Oracle Utilities Cloud Services

Implementation Guide

For 21B Releases

F43382-01

August 2021

ORACLE

Oracle Utilities Customer Cloud Services 21B Integration Guide

Contents

Chapter 1

Introduction

This document provides information about implementation of Oracle Utilities cloud service, including:

- Oracle Utilities Billing Cloud Service
- Oracle Utilities Customer Cloud Service
- Oracle Utilities Meter Solution Cloud Service
- Oracle Utilities Operational Device Cloud Service
- Oracle Utilities Rate Cloud Service
- Oracle Utilities Work and Asset Cloud Service

The topics described in this document include:

- Part One: Implementation Guidelines
- Part Two: Data Upload Support
- Part Three: File-Based Integration
- Part Four: Oracle REST Data Services
- Part Five: Product-Specific Integrations

Part One

Implementation Guidelines

This section describes global implementation guidelines that apply to all Oracle Utilities cloud services running on Oracle Cloud Infrastructure (OCI), which includes

- Oracle Utilities Billing Cloud Service
- Oracle Utilities Customer Cloud Service
- Oracle Utilities Meter Solution Cloud Service
- Oracle Utilities Operational Device Cloud Service
- Oracle Utilities Rate Cloud Service
- Oracle Utilities Work and Asset Cloud Service

Note that these cloud services are all based on the Oracle Utilities Application Framework (OUAF), which supports many different configuration and extension methods, almost all of which are available for use in the cloud. This section provides recommendations for many aspects of set-up and operation of the services. Note that it assumes familiarity with OUAF concepts and tools.

In a nutshell, the top cloud service implementation rules to be aware of are the following:

- Use Groovy code in Scripts (not Java)
- Use existing data structures to extend the base model - such as Characteristics and the Fact table
- Use plug-in driven batch can be used in many scenarios for data fixes - this will ensure proper data validation

The guidelines in this are intended to help implementers to configure and run their cloud services efficiently.

This section include the following chapters:

Chapter 2

Security and Access

This chapter provides implementation guidelines related to security and access, including:

- Identity Management
- Server Access

Identity Management

Use of Identity Cloud Service

In Oracle Cloud Infrastructure, cloud services are provisioned using Oracle Identity Cloud Service (IDCS) to manage user creation, application access, passwords, etc. This service at the 'Foundation' tier is included with the Oracle Utilities cloud service subscription. See Administering Oracle Identity Cloud Service for more information on IDCS.

By default Oracle Identity Cloud Service allows access to the application front-end from any IP address. There are capabilities in IDCS to add sign-on policies that allow or deny IP addresses through the use of allowlists (though some features may require 'Standard' tier licensing).

User Provisioning with Oracle Identity Cloud Service

Application users are added through Oracle Identity Cloud Service, which manages the user lifecycle (i.e. you can disable a user, or reset a user's password in IDCS). The access rights of the user within the application are controlled using the settings on the cloud service User record. Oracle Identity Cloud Service uses Application Roles and Groups: a user must be linked to the Application Roles that they need access to. This linking can also be 'indirect' by linking a new user to a Group which has access. Creation of cloud service User records is done 'just-in-time' - upon the first login to the application, after authentication via Oracle Identity Cloud Service, a call is made to verify access to the application, and using the returned information including the user's IDCS Groups, a template user in the cloud service can be found and used as the 'copy from' source.

Instructions: The security administrator should create an initial User record with full access to the cloud service (including administration functionality). This user should be used to configure "Template Users" and mappings to IDCS Groups. See **Access and Identity Management** in the *Oracle Utilities Cloud Services Administration Guide* for more information. Note that the Cloud Service Foundation also provides several Template Users that have necessary access for process automation.

Server Access

While server access is restricted exclusively to members of the Oracle Cloud Infrastructure and Oracle Utilities Development Operations (DevOps) teams, logs are available to users. See the **Server Logs - Online and Batch** on page 6-7 section for more information.

Chapter 3

Configuration Tools

This chapter describes specific implementation guidelines related to use of Oracle Utilities Application Framework Configuration Tools, including:

- Customization Tools Summary
- Algorithm Types and Algorithms
- Application Environments
- Creating Batch Processes

Customization Tools Summary

While most of each cloud service application's customization options are supported, some are not and others may be limited in certain areas. The table below outlines configuration options and their availability when implementing Oracle Utilities cloud services.

Category	Option	Supported?	Comment
Business Entities	Add custom business objects for product maintenance objects.	Yes	Assuming the Maintenance Object supports Business Object functionality.
	Extend a product business object's structure and rules.	Yes	See the **Algorithm Types and Algorithms** on page 3-3 for more information.
	Add custom maintenance objects.	No	Creation of new tables is not supported. See the **Database Access** on page 7-2 section for more information. Use of the SDK tool to generate Java artifacts is not supported. See the **Algorithm Types and Algorithms** on page 3-3 for more information.
User Interface	Add a custom portal.	Yes	Restricted to a single tab page in 20C and previous versions. Additional tab pages are supported as of release 21A.
	Extend a product portal with custom zones.	Yes	
	Extend a product multi-query search with custom query options.	Yes	
	Customize a product menu. This includes adding new custom menu lines, hiding and reordering lines.	Yes	
	Add custom indexes to support custom queries	No	See the **Database Access** on page 7-2 section for more information.
Batch processes	Add a custom batch process.	Yes	The program cannot be written in Java. See the **Creating Batch Processes** on page 3-6 for more information. The process may only access designated locations in Object Storage. See the **File Access - Cloud Object Storage** on page 7-3 section for more information.

Web Services	Add custom inbound and outbound web services.	Yes	Use Outbound Messaging and Inbound Web Services. The services cannot rely on XSL transformations to occur in the cloud. See the **File Access - Cloud Object Storage** on page 7-3 section for more information.
Reports	Add stored procedures to support custom reports.	No	See the **Database Access** on page 7-2 section for more information.

Algorithm Types and Algorithms

New algorithm types and algorithms can be created during implementation using Scripts. Custom Java-based algorithm types are NOT permitted.

Write custom algorithm types using either Groovy or Oracle Utilities Application Framework's XML-based scripting. Refer to **Defining Algorithms, Plug-In Scripts, and Using Groovy within Scripts** in the *Administrative User Guide* or online help for information about creating algorithms using Groovy.

Key Guidelines of Groovy scripting are:

- Review the third party groovy allowlist (available within the application)
- Be careful with goto statements - it's easy to create endless loops
- Review SQL Function Allowlist (Refer to F1-SQLFunctionWhiteList Managed Content)
- Update/Delete SQL Statements are not allowed
- Explain Plan of the query(s) needs to be examined for all SQLs written in custom code.

When crafting custom SQL queries, you must consider performance. Run explain on all of your SQLs using rule hint before delivering code range scans and nested loops only. Plans with 'table access' are not acceptable. Use the SQL Web Developer toolset to check SQL.

Application Environments

Each cloud service by default comes with three environments designated as Development, Test, and Production. Test and Production are sized as full-sized environments based on the billable metric of the subscription, while Development is a smaller environment.

Customers can request additional non-production environments through the initial sales order or in a subsequent order for an additional subscription. When asking for more environments, the names of the base and additional environment are predefined and cannot be changed.

Environment Names and Codes

Cloud service environments have an environment code and name. The environment code is used to identify the environment and enable migration processes (such as configuration migrations) between the environments. The environment code is also used at installation/provisioning time. The table below lists all the possible environments that can be provisioned for each cloud service.

Environment Code	Name	Type	Default	Additional
DEV	Development	Development	Yes	No
TEST	Test	Test	Yes	No
PROD	Production	Production	Yes	No
DEV01..DEV10	Development 1 .. 10	Development	No	Yes
TEST01..TEST10	Test 1 .. 10	Test	No	Yes

Application / Environment Access and URL Tokens

When environment provisioning is complete, customers / implementers will receive a list of links to the various product environments included with their subscription. There are cases in which cloud service applications need to access other cloud service or on-premises applications or other environments, such as:

- Data/Configuration Migration
- Data Conversion
- Redirecting a user to another application as part of a business process transaction that is integrated across products
- Invoking web services of a different application (another cloud service or SOA for existing SOA-supported cloud integrations)
- Invoking a web service of the same application in a different environment. This type of communication is used to help automate inter-environment processes like configuration migrations. See **Code and Configuration Migration** on page 6-4 for more information.

The following URL Tokens are available for direct navigation or web service calls for each cloud service.

Token	Description
EXT_PUB	Prefix token that should be used to reference external addresses in Message Senders. For example, to reference paymentcorp.payusa.com endpoint URL you would need to use the following notation: @EXT_PUB@paymentcorp.payusa.com.
CCS_WS CCS_ONLINE	Customer Cloud Service address for web service calls. Customer Cloud Service address for online access.
MSCS_WS MSCS_ONLINE	Meter Solution Cloud Service address for web service calls. Meter Solution Cloud Service address for online access.
WACS_WS WACS_ONLINE	Work and Asset Cloud Service address for web service calls. Work and Asset Cloud Service address for online access.
AICS_ONLINE	Analytics Insights Cloud Service (aka DataRaker) address for online access.
INT_WS	Invoke SOA web services (for integration via SOA)
BI_PUBLISHER_ADMIN	BI Publisher address for web services calls.
DEV_WS, DEV01_WS-DEV10_WS TEST_WS, TEST01_WS-TEST10_WS PROD_WS	Web service addresses for all possible environments. For example, DEV_WS can point to Customer Cloud Service in the DEV domain while PROD_WS will point to the same application but in the PROD domain.

Usage Examples:

- The outbound message sender on the Customer Cloud Service configured for invoking a Meter Solution Cloud Service Inbound Web Service service named "ABC" (in production) will be have the URL definition of @MSCS_WS@abc.
- The value of @MSCS_WS@ will be different in each environment so that the same token can be used in all environments, and the runtime value translation will be based on the environment invoking the call.
- Internal-facing tokens such as DEV_WS can be used for inter-domain communications, such as the automation of configuration migration between product domains. These tokens are used by the Process Automation Tool within Cloud Service Foundation.
- External facing addresses will use the @EXT_PUB@ prefix, for example: @EXT_PUB@paymentcorp.payusa.com/api/int01/addPayment

Creating Batch Processes

Custom batch jobs can be written using the plug-in driven batch job functionality supported by the Oracle Utilities Application Framework. There are three broad categories of batch jobs that may be implemented using plug-in driven batch.

- **Ad-hoc Processing**. This covers any batch job that should select records in the system and perform some type of logic for each record.
  - The system provides a Select Records plug-in for retrieving the records in the system based on criteria. This plug-in requires the selection SQL (properly tuned) to be defined as a parameter. Logic in the plug-in script may be used to set filter criteria if needed. The plug-in script may be written using XPath scripting.
  - The system also provides a Process Record plug-in where each record may be reviewed and some appropriate action may be performed. This plug-in may be written in either XPath or Groovy scripting.
  - See also the **Data Fix with Plug-in Driven Batch** on page 6-5 section
- **Extract a Batch of Records**. This covers any batch job that produces an extract of records. The same plug-in spots described for Ad-hoc processing are applicable here. The Select Records plug-in is used for selecting the records eligible for extraction, for example from as staging table. The Process Record plug-in us responsible for returning the data to be written to the extract for each record. This plug-in may be written in either XPath or Groovy scripting. However, because the output of the plug-in is one or more schema objects to include in the extract, XPath scripting may be better suited.
- **Upload Records from a File**. This covers any batch job that needs to read a file and create records in the system based on the content. The system provides a File Upload plug-in spot. This plug-in is responsible for calling appropriate APIs to read the content of the file and store the data in appropriate tables, for example a staging table. This type of plug-in must use Groovy as the APIs are not accessible using the XPath scripting language.

Chapter 4

Data Conversion Guidelines

This chapter provides general guidelines related to data conversion.

Data conversion refers to the migration of data from a client's legacy system (on-premise or cloud) to the application database(s) within Oracle Utilities cloud services. Since no direct access is permitted to the application database, data conversion support is provided to facilitate SQL Loader-based data upload via Cloud Service Foundation tools. Staging tables cleanup is also supported.

Data Conversion Approach

The implementation project is expected to extract the legacy data into flat files and upload these files to a specific location on the cloud, and then to run a sequence of batch processes that moves the data into corresponding tables in the special Staging database schema. The subsequent processing of the staging data, along with its insertion into production tables, is specific for each cloud service. Refer to **Data Conversion and Migration** in the *Administrative User Guide* or the online help for information about data conversion. See **Data Upload Support** in the *Oracle Utilities Cloud Services Implementation Guide* for more information about data upload. Sample upload files are also available from the Oracle Utilities Documentation library for the relevant cloud service.

It is recommended to start with small set of data covering most of the unique / critical scenarios, perform the object / FK validations and try to resolve the data quality conversion issues by comprehensive testing (both online and batches). Gradually increase the data volume to avoid running full scale of converted data early resulting in application errors (invalid data will often lead to a lot of 'noisy errors' that can be avoided with this approach).

Data Conversion Tips

The following high-level tips are important for data conversion efforts:

1: Data Upload Indexes and Constraints

Data conversion is performed by processing legacy data extract files using SQL Loader. During the data upload, the indexes and constraints are disabled and duplicate keys are not validated. See Oracle SQL Loader Documentation for details.

Please ensure that you cleanse the data extract file and remove duplicates prior to the upload.

2: Key Tables in Staging Area

The Key tables in the staging area are not populated automatically. The Key Table data has to be created with the corresponding Environment ID and then uploaded as a separate extract. Refer to **Data Conversion and Migration** in the *Administrative User Guide* or the online help for more information.

3: CLOB Data Upload with Secondary Files

The CLOB data upload with secondary files is not supported when there are multiple CLOB columns in the table. Configure the conversion task type to include CLOB data in the main extract, amend Conversion Master Configuration, and regenerate Conversion Artifacts. Refer to **Data Conversion and Migration** in the *Administrative User Guide* or the online help for more information.

Chapter 5

Integration Guidelines

This chapter provides guidelines related to integration with Oracle Utilities cloud services including:

- Integration Methods
- Integration Middleware
- Allowlisting

Integration Methods

The primary integration methods supported with cloud services are (a) inbound and outbound files, and (b) inbound and outbound web services. Other protocols and methods (JMS, SQL Net, etc.) are not currently supported.

Besides standard Oracle Utilities Application Framework integration modules, no additional extract, transform, and load (ETL) capabilities or middleware are provided with cloud service offerings. Oracle Cloud middleware solutions—such as SOA Cloud Service (available via Platform-as-a-Service) or Integration Cloud Service —need to be licensed to address advanced integration requirements such as complex ETL, orchestration, etc. Alternatively an on-premise middleware solution could be used.

Integration Method: File-Based

Inbound File Processing: Files are uploaded to Object Storage and processed via scheduled batch jobs. Implementation-specific file parsing and processing logic can be introduced using browser-based Oracle Utilities Application Framework tools. Refer to **Uploading Records in the Plug-in Driven Background Processes** section in the *Administrative User Guide* or the online help for more information. Please also review the **File Access - Cloud Object Storage** on page 7-3 section.

Outbound File Processing: File-based extracts can be generated and made available for download and further processing. Implementation-specific file processing and generation logic can be introduced using browser-based OUAF tools. Refer to **Processing System Records in the Plug-in Driven Background Processes** section in the *Administrative User Guide* or the online help for more information.

Large-volume data conversion and loading is supported. See Chapter 4: Data Conversion Guidelines for more information.

Integration Method: Web Services

Web services are supported through Inbound Web Services (IWS) and Outbound Messages. All inbound and outbound web services communication must be HTTPS. Refer to **Inbound Web Services** and **Outbound Messages** in the *Administrative User Guide* or the online help for more information.

Note that in order to call Inbound Web Services, you must provide a user/password for authentication and authorization (the user must be defined in Oracle Identity Cloud Service with the 'AppWebServices' Application Role and as an application User). Inbound Web Services support both SOAP and REST. Outbound Messages may only reference public IP addresses, and those addresses must be on an 'allowlist' (which can be provided to Cloud Operations via a service request ticket).

For integrations that involve outbound synchronization to other systems driven by online activity, real-time synchronous outbound messages are not recommended. Rather use the business object batch monitor processing on a frequent basis to process queued messages. This involves using the deferred monitor batch set on the PENDING state of the Sync Request so that message processing occurs asynchronously.

SSL certificates must be created using certification authority. Self-signed SSL certificates are not supported. Also reference the How to access SOAP and REST Services in Oracle

Utilities Enterprise Cloud Services document on Oracle Support (Document ID 2564697.1).

Upload and attachment of implementation-specific xsl files to process xml payloads is supported through the **Managed Content** portal for relevant product or cloud service. Refer to **Maintaining Managed Content** in the *Administrative User Guide* or the online help for more information.

Integration Middleware

While file-based integration does not require middleware, often real-time integration benefits from the use of a middleware platform to facilitate message delivery, error handling, and data transformation. With Oracle Utilities cloud services, there are several different middleware options which may be useful, and in some cases accelerator code is available to help get an integration up and running quickly. This section describes several middleware options (note: these are not included with your cloud service subscription).

Integration Middleware: Oracle Integration Cloud (OIC)

Oracle Integration Cloud Service (OIC) is an integration platform offered as Software-as-a-Service - it provides a modern web-based user interface to set up integration connection points, and uses application catalogs of available services provides data mapping capabilities and statistics on message flows. The Oracle Integration Cloud suite includes additional analytics and other tools. Oracle Utilities uses OIC as an integration platform to link to the Customer Experience (CX) applications including Oracle Field Service Cloud (OFSC), and is likely to make accelerator packages available. OIC can also handle file-based integration, but is not currently recommended for complex multi-system integration flows. As a cloud offering Oracle supports upgrades to the service, but note that Disaster Recovery is not currently part of the standard offering.

Integration Middleware: SOA Cloud Service (SOACS) - PaaS

This option uses the SOA Suite hosted as a Platform as a Service (PaaS) - thus allowing for full control and development capability. As a PaaS, the customer is responsible for managing the software, updates, etc.

Integration Middleware: SOA Suite On-Premises

This option uses the SOA Suite hosted on premises- thus allowing for full control and development capability. The customer is responsible for managing the software, updates, etc.

Allowlisting

Allowlisting is required to specify allowable access destinations on the public internet.

IP Allowlisting

IP Allowlists enable customers to control how data flows into or out of their SaaS environments.

Outbound Traffic

Outbound traffic is controlled via allowlist of IP addresses. Only HTTPS traffic is allowed to port 443.

Configuring IP Allowlists

To configure IP allowlists, customers must log a service request and follow the steps outlined in the **Cloud Operations** section of the *Oracle Utilities Cloud Services Administration Guide* to provide configuration details.

Chapter 6

Operational Guidelines

This chapter provides guidelines around the operation of Oracle Utilities cloud services once configured to a customer's requirements, including:

- Batch Job Submission
- Batch Job Scheduling
- Level of Service Batch Monitoring
- Code and Configuration Migration
- Customer Facing Alerts
- Data Fix with Plug-in Driven Batch
- Data Cloning/Subsetting
- Information Lifecycle Management / Archiving
- Limits on Time or Size of Certain Services
- Server Logs - Online and Batch
- Cloud Service Release Management
- Regression Testing through Utilities Testing Accelerator

Batch Job Submission

Much of the heavy processing in Oracle Utilities cloud services is done via batch processing, so it's critical to set up batch jobs to run in the most efficient way, with correct parameters.

Key rule: Run all batches with a **Commit Frequency** of 1. This setting is optimal given the way the application's hibernate entity cache works.

Batch jobs can be submitted in one of the following ways:

- Using a manual or timed submission of a batch job using the application base functionality
- Using Oracle DBMS Scheduler to schedule and submit jobs, using the
- Using an on-premise customer batch job scheduler

Refer to **Background Processes** in the *Administrative User Guide* or the online help for more information.

Batch Job Scheduling

Using Oracle DBMS Scheduler

The Oracle DBMS Scheduler is provided with Oracle Utilities cloud services and is the default job scheduling option.

Refer to **Batch Scheduler Integration** in the *Administrative User Guide* or the online help for more information.

However, note that access to the DBMS Scheduler via SQL using SQL Developer is not allowed in the cloud.

The Oracle Utilities Application Framework provides various REST services to directly interact with the DBMS scheduler. While batch operations use the underlying services of these APIs to interact with the DBMS scheduler, there are certain restrictions in calling the REST services from outside the system or via a third party integration. Please review the Batch Scheduler Integration for Oracle Utilities Application Framework technical reference paper in Oracle Support (Document ID 2196486.1) for more information.

A copy of the batch job stream definitions created in batch operations is kept in the Oracle Utilities Application Framework along with publishing them to DBMS scheduler. Any changes to such batch job stream definitions such as step changes, schedule changes, and so on must be done through the batch operations user interface. The Oracle Utilities Application Framework REST services should not be used directly for changing the definitions as this can cause the Oracle Utilities Application Framework job stream definitions go out of sync with the definitions inside the DBMS scheduler. This can result in a risk of losing historical data tracking, logging, risk of losing definitions and re-work etc. It is advised that only those Oracle Utilities Application Framework DBMS Scheduler REST services that pertain to handling job stream runs, not definitions be used from outside of the system or via third party integrations. Such services involve querying on the job stream run status, details, make adhoc submissions of the job stream run, canceling a job stream run and getting past job stream runs.

Cloud customers can use a set of REST APIs to set up and manage their batch scheduling using the Oracle DBMS Scheduler. The available DBMS scheduler APIs can be viewed by searching for Business Services that start with F1-DBMS.

Partial DBMS Updates

Partial updates to the Database Management Schedule (DBMS) chain are not allowed.

Pausing DBMS Jobs

Holding the execution of DBMS jobs is currently not supported.

Scheduling Batch Jobs

Scheduling a batch job to be effective in future is currently not supported.

Level of Service Batch Monitoring

The Oracle Utilities Application Framework provides a Health Check feature that reports on a configurable set of 'Level of Service' algorithms, which can check on various conditions of particular Batch Controls and Batch Job Streams. Results of the Health Check are given in this form (similar to http return codes):

- 200 - All Checks Successful (i.e. no warnings, no errors)
- 203 – Warning - Non-Critical Function Degraded
- 500 – Error - One or More Critical Functions Degraded

The Health Check can be brought up online at any time and can also be polled regularly from outside of the service via the Inbound Web Service F1-HealthCheckREST.

Level of Service Algorithms are available at the Batch Control as well as Batch Job Stream level, most of which can be set up with parameters to fine tune the error and warning conditions. The delivered algorithm types are:

Level of Service Algorithms – Batch Control:

- F1-BAT-ERLOS: Report Batch Jobs in Error
- F1-BAT-LVSVC: Evaluate Error Count and Time Since Completion
- F1-BAT-RTLOS: Compare Total Batch Run Time to Threshold
- F1-BAT-TPLOS: Compare Throughput to Threshold
- K1BATJNSXM: Batch Job not Started in X Minutes
- K1BATJPLNR: Job Processed Low Number Of Records
- K1BATLOSRTL: Batch Job ran too long (Relative Run)
- K1BATLOSTTL: Batch Job throughput too low (relative run)

Level of Service Algorithms – Batch Job Stream:

- K1BJSDJNSXM: Batch Job Stream Not Started in X Minutes
- K1BJSDJRTL: Job Stream Ran Too Long (Relative Run)

- K1-BJSD-LVSV: Batch Job Stream Has Failed

Refer to **Service Health Check** in the *Administrative User Guide* or the online help for more information.

Code and Configuration Migration

All automated configuration migration between environments should be done using Content Migration Assistant (CMA) provided with the Oracle Utilities Application Framework. An automation option for configuration migration, called the Process Automation Tool, is supplied through Cloud Service Foundation (CSF) included in all Oracle Utilities cloud services.

What is Cloud Service Foundation?

Cloud Service Foundation (CSF) is an add-on companion product for Oracle Utilities cloud services that provides automation for various generic processes, such as configuration migration. It is installed on top of the Oracle Utilities Application Framework in the same primary application installation in the cloud.

Infrastructure Process Types are provided out-of-the-box to support CMA Accelerator Load and CMA Migration processes (extract from one cloud environment and import into target). More information about CMA and Process Automation Tool is available in the relevant Administrative User Guide for each cloud service.

Refer to **Content Migration Assistant (CMA)** in the *Administrative User Guide* or the online help and **Process Automation Tool** in the *Oracle Utilities Cloud Service Foundation Administrative User Guide* for more information.

The Process Automation Tool provides several migration requests with the base package that orchestrate migration of objects based on appropriate criteria. Alternatively, you can create custom migration requests to select appropriate records based on specific business requirements. To see the migration requests provided by the product, log in to the relevant cloud service application and navigate to the migration request page by selecting **Admin**, then **Implementation Tools**, then **Migration Request**, then **Search**.

Customer Facing Alerts

Customers must be notified if certain errors or issues arise that will require attention. The most common example is batch alerts.

Customers need to know if important batch processes have failed or have not performed that work they were supposed to do. This could be critical for regular nightly batch processes but is also useful for daily or other scheduled batch processing. Instead of manual monitoring of important processes to make sure they worked as planned, the system has the ability to respond to a REST call that inquires about the overall status of the system processing. That service is called System Health Check and support both batch related checking (via the Level of Service algorithms that are plugged into Batch Controls) as well as batch job stream checking (via the Level of Service algorithms that are plugged into batch job stream definition).

In addition to the system health check service, an external probe can be set up that will invoke the REST call to the system from time to time and initiate an email notification to a configurable set of email addresses so that customers don't have to do it themselves manually.

A sample System Health Check Probe like this exists and is available to customers and implementers for educational purpose only. This sample is NOT supported by Oracle Support). See the Oracle Utilities System Health Check Probe document on Oracle Support (Document ID 2711546.1) for more information.

Data Fix with Plug-in Driven Batch

Inevitably there are fixes required that cannot be done by users through online tools, either due to the complexity of the issue or the volume of data.

Data fixes can often be performed using a Plug-in Driven batch which affords the following benefits:

1. Development and testing can be done in the development environment without the need for Service Requests
2. The fix will be applied through the application layer ensuring that it is well validated
3. Creating a plug-in driven batch is a relatively straight forward process that only requires SQL and scripting knowledge

For a more in depth look at how a plug-in driven batch can be used to execute a data fix, refer to **Plug-in Driven Background Processes** in the *Administrative User Guide*.

In some cases the ability to create plug-in driven batch approach will be constrained - for example you cannot change many status values directly via a Business Object update. In such cases, a service request will need to be logged with the required SQL update statement (and expected results - such as number of rows impacted). See **Cloud Operations** the *Oracle Utilities Cloud Services Administration Guide* for more information.

Data Cloning/Subsetting

Data subsetting is the process of moving data from the production environment to other environments (such as TEST or DEV environment). There are two processes for data subsetting: full-volume and partial-volume.

Full Volume - Clone

The TEST environment is a full-sized environment and can host a complete copy of the production database. A service request should be submitted to request the refresh of TEST from PROD. The two environments must be at the same version/patch level.

After the data refresh, any configurations and scripts that were in TEST and not in production will need to be re-applied to TEST.

The full volume clone can also be used to migrate a 'gold' configuration environment (DEV-sized) to a Test environment or to Production during implementation.

See **Cloud Operations** the *Oracle Utilities Cloud Services Administration Guide* for more information about submitting data cloning service requests.

Partial Volume - Subset

The DEV environment is not a full-sized environment and can only hold a subset of the production database. A test data subsetting tool is on the roadmap to pull a subset of the production data into DEV, but there is currently no tool available to create this subset. For now, all data in DEV will need to be manually generated.

Information Lifecycle Management / Archiving

Information Lifecycle Management (ILM) is configured by default, with all partitioning pre-configured at provisioning time. Partitions are date-based and monthly, and vary by type of data (very high volume Meter Data Management data such as Initial Measurement Data is set for just a three month lifetime, while most Customer Care and Billing transaction data is set for 48 months). Customers will need to run 'Add Partition' jobs (F1-ILMAD and F1-ILMSV) on a regular basis to ensure that new monthly partitions are created. Facilities for self-service removal of 'old' partitions is not yet available.

At this time, customers should simply ensure partitions are kept up to date for eventual archive.

Exclusions: WACS does not yet support ILM (i.e. W1 prefix tables are not partitioned).

Limits on Time or Size of Certain Services

Oracle Utilities cloud services have some time and data size limits.

In the online application, pages have a default timeout limit of two minutes. This is to prevent an endlessly running transaction, and is often an indication that there is an underlying performance issue (for instance a slow-running SQL) or inherent limit on how much work can be accomplished in the time period (such as attempting to generate a bill online for an account with hundreds of service agreements, which should be done in batch).

There is also a number of pre-set limits in BI Publisher for reporting:

Process/Report	Limit
Execution of SQLto build a report (Scheduled (offline) report output):	30 minutes
XML format report output	500MB
CSV format report output	1,000,000 Rows
Online/Browser report output	300MB
Scheduled (offline) report output	500MB

Server Logs - Online and Batch

The Oracle Utilities Application Framework creates log files for various processes such as web server, application server, and batch processes. Since log files may contain both technical and personal private information intended to be accessible by different people, the logs have been split with this technical and "application user" separation in mind.

Application users are able to view the web and application server user logs online by pressing the "Show User Log" button when running the application with "debug=true" in the URL (see **Debug Mode** in the *Administrative User Guide*). The batch logs can be downloaded on the **Batch Run Tree** page for each thread via the **Download stdout** and **Download stderr** links.

The application user logs accessible to the user through the above method may not contain certain internal technical details (for example, table structure, SQL, or other internal code-related logging).

Cloud Service Release Management

Oracle Utilities cloud services include software releases & updates, Infrastructure Updates, Maintenance Packs and Hotfixes which Customer are obligated to take in timely manner.

Software Releases & Updates

Staying current with software updates also directly benefits the customer:

- Access to latest functionality, means needing fewer customizations.
- New features are typically 'opt-in' so you can choose when to configure & use them
- Having all available patches applied, means discovering fewer problems yourselves, and makes it easier for customer support to replicate any issues you do experience
- Having all security patches applied, means less risk of security breaches.

Oracle Utilities plans for three releases a year: April, August, December

- Releases introduce new features and may make improvements and changes to existing functionality
- The releases are labeled using the format YY followed by A, B or C (A=April, B=August, C=December). For example, the August 2021 release is called '21B'.
- Product documentation is made available in advance of each update, including a New Feature Summary about one month before general availability.

Each release has a 'lifespan' of one year – i.e. end of life is one year after release,

Customers must operate a Generally Available release of the Oracle Cloud Service. General Availability and End of Life (EOL) dates are published in the Oracle Utilities Program Documentation.

A new release is applied as an update of each environment (non-production first for testing), retaining all configuration and data.

Infrastructure Updates

The underlying technology platform receives monthly updates

- This technology platform is independent of the actual software release (e.g. 21B) that a customer is running
  - This includes technologies like database, operating system, etc.
  - This also includes services like monitoring, alerting, logging, etc.
- All customers receive these platform updates every month
- No exceptions

Maintenance Packs (First 3-4 months of release)

For each release, Oracle Utilities provides bug fixes and patches.

- The standard mechanism for this is a Maintenance Pack (MP), which bundles up product fixes on the latest release on a monthly basis.
  - Uptake of Maintenance Packs is required by all customers still implementing the service (in other words, more than 4 months from go-live - we call this the 'implementation' lane)
  - Each Maintenance Pack is identified with the month and a sequential 'build' number
  - Release notes for each Maintenance Pack are published in My Oracle Support with detailed information about each fix – what's changed, how to verify it.
  - Maintenance Packs are pushed out to chosen environments on the 1st weekend (for early adopter environments) and 3rd weekend (for later adopter environments including 'Production') of each month.

Hotfixes for Critical Issues

For more time-sensitive critical application fixes there is a 'hotfix' mechanism that can provide product patches off the monthly maintenance cycle.

- Once Maintenance Packs stop for a release, we move to a 'hotfix-only' model (known as the 'cutover' lane and 'production' lane)
- Note that hotfixes are cumulative, so customers are not able to 'pick and choose'
  - If a customer needs a fix they will get all hotfixes available up to that point
  - If a customer does not need any patches after MPs end, they do not have to take any
- Hotfixes are not used to backport changes; they are intended to address only critical issues which customers have reported.
- Hotfixes are sometimes issued while Maintenance Packs are still coming out (and are then rolled into the next Maintenance Pack)
- Hotfixes will continue as necessary to the End of Life of the release (12 months from General Availability).

Oracle Recommended Approach to Planning for Updates:

- Oracle recommends a 'three lane' approach to support customers thru the different stages of their lifecycle:
- **Implementation** lane: During most of the implementation project – early adoption of new releases as they become available
    - Uptake and use latest release with latest functionality, full maintenance packs each month
- **Cutover** lane: When approaching Go-live – use the extended 'runway' on the target release to stabilize
    - Slow down rate of change and stabilize with any necessary hotfixes
- **Production** lane: For live production customers only
    - Uptake releases late, take only critical hotfixes, stay on 4-month update cycle

For more details, go to the Oracle Utilities Documentation page, and click the Release Schedule link for the relevant cloud service.

Regression Testing through Utilities Testing Accelerator

Oracle Utilities Testing Accelerator (UTA) comprises test automation accelerators for the automated testing of Oracle Utilities applications. It is a framework based on Java and Selenium for creating the web services and user interface automation scripts.

Oracle Utilities Testing Accelerator contains out-of-the-box product-specific components used to build new test flows in Oracle Utilities Testing Accelerator Workbench to test the Oracle Utilities applications. These out-of-the-box components correspond to specific business entities, such as business objects, service scripts, or business services used for interfacing with the application. Users can use these components as available or can extend them. Users can also create new components to be used to create flows.

Refer to the Oracle Utilities Testing Accelerator documentation for more information.

Chapter 7

Data Access and Analytics

This chapter provides guidelines related to data access and analytics, including:

- Analytics/BI Publisher
- Database Access
- Reports and Queries
- File Access - Cloud Object Storage

Analytics/BI Publisher

Oracle Utilities Analytics Visualization is included in the cloud service subscription, available via a separate URL for each environment.

BI Publisher is available and included in the service as a reporting/query tool.

SQL Web Developer is also available and included in the service for querying the database (see **SQL Developer Web** in the *Oracle Utilities Cloud Services Implementation Guide* for more information).

Data extraction is supported via the Generalized Data Extract functionality and or DataConnect (CCS & MSCS only) may be a starting point for extraction of data for a BI/ reporting tool such as Cognos.

Database Access

No direct access is permitted to the application database either through Toad, SQL Developer, or command line utilities. This also means that you **cannot** create new tables or related data including new Maintenance Objects, custom audit tables, and database links.

Query access is supported both in BI Publisher and SQL Web Developer (see **SQL Developer Web** in the *Oracle Utilities Cloud Services Implementation Guide*), which are available as part of the cloud deployment. For more information about BI Publisher, see http://www.oracle.com/technetwork/middleware/bi-publisher/documentation/index.html.

BI Publisher deployment includes a JDBC data source configured with credentials that allow access to read-only synonyms in the production schema.

Note that in some cases it may be feasible to create a custom zone in the application to provide online display to view data.

Reports and Queries

Reports and queries can be run using BI Publisher, which is included with the cloud service deployment. BI Publisher deployment includes a JDBC data source configured with credentials that allow access to read-only synonyms in the production schema. Note that there are several output formats, and we have found that PDF performs best for larger reports.

The cloud services also offer the option of using SQL Web Developer and Oracle Utilities Analytics Visualization.

File Access - Cloud Object Storage

All inbound and outbound file-based data is staged in Oracle Cloud Object Storage (a separate service that the customer must license).

Cloud Object Storage involves creation of a set of compartments and buckets (each compartment can have many buckets, and have child compartments), and the compartments are represented within the application as values for the File Storage Configuration (F1-FileStorage) extendable lookup. If the customer creates a new compartment, a new value needs to be specified in the extendable lookup, with OCID references to the user/tenancy/compartment. Once that is set up, the application can reference particular buckets as needed.

The format for Object Storage paths is as follows:`file-storage://<Extendable Lookup value for Compartment>/<bucket> - example: file-storage://OS-SHARED/CMA-Files`

Typically there are a few places where a 'path' to an Object Storage bucket can be specified, such as on batch job parameters, and some Master Configurations. Refer to the **Object Storage Setup** in the *Oracle Utilities Cloud Services Administration Guide* for more information. Refer to the Object Storage documentation for more information about Oracle Cloud Object Storage.

Uploading and Downloading File To and From Object Storage

There are two main options of exchanging files between Oracle Cloud Object Storage and the outside world.

- The Oracle Infrastructure Console User Interface which allows authorized users to upload or download files to and from object storage buckets.
- The object storage APIs which allow other applications to interact with object storage and exchange files as well as other actions.

When customers or implementers need to upload or download files in bulk, the option of the Oracle Infrastructure Console User Interface can be cumbersome.

There are Oracle and 3rd party tools that customers can install (for example the Oracle Storage Gateway) that offer integration solutions for file exchange with object storage.

The Oracle Integration Cloud (OIC) offers an Object Storage 'adapter' which allows OIC to move files/objects in or out of Object Storage. This uses the OIC REST Adapter.

Any other client that can make REST calls could also interact with Object Storage.

Part Two

Data Upload Support

This section provides data upload and implementation information relevant to the products included in Oracle Utilities Cloud Services. Most of the information is generic and applies to functionality that is available in each of the products as part of the Cloud Service Foundation. There are also some conversion tools that are documented with each specific product. (The specific products are referred to in this document as "products" or "applications".)

This section includes:

Chapter 8

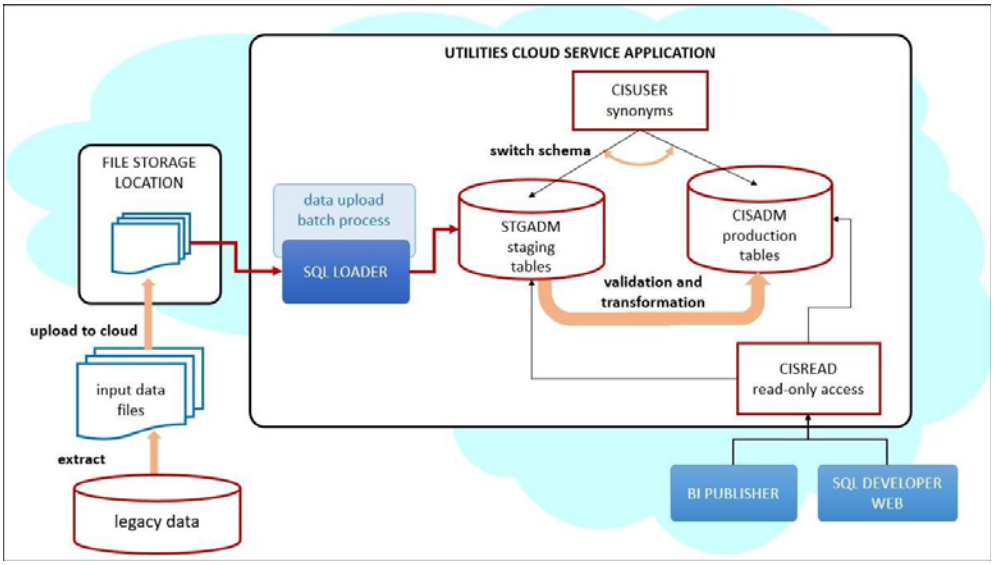
Data Upload Support Overview

This chapter includes:

Conversion Process Overview

The goal of the Conversion Process is to migrate data from a legacy application into a target environment, and to begin running the application in the cloud. Due to cloud-related technical restrictions, legacy data cannot be uploaded directly into the software-as-a-service (SaaS) database.

Legacy data must be extracted into file(s) and compressed. The data files are uploaded to the cloud file storage location and then loaded into the target "staging" tables using Oracle SQL Loader. The data is validated, transformed, and finally inserted into "production" tables. Oracle Utilities cloud services include various tools supporting ad hoc SQL inquiries and reconciliation reports on both staging and production data.



Implementation Effort

Implementers are expected to perform the following tasks for data conversion:

- Analyze the legacy data and decide what portion of it should be converted
- Map the legacy data to target Oracle Utilities Application Framework (OUAF) / Application data
- Develop legacy data extract process and produce input data files
- Adjust default data upload setup in OUAF / application, if needed
- Rehearse data upload and fine-tune configurations and/or legacy data extract, if needed
- Create reconciliation reports in BI Publisher
- Use uploaded data to try the subsequent conversion flow(s); bring the end-to-end conversion flow to perfection
- Execute the final conversion data upload run, a.k.a. cut-over
- Execute the application's data conversion processes.
- Disable conversion activities in the environment

What Is in the Newly Provisioned Environment?

The production instance is available for conversion.

Conversion activities do not co-exist well with the rest of the implementation. The massive data uploads, table truncation, and switching schema could disrupt business configurations development and testing. The production environment is the best candidate for conversion.

In the newly provisioned instance, the staging area in the database is created according to application specifications. The BI Publisher instance and SQL Developer Web / Oracle REST Data Services are connected to production and staging data.

The environment contains pre-configured conversion data upload setup.

The default configurations are suitable for typical table volumes and common data formats. If your implementation does not include extremely large data volumes, special data formats, or other idiosyncratic requirements, the default setup can be used "as is".

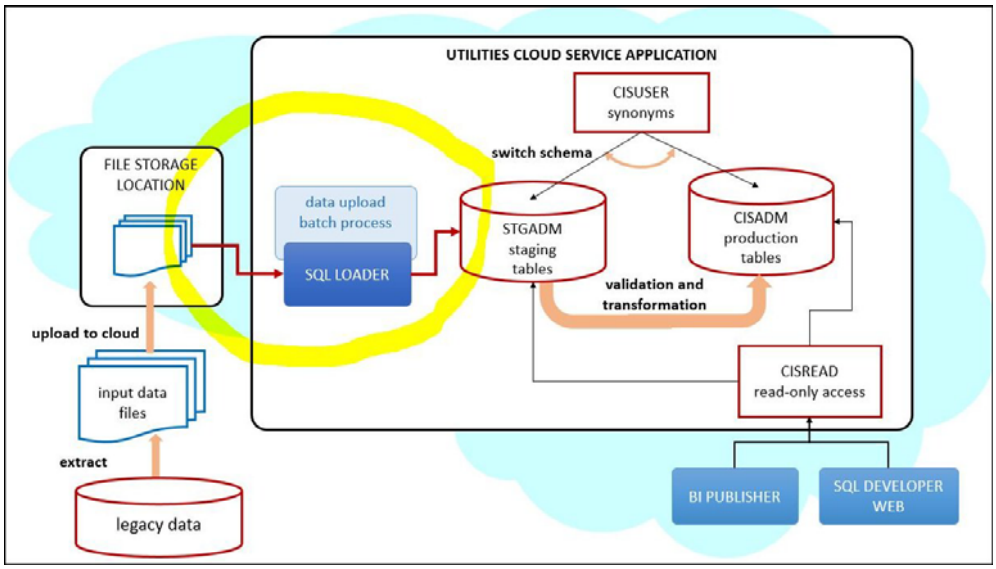
Chapter 9

Data Upload Support On Cloud

This chapter provides an overview of data upload support in Oracle Utilities cloud services, including:

Overview

The highlighted portion of the flow shown below is supported by Oracle Utilities Cloud Service Foundation (CSF). The legacy data extract and the input file creation belong to the implementation. The business application provides conversion validation and transformation processes, as well as the definition(s) of the staging area.



Provided by Cloud Service Foundation

Oracle Utilities Cloud Service Foundation (CSF) features metadata-driven configurable and customizable data upload with SQL Loader. It also provides support for basic database operations such as table clean-up (truncate), index enable/disable, and some others.

SQL Loader is an Oracle database utility that allows users to load data from external files into target DB tables. See Oracle DB SQL Loader Documentation for details.

The load of the Input Data File is performed according to the instructions recorded in a Control File. The Control File contains load options and parameters and also a list of data fields with formatting and parsing instructions.

For data upload on cloud, Control Files are pre-generated based on the metadata and conversion configurations and stored in the system.

Cloud Service Foundation allows users to generate control files, and it also provides a batch process that consumes the Input Data File(s), reads the pre-generated Control File and calls SQL Loader.

Cloud Service Foundation delivers the following:

- Batch processes. CSF batch controls are "generic", with no default value for the parameter that specifies target table or maintenance object (MO). These batch controls are used mostly for development and testing purposes. Applications are likely to supply "specific" batch controls for each target table or MO.
    - Batch Controls: Load Data into Table or Maintenance Object, Truncate Table or MO's Tables, Disable/Enable Indexes, Disable/Enable Triggers, Update Statistics, Populate Key Table, Cleanup Key Reference and XML resolution Tables, Generate Conversion Artifacts (bulk), and few others.
- Services accessible via online UI.
    - Switch Schema - executes the stored procedure that is re-directing the CISUSER synonyms between staging and production.
    - Generate Conversion Artifacts - creates input data file specifications and SQL Loader control files for specific converted objects. The artifacts are generated based on the metadata and according to the conversion data upload setup
- The instance of BI Publisher that is connected to the database with read-only access to the staging and production schema tables.
- The instance of SQL Developer Web that is connected to the database with read-only access to the staging and production schema tables.
- Predefined Configurations
    - Data delimiters and data format strings for date and date/time fields (Extendable Lookups)
    - Default Conversion Instructions (Conversion Task Types) for typical Table, MO and Key Table.
    - SQL Loader Control File fragments (Managed Content) for parallel and non-parallel load
    - Conversion Data Upload Master Configuration with default setup
    - User Groups and Template Users for conversion (suggested setup)

Provided by Applications

Each application comes with its own Conversion Accelerator that includes admin and system data for suggested upload configurations. Applications also provide a set of processes and tools to validate and transform the uploaded "staging" legacy data into real production form.

Application Conversion Tool

The Conversion Tool is usually comprised of processes, services, and configurations that support necessary legacy data validation and transformation.

The application connects to the @application_user schema in the database. This schema contains synonyms to the actual tables and views. Let's assume the legacy data has to be loaded into a table TABLEXXX. From the application perspective, the data upload process inserts the data into @application_user.TABLEXXX. The meaning of the synonym @application_user.TABLEXXX could be @production_schema.TABLEXXX or @staging_schema.TABLEXXX. Once the *switch_schema()* stored procedure is executed, the assumption is that the synonyms in @application_user schema are set to point to the "staging" tables, and the data upload may begin.

Possible approaches to the staging data upload into target tables are described below.

Approach: The legacy data is uploaded into a set of tables in the staging schema (STGADM).

Upon successful legacy data upload, the sequence of batch processes performs object-level validation and FK validation of the data in the staging tables, generates new keys and finally inserts the data into production tables.

Approach: The legacy data is extracted and loaded into Initial Sync table(s) in the production schema (CISADM). This approach is relevant under special circumstances such as extremely large data volume. Another reason for loading data directly into production could be a migration of existing Oracle Utilities application from on-premise to cloud, when the legacy data is actually a valid application data, conforming to target data formats and standards and there is no need for additional validations and key generation.

> NOTE: the Switch Schema has to be performed anyway in order to set the internal system indicator to Staging and allow the data upload

A custom control file should be created in order to insert the data into production tables..

Upon successful legacy data upload, the data can be further processed using regular application processes.

Application Accelerators

The accelerator usually contains suggested configurations for data upload support. It may include:

- Conversion Instructions for tables/maintenance objects with special data requiremenets (Conversion Task Types)
- Alternative Control File fragments and custom Control Files (Managed Content)
- Conversion Master Configuration

- Specific Batch Controls for each converted Table or MO, suggested batch job/ batch streams for suggested conversion activities orchestration
- Sample reconciliation reports
- Other system, admin, and/or configuration data

Chapter 10

Data Upload Design

There are several aspects implementation should consider when designing the legacy data extract processes and creating the Input Data Files. The data upload process is very flexible and configurable, and can be fine-tuned to address both application and client data specifics.

This chapter provides information about designing extract processes, including:

- Extract/Upload by Table or Maintenance Object
- CLOB Data in a Secondary File
- Multiple Data Files for Single Table or MO Upload

Extract/Upload by Table or Maintenance Object

The SQL Loader allows users to insert data into one or multiple tables from a single input file. Choose the more convenient option, depending on the structure of the legacy data (source), data volumes, and extract technique:

- **Table-level**. Extract file contains data for the single table. The data is loaded into a table in the OUAF/ application database.
- **Maintenance Object-level**. Extract file contains data for the entire object. The data is loaded into a set of tables that represent the corresponding Maintenance Object in the OUAF/ application database.

Both options are supported in Cloud Service Foundation. Generate the artifacts and review the differences in the specifications.

The table below illustrates the difference between Table and Maintenance Object data file:

Target Object	Table: CI_PER Data file contains records for a single table.	Maintenance Object PERSON: Tables: • CI_PER • CI_PER_NAME • CI_PER_ID ….etc Data file contains records for multiple tables within Maintenance Object. Table name serves as "record type" qualifier.	
Input Data File Layout	1234, IND, Doe,…	CI_PER	1234, IND, Doe,…
	5678, IND, Moon,...	CI_PER	5678, IND, Moon,...
	9063, BUS, ABC Corp,..	CI_PER	9063, BUS, ABC Corp,..
		CI_PER _ID	1234, SSN,72346781
		CI_PER _ID	5678, SSN, 87635241
		CI_PER _ID	9063, EIN, 09182835
		CI_PER _ID	9063, TID, 82528555
		CI_PER _NAME	1234, Doe, Mary
		CI_PER _NAME	5678, Moon, Barry

CLOB Data in a Secondary File

CLOB data can be supplied as part of the record in the "main" data file or as a secondary file. Once again, the decision should be made based on the source data volumes, extract techniques, and the availability of the CLOB data in most records.

- If most of the records have CLOB column(s) populated, and/or the CLOB field often contains large amount of data, it may make sense to use a secondary file.
- Otherwise, if the CLOB column(s) are rarely populated and/or the CLOB field rarely contains large amount of data, you may choose to include the CLOB data in the record.

  Note: If supplied as secondary file, the CLOB data file has to contain exactly as many records as the main file. This means that a line has to be added even for empty CLOB fields.

Both options are supported. The definition is controlled by the Conversion Instruction (Conversion Task Type).

Multiple Data Files for Single Table or MO Upload

The Cloud Service Foundation data upload process supports the upload into single target (table or maintenance object) from multiple data files. For example, instead of extracting a large Payment table into a single *payment.csv* file, you can split the extract into *payment1.csv*, *payment2.csv*, *payment3.csv*, and so on.

It is recommended to keep the file size under 2 gigabytes. The number of files is unlimited. Naming conventions apply. See the online help for more details.

Chapter 11

Preparing for Conversion

This chapter describes how to prepare an environment and legacy data for conversion with Oracle Utilities cloud services, including:

- Preparing Environment for Conversion
- Preparing Legacy Data Extract for Upload

Preparing Environment for Conversion

Preparing an environment for conversion involves the following:

- Set Up Conversion Security
- Prepare Environment for Conversion

Set Up Conversion Security

Conversion activities comprise massive data manipulations and database operations such as disabling / enabling indexes, truncating tables, and other operations. Whoever works on the conversion project deals with the real client's data and may have access to sensitive customer information. Therefore it is important to determine implementer's roles and responsibilities in advance, and to provide the user with the appropriate authorization level.

Use the pre-configured user groups *Conversion Administration*, *Conversion Development*, and *Conversion Operations*, along with the corresponding Template Users K1CNVADM, K1CNVDEV and K1CNVOPR. Alternatively, design and define your own conversion user authorization setup.

Prepare Environment for Conversion

- Enable conversion activities in the environment.
  - Run K1-CNVEN batch.
- Import the Conversion Data Upload Accelerator, if it was supplied by the application.
- Generate conversion artifacts.
  - To generate artifacts for all eligible tables and/or maintenance objects, submit a batch job for the K1-CNVAG batch control and use batch parameters to specify the scope for the generation: everything, Tables only or Maintenance Objects only.
    - As a result, new Conversion Task is generated for each Table and each Maintenance Object eligible for conversion. The artifacts are linked to the Conversion Tasks as attachments
  - To generate artifacts for an individual table or maintenance object, select **Admin**, select **Conversion Support**, and select **Generate Conversion Artifacts**. Choose "Table" or "Maintenance Object" and run the generator.
    - As a result, a new Conversion Task is generated for the selected table or maintenance object. The artifacts are linked to the Conversion Tasks as attachments.
  - Query Conversion Tasks that were created for various Tables and Maintenance Objects and explore the generated artifacts:
    - **Input Data File Specifications**. This file contains the detailed field by field formatting instructions and other notes about the expected contents of the input data file. Use these instructions when preparing the legacy data extract.
    - **Control File**. This file is used by SQL Loader during the data upload

        - **File List**. This file lists the name of the input files that has to be prepared for the data upload.
            - Multiple data files for a single object (Table or Maintenance Object) are expected if the data is being uploaded contains CLOB columns AND the upload is configured to load CLOB from secondary file.
    - Switch schema to redirect the application to the staging data area.
        - Use the menu to navigate to the generator by selecting **Admin**, then selecting **Conversion Support**.
    - Truncate tables in the staging data area to ensure that you will be uploading the data into clean empty tables.
    - Disable indexes in the staging data area. This is required because SQL Loader is not capable of implicitly disabling partitioned indexes during the data upload
    - Disable triggers in the staging data area.

The environment is now ready for the legacy data upload:

- Conversion is enabled
- SQL Loader Control Files have been generated, and
- Synonyms in the database schema point to the staging data area tables.

Notes:

- Conversion activities are possible as long as conversion is enabled in the environment. Once the legacy data is successfully migrated, you should disable conversion by running the K1-CNVDS batch. By doing this you set an internal indicator that is queried by conversion-related processes, such as switch schema, data upload, table cleanup, and index/statistics update. These processes will only run when conversion is enabled.

  Important: The Disable Conversion process should be executed ONLY ONCE right before the system is ready for go live. It is one-time event and is irreversible. Once disabled, conversion activities cannot be fully re-enabled as the assumption is that the re-enabling is happening while the application is running live in production. Enabling conversion after it has been disabled will result in the application running in the Incremental Conversion mode with its limitations.
- Switching the schema sets an internal flag that indicates whether the synonyms are pointing to "staging" or "production" area. The data upload is only allowed when the application is running in a "staging" mode.
- It is recommended to perform truncate operations at the maintenance object level as it will prevent leaving orphan records in the database. When truncating tables one by one always truncate child tables first.

Preparing Legacy Data Extract for Upload

The legacy data mapping and extract will vary from one customer to another. The files created as a result of the extract process should conform to the specifications generated above. The resulting data extract files should be:

- Created according to the specifications
- Named according to the naming convention (see the online help and the specifications for more details)
- Optionally, the file might be compressed with gzip or zip (see the online help for details)

Special Data Considerations:

Oracle Utilities Cloud Services provide support for Information Lifecycle Management (ILM) and Data Archiving.

All ILM-enabled objects contain the following fields:

- ILM Date (ILM_DT)
- ILM Archive Switch (ILM_ARCH_SW).

The ILM and Data Archiving functionality is controlled by the combination of these two fields.

- The ILM Date field is used in conjunction with partitioning to group data by age.
- The ILM Archive Switch is set by a background process when a record meets the business rules specific to the record's Maintenance Object that indicates the record is eligible to be archived.

See **Information Lifecycle Management** in the application's *Administrative User Guide* for more information about how these fields are used.

When preparing the legacy data extract for a target table, perform the following steps:

- Access the Oracle Utilities application, search for a Conversion Instructions Conversion Task Type (see **Conversion Task Types** in the *Oracle Utilities Cloud Service Foundation Administrative User Guide*) for the target table or maintenance object and review the input data specifications. Determine if the field list contains the fields named ILM_DT and ILM_ARCH_SW.
- In the data extract, populate the ILM_ARCH_SW field as follows:
  - Set the field with a value of "Y" for high-volume tables. In specific, the ILM_ARCH_SW field MUST be set to "Y" for the following tables used with Oracle Utilities Customer Cloud Service and Oracle Utilities Meter Solution Cloud Service:
    - D1_DVC_EVT (Device Event)
    - D1_INIT_MSRMT_DATA (Initial Measurement Data)
    - D1_USAGE (Usage Transaction)
  - Set the field with a value of "N" for all other tables
- For the ILM_DT field, the ILM Date Fields table below lists the recommended column whose value should be used to populate the ILM _DT for conversion data upload.

- Locate your target table name in the list and determine how the ILM_DT field should be populated
- If the table is not listed, please contact Oracle Utilities support.

ILM Date Fields

Table Name	ILM DT Initial Load
CI_TD_ENTRY	CI_TD_ENTRY.CRE_DTTM
F1_SYNC_REQ_IN	F1_SYNC_REQ_IN.CRE_DTTM
F1_OUTMSG	F1_OUTMSG.CRE_DTTM
F1_SVC_TASK	F1_SVC_TASK.CRE_DTTM
F1_OBJ_REV	F1_OBJ_REV.STATUS_UPD_DTTM
F1_BUS_FLG	F1_BUS_FLG.CRE_DTTM
F1_REMOTE_MSG	F1_REMOTE_MSG.CRE_DTTM
F1_STATS_SNPSHT	F1_STATS_SNPSHT.CRE_DTTM
F1_ERASURE_SCHED	F1_ERASURE_SCHED.STATUS_UPD_DTTM
F1_PROC_STORE	F1_PROC_STORE.STATUS_UPD_DTTM
F1_GNRL_AUDIT	F1_GNRL_AUDIT.CRE_DTTM
D1_ACTIVITY	D1_ACTIVITY.CRE_DTTM
D1_COMM_IN	D1_COMM_IN.CRE_DTTM
D1_COMM_OUT	D1_COMM_OUT.CRE_DTTM
D1_DVC_EVT	D1_DVC_EVT.CRE_DTTM
D1_COMPL_EVT	D1_COMPL_EVT.CRE_DTTM
D1_INIT_MSRMT_DATA	D1_INIT_MSRMT_DATA.CRE_DTTM
D1_USAGE	D1_USAGE.CRE_DTTM
D1_USAGE_EXCP	D1_USAGE_EXCP.CRE_DTTM
D1_VEE_EXCP	D1_VEE_EXCP.CRE_DTTM
D1_ACTIVITY	D1_ACTIVITY.CRE_DTTM
CI_ADJ	CI_ADJ.CRE_DT
CI_APPR_REQ	MIN(LOG_DTTM) on CI_APPR_REQ_LOG for given APPR_REQ_ID
CI_BILL	CI_BILL.CRE_DTTM
CI_BSEG	CI_BSEG.CRE_DTTM
CI_STM	CI_STM.STM_DT
C1_OFFCYC_BGEN	C1_OFFCYC_BGEN.STATUS_UPD_DTTM
CI_BILL_CHG	CI_BILL_CHG.START_DT
CI_CASE	MIN(LOG_DTTM) on CI_CASE_LOG table for given CASE_ID
CI_FA	CI_FA.CRE_DTTM

Table Name	ILM DT Initial Load
CI_ENRL	CI_ENRL.START_DT
CI_PAY_EVENT	CI_PAY_EVENT.PAY_DT
CI_PAY	CI_PAY_EVENT.PAY_DT
CI_MATCH_EVT	CI_MATCH_EVT.CREATE_DT
C1_USAGE	C1_USAGE.CRE_DTTM
C1_CUST_REL_REQ	C1_CUST_REL_REQ.CRE_DTTM
CI_CC	CI_CC.CC_DTTM or CI_CC.LETTER_PRINT_DTTM
CI_MR	CI_MR.READ_DTTM
C1_PA_RQST	C1_PA_RQST.CRE_DTTM
C1_CS_RQST	C1_CS_RQST.CRE_DTTM
C1_CS_REQ_ACCT	C1_CS_REQ_ACCT.?
C1_CS_REQ_CONT	C1_CS_REQ_CONT.CRE_DTTM
C1_CS_RQST_CONT_PROD	C1_CS_RQST_CONT_PROD.CRE_DTTM
C1_CS_REQ_PER	C1_CS_REQ_PER.CRE_DTTM
C1_CS_REQ_CVS_LOC	C1_CS_REQ_CVS_LOC.CRE_DTTM
C1_CS_REQ_PREM	C1_CS_REQ_PREM.CRE_DTTM
C1_MKTMSG_CHG	C1_MKTMSG_CHG.MKT_CHG_DT
C1_MKTMSG_PAY	C1_MKTMSG_PAY.MKT_PAY_DT
C1_MKTMSG_USG	C1_MKTMSG_USG.MKT_USG_DT

Chapter 12

Data Upload Steps

This chapter describes the steps involved in data upload, including:

- Upload Data into a Table or Maintenance Object
- Data Upload Orchestration

Upload Data into a Table or Maintenance Object

The data upload stage may begin only after conversion artifacts have been generated.

Review Input Data File Spec

- Retrieve the Conversion Task associated with the Table XXX
  - Navigate to **Admin**, then **Conversion Support**, then **Conversion Task Query** and select the "Table/Maintenance Object" **Query Option**.
  - Use search to populate either Table or Maintenance Object search criteria
  - From the search results, pick the latest entry.
- Load Conversion Task and locate a collection of Attachments.
- Find an attachment that represents *Input File Specification*.
- Click on the context menu to launch **Attachment** view the attachment contents.

Create Input Data File(s)

The specification defines the expected input data record format. The data fields are listed in the order it expected to appear in each record. For each field, the specification contains the data type, size and format. The specification also describes:

- Data delimiter
- Enclosing characters (to enclose a single blank that will represent empty non-nullable field)
- Date and date time formats
- CLOB data delimiter
- Expected name(s) for the secondary data file(s)

Extract the legacy data into a file according to the specification.

Each line in the file should represent a row in the target table. In the maintenance object-level extract, each record represents a row in one of the maintenance object tables and the first 30 characters in each line contains the table name.

If CLOB data is to be provided as secondary file, create CLOB data files.

Note: the SQL Loader treats invalid secondary file differently than a missing secondary file:

- If the secondary file is missing, the process will report an error.
- If the CLOB data in the secondary file is invalid, the CLOB field in the target table will be initiated into NULL or blank.

The input data file might be supplied uncompressed or compressed. Supported compressed formats include gzip and zip (See online help for details).

Switch Schema

Navigate to **Admin**, then **Conversion Support**, then **Switch Schema**, and select "Conversion" from the drop-down list and click **OK**.

Cleanup Target Table

Run the K1-SCLTB batch process, specifying the target table or maintenance object as a parameter.

Upload Data

Upload the input data file created above to the Object Storage location

Run the K1-CNVLD batch process, specifying the target table or maintenance object as a parameter. Detailed description of data upload parameters can be found in the online help.

Populate Key Table(s)

According to OUAF DB design standards, a corresponding Key Table exists for each table with system-generated or sequential primary key. Under normal circumstances, the key tables are populated when an application creates a "main" record. In a conversion situation, where the data is inserted directly into the database, there are two possibilities to populate the Key Table:

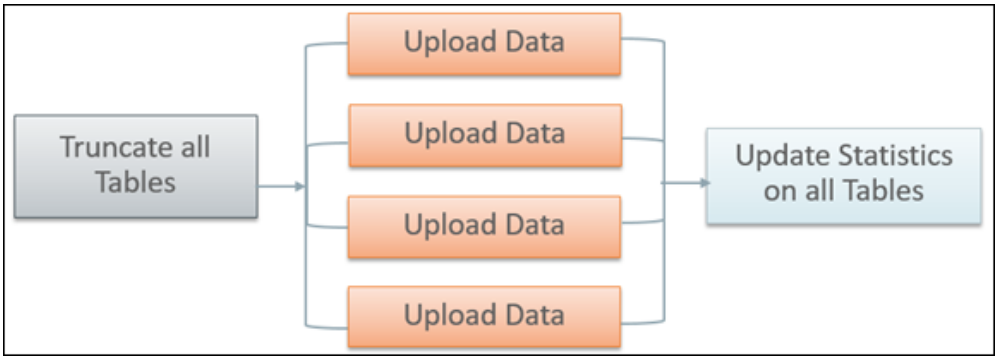
- Create an input data file for the Key Table and upload it using the same batch K1-CNVLD.
- Populate the key Table programmatically, by running K1-CPKTB after successful "main" table or MO data upload. This batch can be used for both Table and MO-level upload.

Data Upload Orchestration

The SQL Loader is running in multiple threads and therefore it is not performing table truncation before loading (command APPEND). Hence, the target tables should be truncated prior to the load. For better performance the indexes have to be disabled before the load and re-enabled/statistics updated after the load. The batch jobs can be organized into various chain structures, as shown in the examples below.

Single Table Upload

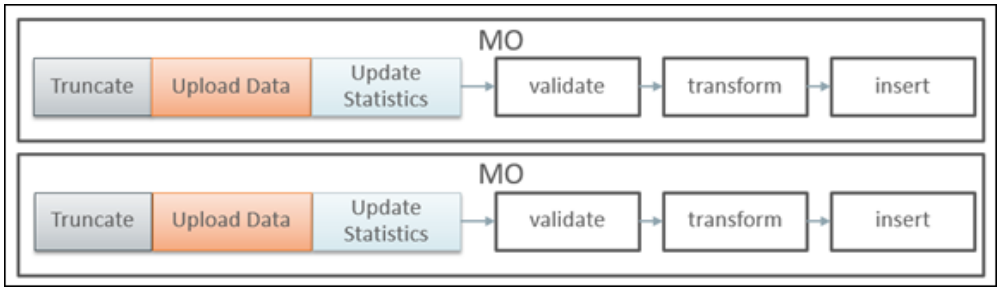
Multiple Tables or MOs Upload



There are multiple strategies to orchestrate the entire conversion run and to build the optimal sequence of the conversion processes. Below are some of the many possibilities:

- upload all legacy data extract files simultaneously, then run the subsequent validation and transformation processes for the converted object in a certain order of precedence, to preserve referential integrity
- begin the upload of very large tables in advance, so all upload is finished simultaneously, then validate & transform
- include legacy data upload batch(es) in the batch job chain for the target object
- upload some of the data by maintenance object, some table by table
- process maintenance objects end-to-end simultaneously, if there are no inter-dependencies

Full Conversion Chain per MO, Parallel Run



Upload All + Subsequent Validate/Transform MOs

Chapter 13

Customizing Data Upload

Conversion-related configurations define the expected extract file layout and the SQL Loader run-time upload options and parameters. SQL Loader's Control Files are generated based on these configurations.

The Batch Job/Batch Job Chain setup defines the overall orchestration of the conversion process flows.

This chapter describes customizations to the data upload process including:

- Why Customize
- When to Customize
- What to Customize
- How to Customize
- Tips and Important Mistakes to Avoid
- Sample Artifacts and Data Files

Why Customize

There are several reasons for customizing conversion configurations, including:

- fine-tuning data upload performance
- handling unusual data volumes
- marking additional table(s) as eligible for conversion
- reducing creation of unnecessary input files

When to Customize

The layout of the legacy extract files should be finalized as soon as possible, to provide enough time for the extract process development.

The setup of the batch job chains is less critical at the beginning of the project. The initial suggested setup is likely to be included in the application Conversion Accelerator. Adjust the initial setup after you've performed the trial uploads of the actual data, assessed the performance and figured the optimal flows.

What to Customize

Control File

The majority of the customizations affect the contents of the generated Control File and the corresponding input data file specifications. The configurations are stored on the Conversion Task Types that represent Conversion Instructions.

- Customizing the Control File's load options and parameters may improve upload performance
- Fully customized Control File allows you to use alternative record parsing and other advanced SQL Loader configuration techniques.
- When CLOB data is supplied as Secondary Files, the system is expecting the input data files to exist and be named following the specific naming convention.
- For example, if the table has multiple CLOB fields, for every CLOB field that was not excluded from conversion, the system is expecting the secondary file's name to be suffixed with _<*CLOB Field Name*>. See the online help for more details.

Data Delimiters and Enclosing Characters. Examine the default Conversion Instructions (Conversion Task Type) setup. Either select another delimiter from the existing list or add new value to the Extended Lookup.

CLOB as Secondary File? The indicator is defined on Conversion Instructions (Conversion Task Type).

Applicable when CLOB is supplied as Secondary File:

CLOB Columns Included in Conversion. By default, the control file is generated as if all CLOB fields are part of the converted data. The legacy data does not necessarily

contain data for all CLOB fields, hence there is no reason to create empty files. The list of excluded CLOB columns is is defined on Conversion Instructions (Conversion Task Type). Create new Conversion Instructions (Conversion Task Type) for the Table or MO with multiple CLOB fields and specify the exclusion list.

Control File "Header" - Load Options. A text stored as Managed Content. Contains the control file's fragment with options and load parameters. You can amend the options according to SQL Loader documentation. Examine the entries delivered with the product.

> **Note**: The text contains several substitution parameters prefixed with %. The substitution happens at generation time or at run time. Preserve them while creating a custom Control File header.
>
> If you wish to amend the load options and parameters only, create a new Managed Content entry. Modify default Conversion Instructions (Conversion Task Types) or create new ones and add Override Instructions to Conversion Master Configuration. Run Conversion Artifact Generator and create new customized Control File. See the online help for more details.

Custom Control File. A text stored as Managed Content and representing the entire Control File, including load options, parameters and the field list.

> **Note**: Preserve substitution parameters (see the note above). The input data file specifications are not generated when the Custom Control File is used. Make sure that the fields in the input data files correlates to the field's list in the custom Control File.

Additional Customization Items

Table's Conversion Eligibility. The table is considered eligible for conversion according to the indicator on the Metadata Table record. It is a system data and cannot be modified by the implementation. In order to make a non-converted Table eligible for Conversion, you should add an entry to the *Override Conversion Eligibility* list on the Conversion Master Configuration.

Conversion Orchestration. The suggested setup of the Batch Controls, Batch Jobs, and Chains is usually included in the application Conversion Accelerator. Adjust this setup by fine-tuning the number of threads, the chain structure(s) and other batch job parameters.

How to Customize

Configurations can be amended on several levels:

- To modify the configuration globally, amend the default Conversion Instructions (Conversion Task Type) that is referenced on *Conversion Data Upload* Master Configuration
- To modify the option globally for all tables, amend the default Conversion Instruction for Table (Conversion Task Type) that is referenced on *Conversion Data Upload* Master Configuration

- To modify the option globally for all maintenance objects, amend the default Conversion Instruction for MO (Conversion Task Type) that is referenced on *Conversion Data Upload* Master Configuration
- To modify the option for a specific table(s) or maintnance objects, create new Conversion Instruction (Conversion Task Type) and add the Override Instruction for Table or MO on *Conversion Data Upload* Master Configuration
- To make a non-converted table eligible for conversion, add it to the Override Conversion Eligibility list on *Conversion Data Upload* Master Configuration

IMPORTANT! Regenerate Conversion Artifacts to apply the configuration changes. Download the updated input file specifications.

Tips and Important Mistakes to Avoid

Issue	Details
Run the process against the right target.	The data upload only runs if the environment is pointing to the STAGING schema. Navigate to Conversion Support ' Switch Schema. On the popup screen the current schema is displayed. Make sure the current schema is **Staging**.
Provide data files according to the specifications Regenerate the artifacts after modifying the data upload configurations	SQL Loader loads the data according to the Control File. Input Data File Specifications describe what is expected from the input data file: • Names of the data files • Data format for all fields • Data delimiters to be used in the input data file Every time the configuration has changed the artifacts must be regenerated in order to keep the configurations and the input data specifications in sync.
Provide input data files with CLOB data IF NECCESSARY	Conversion Instruction defines whether CLOB data is provided as part of the main file or as a separate file. The system expects the data files to be provided according to this definition. Open the Input Data Specifications and read carefully. If the specification mentions that CLOB is to be provided as a secondary file, this is what Control File would inspect. If you wish to include CLOB data in the main file, verify that the Conversion Instruction is set correctly. **If the configuration was modified you must regenerate the artifacts.**

Issue	Details
Avoid creating unnecessary data files for CLOB columns	By default the system expects the data to be provided for all target table columns. If the table contains multiple CLOB columns AND the CLOB data is provided as a secondary file, it means one input data file per column. To exclude unnecessary CLOB columns for a table or maintenance object, configure Conversion Instructions using the *K1-ConvArtMultClobMOTaskType* or *K1-ConvArtMultClobTblTaskType* business object and specify the Override Conversion Instruction on the Master Configuration Regenerate conversion artifacts and examine the input data specifications after changing the configuration
Avoid truncating the entire staging data unintentionally	The K1-SCLTB batch process allows you to truncate a specific table or maintenance object in the STAGING schema. The K1-CLNTB batch process allows you to truncate a specific table or maintenance object in the PRODUCTION schema. If submitted without input parameter specifying a table or maintenance object, these batches will process all tables eligible for conversion. This means that all your staging data will be wiped out at once.
Clean up duplicate PK values before the data upload	Indexes and constraints are disabled during data upload in order to boost performance. De-duplication during the data upload is not supported out-of-the-box • SQL Loader direct path upload doesn't perform duplicate check • No direct database access means no possibility to modify data via direct SQL after the upload Keep track of the legacy data that has been already uploaded If you re-upload the same data again, always clean up the target table(s).

Issue	Details
The business configurations and admin data has to be finalized and populated in Production prior for legacy data upload Populate the legacy data extract with valid FK references to the admin/control data	**Once uploaded, the staging data cannot be "massaged"/modified thru direct SQL** (that because no database access is possible on cloud) Hence the overall conversion project steps are: • Design, test and complete business configurations. During this stage, multiple trial data uploads with dummy data could be performed • Populate admin data in Production • Create legacy data extract with valid admin data FK References • Upload data into staging tables
Key Tables are not populated implicitly	The Key Tables in the staging schema tables are not populated automatically when the legacy data is uploaded into "main" tables. Upload the data into Key Tables separately or use the batch program provided by Cloud Service Foundation.
Override Conversion Eligibility is supported on Table level only	The conversion eligibility is overridden for individual tables. Override the eligibility for all the tables that belong to the maintenance object if you decided to convert the entire maintenance object. **Note**: Overriding a table's conversion eligibility doesn't mean that the staging schema is automatically updated. It only means that the data upload processes will threat this table as a valid target table

Loading Data Directly into the Production (CISADM) Schema	The following configuration steps are required to load data directly into the Production (CISADM) schema tables:
	• Create a custom Control File for the target table.
	  • Generate the control file with default conversion instructions and copy the contents.
	  • Modify the INTO... clause to add 'CISADM.' in front of a target table name
	  • Create a new Managed Content entry. Copy the entire control file text and save.
	• Create a new Conversion Task Type for the target Table
	• Specify the new Managed Content as an Override Control File.
	• In the Data Upload Support Master Configuration, create an Override Table Instruction entry. Specify the target table and the new Conversion Task Type.
	• Generate Conversion Artifacts for the table.

Issue	Details

Loading Very Large Data Volumes	Avoid SQL-based conditions in the control file when loading very large data volumes. The default values and SQL-based conditions will cause SQL Loader to switch to the conventional path load which performs row-by-row inserts.
	
	The best results are achieved with a direct path load.
	
	More threads doesn't necessarily mean better performance. The optimal overall data load performance is achieved when the threads (and their corresponding SQL Loader processes) are targeting different partitions.
	
	Additional guidelines:
	• Partitioning by month is required for best performance
	• Load multiple months in parallel for best performance & scalability
	  • Start with ONE thread per month
	  • Increment number of threads per month. If performance does not increase, try a smaller increment or stay with your last best. For example: loaded 12 months data with 48 threads, 4 threads/month
	• Large data files are preferable
	  • Many small files have the overhead of spinning up new SQL*Loader process for each file
	  • Set longer SQL timeout on the data upload batch process
	• Disable indexes before loading
	• Rebuild indexes after direct path load
	• Reduce or stop the activities in the environment when performing the massive data upload

Sample Artifacts and Data Files

To assist implementers with the conversion and data upload process, multiple sample artifacts and data files are available. The sample files are provided with your cloud service documentation. The samples illustrate various data upload scenarios for table- and MO-level upload. Within the master samples zip file, there are multiple zip archives, each of which contain the following:

- Control file, generated
- Input Data File Specification, generated
- Sample Data File, created according to the specification

The table below provides more details on each of the sample artifacts available.

Target Object	Sample Description
Interval Data Set (INT_DATA_SET)	Regular maintenance object, CLOB field as a secondary file. Configuration: Conversion Task Type K1-CNV-MO Multiple data files (3)
MO Customer Contact (CUST_CONTACT)	Regular maintenance object, CLOB fields in the main file. Configuration: same as Conversion Task Type K1-CNV-TABLE, but the *CLOB as Secondary File* indicator set to false.
Table Meter Read (CI_MR)	Regular table, CLOB field as a secondary file. Configuration: Conversion Task Type K1-CNV-TABLE.
Table Adjustment (CI_ADJ)	Regular table, CLOB field in the main file. Configuration: same as Conversion Task Type K1-CNV-TABLE, but the *CLOB as Secondary File* indicator set to false.
Table Initial Sync Request (F1_SYNC_REQ_IN)	Table with Multiple CLOBs as secondary files. Configuration: For table with multiple CLOBs, the special Conversion Task Type was created based on the K1-ConvArtMultClobTblTaskType business object. Override Control File (Managed Content) was created and used as a custom Control File. Review the sample and note that there is a conditional input data selection. Only records with BO = W1-CompositeSyncReqGISAsset would be uploaded. A custom Control File is necessary if you have a requirement to manipulate the data during upload. Input Data File Specification: Since the Control File is fully custom, including the field list, the generated specification is describing expected file name(s) only. The data field formats, delimiters, sizes, and any other information related to the Input Data File layout should be determined based on the custom Control File.

Part Three

File-Based Integration

The next three chapters describe how implementations can integrate and exchange information from Oracle Utilities Cloud Services to other applications and vice versa through file based integration. File upload and download processes are used by some implementation for data connect, payment upload, letters extract, financial extracts, other business processes.

Oracle Utilities Cloud Services can exchange data files from one application to another by:

- Directly accessing Oracle Cloud Object Storage
- Integrating with Oracle Integration Cloud.

  For more information about this approach, refer to the Oracle Integration Cloud documentation at https://docs.oracle.com/en/cloud/paas/integration-cloud/

These chapters provides information on how Oracle Utilities Cloud Services, specifically Oracle Utilities Customer Cloud Service (CCS), access data files from Oracle Cloud Object Storage, including:

- Chapter 14: Object Storage Connection Management
- Chapter 15: File Export Sample Implementation
- Chapter 16: File Import Sample Implementation

Chapter 14

Object Storage Connection Management

This chapter outlines how to manage connections between Oracle Utilities cloud services and Oracle Object Storage, including:

- Oracle Object Storage Setup
- Oracle Utilities Cloud Service Configuration for Object Storage Connection
- Register API Key to Oracle Cloud Object Storage

Oracle Object Storage Setup

Before initiating a file transfer from an Oracle Utilities cloud service to Oracle Cloud Object storage or vice versa, you must first make sure that the basic administration tasks in Oracle Cloud Infrastructure related to Object Storage have been completed properly, and that the compartments and buckets where the import and export files are stored have been setup.

For more information on Oracle Cloud Object Storage setup for Oracle Utilities Cloud Services, including Oracle Utilities Customer Cloud Service (CCS), please see the *Oracle Utilities Cloud Services Object Storage Setup Guide*.

Oracle Utilities Cloud Service Configuration for Object Storage Connection

Authentication and connection between the Oracle Utilities cloud service and Object Storage enables batch processes to import and export files from and to Object Storage locations. Setting up this authentication requires the following in your Oracle Utilities cloud service:

- Creating API Keys
- Creating An Object Storage Connection

Refer to the *Oracle Utilities Cloud Services Object Storage Setup Guide* for details concerning setting up Keys and Key Rings, an object storage connection configuration, and registering the API key.

> **Note**: You can use the same API Keys and Object Storage Connection setup for both import and export process.

Creating API Keys

Create a key ring for the cloud service environment. The key ring should be active and should have a set of private/public encryption key pairs. This key ring will be included in the Object Storage Connection Configuration.

Creating Key Rings and Pairs

Authentication between the Oracle Utilities cloud service and Oracle Object Storage requires an API signature key. See **API Key Management** in the *Oracle Utilities Cloud Services Object Storage Setup* Guide for more information.

API key rings and key pairs are maintained in the **Key Ring** portal in the cloud service. This portal contains the following zones:

- **Key Ring**: Displays basic information about the key ring
- **Key Pairs**: Displays a list of key pairs for the current key ring

Key rings are defined by the following:

- **Key Ring**: A unique code for the key ring
- **Key Ring Class**: Signature (default)

- **Status**: The current status of the key ring.

  Note: Key pairs can only be generated for Active key rings. Once a key ring has been deactivated, you can no longer create key pairs for that ring.

- **Description**: A name for the key ring (this will be referenced in the File Location extendable lookup, see below)

Once the key ring is created, you need to generate and activate the key pair. Click **Generate Key** to generate a key pair for the key ring.

Key Pairs are defined by the following:

- **Sequence**: The sequence of the key pair (the order in which the key pair was created)
- **Creation Date/Time**: The date/time when the key pair was created
- **Key Status**: The current status of the key pair. Key pairs are inactive when first created.
- **Public Key**: Click **View** to open a dialog box containing the public key.
- **Action**: Click **Activate** to activate an inactive key pair.

Click **Activate** in the **Actions** column in the **Key Pairs** zone. A dialog box opens displaying the following message: "Warning(s): Activating a key assumes that you have already registered the public key with the appropriate third parties. Press **Cancel** to abort." Click **OK** to activate the key. The **Key Status** column will change to "Active".

Note: Be sure to register the API Key with Object Storage by copying the public key to Oracle Identification and Access Management. To copy the public key, click **View** in the **Actions** column in the **Key Pairs** zone, and select and copy the text in the **View Public Key** dialog box. Refer to **Register API Key to Oracle Cloud Object Storage** on page 14-4 for more information.

Creating An Object Storage Connection

Create an object storage connection via the File Storage Configuration extendable lookup (F1-FileStorage). This defines the Object Storage location where the files will be stored.

Creating File Storage Extendable Lookup Values

Apart from the authentication, the cloud service also needs information about the Object Storage locations to be used. Object Storage locations are defined by values in the File Storage Configuration (F1-FileStorage) extendable lookup. These file storage configurations will be referenced by the batch processes that will import or export records.

Values for the File Storage Configuration extendable lookup are defined by the following:

- **Value**: A unique code for the extendable lookup value. This value will be referenced as a batch control parameter value.
- **Description**: A description of the extendable lookup value
- **Status**: The current status of the value. Select "Active".
- **File Storage Details**: This section defines details for the object storage location, including:

- **File Adapter**: The type of file adapter for the location. Select "Oracle Cloud Object Storage".
- **User**: The user Oracle Cloud ID (ODIC) for the object storage location
- **Tenancy**: The tenancy Oracle Cloud ID (ODIC) for the object storage location
- **Compartment**: The compartment Oracle Cloud ID (ODIC) for the object storage location
- **Namespace**: The namespace for the object storage location
- **Key Ring**: The Key Ring you created earlier
- **Region**: The region of the object storage tenancy for the connection (Values for this field are defined in the F1_REGION_FLG lookup.

Refer to **External File Storage** in the *Oracle Utilities Application Framework Administrative User Guide* and the *Oracle Utilities Cloud Services Object Storage Setup Guide* for more information.

Register API Key to Oracle Cloud Object Storage

Once the key ring and key pair have been created in the Oracle Utilities cloud service, copy the public key from the key pair and add the public key to the **User API Key** in Oracle Identification and Access Management (IAM) See the **User API Keys** section in the **Security and Access Management** section of the **Managing Object Storage** chapter in the *Oracle Cloud Infrastructure Services* documentation.

Chapter 15

File Export Sample Implementation

This chapter provides an example of implementing a file export process, which includes:

- Creating a File Export Batch Process
- Configuring the Export Process

Creating a File Export Batch Process

Oracle Utilities cloud services provide a way to extract system information into a file using a file-export batch process. This process can be configured to export and store extracted files in an Object storage location.

Oracle Utilities cloud services provide a sample Batch Control (F1-PDBEX) which supports file export functionality and which can be used as 'template' to implement custom file export functionality as needed by specific implementations. Along with this out of the box batch control, cloud services also provide related objects (such as scripts.) that also can be used as templates while creating custom export processes.

Please note that the main data extraction logic lies within the script as described below. Customer can look at and copy the F1-GenProcEx script to implement their own data extraction logic.

This guide will follow a "bottom up" approach regarding the creation and configuration of cloud service data and objects to facilitate the file export process. The first step is to create a data area and script using that data area. Other objects for the file export will be created next.

For this sample implementation, we will export Premise information from Customer Cloud Service to Object Storage. This involves creating two algorithms (one for file export and one for selecting records to export) and a batch control.

To create a File Export Algorithm:

1. Create a data area that defines the schema for holding premise information.
2. Create a Plug-in script with the *Batch Control - Process Record* Algorithm Entity. This script may be based on the F1-GenProcEx script and modified as needed. Make sure to use the data area created above to hold premise information.

   The script will be used by the algorithm that will perform the file export.
3. Create an Algorithm Type similar to F1-GENPROCEX, based on the plug-in script created in the last step.
4. Create an algorithm based on the algorithm type.

To create a record selection algorithm:

1. Duplicate the F1-GENPROCSR algorithm to create a custom algorithm that will be used for selecting records.
2. Update the algorithm's parameters.
   - Use the **SQL** parameter name to define the new query for retrieving records from the cloud service database.
   - Keep the **Batch Strategy** parameter as *THDS*.
   - Define the key field on the query under the **Key Field** parameter.

To create a batch control:

1. Duplicate the F1-PDBEX batch control to create a new batch control. Navigate to **Algorithms** tab on the batch control and replace the existing algorithms with the file export and record selection algorithms created above.
2. This newly created batch control contains parameters for file storage that must be modified as described in the following section.

Configuring the Export Process

This section describes the setup needed to enable export processing, including establishing communication between the Oracle Utilities cloud service and Object Storage, configuring batch parameters, and testing the export process.

- Setting Up Communication Between Cloud Service and Object Storage
- Configuring File Export Batch Parameters
- Testing the Export Process

Setting Up Communication Between Cloud Service and Object Storage

The export process requires authentication and communication between the Oracle Utilities cloud service and Object Storage. See Chapter 14: Object Storage Connection Management for more information about setting up this communication and authentication.

Configuring File Export Batch Parameters

The next step is to configure the following parameters of the file export batch control created earlier:

- **File Name**: the name of the exported file
- **File Path**: the path to file location in Object Storage. The format for the path is as follows:

  file-storage://<File Location>/<Bucket>

  where:

  - **<File-Location>**: The File Storage Configuration extendable lookup value defined for that file. This will include the compartment identification.
  - **<Bucket>**: The object storage bucket in the compartment that is defined as part of the File Storage Configuration extendable lookup value.

  For example, the "File-Export" bucket in a compartment that is referenced in the "AB-Export" File Storage Configuration extendable lookup value can be referenced as:

  "file-storage://AB-Export/File-Export"

- **Other Parameters**: The batch control supports other optional parameters, including:

  - **XML Root Name**: used to declare the name of root-node in generated xml (when exporting files in xml format)
  - **File Delimiter**: used to define the delimiter used in delimited files

  Refer to the batch control for information about other parameters.

Testing the Export Process

The last step is to test the export process.

To test the export process:

1. Run the file export batch process.
2. Navigate to the **Batch Run Tree** portal to verify the job has run successfully.
3. Navigate to targeted bucket inside Object Storage and verify the exported file.

Chapter 16

File Import Sample Implementation

Oracle Utilities cloud services include a batch control that can be used as a template for creating batch controls to import data from a file to the application. This template batch control is called Plug-in Driven File Upload Template (F1-PDUPL).

To use this template, an implementation can duplicate the F1-PDUPL batch control and provide the required algorithm for the "File Upload" system event. The algorithm associated with the batch control is responsible for using provided APIs to read the content of the file and store the data in appropriate table(s) such as a staging table or the FACT table. (we will use the FACT table in this sample).

The plug-in scripts written to implement this type of algorithm must use the Groovy script engine version as the APIs are not accessible using the XPath scripting language. The sample plug-in scripts provided illustrate using the various available APIs to upload a flat file, xml file or a delimited file. Implementation can write their own plug-in scripts to handle their specific file upload needs.

The following steps summarize how to implement a new file import background process:

- Identifying Upload File Content Data
- Uploading File to Oracle Cloud Object Storage
- Creating a File Import Batch Process
- Configuring the Import Process

For more information on how to use Plug-in Driven background processing for import and upload, refer to the following section in the *Oracle Utilities Application Framework Administrative User Guide*:

Background Processes

Understanding Background Processes

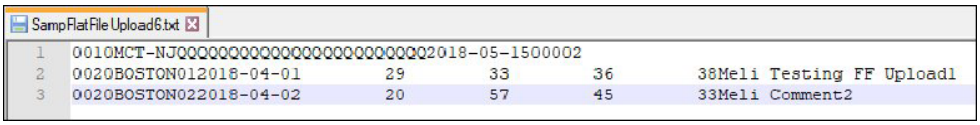
Plug-in Driven Background Processes

Uploading Records

Identifying Upload File Content Data

An important step in creating an import process is to define the format of the data files you plan to import, identify the different values in the file and map the data to fields in one or more appropriate tables in the application.

For example, the SampFlatFileUpload6.txt file has the following content



This sample data contains 'degree day' data, including a header record and two data records.

The header record contains the following components (the length of the field is shown in parenthesis):

- Record Type (4) (Value: 0010)
- Source (30) (Value: MCT-NJ)
- Date Transmitted (10) (Value: 2018-05-15)
- Number of Records (5) (Value 00002)

Individual data records have the following components:

- Record Type (4) (Value: 0020)
- Area (8) (Value: BOSTON01, BOSTON02)
- Degree Date (10) (Value: 2018-04-01, 2018-04-02)
- Degree Day (10) (Value: 29, 20)
- Minimum Temperature (10) (Value: 33, 57)
- Average Temperature (10) (Value: 36, 45)
- Maximum Temperature (10) (Value: 38, 33)
- Comments (254) (Value: Meli Testing FF Upload..., Meli Comment2)

Uploading File to Oracle Cloud Object Storage

Once you have a sample file in the correct format, you need to upload the file to the location in Object Storage from where you plan to upload and import your data.

Creating a File Import Batch Process

Creating a file import process involves creating the following components in the Oracle Utilities cloud service:

- Plug-In Script
- Algorithm Type and Algorithm
- File Upload Batch Control

Plug-In Script

The first step is to create a plug-in script that will process the data in the upload file. This plug-in script should be created for the "Batch Control - File Upload" **Algorithm Entity**.

You can use sample plug-in scripts provided or create a new plug-in script with logic required for reading the record and identifying each record detail to properly create the insert statements for storing the data in the appropriate application tables.

The sample plug-in scripts provided illustrate how to call the supplied APIs for processing different types of source data including fixed position, comma delimited, and XML formats.

Sample Plug-in Scripts for Algorithm Entity: Batch Control - File Upload

Plug-In Script	Description
F1UplSmplFlt	Sample Flat File Upload Script
F1UplSmplDlm	Sample Delimited File Upload
F1UplSmplXML	Sample XML File Upload

Note: The F1UplSmplFlt sample script is designed to work with the sample data above.

Algorithm Type and Algorithm

The next step is to create an Algorithm Type and Algorithm that use the plug-in script you created above. In this sample file import implementation, the source data uses the fixed position format, so the Algorithm Type and Algorithm should use the F1UplSmplFlt script.

To create a new algorithm:

- Create an Algorithm Type using the F1UplSmplFlt plug-in script.
- Create a corresponding Algorithm for the Algorithm Type created above.

File Upload Batch Control

The last part of the cloud service configuration is to create a batch control that will use the new algorithm to import the data. You can create a new batch control by duplicating a template batch control and reference the new algorithm. The base product includes the Plug-in Driven File Upload (F1-PDUPL) batch control which can be used as a template.

To create a new batch control:

1. Duplicate the F1-PDUPL sample batch control to create your own batch control.
2. In the **Algorithms** tab, define the algorithm created above for the "File Upload" **System Event**.
3. Define default parameters for the batch control, if required.

Configuring the Import Process

This section describes the setup needed to enable file import processing, including establishing communication between the Oracle Utilities cloud service and Object Storage, configuring batch parameters, and testing the process.

The following steps will enable file import batch processing to run and import the data from the import file to the appropriate application tables in the Oracle Utilities cloud service:

- Setting Up Communication Between Cloud Service and Object Storage
- Configuring File Import Batch Controls
- Testing the Import Process

Setting Up Communication Between Cloud Service and Object Storage

The export process requires authentication and communication between the Oracle Utilities cloud service and Object Storage. See Chapter 14: Object Storage Connection Management for more information about setting up this communication and authentication.

Configuring File Import Batch Controls

The next step is to configure the following parameters of the file import batch control created earlier:

- **File Name**: The name of the file to import.
- **File Path**: the path to the file location in Object Storage where import files will be located. The format for the path is as follows:

  file-storage://<File Location>/<Bucket>

  where:

  - **<File-Location>**: The File Storage Configuration extendable lookup value defined for that file location. This will include the compartment identification.
  - **<Bucket>**: The object storage bucket in the compartment that is defined as part of the File Storage Configuration extendable lookup value.

  For example, the "File-Import" bucket in a compartment that is referenced in the "INT-UPLOAD" File Storage Configuration extendable lookup value can be referenced as:

  file-storage://INT-UPLOAD/File-Import
- Refer to the batch control for information about other parameters.

Testing the Import Process

The last step is to test the import process using the sample data.

To test the import process:

1. Run the file import batch process.
2. Navigate to the **Batch Run Tree** portal and make sure the batch process ran successfully.
3. Check that the records have been added to the FACT table.

Part Four

Oracle REST Data Services

This section describes how to use Oracle REST Data Services with Oracle Utilities cloud services. This includes

- Chapter 17: SQL Developer Web
- Chapter 18: REST APIs

Refer to the Oracle REST Data Services documentation for more information about Oracle REST Data Services.

Chapter 17

SQL Developer Web

Oracle SQL Developer Web is a part of Oracle REST Data Services, and is the web-based version of Oracle SQL Developer that enables you to connect to an Oracle database and execute queries and scripts, create database objects, build data models, and monitor database activity.

Oracle Utilities cloud services use Oracle SQL Developer Web to connect to a cloud service database to execute read-only queries on various database schema objects.

Please refer the Oracle REST Data Services documentation for more information about using Oracle SQL Developer Web.

Users must be assigned to the "SQL Developer Web Online User" application role in order to use SQL Developer Web with Oracle Utilities cloud services. See **Pre-Defined Application Roles** in the *Oracle Utilities Cloud Services End User Provisioning Guide* for more information about application roles used with Oracle Utilities cloud services.

Access is provided to both CISREAD and STGADM database schemas to perform select/read-only queries.

- The CISREAD schema can be used to perform select and read-only queries of the production database.
- The STGADM schema can be used to perform select and read-only queryies of the staging database (used with data migration and conversion).

Chapter 18

REST APIs

Oracle REST Data Services also provides REST APIs that can be invoked via cURL to connect to an Oracle database and perform operations.

Users must be assigned to the "REST Enabled SQL" application role in order to use REST APIs with Oracle Utilities cloud services. See **Pre-Defined Application Roles** in the *Oracle Utilities Cloud Services End User Provisioning Guide* for more information about application roles used with Oracle Utilities cloud services.

The following is an example syntax for cURL command.

```
curl -i -X POST --user <username>:<password> --data-binary "<SQL
statement>" -H "Content-Type: application/sql" -k <Oracle REST URL>
```

Contact your system administrator for Oracle SQL Developer Web and REST service URLs.

No additional configuration is required to use Oracle SQL Developer Web or REST services.

Part Five

Product-Specific Integrations

This section describes product-specific integrations available for use with Oracle Utilities Cloud Services. This includes:

- Chapter 19: Customer Cloud Service Receipt Printing

Chapter 19

Customer Cloud Service Receipt Printing

This chapter describes how to configure Oracle Utilities Cloud Services to support integration with a Point Of Sale (POS) printer for printing of receipts related to the following payment transactions:

- Payment Event
- Payment Event Quick Add
- Payment Quick Add

Refer to the Oracle Utilities Cloud Services *Business User Guide* for more information about these payment transactions.

Configuration to support this functionality includes:

- Printer Installation
- Oracle Utilities Cloud Services Configuration

Notes:

- The instructions in this document are based on a specific sample printer, the Epson TM-H6000IV-DT Series.

Printer Installation

This implementation requires installation of a Javascript printer library and SDK on a server that is accessible from the user's web browser (the connection to the printer will be directly from the web browser rather than from the application server).

For example, when using the sample Epson TM-H6000IV-DT Series printer, the following Javascript library would be installed on the server:

```
epos-x.x.x.js
```

> **Note**: If the Oracle Utilities application or cloud service is using HTTPS, the Javascript library should be also be served over HTTPS to avoid mixed-content errors. SSL should be enabled on the printer. Use port 8043, to connect to the printer over HTTPS.

Refer to the printer documentation for specifics regarding installation and set up of the printer driver and library, as well as use of the printer's API library.

> **Note**: The SDK, driver and documentation for the sample Epson TM-H6000IV-DT Series printer can be found here.

Note the file name and path of the Javascript printer driver. You will need to reference this in the UI maps that generate the print dialog box from the payment transaction portals (see Configuring and Updating UI Maps and BPA Scripts).

Oracle Utilities Cloud Services Configuration

Configuration of Oracle Utilities Cloud Services includes the following:

- Configuring the Point of Sale Printer Integration Master Configuration
- Configuring and Updating UI Maps and BPA Scripts
- Configuring and Updating Tender Sources

Configuring the Point of Sale Printer Integration Master Configuration

To enable printing from the three payment transactions, you must define the BPA script used to launch the print dialog box from each type of transaction in the Point of Sale (POS) Printer Integration (C1-PointOfSaleIntegConfig) master configuration. The base product provides three processing types (one for each payment transaction that supports printing) and corresponding sample BPA scripts:

Processing Type	Sample BPA Script
POS Printing - Payment Event	Payment Event Add Print (C1-PEAddPrt)
POS Printing - Payment Event Quick Add	Payment Event Quick Add Print (C1-PEQAddPrt)
POS Printing - Payment Quick Add	Payment Quick Add Print (C1-PyQAddPrt)

Define a BPA script for each of the processing types you want to support.

Configuring and Updating UI Maps and BPA Scripts

The BPA scripts referenced on the Master Configuration each reference a UI map that's used to define the print dialog box. The base product provides sample UI maps for each of the sample BPA scripts listed above:

Sample BPA Script	Sample UI Map
Payment Event Add Print (C1-PEAddPrt)	Payment Event Print Control (C1-PayEventAddPrint)
Payment Event Quick Add Print (C1-PEQAddPrt)	Payment Event Quick Add Print (C1-PayEventQuickAddPrint)
Payment Quick Add Print (C1-PyQAddPrt)	Payment Quick Add Print (C1-PaymentQuickAddPrint)

Since the sample UI maps contain printer-specific information, the UI maps and the referencing BPA scripts must be copied and configured accordingly.

The UI maps must be updated to include specific information about the printer being used, including:

- Javascript Printer Library
- Printer Actions and Text Composition and Formatting

Javascript Printer Library

Each UI map must be updated to reference the file name and path of the Javascript printer library installed on the printer server (see Printer Installation).

Update the appropriate UI Map for each of the processing types you want to support. The printer library should be defined in a <script> element in the <head> element of the UI map's schema. Replace the ePOS-Javascript-SDK-path placeholder text in the UI map with the path of the ePOS.

Example: The sample below shows the location of the ePOS path placeholder in the UI map code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/TR/xhtml1/strict">
    <head>
        <title oraMdLabel="TITLE_CIPPEQPCTL"></title>
        <link href="cisDisabled.css" type="text/css"
rel="stylesheet" />
        <script type="text/javascript" src="c1/ccbUtil/
c1Utils.js"></script>
        <script type="text/javascript" src="ePOS-Javascript-SDK-
path"></script>
        <oraInclude map="C1-POSPayAmountFormatterJS"/>
        <script type="text/javascript">
```

Printer Actions and Text Composition and Formatting

Each UI map must also be updated to include the specific printer actions and appropriate text composition and formatting for each type of receipt you wish to print.

Use the printer library API to generate code for the printer actions and formatting you wish to use, and include that code in the UI map. For example, the Payment Event Quick Add Print (C1-PayEventQuickAddPrint) UI map includes the following functions for connecting to the printer, composing receipts, and printing receipts:

```
function printReceipt() {
// Connect to printer
    isPrintReceipt = true;
    connect();}

function connect() {
    // Get IP address from Tender Source
    id('tenderControlId').value = myPageData.TNDR_CTL_ID;
    oraInvokeSS('C1-GetPrntIP','getPrinterIP');
    var tenderSource = id('tenderSource').value;
    ipAddress = id('printerIPAddress').value;
    port = id('printerPort').value;
    if (ipAddress == '' || port == '') {
        // ERROR: IP and/or port missing
        main.showErrorMessage(11111, 80701, null, null, window,
[myPageData.TNDR_CTL_ID,tenderSource]);
        return;
    }
    ePosDev.connect(ipAddress, port, callback_connect);
}

function callback_connect(resultConnect) {
    if ((resultConnect == 'OK') || (resultConnect ==
'SSL_CONNECT_OK')) {
        if (isPrintReceipt) {
            ePosDev.createDevice('local_printer',
ePosDev.DEVICE_TYPE_PRINTER, {'crypto' : true, 'buffer' : false},
callback_createDevice);
        } else {
            ePosDev.createDevice('local_printer',
ePosDev.DEVICE_TYPE_HYBRID_PRINTER, {'crypto' : true, 'buffer' :
false}, callback_createDevice);
        }
    } else {
        // ERROR: Connection to the printer can not be made
        main.showErrorMessage(11111, 80702, null, null, window,
[resultConnect]);
    }
}

function callback_createDevice(deviceObj, resultCreateDevice) {
    retry++;
    if (resultCreateDevice == 'OK') {
        printer = deviceObj;
        retry = 0;
        if (isPrintReceipt) {
            composeReceipt();
            printer.addFeedLine(8);
            printer.send();
        } else {

printer.EndorsePrinter.addTextFont(printer.EndorsePrinter.FONT_B);
            composeEndorsement();
            printer.EndorsePrinter.send();
        }
        disconnectPrinter();
    } else if (resultCreateDevice == "DEVICE_IN_USE" ){
```

```
        if (retry < 5 ) {
            setTimeout(function () {
                ePosDev.createDevice('local_printer',
ePosDev.DEVICE_TYPE_PRINTER, {'crypto' : true, 'buffer' : false},
callback_createDevice);
                }, 3000);
        }
    } else {
       // ERROR: Connection to the printer can not be made.  Create
device error.
        main.showErrorMessage(11111, 80703, null, null, window,
[resultCreateDevice]);
    }
}

function composeReceipt() {
    printer.addTextAlign(printer.ALIGN_CENTER);
    printer.addText(myPageData.USER_ID);
    printer.addText(' ');
    printer.addText(myPageData.TNDR_CTL_ID);
    printer.addText(' ');
    printer.addText(main.convertInternalDateToLo
    cal(myPageData.PAY_DT));
    printer.addText('\n\r\n');
    var receiptList = main.model.getList('PRINT');
    // Set max width of the printer output
    setMaxWidth(isPrintReceipt);
    // Determine maximum payment amount length. This is used for
formatting the payment amount
    setMaxPayAmountLength(receiptList,myPageData.CURRENCY_CD);
    for (var x = 0; x < receiptList.elements.length; x++) {
        var elem = receiptList.elements[x];
        if (elem.PAY_EVENT_ID != '' || elem.ACCT_ID != '' ||
            elem.PAY_AMT != 0 || elem.TENDER_TYPE_CD != '') {
            printer.addTextAlign(printer.ALIGN_LEFT);
            printer.addText(elem.ACCT_ID);
            printer.addText(' ');
            printer.addText(elem.ACCT_CHECK_DIGIT);
            printer.addText('\n');
            var tndrDesc = '';
            var parameters = {
                FK_REF_CD: 'C1-TNDTY',
                FK_VALUE1: elem.TENDER_TYPE_CD
            };
            var result = oraGetFkRefInfoFromServer(parameters);
           if (result != null && result != '' && result.INFO_DESCR
!= null) {
                tndrDesc = '  ' + result.INFO_DESCR;
            }
           // Determine if payment amount will be on the same line
as tender type
            var isWrap = shouldWrap(tndrDesc);
            printer.addText(tndrDesc);
            if (isWrap) {
                printer.addText('\n');
            }
            var payAmt =
main.convertInternalMoneyToLocal(elem.PAY_AMT,
myPageData.CURRENCY_CD);
          // Pad the payment amount with the appropriate spaces to
ensure that amounts are right-aligned
```

```
            printer.addText(getPayAmountPadding(tndrDesc, payAmt,
isWrap) + payAmt);
            printer.addText('\n');
      }

}
}
```

Refer to the printer documentation for specifics regarding use of the printer's API library.

> **Note**: The SDK, driver and documentation for the sample Epson TM-H6000IV-DT Series printer can be found here.

Refer to the Payment Event Quick Add Print (C1-PayEventQuickAddPrint) UI map for additional sample functions and text composition examples.

Configuring and Updating Tender Sources

You must also update the Tender Sources used for the payment transactions you need to support.

The functions defined in the UI maps access the printer library (installed on the printer server, see Printer Installation) via the printer's IP address and port number.

> **Note**: The UI map derives the printer IP address and port from the Tender Source using the Get Printer IP Address and Port (C1-GetPrntIP) service script.

The printer's IP address and port number should be specified in the **Printer ID Address** and **Port Number** fields on the Tender Sources used by the Tender Controls of the payment transactions for which printing is supported.

> **Note**: The default port can vary by printer model. For instance, the default port for an Epson TM-H6000IV-DT is 8008.

Refer to the *Business User Guide* for more information about Tender Sources.