Billing and Usage (CBU) objects.
- *Communications Billing and Usage Reference*
  Provides detailed descriptions of fields and tables in the Communications Billing and Usage (CBU) database.

- *Customer Interaction Datastore Reference*
  Provides detailed descriptions of fields and tables in the Customer Interaction Datastore (CID).
- *Integration Services Framework API Reference*
  Describes usage syntax of the set of APIs to program connectors and other components of the Intelligent Synchronization Framework (ISF).
- *Integration Services Framework Message Cache Reference*
  Provides detailed descriptions of fields and tables in the Intelligent Synchronization Framework (ISF) Message Cache.
- *Integration Services Framework Script API Reference*
  Describes usage syntax of the Intelligent Synchronization Framework (ISF) script APIs to program the ISF connectors.
- *JavaServer Page Framework for Internet Application API Reference*
  Describes usage syntax for the JavaServer Page Framework for Internet Application (JFN) APIs. These APIs are used to build JSPs using the JFN. This framework provides basic application functions and services as the foundation of user interfaces.
- *Logger Message Reference*
  Provides detailed descriptions of the Self-Service Solutions Platform log messages.
- *QRA API Reference*
  Describes usage syntax for the Query, Reporting, and Analysis (QRA) Engine APIs. These APIs are used to build reports.
- *UTIL API Reference*
  Describes usage syntax for the UTIL package used by different components of the Self-Service Solutions Platform. This package contains a set of utilities including the logger.

Self-Service Solutions Applications Documentation

Each Self-Service Solutions Application comes with a comprehensive set of documentation covering the installation, configuration, and use of the product. The application documentation expands and complements the Self-Service Solutions Platform documentation.

The documentation set is divided into the following categories:

- **Manuals**: These manuals cover installing and using the application.
- **Reference**: These reference documents contain information about APIs, databases, configuration files, and so on. These documents are delivered in HTML.

Self-Service Solutions Application Manuals

A full set of these manuals is available for each Self-Service Solutions Application (Business, Channel, Consumer, andCSR Portal). The documentation set includes the following (in alphabetical order):

- *Business Objects Model Reference*
  Provides a detailed description of the models and entities that make up the Self-Service Solutions Application.
- *Configuration and Development Guide*
  Provides instructions for configuring and developing Self-Service Solutions Application features.
- *Introduction*
  Provides a high-level architectural and functional description of the Self-Service Solutions Application. It covers common features, order management, account management, and bill presentment.

- *Feature Reference*
  Describes the logic and provides use cases for the functional domains of the application.
- *Out-of-the-Box Reference Guide*
  Describes the Self-Service Solutions Application Out-of-the-Box release.
- *Self-Service Installation Guide for Comverse ONE*
  Provides detailed installation, configuration, and deployment instructions for the Self-Service Solutions Application alongside other elements of the Comverse ONE solution.
- *Self-Service Installation and Deployment Guide*
  Provides detailed installation, configuration, and deployment instructions for the Self-Service Solutions Application.
- *User Guide*
  Provides instructions for navigating and using the Self-Service web application. For Business Self-Service and CSR Portal only.

Self-Service Solutions Application References

A full set of these references is available for each Self-Service Solutions Application.The reference documentation set includes the following (in alphabetical order):

- *API Reference*
  Describes usage syntax for the Self-Service Solutions Application APIs. These APIs are used to program the user interface and manage data.
- *Invoice Schema Reference*
  Describes the invoice schema reference of the Self-Service Solutions Application.
- *Presentation Layer Page Flow Reference*
  Describes the page flows of the Self-Service Solutions Application.
- *Specification Entity Relationship Diagrams*
  Provides diagrams describing the actors, use cases, user activity, and storyboard in IBM Rational Rose format.

Self-Service Solutions - Separately Licensed Products

Documentation available with optional, separately-licensed premium products in the Comverse Self-Service Solutions is listed below.

Online Catalog Manager

Online Catalog Manager (OCM) documentation includes the following (in alphabetical order):

- *Introduction to the Online Catalog Manager*
  Provides a high-level architectural and functional description of the Online Catalog Manager.
- *Online Catalog Manager Getting Started Guide*
  Describes the best way to build product catalogs in the Online Catalog Manager. This manual is a template for creating end-user documentation.
- *Online Catalog Manager Installation and Configuration Guide*
  Provides installation and configuration instructions for the Online Catalog Manager.
- *Online Catalog Manager User Documentation Template*
  Describes the use of the Online Catalog Manager. This manual is a template for creating end-user documentation. This manual covers many common concepts and procedures of the OCM.
- *Online Catalog Manager User Guide*
  Provides a detailed description of the concepts and use of the Online Catalog Manager. The topics include:

- Managing Media Files
- Managing Offers
- Managing Prices
- Managing Products
- Managing Properties
- Managing Reference Data
- Publishing

Chapter 1

Introduction

Welcome

Welcome to the *Bulk Provisioning Guide*, which documents two batch utilities: CCBatch and the Bulk Provisioning Utility.

Who Should Use This Manual

This document is intended for use by back office provisioning users (not Back Office GUI users). Customer service representatives are not expected to use this manual.

New Features for This Release

This section provides list of changes to the *Bulk Provisioning Guide*. Changes include features that impact the *Bulk Provisioning Guide* as well as corrections and improvements to the document. Changes are listed in reverse chronological order (the most recent release appears first).

Comverse ONE 3.5 RT TR2.0

The following is a list of new features in the Comverse ONE 3.5 RT TR2.0 release that impact this document:

- Service Provider Support. This feature introduces the concept of market offer group. A market offer group is essentially a division within a reseller, giving an added dimension to the reseller, and can be used for purposes such as campaigns to promote subscriber activity. Market offer group has been added as an attribute to voucher definitions and appears in various CCBatch input files. For information, see “Normal Voucher Load Input Batch File,” on page 16, “Fast Voucher Load Input Batch File,” on page 18, and “Template File anonymousSubscriberCreate.ccbatch,” on page 24.

Comverse ONE 3.5 RT TR1.0

The following is a list of new features in the Comverse ONE 3.5 RT TR1.0 release that impact this document:

- Unified cards have been introduced to Comverse ONE. Unified cards require that anonymous accounts exist in the database. To meet this requirement, the following enhancements have been made:
  - The bulk load utility has been introduced to ensure that a minimum number of anonymous accounts always exists in the database for any combination of offer ID and reseller ID. For information, see Chapter 4, “Bulk Load Utility (RT),”
  - The bulk load utility passes its own input file (distinct from the traditional `BulkProvisioningInputFile.txt`) to the bulk provisioning tool. Because there now exist two input files, callers must include the input file name on the command line when invoking the bulk provisioning tool.

NOTE A unified card can be used either as (1) a recharge voucher (to replenish an existing subscriber account), or (2) as a calling card, allowing subscribers to make prepaid calls.

What is CCBATCH?

The CC Batch application enables bulk creation of recharge vouchers and subscribers.

NOTE CC Batch creates single subscriber (1SA) accounts only.

CC Batch processes files containing commands that create vouchers and subscribers. CC Batch interfaces with the Comverse ONE solution Unified API to provide secure login and other services.

CC Batch parses an input file from Card Generator, invokes the corresponding Unified API methods, and generates a log file that contains the API's responses to the executed commands as well as any error response. Optionally, the operator can choose to generate a separate log file that contains the error response.

What is the Bulk Provisioning Utility?

This utility allows you to create anonymous accounts to support the pre-activation of pre-paid SIM cards. It instantiates an order that does the following:

- Creates anonymous accounts and subscribers,
- Assigns bundles or primary offers,
- Associates MSISDN and SIM/IMSI with the subscribers.
- If requested, provisions the network to activate SIM cards via a workflow.

What is the Bulk Load Utility?

The bulk load utility creates anonymous accounts for use by Comverse ONE unified cards.

Organization of This Document

This document is organized as follows:

- Chapter 1, "Introduction," provides an introduction to this book.
- Chapter 2, "CC Batch," explains how to configure and run the CCBatch application.
- Chapter 3, "Bulk Creation of Anonymous Accounts," explains how to run the Bulk Provisioning Utility. The chapter also explains the utility's output files and provides troubleshooting guidelines.
- Chapter 4, "Bulk Load Utility (RT)," explains how to run the Bulk Load Utility.

Chapter 2

CC Batch

Overview

The CC Batch application enables bulk provisioning of recharge vouchers and subscribers.

It is installed on, and run from, a Windows-based computer.

NOTE CC Batch creates single subscriber (1SA) accounts only.

CC Batch processes files containing commands that create vouchers and subscribers. CC Batch interfaces with the Comverse ONE solution Unified API to provide secure login and other services.

CC Batch parses an input file from Card Generator, invokes the corresponding Unified API methods, and generates a log file that contains the API's responses to the executed commands as well as any error response. Optionally, the operator can choose to generate a separate log file that contains the error response.

SEE See the Comverse ONE solution *Voucher & Recharge Guide* for more information on the Card Generator and the files it creates for use by CCBatch.

NOTE CCBatch only supports regular and fast voucher bulk loading and anonymous subscriber bulk loading. CCBatch *cannot* be used for bulk creation of calling circles or bonus awards.

Roles

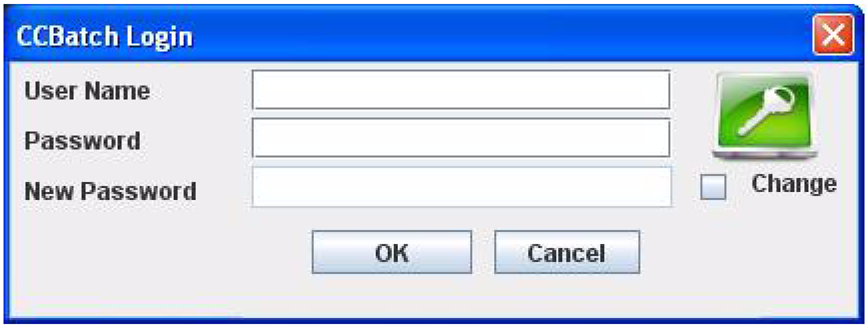
A user must have the CC_BATCH_ACCESS role in order to use the CC Batch application.

SEE For more information on provisioning user roles, refer to the Comverse ONE solution Security Platform Operations Guide.

CC Batch Log In

1. Run the CC Batch application by clicking the **launchapp.bat** icon or issuing the **launchapp.bat** command from the command line.
2. The CC Batch Login window, shown in Figure 1, appears above the main CC Batch window, which is disabled until login is complete.

Figure 1 CC Batch Login Window



3. Enter the User Name and Password.
4. If desired, enter a new password in the New Password field.
5. Click **OK**. The CC Batch main window appears.

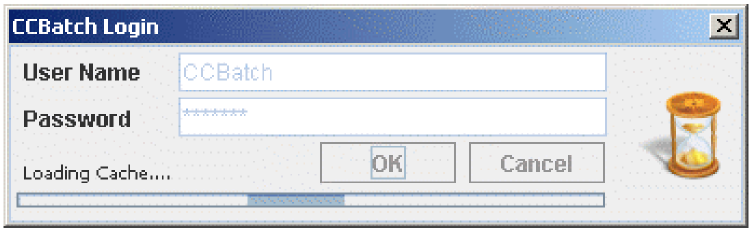
NOTE The CC Batch main window only becomes active if the user has the CC_BATCH_ACCESS privilege. If the user does not have this role, an error message appears and the CC Batch Login window prompts the user to reenter the User Name and Password.

Cache Load

The CC Batch application loads versioned and non-versioned information to the Unified API cache during its initial startup. When a new version becomes current, or when non-versioned data changes, CC Batch continues to use the old information for the life of the application session.

If the application session restarts, CC Batch restores the Unified API cache to support the new data. The cache is loaded after validating and processing the configuration file. Figure 2, “CC Batch Unified API Cache Load” shows the CC Batch login screen while the application is loading data to the Unified API cache.

Figure 2 CC Batch Unified API Cache Load



User Name: CCBatch

Password: *******

Loading Cache....

Timeouts

If the CCBatch application is idle for a certain period of time, the application prompts the user for re - authentication. This is referred to as a *soft* timeout. The length of the idle period before a soft timeout is configured in the Security Server.

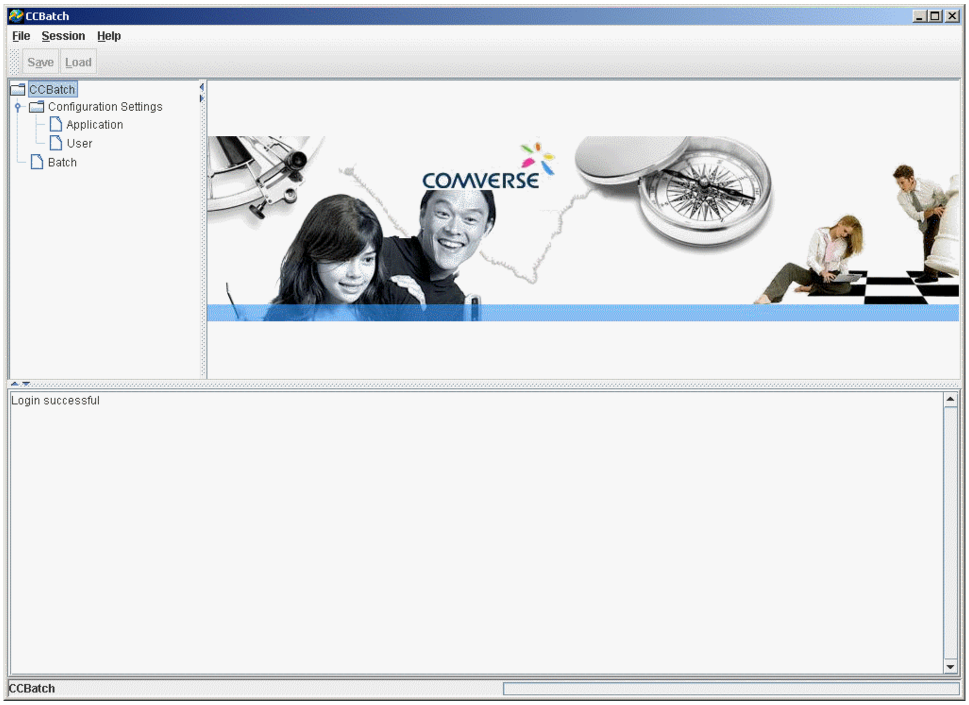
For each user login, Security Server also provides a specified interval of time known as a session. After the expiration of the session, the CCBatch application exits and the user must re-launch the application. This is referred to as a *hard* timeout.

CC Batch Interface Components

CC Batch Main Window

When the user has successfully logged in, the main window appears, as shown in Figure 3.

Figure 3 CC Batch Main Window



Login successful

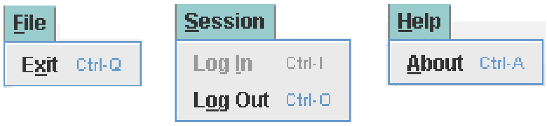
The CC Batch main window is divided into the following sections:

- **Top Panel**: Contains the Menu Bar
- **Tree View**: At the left of the window area.
- **Right panel**: Gray area at right, displays configuration settings.
- **Console Window**: White area at bottom, displays activity.
- **Status Bar**: At bottom edge of window.

Menus

CC Batch provides the menus shown in Figure 4.

Figure 4 CC Batch Menus



File Menu

- **Exit**: Click to exit the CC Batch application.

Session Menu

- **Log In**: Click to log in to the CC Batch application. The User Name of the logged in user displays on the status bar of the CC Batch window.
- **Log Out**: Click to log out of the CC Batch application.

Help Menu

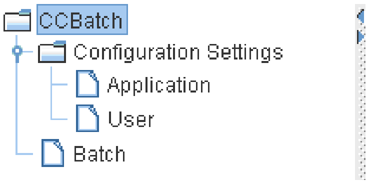
- **About**: Provides information related to the CC Batch application.

Tree View

The CC Batch Tree View, shown in Figure 5, has two nodes:

- **Configuration Settings**: Click to display configuration information in the right panel of the CC Batch Main Window. The Configuration Settings node has two child nodes:
  - **Application**: Click to display application settings.
  - **User**: Click to display user settings.
- **Batch**: Click to display the load batch screen in the right panel of the CC Batch Main Window

Figure 5 CC Batch Tree View

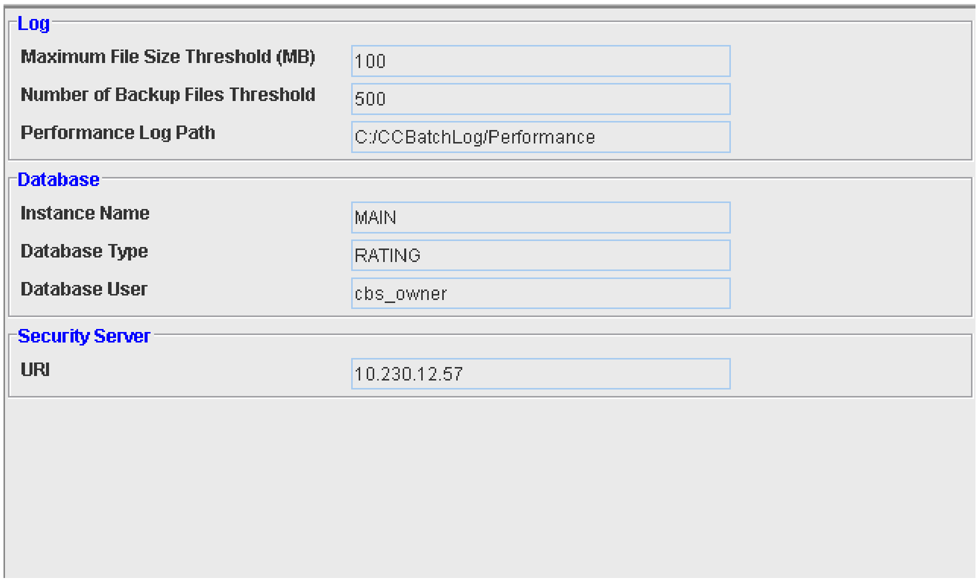


Application Settings Window

The Application Settings window displays the settings configured in the CC Batch Application configuration file (CCBatchApplication.config.xml). Figure 6, "CC Batch Application Settings Window" shows the read-only display visible in this window.

NOTE The Application Settings window does not display the settings configured in the user-specific configuration file (<UserName>_CCBatch.config.xml)

Figure 6 CC Batch Application Settings Window



Log

Maximum File Size Threshold (MB): 100

Number of Backup Files Threshold: 500

Performance Log Path: C:/CCBatchLog/Performance

Database

Instance Name: MAIN

Database Type: RATING

Database User: cbs_owner

Security Server

URI: 10.230.12.57

Log

- **Maximum File Size Threshold (MB)**: Maximum file size of log files in megabytes. This setting is specified in the User Settings window.
- **Number of Backup Files Threshold**: Maximum number of backup files that can be maintained. This setting is specified in the User Settings window.
- **Performance Log Path**: Path to the performance log.

Database

- **Instance Name**: Name of the database to which voucher batches are loaded as configured in tnsnames file.
- **Database Type**: Database Identifier.
- **Database User**: Database user name.

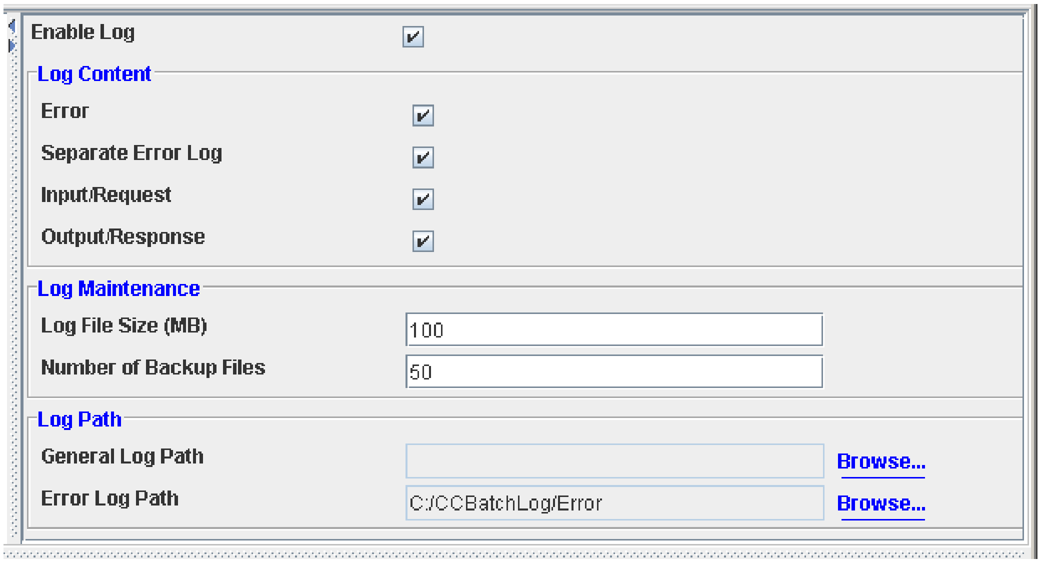
Security Server

- **URI**: IP address or name of the Comverse ONE solution Security Platform.

User Settings Window

Figure 7 shows the configurable settings accessible in this window.

Figure 7 CC Batch User Settings Window



Enable Log ☑

Log Content

Error ☑

Separate Error Log ☑

Input/Request ☑

Output/Response ☑

Log Maintenance

Log File Size (MB): 100

Number of Backup Files: 50

Log Path

General Log Path:

Error Log Path: C:/CCBatchLog/Error

- **Enable Log**: Select to generate log files during Batch load process.

Log Content

- **Error**: Log all errors and exceptions.
- **Separate Error Log**: Generate error logs in separate error log file and not in general log file. If the Errors checkbox under Log Type is not checked, this checkbox is disabled.
- **Input/Request**: Log all input requests in General log file.
- **Output/Response**: Log all output responses in General log file.

NOTE The Separate Error Log checkbox is enabled *only* if the **Error** checkbox is selected.

Log Maintenance

- **Log File Size (MB)**: Maximum size of the log and error log files. Cannot be greater than the Maximum File Size Threshold specified in the Application Settings window.
- **Number of Backup Files**: Maximum number of backup log files. Cannot be greater than the Number of Backup Files Threshold specified in the Application Settings window.

NOTE If the user has not specified values for **Log File Size** and **Number of Backup Files**, the default values specified in the CC Batch-Application configuration file (CCBatchApplication.config.xml) or user-specific configuration file (<UserName>_CCBatch.config.xml) are used.

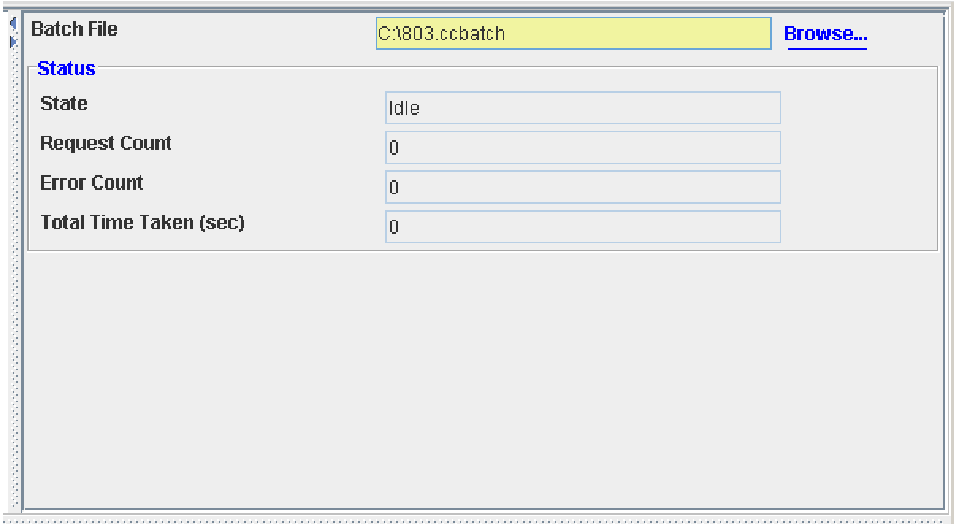
Log Path

- **General Log Path**: Specifies the path to the general log file.
- **Error Log Path**: Specifies the path to the error log file.

Batch Window

Figure 8 shows the window that appears when the user clicks the Batch node in the CC Batch Tree View.

Figure 8 CC Batch Batch Window



Batch File: C:\803.ccbatch

Status

State: Idle

Request Count: 0

Error Count: 0

Total Time Taken (sec): 0

- **Batch File**: Path to the batch file to load.

NOTE Ensure that the batch file entered has the .ccbatch extension.

- **State:** Displays the status of the batch operation**:**
  - **Idle**: Before starting and after completing the batch operation.
  - **Validating Input File**: While CC Batch validates the input file.
  - **Batch Import in Progress**:

    - Parsing the input file: While parsing the XML nodes to corresponding objects.

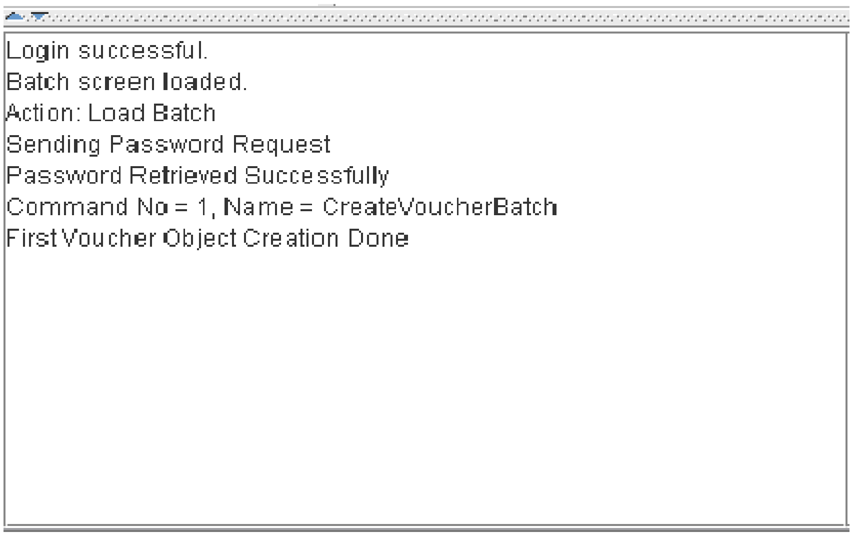
    - Executing batch command: While executing command by invoking Unified API method.
  - **Batch Cancelled**: When user cancels the batch load process.

- **Request Count**: Displays the number of commands in the Batch file to be executed.
- **Error Count**: Displays the number of commands that failed on execution.
- **Time Taken**: Displays the number of seconds that elapsed while loading the batch; displayed on completion of the batch load.

Console Window

The console window displays the results of entered commands. Figure 9 shows a typical console window output.

Figure 9 CC Batch Console Window



```
Login successful.
Batch screen loaded.
Action: Load Batch
Sending Password Request
Password Retrieved Successfully
Command No = 1, Name = CreateVoucherBatch
First Voucher Object Creation Done
```

Shortcut Keys

Table 4 shows the keyboard shortcuts available in the CC Batch application.

Table 4 CC Batch Shortcut Keys

Hot Keys	Functions
Alt + F	Opens File menu
Alt+F+X, Ctrl + Q	Exits application
Alt + S	Opens Session menu
Alt+S+I, Ctrl + I	Opens Login dialog box
Alt+S+O, Ctrl + O	Logs out currently logged in user
Alt + H	Opens Help menu
Alt+H+A, Ctrl + A	Opens About dialog box
Alt + L	Loads Batch specified file to the database
Alt + C	Cancels loading of Batch file
Alt + A	Saves all settings currently provided in GUI to Configuration file.

CC Batch Log Out and Exit

1. Select **Log Out** from the **Session** menu.
2. Select **Exit** from the **File** menu.

CC Batch Files

User-Specific Configuration File

Each time a user logs in, CC Batch checks for a configuration file specific to that user. If a user-specific configuration file is found, CC Batch uses the configuration values from that file. Otherwise, CC Batch uses the *default* configuration file, CCBatchApplication.config.xml, and creates a new configuration file for the specific user <UserName>_CCBatch.config.xml. All of the default user settings set in the application configuration window are copied to the new user-specific configuration file.

If a CC Batch user clicks the Save Settings button, changes made to the user-settings window are saved to the user-specific configuration file.

Example

CCBUser_CCBatch.config.xml

The CC Batch configuration file, CCBatchApplication.config.xml, contains the Unified API configuration settings and the attributes specific to the CC Batch Application.

On every login, CC Batch checks for the configuration file specific to the logged in user. If the specific configuration file is present, CC Batch uses values from that file. Otherwise, it uses the default configuration file to create a new user-specific configuration file.

Each user has their own configuration file stored locally on the machine on which CC Batch is installed. The file contains all the settings specific to that user, except the password.

The contents of the user-specific configuration file are:

- User Name
- Generate Log file
- Generate Error in log file
- Generate Input in log file
- Generate Output in log file
- Generate Separate Error in log file
- Max File Size of log file
- Number Of Backup Files
- General Log Path
- Error Log Path
- Last Batch File Used

Sample Configuration File

```
<?xml version="1.0" encoding="UTF-8"?>
<CCBatchUserSettings>
    <UserName>CCBatchUser</UserName>
    <GenerateLog>true</GenerateLog>
    <GenerateError>true</GenerateError>
    <GenerateInput>true</GenerateInput>
    <GenerateOutput>true</GenerateOutput>
    <GenerateSeparateError>true</GenerateSeparateError>
    <MaxFileSize>100</MaxFileSize>
    <NumberOfBackupFiles>50</NumberOfBackupFiles>
    <LastBatchFileUsed>E:\10000.ccbatch</LastBatchFileUsed>
    <GeneralLogPath>C:/CCBatchLog/General</GeneralLogPath>
    <ErrorLogPath>C:/CCBatchLog/Error</ErrorLogPath>
</CCBatchUserSettings>
```

The user-specific configuration file is created in the /conf directory at application path, typically C:\Program Files\Comverse\CCBatch\conf

Every user must have their own User Name created on the Comverse ONE solution Security Platform under the CC Batch realm. If the CC Batch username is shared among different users, they must share the same settings.

Input Batch Files

The Card Generator creates the batch input files for CCBatch when it creates a batch of vouchers. . The files are in XML format and have the `.ccbatch` extension.

There are two kinds of files:

- Normal Voucher Load
- Fast Voucher Load

NOTE Even though Normal Voucher Load is much slower than Fast Voucher Load, CCBatch supports it for customers upgrading to Comverse ONE from RTB, which did not have the Fast Voucher Load functionality.

Normal Voucher Load Input Batch File

The input file contains a root node called <Batch>, followed by a node specifying the type of operation to be performed. The following is an example of such an input file:

```
<Batch>
  <voucherCreate>
        <VoucherObject>
            <batchNumber>10003</batchNumber>
            <serialNumber>12099</serialNumber>
            <codeNumber>105526044896</codeNumber>
            <acctExpOffset>5</acctExpOffset>
            <changePhonebook>0</changePhonebook>
            <currencyUnit>US Dollars</currencyUnit>
            <distributor />
            <expirationDate>2008-10-15T00:00:00+05:30</expirationDate>
            <expireOffset >0</expireOffset>
            <faceValue>
                <longValue />
                <value>500.00</value>
            </faceValue>
            <identityId />
            <orderNumber />
            <prominExpOffset1 />
            <prominExpOffset2 />
            <prominValue1 />
            <prominValue2 />
            <rechargeServerId />
            <shipDate />
            <spName>Global Reseller</spName>
            <mogName>MarketOfferGroup1</mogName>
            <state>1</state>
            <subscriberId />
            <ucardCos />
            <ucardSp />
            <useDate />
            <voucherType>0</voucherType>
        </VoucherObject>
  </voucherCreate>
</Batch>
```

NOTE CC Batch supports Subscriber and Voucher operations in the same Input file.

CC Batch Log Files

CC Batch has the same logging capabilities as Unified API and the log file format of CC Batch is same as the Unified API method responses. Log maintenance is done using configuration settings.

The CC Batch log files contain the following information:

- Date time stamp (date hour:min:sec) in the header on both request and response
- Request message
- Response message
- Username

General Log

The General Log files are in XML format and contain details about errors, input requests, and output responses.

- **Default Path**: C:/CCBatchLog/General directory
- **Naming Convention**: ccbatch-trace-general.log

Error Log

The Error Log files are in XML format and contain details about input requests and output responses for the commands which encountered errors during Batch Load process.

- **Default Path**: C:/CCBatchLog/Error directory
- **Naming Convention**: ccbatch-trace-error.log

Developer Log

The Developer Log files are not in XML format and contain details about the method name, Unified API logs and other required information. The Developer logs are similar to Unified API Developer logs.

Default Path

- **Default Path**: C:/CCBatchLog/Debug directory
- **Naming Convention**: ccbatch-trace-debug.log

Fast Voucher Load

Fast Voucher Load is CC Batch's high-performance functionality for bulk-loading recharge vouchers to the database. All vouchers in the batch use the common data in the first row.

CC Batch responds to the VoucherFast command by retrieving the Voucher Key from the database using the first row of data. The SQL Loader application, installed with the Comverse ONE solution, creates all of the cards in the batch using the Voucher Key. The Voucher Key is a mandatory field for Voucher Creation and is an autonumber value in the database.

SQL Loader Utility

SQL Loader (sqlldr) is the utility for high-performance bulk data loads. The data can be loaded into the database from any file.

Database Interaction

SQL Loader uses database services for loading the data into the database.

For the database interaction of the SQL Loader utility to operate correctly, the following configuration tasks must be performed:

- The tnsnames file must be configured with the details of database servers used by the Oracle client. The following example shows a typical tnsnames file:

```
MAIN=
(DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST= 10.150.12.238)(PORT= 1521))
    (CONNECT_DATA = (SID =main))
 )
HIST=
(DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST= 10.150.12.238)(PORT= 1521))
    (CONNECT_DATA = (SID =hist))
 )
```

- TNS Service name must be configured in the CC Batch configuration file, which is used by SQL Loader
- The Comverse ONE solution Security Platform must be configured to provide the database password for the User Name, DBType, and DBInstance, which are configured in the CC Batch application configuration file and used by SQL Loader.

Fast Voucher Load Input Batch File

The following example shows a typical input file:

```
<Batch>
    <CreateVoucherBatch>
        <Common>
            <BatchNumber>100001</BatchNumber>
            <CardId>100001</CardId>
            <State>Idle</State>
            <SerialNumber>100001</SerialNumber>
            <batch_count>100000</batch_count>
            <Reseller>Airtel</Reseller>
            <mogName>MarketOfferGroup1</mogName>
            <FaceValue>100000</FaceValue>
            <CurrencyUnit>United States Dollar</CurrencyUnit>
            <date_offset>1000</date_offset>
            <ExpirationDate>2008-04-19</ExpirationDate>
            <create_date>2007-04-19</create_date>
            <Distributor></Distributor>
            <ChangePhonebook>0</ChangePhonebook>
        </Common>
        <Vouchers>
            <CodeNumber>691692770246</CodeNumber>
            <CodeNumber>183309235075</CodeNumber>
            <CodeNumber>095682446233</CodeNumber>
            <CodeNumber>069848061611</CodeNumber>
            <CodeNumber>988832337617</CodeNumber>
            <CodeNumber>708839920870</CodeNumber>
            <CodeNumber>646950770994</CodeNumber>
            <CodeNumber>386489165357</CodeNumber>
            <CodeNumber>363939108213</CodeNumber>
        </Vouchers>
    </CreateVoucherBatch>
</Batch>
```

Creating the First Recharge Voucher and Retrieving its Voucher Key

Database Tables

In the database, recharge vouchers are loaded into two tables:

- **RECHARGE_CARD table**: Contains data for individual cards, such as batch number, serial number, card code, and so on.
- **VOUCHER_DEFINITION table**: Contains common information on the card batch, such as currency, face value, and so on

Each recharge card inserted in the `RECHARGE_CARD` table requires a reference from the `VOUCHER_DEFINITION` table, called the voucher key. The SQL Loader utility only loads the bulk data; it does *not* supply this reference.

CC Batch creates the first voucher in the batch using the normal Unified API voucher creation process. The voucher key for this voucher is then used for the complete batch, i.e. all other cards in the batch will have the same voucher key.

NOTE

Operators *must* be careful to ensure that the combination of Reller and Reseller ID specified for a card batch in the Card Generator configuration or System Options is accurate. Card Generator does *not* validate this pair before creatina g a new batch, Neither does CC Batch perform any data validation on batch as it loads it.

LoadVoucher.ctl file

The SQL Loader control file, LoadVoucher.ctl, describes how the batch data is loaded. It specifies the format of the data, including the table name, column name, column data types, field delimiters, and so on. The following example shows a typical LoadVoucher.ctl file:

```
Load Data
infile "./datafile.dat"
badfile "C:/ErrorFastVoucherLoad.bad"
append
into table RECHARGE_CARD
fields terminated by "|"
(
BATCH_NUMBER,
SERIAL_NUMBER,
CODE_NUMBER,
EXPIRATION_DATE         DATE "YYYY-MM-DD",
DISTRIBUTOR,
STATE,
VOUCHER_KEY
```

Fast Voucher Load Bad File

When at least one record from the input file is rejected, the Fast Voucher Load process creates a Bad File (ErrorFastVoucherLoad.bad) that contains the rejected data records. A record could be rejected for reasons that include a non-unique key or a required column being null. The following example shows a typical Bad File:

```
100001|199997|342159845791|2008-04-19|none|1|468|
100001|199999|117568328565|2008-04-19|none|1|468|
```

Fast Voucher Load Log File

After the job is complete, SQL Loader creates a log file that contains information about the job execution. Summary information at the bottom of the log includes CPU time and elapsed time. The following example shows a typical log file:

```
SQL*Loader: Release 8.1.7.0.0 - Production on Thu Sep 13 18:30:27 2007

(c) Copyright 2000 Oracle Corporation.  All rights reserved.

Control File:   C:\LoadVoucher.ctl
Data File:      C:\datafile.dat
  Bad File:     C:/ErrorFastVoucherLoad.bad
  Discard File:  none specified
```

```
 (Allow all discards)

Number to load: ALL
Number to skip: 0
Errors allowed: 500000
Bind array:     64 rows, maximum of 65536 bytes
Continuation:    none specified
Path used:      Conventional
Silent options: FEEDBACK, ERRORS and DISCARDS

Table "PPS_OWNER"."RECHARGE_CARD", loaded from every logical record.
Insert option in effect for this table: APPEND

   Column Name                  Position   Len  Term Encl Datatype
------------------------------ ---------- ----- ---- ---- ---------------------
BATCH_NUMBER                        FIRST     *   |       CHARACTER
SERIAL_NUMBER                        NEXT     *   |       CHARACTER
CODE_NUMBER                          NEXT     *   |       CHARACTER
EXPIRATION_DATE                      NEXT     *   |       DATE YYYY-MM-DD
DISTRIBUTOR                          NEXT     *   |       CHARACTER
STATE                                NEXT     *   |       CHARACTER
VOUCHER_KEY                          NEXT     *   |       CHARACTER


Table "PPS_OWNER"."RECHARGE_CARD":
  0 Rows successfully loaded.
  99999 Rows not loaded due to data errors.
  0 Rows not loaded because all WHEN clauses were failed.
  0 Rows not loaded because all fields were null.


Space allocated for bind array:                  65016 bytes(36 rows)
Space allocated for memory besides bind array:        0 bytes

Total logical records skipped:          0
Total logical records read:         99999
Total logical records rejected:     99999
Total logical records discarded:        0

Run began on Thu Sep 13 18:30:27 2007
Run ended on Thu Sep 13 18:40:03 2007

Elapsed time was:     00:09:36.14
CPU time was:         00:00:14.36
```

Fast Voucher Load Process Sequence

The complete sequence of the Fast Voucher Load process is as follows:

1. The user provides the input batch file generated by Card Generator to CC Batch for loading.
2. CC Batch validates the input file and creates the voucher object for the first voucher.
3. A first voucher creation request is sent to database via the Unified API libraries, which are part of the CC Batch application.
   - If the voucher creation is *unsuccessful*, the error is returned to the user and logged, and the fast voucher loading process terminates.
   - If the voucher creation is *successful*, a request to retrieve the voucher key for the first voucher of the created batch is sent to the user.
4. After the voucher key is retrieved, the CC Batch application prepares the data file to load the data using SQL Loader, updates the control file with the path for the data file, and prepares the batch file for SQL Loader invocation.
5. CC Batch removes old log files, BadFiles and flag files from the system.
6. CC Batch executes the batch file, which invokes the SQL Loader utility.
7. SQL Loader utility performs the voucher loading.
8. When the batch process is complete , CC Batch analyses the logs generated by the SQL loader utility and displays the results.

CAUTION

If you are attempting to perform a *Fast Voucher Load* operation but the RATING_DB password for the DB schema is not configured at the Security Server, CCBatch will raise an error which will be written to the CCBatch log, the console and the Batch Load screen.
The *entire* batch load operation will then be canceled.

Changing the State of a Batch of Vouchers

The CCBatch user can change the state of a batch of vouchers using the input file voucherModify.ccbatch, found in the input-file-templates folder. The batch file is specified in the input file by it's **batchNumber** and delimited by a range of serial numbers, beginning with **startSerial** and ending with **endSerial** in the input file.

Roles

The CCBatch user must have the CHANGE_VOUCHER_STATE role.

voucherModify.ccbatch

```
<Batch>
  <voucherModify>
        <batchNumber></batchNumber>
        <!-- newState value should be state Id -->
        <newState></newState>
        <startSerial></startSerial>
        <endSerial></endSerial>
  </voucherModify>
</Batch>
```

After a CCBatch user has loaded a batch of vouchers into the voucher database, he may use this modify capability to change the state of the batch from the initial **Idle** one to (typically) the **Shipped** or **Active** state.

NOTE If there is a problem with a batch of vouchers or individual vouchers, (e.g. a voucher has been stolen), the voucher state can be changed to **Disqualified** or **Stolen** using CCBatch or the Unified API. Once this has been done to a voucher, there is no way to re-qualify it.

Modifying the Voucher Details of a Batch of Vouchers

The CCBatch user can modify the Order Number, Distributor, Ship Date and Expiration Date of a batch of vouchers using the input file template voucherModifyExt.ccbatch, which can be found in the input-file-templates folder. The batch file is specified in the input file by its **batchNumber** and delimited by a range of serial numbers, beginning with **startSerial** and ending with **endSerial** in the input file.

Roles

The CCBatch user must have the MODIFY_VOUCHER role.

voucherModifyExt.ccbatch

```
<Batch>
  <voucherModifyExt>
            <com.comverse.api.csm.base.data.VoucherObject>
                        <distributor></distributor>
                        <expirationDate></expirationDate>
                        <orderNumber></orderNumber>
                        <shipDate></shipDate>
                        <state></state>
            </com.comverse.api.csm.base.data.VoucherObject>
        <batchNumber></batchNumber>
        <startSerial></startSerial>
        <endSerial></endSerial>
  </voucherModifyExt>
</Batch>
```

Anonymous Subscriber Load

Anonymous Subscriber load creates single-subscriber (1SA) accounts. Anonymous subscribers are required to enable a provider’s customers to initiate new service or modify existing services through web-based Self Care.

A template for anonymous subscriber creation input files, with the file name anonymousSubscriberCreate.ccbatch, is included with CCBatch. The file is found in the input-file-

templates folder. By default, that is located at
`<C:\Program Files\Comverse\CCBatch\input-file-templates>`

Roles

The CCBatch user must have the CREATE_SUBSCRIBER role.

Template File `anonymousSubscriberCreate.ccbatch`

```
<Batch>
   <subscriberCreateFromOffer>
    <AccountObject>
       <acctSegId>1</acctSegId><!-- Provided by the user -->
    </AccountObject>
    <SubscriberObject>
       <serviceFname>Subscriber</serviceFname><!-- Provided by the user -->
       <serviceLname>Subscriber</serviceLname><!-- Provided by the user -->
       <acctAccessAllowed>false</acctAccessAllowed>
       <languageCode>1</languageCode><!-- Provided by the user -->
       <notificationLanguage>1</notificationLanguage><!-- Provided by the user -->
       <currencyCode>1</currencyCode><!-- Provided by the user -->
       <inRatingDb>false</inRatingDb>
       <liabilityRedirectFlag>false</liabilityRedirectFlag>
       <pinChangeNeeded>false</pinChangeNeeded>
       <primaryOfferId>1</primaryOfferId><!-- Provided by the user -->
       <privacyLevel>0</privacyLevel>
       <ratingState>1</ratingState><!-- Provided by the user -->
       <resellerId>0</resellerId><!-- Provided by the user -->
       <revRcvCostCtr>1</revRcvCostCtr>
       <serviceFranchiseTaxCode>1</serviceFranchiseTaxCode><!-- Provided by the
user -->
       <displayAddressId>2</displayAddressId><!-- Provided by the user -->
      <mogId></mogId>
    </SubscriberObject>
    <OfferIdentifier>
       <attribs>0</attribs>
       <offerId>1</offerId> <!-- Provided by the user -->
       <resellerVersionId/>
       <languageCode>1</languageCode><!-- Provided by the user -->
    </OfferIdentifier>
    <LiabilityRedirectionInstanceObject/>
    <subscriberExternalIdList>
       <SubscriberExternalIdObject>
          <serviceExternalId>10000000</serviceExternalId><!-- Provided by the user
-->
          <serviceExternalIdType>1</serviceExternalIdType>
       </SubscriberExternalIdObject>
    </subscriberExternalIdList>
    <autoInvPlaceholderList/>
    <manInvPlaceholderList/>
    <AdrAddressObject>
       <address1>test</address1><!-- Provided by the user -->
       <address2>test</address2><!-- Provided by the user -->
       <city>test</city>    <!-- Provided by the user -->
       <addressTypeId>2</addressTypeId>
       <state>test</state>  <!-- Provided by the user -->
    </AdrAddressObject>
    <OrderIdentifier/>
    <waiveActivation>true</waiveActivation>
    <effectiveDate/>
    <autocommitOrder>true</autocommitOrder>
    <generateWorkflow>false</generateWorkflow>
    <SubscriberInfoToRetrieve/>
   </subscriberCreateFromOffer>
</Batch>
```

Database Tables

CCBatch loads the anonymous subscriber data into these two dynamic tables:

- **ACCOUNT_SUBSCRIBER table**: Stores subscriber and account information
- **EXTERNAL_ID_EQUIP_MAP table**

CC Batch Error Handling

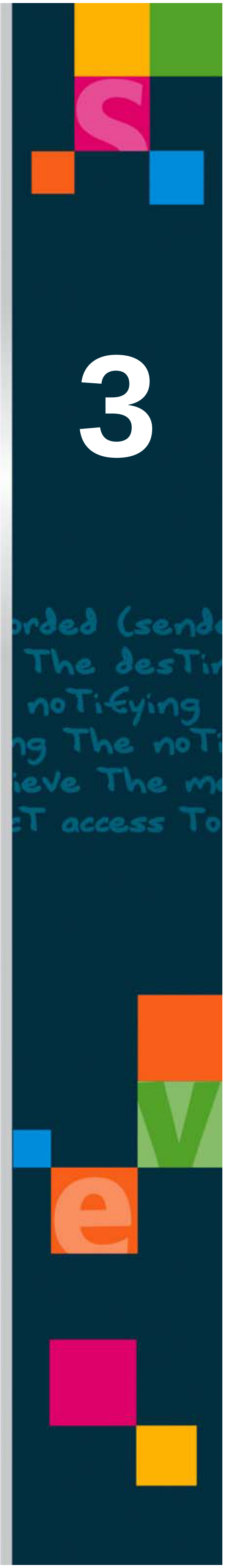
Table 5 shows the CC Batch error messages and a description of the cause of each error.

Table 5 CC Batch Error Messages

Error Message	Description
Invalid User/Password.	Login Failed.
User needs to have CC_BATCH_ACCESS role in order to use the Application.	Logged in user does not have the CC_BATCH_ACCESS permission.
Unable to open/read CCBatchApplication.config.xml file.	Configuration file is not present in the application path or it is not readable.
CCBatch: Unable to set user config.	User-specific configuration file is not present in the application path or it is not readable.
Log file size should be in the range (1 to 500)	Max file size in the configuration file is more than 100 MB.
Number of backup files should be in the range (1 to 100)	Number of backup file in the configuration file is more than 500.
Unable to open/read Batch file.	Input file provided in the batch file is not available in the specified path.
Input file should be a .CCBatch file.	CC Batch File Format is set to CC Batch format and the input file is not in CC Batch format.

Chapter 3

Bulk Creation of Anonymous Accounts



Overview

The Bulk Provisioning Utility allows you to create anonymous accounts to support the pre-activation of pre-paid SIM cards. It instantiates an order that does the following:

- Creates anonymous accounts and subscribers,
- Assigns bundles or primary offers,
- Associates MSISDN and SIM/IMSI with the subscribers.
- If requested, provisions the network to activate SIM cards via a workflow.

The bulk provisioning utility is initiated via a shell script on a UNIX machine

Understanding the BulkProvisioningInputFile.txt Input File

The bulk provisioning utility uses an input file (`<`*`BULK-TOOLROOT-DIR`*`>/input/BulkProvisioningInputFile.txt)`, which allows you to provide input parameter values (such as currency and language code for the account and subscriber) to the bulk provisioning utility. The file consists of two sections:

- **The Dynamic Section**: You will most likely populate this section with new values each time you run the bulk provisioning utility.
- **The Static Section**: Static section values need to be provided the first time you run the bulk provisioning utility. These values can also be modified before any run. These parameters are in the static section because you do not need to modify their values very often.

Input parameter descriptions are provided in the following sections.

Input Parameters in the Dynamic Section

Table 6 provides descriptions of the input parameters in the dynamic section of the `BulkProvisioningInputFile.txt` input file, and indicates validation checks that the bulk provisioning utility performs on the parameters.

Table 6 BulkProvisioningInputFile.txt Parameters: Dynamic Section

Parameter	Required?	Description	Validation check
`count`	Yes	The number of anonymous accounts/services you are requesting the utility to create. The utility will calculate the actual count, based on the following search criteria: - range of MSISDN, - range of SIM, - vanity code, - location of MSISDN and - processing location of SIM	The total number of accounts that can be successfully created is calculated and displayed in the validation report. The count will either be equal to the count requested or, if inventories are not available, less. The user will then have the option to either accept the calculated count or to exit.

Table 6 BulkProvisioningInputFile.txt Parameters: Dynamic Section (Continued)

Parameter	Required?	Description	Validation check
`lowerValue.MSISDN.external_id, higherValue.MSISDN.external_id`	Yes	Search criteria. The lower and upper values of MSISDN inventory range. This range will be used to assign inventory. Only “available” inventory will be used.	No validation checks performed on this parameter.
`vanity.code.MSISDN`	No	Search criterion. Vanity code for MSISDN inventory. (The parameter value can be blank on the right side of “=” or you can comment out the parameter name using #.)	If provided, the vanity code must exist in the database.
`lowerValue.SIM.external_id, higherValue.SIM.external_id`	Yes	Search criteria. The lower and upper values of the SIM card serial number range. Only “available” inventory will be used.	No validation checks performed on this parameter.
`hlr.id`	No	HLR ID for SIM inventory. The bulk provisioning utility updates this parameter in the ‘annotation_2’ field of the SIM card inventory element, `INVD_MAIN`. (The parameter value can be blank on the right side of “=” or you can comment out the parameter name using #.)	HLR ID must exist in the database.
`account.bundleId`	No	Account level bundle ID. If this value is provided, an instance of this bundle will be associated with the account. (The parameter value can be blank on the right side of “=” or you can comment out the parameter name using #.)	In order to get the bundle/offer assigned, the Primary Offer/subscriber bundle/account bundle configuration *must* exist in the database *and* the account segment and account category must be eligible for the account bundle.
`subscriber.bundleId`	No**	If this value is provided, an instance of this bundle will be associated with the subscriber. (The parameter value can be blank on the right side of “=” or you can comment out the parameter name using #.) **NOTE: You *must* have *either* `subscriber.primary.ID`*or* `subscriber.bundle ID`but *not* both.	In order to get the bundle/offer assigned, the Primary Offer/subscriber bundle/account bundle configuration *must* exist in the database *and* the account segment and account category must be eligible for the subscriber bundle.

Table 6 BulkProvisioningInputFile.txt Parameters: Dynamic Section (Continued)

Parameter	Required?	Description	Validation check
`primary.offerId`	No**	If this value is provided, an instance of this primary offer will be associated with the subscriber. (The parameter value can be blank on the right side of "=" or you can comment out the parameter name using `#.`) **NOTE: You *must* have *either* `subscriber.primary.ID`*or* `subscriber.bundle ID`but *not* both.	In order to get the bundle/offer assigned, the Primary Offer/subscriber bundle/account bundle configuration *must* exist in the database *and* the account segment and account category must be eligible for the Primary Offer.
`box.type`	No	Prepaid box type as free-form text for MSISDN inventory. Examples: blue box, red box. The bulk provisioning utility updates this parameter in the 'annotation_1' field of the SIM card inventory element, `INVD_MAIN`. (The parameter value can be blank on the right side of "=" or you can comment out the parameter name using #.)	No validation checks performed on this parameter.
`location.MSISDN`	No	Search criterion. ID of the location that the MSISDN is to be taken from. (The parameter value can be blank on the right side of "=" or you can comment out the parameter name using `#).`	Location ID must exist in the database.
`location.SIM`	No	ID of the location that the SIM is to be taken from. (The parameter value can be blank on the right side of "=" or you can comment out the parameter name using #.)	Location ID must exist in the database.
`account.segment`	Yes	Account segmentation key, which controls which CSRs can view an account's information.	The account segment must exist in the database, i.e. this is the system-defined `acct-seg-id` in `ACCT_SEG_REF`
`currency.code`	Yes	Currency code for the account and the subscriber.	The currency code must exist in the database.
`notification.language`	Yes	Subscriber notification language code.	The notification language must exist in the database
`language.code`	Yes	Language code for sending notification.	The language code must exist in the database

Table 6 BulkProvisioningInputFile.txt Parameters: Dynamic Section (Continued)

Parameter	Required?	Description	Validation check
reseller.version	No	Reseller version ID. If not provided the latest reseller version id will be used.	The reseller version must exist in the database Used to find the specified offer and bundle information.

Input Parameters in the Static Section

Table 7 describes salient parameters in the static section of the BulkProvisioningInputFile.txt input file. They are located here because the user does not have to modify them very often.

For anonymous accounts, the subscriber's contact, name and address fields can be left blank.

Table 7 BulkProvisioningInputFile.txtInput Parameters: Static Section

Parameter	Required?	Description
generate.workflow	No	Turns workflow generation on or off. Valid values: ■ true (default) ■ false If true, a WorkPoint job will be generated.
account.category	Yes	Account category identifier, defined in the ACCOUNT_CATEGORY_REF table.
subscriber.address1	Yes	Subscriber address - first line.
subscriber.address2	Yes	Subscriber address - second line.
subscriber.state,	Yes	Subscriber state or province.
subscriber.city	Yes	Subscriber city.
subscriber.address.type.id	Yes	Subscriber address type.
MSISDN.inventory.type.id	Yes	Inventory type for MSISDN.
SIM.inventory.type.id	Yes	Inventory type for SIM.
MSISDN.external.id.type	Yes	External ID type for MSISDN. Needed to assign Inventory placeholder when creating a subscriber.
SIM.external.id.type	Yes	External ID type for SIM. Needed to assign Inventory placeholder when creating a subscriber.

Populating the Input File

You are limited to supplying values for parameters already present in the input file. You cannot add additional parameters.

To modify the input file, follow these three steps:

1. Change to the <*BULK-TOOLROOT-DIR*>\input directory.
2. Open the BulkProvisioningInputFile.txt file
3. Populate the variables. For guidance, see Table 6 and Table 7.
4. Save and close the file.

Configuring Security, Runtime, and Logging Properties

Security and Runtime

Use the `config/BulkTool.properties` file to configure security and run properties. To configure the file, follow these steps:

1. Change to the `<BULK-TOOLROOT-DIR>/config` directory.
2. Open the `BulkTool.properties` file.
3. Configure the following security server properties:
   - `SEC_USER_NAME`
   - `SEC_USER_PASSWORD`
   - `SEC_USER_REALM`

NOTE The user name, password and realm parameters are pre-populated with default values. The user needs to replace those with values appropriate to his production environment. This is typically a one-time operation, unless there is a change in the user’s security settings.

4. To add IMSI external ID, a new parameter, ADD_IMSI_EXTID, has been added.

   To create external ID for IMSI,. set the value of this parameter to “true”.
5. To improve the performance of the bulk provisioning utility, you may modify the `NUM_WORKER_THREADS` property.

CAUTION Do *not* modify *any* other parameters in the BulkTool.properties file.

6. Save and close the `BulkTool.properties` file.

Logging

Use the `config/log4j.xml` file to control logging properties, such as the verbosity of the reporting, the maximum size of a log file and the number of old log files that the system will retain.

To configure the file, follow these steps:

1. Change to the `<BULK-TOOLROOT-DIR>/config` directory.
2. Open the `log4j.xml` file.
3. To set the verbosity level, edit the two highlighted lines in the extract below.

Possible values for logging verbosity level

Table 8 Setting Logging Verbosity in ‘log4j.xml’ file

Setting	Description
fatal	*Least* verbose. Utility logs only Fatal Error messages
warning	Utility logs Warning + Fatal Error messages
error	Utility logs Error + Warning + Fatal Error messages
info	Utility logs above messages + gives basic processing info
trace	*Most* verbose. Utility logs all above messages + gives very detailed processing info

Figure 10 Extract from Logging Configuration File ‘log4j.xml’

```
<appender name="bulk_tool_appender"
  class="org.apache.log4j.RollingFileAppender">
   <!-- **** File name must match SAPILOGNAME value in
        Install.properties       **** -->
   <param name="File" value="BulkTool.log" />
   <param name="BufferedIO"   value="true" />
   <param name="Append" value="true" />
   <param name="MaxFileSize" value="1MB" />
   <param name="MaxBackupIndex" value="10" />
   <param name="Threshold" value="error"  />
   <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d [%p] %c{2} | %m |
       %t%n"/>
   </layout>
</appender>
<!-- ================================================================ -->
  <!-- possible logger threshold level values: --->
    <!-- trace < debug < info < warn < error < fatal -->
<!-- ================================================================ -->

<logger name="com.comverse.bulkTool.bulkProvisioning">
   <level value="error"  />
</logger>
```

Output File

Table 9, on page 37 shows the format of the generated output file (one line per account)

Understanding the Bulk Provisioning Utility's Operation Modes

The bulk provisioning utility is initiated via a shell script on a UNIX machine and can be run in either of two modes:

- **Validation Mode**, in which the system validates input parameters and produces a validation report.

CAUTION

If *all* inputs are valid *and* the Primary Offer/Subscriber Bundle ID/Account Bundle ID is compatible, the utility will proceed *directly* to Execution Mode.

However, if *any* input parameter is invalid *or* if the Primary Offer/Subscriber Bundle ID/Account Bundle ID is not compatible, then the utility will *exit.*

- **Execution Mode**, in which accounts and subscribers are created and SIM cards are provisioned on the network. The system creates a report that shows which account/subscriber pair is associated with which primary offers and MSISDN, SIM and IMSI numbers.

Regardless of mode, after the bulk provisioning utility runs, it creates the following directories (if they do not already exist):

- `work`
- `error`
- `report`
- `done`

These directories are explained in the following sections.

Running the Bulk Provisioning Utility In Validation Mode

NOTE

If execution permission is not set on the `RunBulkTool.sh` script, use the following command to set execution permission:

```
chmod 755 RunBulkTool.sh
```

To run the bulk provisioning utility in validation mode, change directory to the *`<BULK-TOOLROOT-DIR>`*`/` directory and issue the following command:

```
RunBulkTool.sh -v BulkProvisioningInputFile.txt
```

NOTE

In validation mode, the bulk provisioning utility only *validates* input parameters in the `BulkProvisioningInputFile.txt` input file. No accounts or subscribers are created in the database, and inventory does not change its state.

Results of a Validation Mode Run

After running, the bulk provisioning utility writes a report to the console and also writes a validation report file to the `error` directory, using the following naming convention:

```
BulkTool.error.<PID>.<dd-mm-yyyy-hh-min-ss>
```

Contents of the Validation Report

After you view the validation report, you can decide whether to run in execution mode or modify input parameters in the `BulkProvisioningInputFile.txt` file.

Following is sample content from a typical validation report file:

```
|     =================================================================
|     |                 Input Validation Report                        |
|     =================================================================
|     Reseller version id                  :  1 is InValid.
|     Language code                        :  1 is Valid
|     Currency code                        :  1 is InValid
|     Notification language code           :  1 is Valid
|     Account Segment                      :  1 is Valid
|     Subscriber Primary offer Id          :  100 is InValid
|     Subscriber Primary offer Id          :  100 is not compatible
|     SIM Hlr Id                           :  100 is InValid
|     Number of account expected           :  5
|     Account can be created               :  0
|     -----------------------------------------------------------------
```

NOTE If *all* inputs are valid *and* the Primary Offer/Subscriber Bundle ID/Account Bundle ID is compatible, the utility will proceed directly to Execution Mode. However, if *any* input parameter is invalid *or* if the Primary Offer/Subscriber Bundle ID/Account Bundle ID is not compatible, then the utility will *exit.*

Running the Bulk Provisioning Utility In Execution Mode

NOTE If execution permission is not set on the `RunBulkTool.sh` script, use the following command to set execution permission:

```
chmod 755 RunBulkTool.sh
```

To run the bulk provisioning utility in execution mode, change directory to the *<BULK-TOOLROOT-DIR>/* directory and issue the following command:

```
RunBulkTool.sh -e BulkProvisioningInputFile.txt
```

In this mode, the bulk provisioning utility creates accounts and subscribers in the database.

Results of an Execution Mode Run

In execution mode, the bulk provisioning utility does the following:

1. Copies the input file from the `input` directory to the `work` directory as `BulkProvisioningInputFile.txt.`*<PID>* and reads input from this file.

2. After all requests are finished, the input file is moved to the `done` directory as `BulkProvisioningInputFile.txt.<PID>.<dd-mm-yyyy-hh-min-ss>`.
3. Creates a report in the `report` directory as `BulkTool.report.<PID>.<dd-mm-yyyy-hh-min-ss>`.
4. Creates `BulkTool.log` file in the `<BULK-TOOLROOT-DIR>`.

Contents of the Report File

The report file is comma delimited, with one line per account. Following is a sample line from a typical report file:

```
-------------------------------------------------------------------------
101206,9722,,7778,7778,0,100,Basic Cable Offer,,11081124,22060,42070
-------------------------------------------------------------------------
```

Missing or null values are represented as `,,`

Table 9 Bulk Provisioning Utility Output File Format

Number	Parameter
1	Account External ID
2	Account Internal ID
3	Account Bundle ID
4	Subscriber External ID
5	Subscriber Internal ID
6	Subscriber Internal ID Resets
7	Primary Offer ID
8	Primary Offer display name
9	Subscriber Bundle ID
10	MSISDN Number
11	SIM Number
12	IMSI Number

The log level is controlled by the `log4j.xml` file (in the `<BULK-TOOLROOT-DIR>/config folder. See` "Logging," on page 33

If the generated order is in a completed state, then all accounts and subscribers will have been created in the customer and rating databases, and inventories will be in the assigned state.

Trouble Shooting

Table 10 lists some common problems and their recommended solution.

Table 10 Bulk Provisioning Utility Troubleshooting Information

Problem	Recommended Solution
You cannot connect to the Unified API application server.	Verify that you set the correct application server IP address and port for the `java.naming.provider.url` element in the `CCBSClientConfiguration.xml` file.

Table 10 Bulk Provisioning Utility Troubleshooting Information (Continued)

Problem	Recommended Solution
You cannot authenticate the client.	Verify the following settings in the CCBSConfiguration.properties file in Unified API client directory: ■ The security.server.ip property must contain the correct security server IP address. ■ The security.authorization.enabled property must be set to true.

You receive the following error message: User information is not set in User Context	Check the CCBSClientConfiguration.xml file in Unified API client directory. Verify the threaded and enabled elements are set to true as follows:
	`<usercontext>`
	`  <threaded>true</threaded>`
	`</usercontext>`
	`<defaults>`
	`  <enabled>true</enabled>`
	`</defaults>`

You do not have enough logging information in the BulkTool.log file.	Modify the log4j.xml file in the <*BULK-TOOLROOT-DIR*>/config directory.
Validation fails	■ Check that all parameters in the input file. BulkProvisioningInputFile.txt, are entered correctly and are valid. ■ Check the CCBSClientConfiguration.xml file. ■ Check the CCBSConfiguration.properties file. ■ Verify that the wlfullclient.jar file is present in the <*home*>/BEA/wlserver_10.0/server/lib/wlfuillclient.jar directory.

Executing the Bulk Provisioning Utility from a Remote Box

If you don't install Weblogic server or client on a remote box, you need to copy the following jar files from the Weblogic server to your remote box.

- license.bea
- weblogic.jar
- webserviceclient+ssl.jar
- wlfullclient.jar, wseeclient.jar

Chapter 4

Bulk Load Utility (RT)

Real Time only	This entire chapter pertains to Real Time only.

Overview

The bulk load utility ensures that the proper number of anonymous accounts for any offer ID/reseller ID pair exist in the database to support unified cards. The bulk load utility is intended to be installed on a UNIX machine and run on a periodic basis. The bulk load utility is actually a shell script (`call_sapi_bulktool.sh`) that creates an input file for the bulk provisioning utility.

The bulk load utility must be installed on the same server where the bulk provisioning tool is located. The input file to the bulk load utility is explained in the following sections.

Understanding the BulkProvisioning.txt Input File

The `BulkProvisioning.txt` input file is in the `<`*`BULK-TOOLROOT-DIR`*`>/input/` directory. The file consists of two sections:

- **The Dynamic Section**: You will most likely populate this section with new values each time you run the bulk load utility.
- **The Static Section**: Static section values need to be provided the first time you run the bulk load utility. These values can also be modified before any run. These parameters are in the static section because you do not need to modify their values very often.

Input parameter descriptions are provided in the following sections.

Input Parameters in the Dynamic Section

Table 11 provides descriptions of the input parameters in the dynamic section of the `BulkProvisioning.txt` input file, and indicates validation checks that the bulk load utility performs on the parameters.

Table 11 BulkProvisioning.txt Input Parameters: Dynamic Section

Parameter	Required?	Description

`lowerValue.External_id`
`higherValue.External_id`
	Yes	These values are combination of set of values in the following format:
		`<Reseller_id>'|'<Offer_Id>'|'<external_id>`
		where:
		- `Reseller_id` identifies the reseller
		- `Offer_Id` identifies the primary offer id which is used for creating 1SA accounts
		- `external_id` will be different for lower value and higher value.
		Example:
		`lowerValue.External_id=0|29|123456`
		`higherValue.External_id=0|29|123460`

`currency.code`	Yes	Type currency used while creating the single subscriber account. Current default value is 1 (US Dollars).

Table 11 BulkProvisioning.txt Input Parameters: Dynamic Section (Continued)

Parameter	Required?	Description
`language.code`	Yes	language used for creating single subscriber accounts. Current default value is `1` (English).
`notification.language.code`	Yes	language used for sending notification. Default value is `1` (English).
`reseller.version`	Yes	Reseller version ID. Default value is `1`.
`CALLING_CARD.external.id.type`	Yes	Default value is `4` (calling card).
`account.segment`	Yes	Account segmentation key, which controls which CSRs can view an account's information. Default value is `1`.

Input Parameters in the Static Section

Table 12 describes the parameters in the static section of the `BulkProvisioning.txt` input file. They are located in the static section because the user does not have to modify them very often.

Table 12 BulkProvisioning.txt Input Parameters: Static Section

Parameter	Required?	Description
`account.category`	Yes	The account business category. Default value is `1`.

`subscriber.address1`
`subscriber.address2`
`subscriber.state`
`subscriber.city`
`subscriber.address.type.id`
	Yes	Various parts of the subscriber's address.

`address.type.id`	Yes	Address type. Default value is `2` (free-form).
`rating.database`	Yes	Rating database ID, from `SERVER_DEFINITION` table. Used for only validation and inventory reserve operation. Default value is `9`.

Populating the Input File

You are limited to supplying values for parameters already present in the input files. You cannot add additional parameters.

To modify the input file, follow these three steps:

1. Change to the `<`*`BULK-TOOLROOT-DIR`*`>\input` directory.
2. Open the file `BulkProvisioning.txt` file.
3. Populate the variables. For guidance, see Table 11 and Table 12.
4. Save and close the file.

Running the Bulk Load Utility In Validation Mode

NOTE

If execution permission is not set on the `RunBulkTool.sh` script, use the following command to set execution permission:

```
chmod 755 RunBulkTool.sh
```

To run the bulk load utility in validation mode, change directory to the *<BULK-TOOLROOT-DIR>/* directory and issue the following command:

```
RunBulkTool.sh -v BulkProvisioning.txt
```

Running the Bulk Load Utility In Execution Mode

NOTE

If execution permission is not set on the `RunBulkTool.sh` script, use the following command to set execution permission:

```
chmod 755 RunBulkTool.sh
```

To run the bulk load utility in execution mode, change directory to the *<BULK-TOOLROOT-DIR>/* directory and issue the following command:

```
RunBulkTool.sh -e BulkProvisioning.txt
```