

Oracle Utilities Smart Grid Gateway

Configuration Guide

Release 2.0.0.2

E20535-03

October 2011

Oracle Utilities Smart Grid Gateway/Smart Grid Gateway Installation and Configuration Guide, Volume 1,
Release 2.0.0.2

E20535-03

Copyright © 2011 Oracle and/or its affiliates. All rights reserved.

Primary Author: Lou Prosperi

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

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

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

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

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

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

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Chapter 1

Overview.....	1-1
What Is This Book?.....	1-2
Other Documentation.....	1-3
Architecture Overview	1-5
Oracle Utilities Application Framework Configuration Tools.....	1-6
“Lite” Business Objects	1-6
Data Areas.....	1-6
Algorithms.....	1-6
Entity Naming Conventions	1-7
Configuration Process Overview	1-8
Basic Configuration Steps - Design Your Business Objects.....	1-8
Basic Configuration Steps - Create Your Business Objects	1-8
Basic Configuration Steps - Create Portals and Zones	1-9
Basic Configuration Steps - Create Master Data.....	1-9

Chapter 2

General Configuration	2-1
Installation Options	2-2
Base Time Zone	2-2
Installation Algorithms.....	2-2
Master Configuration	2-3
Hijri to Gregorian Date Mapping.....	2-3

Chapter 3

Setting Up Admin Data	3-1
Understanding Admin Data.....	3-2
General Admin Data	3-2
Device Management Admin Data.....	3-3
Device Installation Admin Data	3-4
Device Communication Admin Data	3-5
VEE Rule Admin Data	3-6
Admin Setup Reference Tables.....	3-7
Setup Sequence.....	3-7
Administration Setup and Maintenance	3-7

Chapter 4

Devices, Measuring Components, and Device Configurations	4-1
Understanding Devices, Measuring Components, and Device Configurations.....	4-2
Devices	4-2
Measuring Components.....	4-2
Device Configurations	4-6
Devices In Detail.....	4-7
Maintenance Object - D1-DEVICE.....	4-7
Meter Data Framework Base Package Device Business Objects	4-7

Configuration Options	4-8
Example Device - D1-SmartMeter.....	4-8
Device Configurations In Detail.....	4-10
Maintenance Object - D1-DVCCONFIG.....	4-10
Meter Data Framework Base Package Device Configuration Business Objects	4-10
Configuration Options	4-11
Example Device Configuration - D1-DeviceConfiguration	4-11
Measuring Component Types In Detail	4-13
Maintenance Object - D1-MCTYPE.....	4-13
Meter Data Framework Base Package Measuring Component Type Business Objects.....	4-14
Configuration Options	4-15
Example Measuring Component Type - D1-IntervalChannelTypePhysical	4-16
Measuring Components In Detail	4-18
Maintenance Object - D1-MEASRCOMP	4-18
Meter Data Framework Base Package Measuring Component Business Objects.....	4-19
Configuration Options	4-20
Example Measuring Component - D1-IntervalChannel.....	4-21
Configuring Devices, Measuring Components, and Device Configurations	4-23
Configuring Custom Devices.....	4-23
Configuring Custom Device Configurations	4-23
Configuring Custom Measuring Component Types	4-23
Configuring Custom Measuring Components	4-24

Chapter 5

Service Points and Device Installation	5-1
Understanding Service Points and Device Installation.....	5-2
Service Points	5-2
Contacts.....	5-2
Installation Events	5-2
Service Providers.....	5-3
Measurement Cycles	5-4
Service Points In Detail.....	5-6
Maintenance Object - D1-SP	5-6
Meter Data Framework Base Package Service Point Business Objects	5-6
Configuration Options	5-7
Example Service Point - D1-ServicePoint	5-8
Contacts In Detail	5-9
Maintenance Object - D1-CONTACT	5-9
Meter Data Framework Base Package Contact Business Objects.....	5-9
Example Contact - D1-Business	5-10
Install Events In Detail.....	5-11
Maintenance Object - D1-INSTLEVT	5-11
Meter Data Framework Base Package Install Event Business Objects.....	5-11
Example Install Event - D1-SmartMeterInstallEvent.....	5-12
Service Providers In Detail	5-14
Maintenance Object - D1-SVCPROVDR	5-14
Meter Data Framework Base Package Service Provider Business Objects	5-14
Configuration Options	5-15
Example Service Provider - D1-HeadEndSystem	5-15
Processing Methods In Detail	5-17
Maintenance Object - D1-PROCMEHTD	5-17
Meter Data Framework Base Package Business Objects.....	5-17
Configuration Options	5-18
Example Processing Method - D1-HowToCreateMCInformation	5-20
Configuring Service Point and Device Installation Objects	5-21

Configuring Custom Service Points	5-21
Configuring Custom Contacts	5-21
Configuring Custom Install Events.....	5-21
Configuring Custom Service Providers	5-22
Configuring Custom Processing Methods	5-22

Chapter 6

Measurement Data.....	6-1
Understanding Initial Measurement Data and Final Measurements	6-2
Initial Measurement Data	6-2
Final Measurements	6-9
Daylight Saving Time Support.....	6-10
Initial Measurement Data In Detail.....	6-13
Maintenance Object - D1-IMD	6-13
Base Package Business Objects	6-13
Configuration Options	6-15
Example Initial Measurement - D1-InitialLoadIMDInterval	6-15
Measurements In Detail	6-17
Maintenance Object - D1-MSRMT	6-17
Base Package Device Business Objects	6-17
Configuration Options	6-17
Example Device - D1-Measurement	6-18
Configuring Initial Measurements and Measurements	6-19
Configuring Custom Initial Measurements	6-19
Configuring Custom Measurements	6-19

Chapter 7

Validation, Editing, and Estimation	7-1
Understanding Validation, Editing, and Estimation	7-2
Overview of the Validation, Editing, and Estimation Process	7-2
VEE Rules and VEE Groups	7-2
VEE Roles	7-6
VEE Exceptions	7-6
VEE Groups In Detail	7-9
Maintenance Object - D1-VEEGROUP	7-9
Meter Data Framework Base Package VEE Group Business Objects	7-9
Example VEE Group - D1-VEEGroup.....	7-10
VEE Rules In Detail.....	7-11
Maintenance Object - D1-VEERULE	7-11
Meter Data Framework Base Package VEE Rule Business Objects	7-11
Configuration Options	7-13
VEE Eligibility Criteria In Detail.....	7-15
Maintenance Object - D1-VEEELIGCR	7-15
Meter Data Framework Base Package VEE Eligibility Criteria Business Objects	7-15
Configuration Options	7-16
Example VEE Eligibility Criteria - D1-VEEEligibilityCriteria	7-16
VEE Exceptions In Detail.....	7-17
Maintenance Object - D1-VEEEXCP	7-17
Meter Data Framework Base Package VEE Exception Business Objects	7-17
Example VEE Exception - D2-VEEException.....	7-17
Configuring VEE Groups, Rules, Eligibility Criteria, and Exceptions.....	7-19
Configuring Custom VEE Groups	7-19
Configuring Custom VEE Rules	7-19
Configuring Custom VEE Eligibility Criteria.....	7-20
Configuring Custom VEE Exceptions.....	7-20
Creating Generic Utility VEE Rules	7-20

Chapter 8

Device Communication and Device Events.....	8-1
Understanding Device Communication	8-2
Activities	8-2
Communications	8-4
Completion Events.....	8-9
Understanding the Command Communication Process	8-10
Understanding Device Events.....	8-12
Device Events.....	8-12
Activities In Detail	8-15
Maintenance Object - D1-ACTIVITY.....	8-15
Meter Data Framework Base Package Activity Business Objects.....	8-15
Configuration Options	8-17
Example Activity - D1-RemoteConnect	8-18
Inbound Communications In Detail.....	8-20
Maintenance Object - D1-COMMIN.....	8-20
Meter Data Framework Base Package Inbound Communication Business Objects	8-20
Example Inbound Communication - D1-CommInLite	8-21
Outbound Communications In Detail.....	8-22
Maintenance Object - D1-COMMOUT	8-22
Meter Data Framework Base Package Outbound Communication Business Objects	8-22
Example Outbound Communication - D1-CommOutLite	8-23
Completion Events In Detail	8-24
Maintenance Object - D1-CEVT	8-24
Meter Data Framework Base Package Completion Event Business Objects.....	8-24
Example Completion Event - D1-ConnectDevice	8-25
Device Events In Detail.....	8-26
Maintenance Object - D1-DVCEVENT.....	8-26
Meter Data Framework Base Package Device Event Business Objects	8-26
Example Device Event - D1-DeviceEvent	8-27
Configuring Device Communication and Device Event Objects	8-28
Configuring Custom Activities	8-28
Configuring Custom Inbound Communications	8-28
Configuring Custom Outbound Communications.....	8-28
Configuring Custom Completion Events	8-28
Configuring Custom Device Events.....	8-29

Chapter 9

Batch Processing.....	9-1
Meter Data Framework Base Package Batch Controls	9-2
Synchronization Request Batch Controls.....	9-4
Meter Data Framework Batch Processing Guidelines	9-5

Chapter 10

Integrating Oracle Utilities Smart Grid Gateway with Other Systems	10-1
Understanding Communications Processing.....	10-2
One-Way Communications: Import of Usage and Events	10-2
Two-Way Communications: Meter Commands	10-4
Invoking Meter Commands From an External System.....	10-7
Defining External System as a Service Provider.....	10-7
Exposing Command Activity Business Objects	10-8
Sending Command Responses to the External System	10-9
Configuring A Middleware Communication Layer	10-11
Creating Custom Smart Grid Gateway Adapters	10-12
Adapters for Importing Usage Readings and Device Events	10-12

Adapters for Issuing and Initiating Commands	10-16
Creating Custom Commands	10-26
Initial Measurement and Device Event XML Formats	10-31
Initial Measurement Data XML Format	10-31
Device Event XML Format	10-40
Chapter 11	
Sample Implementation.....	11-1
Implementation Description and Requirements	11-2
Implementation Steps	11-3
Design and Create Business Objects	11-3
Create Admin Data	11-6
Create Master Data	11-15
Appendix A	
Measurement Services	A-1
Appendix B	
Glossary	B-1
Appendix C	
Base Package Configuration Objects	C-1
Base Package Data Areas	C-2
Base Package Extendable Lookups	C-5
Appendix D	
The Oracle Utilities Smart Grid Gateway “Generic” Adapter	D-1
Overview	D-2
Oracle Utilities Smart Grid Gateway Generic Adapter Data	D-3
Required Data	D-3
Demonstration Data	D-4
Configuring and Customizing the Generic Adapter Oracle Service Bus (OSB) Processes	D-11
Understanding the Generic Adapter Oracle Service Bus (OSB) Processes	D-11
Opening and Deploying the OSB Processes	D-21
Customizing the Generic Adapter Oracle Service Bus (OSB) Processes	D-24
FileParser Interface	D-25
FileParser2 Interface	D-26
Configuring and Customizing the Generic Adapter BPEL Processes	D-27
Configuration and Customization Process Overview	D-27
Editing Configuration Files	D-28
Customizing the BPEL Processes	D-32
Packaging and Deploying the Processes	D-33
Configuring BPEL Security	D-34
Emulating a Head-End System	D-36
Message Driven Bean Configuration	D-37
JMS Configuration	D-37
Message Driven Bean Configuration	D-38
Index	

Chapter 1

Overview

This chapter provides an overview of this configuration guide and an introduction to the Oracle Utilities Smart Grid Gateway application. This includes:

- **What Is This Book?**
- **Architecture Overview**
- **Oracle Utilities Application Framework Configuration Tools**
- **Configuration Process Overview**

What Is This Book?

This guide describes how to configure Oracle Utilities Smart Grid Gateway. It is intended for implementers and system administrators responsible for configuration and initial setup of the application.

Oracle Utilities Smart Grid Gateway is based on the Oracle Utilities Application Framework (OUAF). For information about using and configuring basic Framework functions, see the Oracle Utilities Application Framework documentation. This guide only covers configuration of functions specific to Oracle Utilities Smart Grid Gateway.

The body of this guide presents conceptual information to help you understand how the system works as well as how the various configuration options affect system functionality. Once you have an understanding of the system's capabilities, you can plan your data setup and design any customizations you want to implement.

When you are ready to implement your design, use the **Admin Setup Reference Tables** on page 3-7 in **Chapter 3: Setting Up Admin Data** to guide you through the setup process of admin data. This section lists each object that can be configured, defines any prerequisites for configuration.

Note: The sequence in which you configure system objects is very important. Admin Setup Reference Tables describes admin data dependencies and defines the order in which admin objects should be configured. By following this sequence carefully, you can streamline the configuration process and reduce the amount of time required for setup.

This guide includes the following chapters:

- **Chapter 1: Overview** (this chapter) provides an overview of the Oracle Utilities Smart Grid Gateway architecture and of the configuration tools and process used in implementing the product.
- **Chapter 2: General Configuration** provides an overview of some general configuration options used by the system.
- **Chapter 3: Setting Up Admin Data** describes the different types of admin data that must be set up and defined as part of implementing and configuring Oracle Utilities Smart Grid Gateway.
- **Chapter 4: Devices, Measuring Components, and Device Configurations** provides an overview of devices and measuring components and how they are used in the system, along with technical details concerning device-related maintenance and business objects.
- **Chapter 5: Service Points and Device Installation** provides an overview of service points and device installation-related objects and how they are used in the system, along with technical details concerning related maintenance and business objects.
- **Chapter 6: Measurement Data** provides an overview of initial and final measurement data and how it is used in the system, along with technical details concerning related maintenance and business objects.
- **Chapter 7: Validation, Editing, and Estimation** provides an overview of the validation, editing, and estimation process, along with technical details concerning related maintenance and business objects.
- **Chapter 8: Device Communication and Device Events** provides an overview of the device communication-related objects and how they are used in the system, along with technical details concerning related maintenance and business objects.
- **Chapter 9: Batch Processing** provides a list of the base package batch controls provided with the system.

- **Chapter 10: Integrating Oracle Utilities Smart Grid Gateway with Other Systems** provides a description of how Oracle Utilities Smart Grid Gateway can be integrated with other systems.
- **Chapter 11: Sample Implementation** provides a high-level description of a sample implementation.
- **Appendix A: Measurement Services** provides a list of base package measurement services use by VEE rules and functions.
- **Appendix B: Glossary** is a list of commonly used terms.
- **Appendix C: Base Package Data Areas** provides lists of base package configuration objects that can be leveraged during an implementation.

Other Documentation

This section describes other documentation provided with Oracle Utilities Smart Grid Gateway.

Installation Documentation

Installation documentation describes the steps involved in the installation and initial set up of the system, and includes the following documents:

- Oracle Utilities Smart Grid Gateway Quick Install Guide
- Oracle Utilities Smart Grid Gateway DBA Guide
- Oracle Utilities Smart Grid Gateway Installation Guide

User Documentation

User documentation provides conceptual information and procedures related to working with the various objects used in the system, and includes the following documents:

- Oracle Utilities Application Framework Business Process Guide
- Oracle Utilities Application Framework Administrator Guide
- Oracle Utilities Meter Data Framework User's Guide
- Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr User's Guide
- Oracle Utilities Smart Grid Gateway MV90 Adapter for Itron User's Guide

Adapter Configuration Documentation

Adapter configuration documentation provides technical information related to vendor-specific Smart Grid Gateway adapters, and includes the following documents:

- Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide
- Oracle Utilities Smart Grid Gateway MV90 Adapter for Itron Configuration Guide

Supplemental Documentation

Supplemental documentation provides technical information related to system administration tasks and include the following documents:

- Oracle Utilities Smart Grid Gateway Server Administration Guide
- Oracle Utilities Smart Grid Gateway Batch Server Administration Guide

Oracle Utilities Smart Grid Gateway adapters, such as the Adapter for Landis+Gyr and the MV90 Adapter for Itron use Oracle Service Bus (OSB) and Oracle Business Process Execution Language (BPEL) as middleware components. These tools are part of the Oracle SOA Suite. See the Oracle SOA Suite Documentation library (<http://www.oracle.com/technetwork/middleware/soasuite/documentation/index.html>) for more information about using these tools.

Embedded Help

Oracle Utilities Smart Grid Gateway, like all Oracle Utilities Application Framework applications, provides extensive internal documentation. For example, detailed descriptions of system objects are included in the objects' maintenance portals. The lifecycle of each business object is described on the Lifecycle tab and depicted in flow diagrams on the Summary tab. This information is extremely useful for implementers and system administrators.

Embedded help is provided for all non-obvious fields in most portals and zones. If a field has associated help text, a ? icon appears next to the field when the zone is displayed.

Online Help

Oracle Utilities Smart Grid Gateway also include context-sensitive help for all the user interface screens users will typically work with as they use the system. Online help contains conceptual information and procedures related to working with the various objects used in the system.

The online help is divided into the following three sections:

- Oracle Utilities Application Framework: Describes the features and functions of the application framework (F1)
- Oracle Utilities Meter Data Framework: Describes the features and functions provided in the meter data framework (D1)
- Oracle Utilities Smart Grid Gateway Adapters: Describes the features and functions provided in the smart grid gateway adapters (D3, D4, D5, etc.)

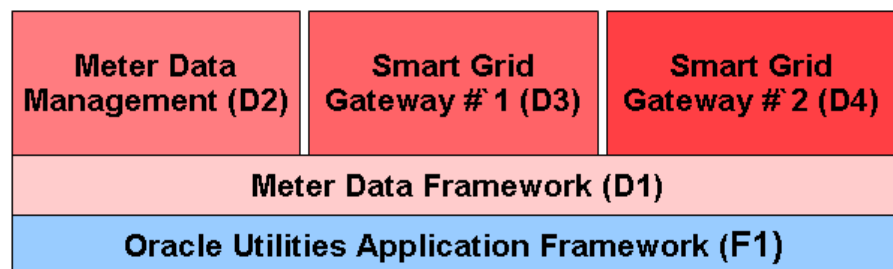
Architecture Overview

Oracle Utilities Smart Grid Gateway is used to maintain information about meters and the service points at which they are installed. The application provides means of recording measurements and events associated with meters in the field as well as the ability to issue commands to meters via head-end systems.

Oracle Utilities Smart Grid Gateway comprises the following functional areas:

- **Device Management:** the maintenance of physical meters in the field
- **Device Installation:** the maintenance of service points and the installation of meters in the field. This includes the means of registering outside systems to provide consumer-specific processing of meter events and activities
- **Device Communication:** the maintenance of communications with head-end systems, including import of usage and events, as well as two-way communications used in issuing meter commands.
- **Validation, Editing, and Estimation:** the maintenance of measurement data and the engine used to validate and modify that data as it comes in

Oracle Utilities Smart Grid Gateway is built upon the Oracle Utilities meter data framework, a framework that provides shared functionality used by Oracle Utilities Meter Data Management, Oracle Utilities Smart Grid Gateway, and other Oracle Utilities products. Oracle Utilities Smart Grid Gateway and the Oracle Utilities meter data framework are built atop the Oracle Utilities Application Framework.



Oracle Utilities Application Framework Configuration Tools

The Oracle Utilities Application Framework (OUAF) configuration tools can be used to create and customize system entities, such as business objects, portals, zones, and UI maps. Refer to the Oracle Utilities Application Framework configuration tools documentation for instructions on using these tools.

This configuration guide does not duplicate the concepts and procedures presented in the Oracle Utilities Application Framework configuration tools documentation; rather, it will identify the specific objects used by Oracle Utilities Smart Grid Gateway that can be configured and customized using the configuration tools, as well as application parameters and objects that can be managed within the application components themselves.

This guide assumes that all individuals responsible for system configuration and implementation will be familiar with the Oracle Utilities Application Framework and will have completed training on the Oracle Utilities Application Framework Configuration Tools.

The following sections discuss some specific topics related to the configuration tools.

“Lite” Business Objects

When a business object is read, the Framework dynamically constructs a SQL statement to retrieve the rows and columns associated with the business object's schema. If a process only needs only a small subset of a business object's elements, a “lite” business object that only references these elements can be used.

These “lite” business objects are used by the business processes (typically the construction of info strings) that only need a small subset of elements. Lite business objects are configured to never allow instances. In other words, they are only used to read existing instances of other business objects.

Later chapters in this book list the various “lite” business objects provided with the base package for each of the types of business objects described (devices, measuring components, service points, etc.)

Data Areas

As described in the Oracle Utilities Application Framework documentation, data areas provide a common schema location for re-used schema structures. Data areas exist solely to help eliminate redundant element declaration. For example, if you have multiple schemas that share a common structure, you can set up a stand-alone data area schema for the common elements and then include it in each of the other schemas.

Many of the base package schemas make use of data areas, and Oracle recommends that you take advantage of data areas where possible to avoid redundant data definition.

See **Appendix C: Base Package Data Areas** for a list of the base package data areas provided with Oracle Utilities meter data framework and Oracle Utilities Smart Grid Gateway.

Algorithms

Many functions in the system are performed using user-defined algorithms (also referred to as plug-ins). For example, user-defined algorithms can be used to perform custom validation, editing, and estimation logic, or retrieve characteristic values for factors.

Custom algorithms allow implementers to modify how the system responds to certain system events. The system provides a means where the custom algorithms can be invoked instead of the base package algorithms provided with the system. For instructions on creating custom algorithms and algorithm types, see the Framework documentation. To view information about specific algorithms provided with the base system, use the Application Viewer (also described in the

Framework documentation). The Application Viewer provides information about the base logic, inputs, and outputs of each algorithm entity or plug-in spot.

Entity Naming Conventions

Oracle Utilities Smart Grid Gateway system uses naming conventions to identify and distinguish entities that belong to different Oracle applications. These conventions can help you locate entities and understand their context.

Each base product uses a 2-character owner code as a prefix for all its entities. For Oracle Utilities Smart Grid Gateway, these prefixes are as follows:

- All Oracle Utilities Application Framework entities start with "F1"
- All Oracle Utilities meter data framework entities start with "D1"
- All Oracle Utilities Smart Grid Gateway adapter entities start with "D3", "D4", "D5", etc.

Oracle recommends that you follow these naming conventions and develop your own set of conventions for the entities you create. If you create new entities, DO NOT use these prefixes; use the prefix "CM" (or some other unique prefix) to identify entities that have been customized.

Configuration Process Overview

This section provides a high-level overview of some of the steps involved in the configuration process when implementing Oracle Utilities Application Framework products such as Oracle Utilities Smart Grid Gateway.

Note: The following sections are a simplification of an involved process, and are provided as guidelines only. Refer to the Oracle Utilities Application Framework documentation for detailed information about business objects and other objects referenced below.

Basic Configuration Steps - Design Your Business Objects

Much of the configuration involved in implementing Oracle Utilities Smart Grid Gateway is centered around the creation of business objects. Nearly every object or set of data used by Oracle Utilities Smart Grid Gateway is defined in business object, including meters and registers (devices and measuring components), service points, contacts, measurement data, validation rules, usage calculation rules, and more.

Given the prominent role that business objects play, one of the most important steps in implementing Oracle Utilities Smart Grid Gateway is identifying the business objects you will need to create to meet the requirements of your implementation. At a high level, this includes the following steps:

1. Identify the data to be defined by each business object

This step defines the maintenance object to be used with the business object, and the data elements to include in the business object's schema. Leverage data areas where possible to minimize redundant data definition.

2. Identify the processing to be performed by each business object

This step determines the specific algorithms/algorithm types, and business services (and related scripts and service programs) that will be perform the processing required by your business objects.

3. Identify how users will access and work with each business object (if applicable)

This defines the portals, zones, navigation options, BPA scripts, etc. you will need to develop to allow users access to your business objects.

Basic Configuration Steps - Create Your Business Objects

After identifying the above information, the next step is to create the business objects used in your implementation. At a high level, this includes the following steps:

1. Create configuration objects

Before you can create your business objects, you must first create the various configuration objects used by each business object, including:

- Application Services
- UI Maps (display and maintenance)
- Navigation Options
- Service Scripts
- Algorithm Types/Algorithms
- BPA Scripts/Business Service
- Other business objects
- Etc.

Where possible, leverage base package objects instead of creating your own to minimize data redundancy.

2. Create the business object

Once the configuration objects used by the business object are in place, you can create the actual business object itself using the Business Object portal, referencing the configuration objects created in step 1 as appropriate.

Later chapters in this book provide examples of many of the base package business objects provided with the system. These are provided to illustrate how the base package objects were designed, and to serve as the basis for the business objects you create as part of your implementation.

Basic Configuration Steps - Create Portals and Zones

If the base package portals and zones are not sufficient to meet the requirements of your implementation, you may have to create your own to allow users to work with your business objects. This can include creating the following:

- Context Menus
- Menus and Menu Items
- Navigation Keys
- Navigation Options
- Portals
- Zones

Basic Configuration Steps - Create Master Data

The “master” data used by Oracle Utilities Smart Grid Gateway includes the various entities used in your implementation, such as devices, measuring components, service points, VEE rules, etc. This data must be created in the system before you can process measurement data and create bill determinants from the data. Creating this data includes the following steps:

1. Create admin “type” data.

Many of the objects used by Oracle Utilities Smart Grid Gateway have corresponding admin “type” objects that are used to define attributes common to instances of that type of object. For example, Device Types are used to define attributes common to devices of a specific type. One of the most important attributes defined by an admin “type” object is the business object that will be used for instances of the object of that type. For example, devices created from a Device Type that references the “D1-SmartMeter” device business object will be based on that business object.

The **Admin “Type” Objects** table below lists the core objects used by Oracle Utilities Smart Grid Gateway and their corresponding admin “type” objects.

2. Create instances of the data.

Once the admin “type” data is in place, you can create the instances of the master data objects used in your implementation. These instances are the individual devices, measuring components, service points, VEE rules, usage subscriptions, etc. that will be used in processing measurement data, calculating usage and bill determinants, and so on.

Admin “Type” Objects.

Object	Admin “Type” Object
Activity	Activity Type
Communication	Communication Type
Contact	Contact Type
Device	Device Type
Device Event	Device Event Type
Device Configuration	Device Configuration Type
Dynamic Option	Dynamic Option Type
Measuring Component	Measuring Component Type
Service Point	Service Point Type
TOU Map	TOU Map Type
Usage Subscription	Usage Subscription Type

Chapter 2

General Configuration

This chapter describes configuration of general components, including the following:

- **Installation Options**
- **Master Configuration**

Installation Options

Installation options define the individual applications installed on your system and identify algorithms used to implement core system functions. These options also define global parameters such as the administrative menu style (alphabetical or functional), the country, language, currency code, as well as the base time zone to use for this implementation.

Installation options are stored in the installation record for your system. Use the **Installation Options - Framework** portal to configure these options. This portal is part of the OUAF and is described in detail in the Framework documentation.

Base Time Zone

The date/time attributes for all time-sensitive application entities, including start and end dates, are stored in the server application time zone in standard time and displayed in that time zone's legal time, which is the standard time adjusted for any seasonal shift.

The server time zone, also referred to as the Base Time Zone, must be correctly specified on the installation options record.

Note: The installation record does not dictate the server time zone, but rather must match it.

Installation Algorithms

Installation algorithms implement global system functions and can be customized for each implementation. The base package supports the following installation options for Meter Data Management-related system events:

- **Geocoding Service:** Responsible for geocoding an address (converting an address to a geocode latitude/longitude pair).
- **Global Context:** Sets global contexts (displayed in the Global Context dashboard zone) based on the value of existing global contexts. For example, if the Service Point is specified, this algorithm sets the Device by finding the most recently installed Device on the service point. It then sets the Measuring Component by finding the most effective Device Configuration and retrieving any measuring component linked to it. It then sets the Usage Subscription by finding the most recent active usage subscription linked to the service point. The contact is set by finding the main contact for the usage subscription.

Master Configuration

Master Configuration is another source of global parameter records used by a system implementation.

The Oracle Utilities meter data products use a Global Configuration record that controls core system functions. This record must be set up for the system to properly operate. See **Admin Setup Reference Tables** on page 3-7 for more information on when to set up this record.

Key concepts related to Master Configuration are discussed in this section. Refer to the embedded help for descriptions of the settings on the Master Configuration page.

Hijri to Gregorian Date Mapping

The Hijri to Gregorian Date Mapping option is used to define the relationship between Hijri dates and Gregorian dates for each year.

Chapter 3

Setting Up Admin Data

This chapter describes the different types of admin data that must be set up and defined as part of implementing and configuring Oracle Utilities Smart Grid Gateway, including:

- **Understanding Admin Data**
- **Admin Setup Reference Tables**

Understanding Admin Data

This section describes the admin data used by the Oracle Utilities meter data framework and related products, including Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway.

General Admin Data

General admin data are types of data used by multiple functional areas.

Exception Types

Exception types define the properties common to many exceptions.

When creating validation, editing, and estimation (VEE) rules, you might create an exception type for each VEE rule. You might also create more general exception types, such as "Insufficient Data" to be used to signify that a measurement didn't have sufficient data for the VEE rule to execute.

Factors

Factor are a centrally stored set of values for use in validation rules, bill determinants calculations, and other processes.

A factor can have different values depending upon some definable attribute of a system object, such as customer size associated with a service point. Examples of factors can include minimum/maximum thresholds, loss factors, etc. Classes of factors are defined that can have numeric values (as in the above examples), or values pointing to profile measuring components, or VEE groups.

A factor's values are effective-dated values - either a number, a profile measuring component, a VEE group, or some custom-defined value - assigned to a factor and associated to the value of some attribute of a system object. For example, consider a service point that can be classified as residential, commercial, or industrial. The tolerance percentage by which a customer's consumption can exceed last month's consumption can be based on the service point category. For this example, factor values for a single factor called "tolerance percentage" could be:
Residential - 20% Commercial - 10% Industrial - 5%.

Service Quantity Identifiers

Service Quantity Identifiers (SQI) are used to further distinguish between measured quantities that have identical UOM/TOU combinations, including situations in which the distinguishing identifier of a UOM is not accurately described as a TOU.

SQIs can also be used as a stand-alone representation of a service quantity that is not measured (one that is not properly described as a UOM) within a usage service quantity collection (such as a billing determinant).

Service Types

Service Types define specific types of service for which usage can be recorded and captured, such as electric, gas, steam, etc.

Time of Use

Time of Use (TOU) periods are modifiers for a given unit of measure that indicate a period of time during which a quantity has been used, such as On-Peak (meaning during a time when the greatest quantity of some consumable is being used), Off-Peak (meaning during a time when the least amount of some consumable is being used), etc.

Units of Measure

Units of Measure (UOM) identify quantities measured and recorded, such as KWH, KW, cubic feet, degrees Celsius, etc. UOMs are based on a specific service type.

Attributes used to define units of measure include the following:

- **Service Type:** The type of service (electric, gas, etc.) measured by the UOM
- **Decimal Positions:** The number of decimal places used when presenting a quantity for this UOM in Usage service quantities
- **Allowed on Measuring Component:** A flag that indicates if the UOM is allowed on Measuring Components
- **Measures Peak Quantity:** A flag that indicates if the UOM is used to measure peak quantities or not. An example of a UOM that measures peak quantities is kilowatts (KW).
- **Magnitude:** A number that indicates the relative size of the UOM as compared to a single unit of the UOM specified under "Base Unit of Measure." For example, megawatt hours (MWH) have a magnitude of 1,000 as compared to a single kilowatt hour (KWH).
- **Base Unit of Measure:** The UOM upon which the current UOM is based. Used in conjunction with magnitude. For example, the base unit of measure for megawatt hours (MWH) with a magnitude of 1,000 would be kilowatt hours (KWH).

Device Management Admin Data

Device management admin data include data that defines “types” of device-related objects.

Device Configuration Types

Device configuration types define the properties of device configurations of this type, including the valid types of measuring components that can be configured for device using configurations of this type.

Device Types

Device types define information about a class of devices, including properties that apply to all devices of a type. Properties defined for a device type can be overridden for an individual device.

Manufacturers

Manufacturers are the companies that makes devices. A device's manufacturer is defined as an attribute of the device itself.

Each manufacturer can have zero or more models defined. Models for a single manufacturer can have diverse service types.

Measuring Component Types

Measuring component types define the most important properties of a measuring component.

Measuring component types define what a measuring component measures (KWH, temperature, etc.), how regularly it measure it, and whether it should be connected to a physical device, or if it's used as a scratchpad measuring component or an aggregator measuring component. Measuring component types also specify how the measuring component's final measurements should be stored, how the measuring component's user-defined values should be calculated, and specific rules governing validation, editing, and estimation (VEE) for measuring components of the type. In addition, measuring component types define display properties and valid attribute values for measuring components belonging to the type.

Some important characteristics defined for measuring component types include:

- **Value Identifiers:** These store the values of UOM, TOU, and SQI that identify the measured amounts for measuring components of this type. Value identifiers specify the quantities stored on the measurement records for measuring components of this type.
- **Valid VEE Groups:** These define the VEE groups considered valid for measuring components of this type.
- **Fallback VEE Groups:** These define default VEE groups that can be used with all measuring components of this type. This alleviates the need to specify the same VEE groups on multiple measuring components of the same type. Each VEE group is designated a VEE group role that indicates when and how the VEE group is used (for initial load, manual override, or estimation).
- **Eligible Profile Factors (interval only):** These define the profile factors that are considered to be eligible for interval measuring components of this type. You can also specify one or more profile factors as a default.
- **Valid Profile Factors for Conversion from Scalar to Interval (scalar only):** These define the profile factors that are considered to be eligible for scalar measuring components of this type when converting scalar measurements to interval measurements. You can also specify one or more profile factors as a default.
- **Valid Scratchpad Measuring Component Types:** These define the scratchpad measuring component types considered valid for measuring components of this type.
- **Display Properties:** Defines how measurement data for measuring components of this type is displayed, including:
 - **Display Configuration:** Details related to how measurements are displayed, including the number of hours of data to display, the default TOU map used, the TOU by Day Profile factor used, and default measurement condition.
 - **Event Bar Profiles:** The event bar profiles used when displaying measurement data for measuring components of this type. Event bar profiles are defined as values for the 360 View Event Bar Profile extendable lookup.
 - **Final Values Overlay Profiles:** The final values overlay profiles used when displaying measurement data for measuring components of this type. Final values overlay profiles are defined as values for the Final Values Overlay Profile extendable lookup.

Measuring component types are described in more detail in **Chapter 4: Devices, Measuring Components, and, Device Configurations**.

Device Installation Admin Data

Device installation admin data includes data used to support the installation of devices.

Markets

Markets define the jurisdictions or regulatory environments in which a service point participates.

Markets also define market relationships for valid service providers and their roles within a market (distributor, etc.). While each service point specifies only one market, a utility may serve more than one market, and different service points throughout the utility's service territory can be linked to different markets.

Service Providers

Service providers are external entities that serve various roles relative to the application.

Service providers can include head-end systems, billing systems to which the application sends bill determinant data, market participants in a deregulated environment, outage management systems that receive meter event data from the application, or other parties that require or provide information to the system.

Service Point Types

Service point types define a specific type of point at which service is delivered.

Specifically, service point types define how the application manages many aspects of the service point's behavior. A service point type may have one or more valid device types defined that limit the types of devices that can be installed at service points of this type.

Contact Types

Contact types define the properties of a class of entities (businesses, persons).

Measurement Cycles

Measurement cycles define the schedule for manual meter reading of devices at service points in that cycle. Measurement cycles can have one or more associated routes used to collect measurements.

When used with smart meters, measurement cycles can also be configured to define when to create usage transactions for usage subscriptions associated to service points in the cycle.

Measurement Cycle Schedules

Measurement cycle schedules define the dates on which devices are scheduled to be read for a given measurement cycle and the routes used to collect measurements for the measurement cycle.

Device Communication Admin Data

Device communication admin data includes data used to support communication with head-end systems.

Activity Types

Activity types define properties common to a specific type of activity.

Activity types include types of communications between an application and a head-end system, such as a connection requests, meter ping requests, or on-demand meter readings, as well as device event types.

Communication Types

Communication types define properties common to a specific type of communication.

Communication types include types of communications between an application and a head-end system, such as notifications (used to notify an head-end system of a command request), or message responses (sent from a head-end system to confirm receipt of a message).

Device Event Types

Device event types define properties common to specific types of events.

Device event types represent different types of events that can take place relative to a device. Examples of device events include power outages, power restoration, tampering alerts, and other events.

Device event types can be defined by the following attributes:

- **Standard Event Name:** the "standard" name of the event type in Smart Grid Gateway. Device vendors may have their own specific names for device events.
- **Device Event Category:** a category (defined as an Extendable Lookup) used to group device event types.
- **Reporting Category:** a category used to group device event types for reporting purposes.
- **Activity Type:** the activity type for activities created for device events of this type.

VEE Rule Admin Data

VEE rule admin data include VEE groups and VEE rules.

VEE Groups

VEE groups are collections of VEE rules that are applied to initial measurement data.

VEE groups can be associated to a specific measuring component, or to a measuring component type (or both). VEE groups associated with a measuring component type are applied to all measuring components of that type, while those associated to a specific measuring component are applied only to that measuring component.

VEE Rules

VEE rules are standard and custom Validation, Estimation and Editing (VEE) rules that perform checking and/or manipulation of initial measurement data.

VEE rules are created for a specific VEE group. For example, if you were configuring two VEE groups and both included a specific VEE rule, you would need to create two instances of the VEE rule, one for each group.

Attributes used to define VEE rules typically include the following:

- **Basic Information:** Basic information about the VEE rule, including its name and description, the VEE group to which the rule belongs, the sequence of the rule within the group, the category, and start and end dates. This information is standard for most VEE rules.
- **Parameters:** The parameters used by the rule. Parameters are specific to each rule.
- **Exception Types and Severity:** Details about how to handle exceptions, including the Exception Type and Exception Severity for exceptions created by the rule.

VEE Rule Eligibility Criteria

VEE rule eligibility criteria are user-definable conditions that could cause a given VEE rule to be applied or skipped. This can involve the evaluation of some attribute of the device or measuring component, or something else entirely.

A VEE rule can have multiple eligibility criteria for determining if the rule should be applied or skipped, based on a user-defined sequence.

Admin Setup Reference Tables

This section lists and describes all objects that must be defined as part of the setup process for Oracle Utilities Smart Grid Gateway. It identifies the order in which objects should be defined and any prerequisites for setup.

Note: All basic Framework setup, including system and database setup and any modifications or extensions to base business objects, must have been completed before beginning setup tasks for Oracle Utilities Smart Grid Gateway. See the Framework documentation for more information.

Setup Sequence

In the setup tables that follow, the **Sequence** column displays the following codes:

L1 = Object has no setup prerequisites and should be defined before L2-L6 objects.

L2 = Object has some L1 prerequisites and should be defined after all L1 objects have been defined and before L3 objects.

L3 = Object should be defined after all L1 and L2 objects have been defined.

L4 = Object should be defined after all L1, L2, and L3 objects have been defined.

L5 = Object should be defined after all L1, L2, L3, and L4 objects have been defined.

Administration Setup and Maintenance

To access the maintenance portals for the objects in this section, do one of the following:

- If you are using functional menus, select **Admin Menu>[Functional Menu]>[object name]**
- If you are using alphabetical menus, select Admin Menu>[object name]

The [Functional Menu] and [object name] are provided in the appropriate columns in the following tables.

Application Framework Setup

Seq	Object	Functional Menu	Description	Prerequisites
L1	Country	General	Your organization's country.	None
L1	Currency	Financial	Your organization's native currency.	None
L1	Display Profile	General	Controls how dates, times, and numbers displayed.	None
L1	Language	General	The language to use for this implementation.	None
L1	Time Zone	General	Your organization's base time zone.	None
L1	To Do Role	General	Used to associate users with To Do entries.	None
L1	Work Calendar	General	The work calendar for your organization, which identifies your public holidays	None
L2	Installation Options	System	Control various aspects of the system. Refer to the Installation Options section earlier in this document.	Time Zone, Language, Currency

Seq	Object	Functional Menu	Description	Prerequisites
L2	Master Configuration	System	Enables an implementation to capture various types of information in the system.	
L2	To Do Type	General	Used to define types of To Do Entries	To Do Role
L2	User	Security	Defines a user's user groups, data access roles, portal preferences, default values, and To Do roles	Language, Display Profile, To Do Roles
L2	User Group	Security	A group of users who have the same degree of security access	User

Oracle Utilities Smart Grid Gateway Setup

Seq	Object	Functional Menu	Description	Prerequisites
L1	Activity Type	Communications	Defines properties common to a specific type of activity	None
L1	Communication Type	Communications	Define properties common to a specific type of communication	None
L1	Contact Type	Customer Information	Defines properties of a class of entities (businesses, persons)	None
L1	Device Event Type	Communications	Defines properties common to a specific types of events	None
L1	Exception Type	Common	Defines properties common to exceptions of a specific type	None
L1	Factor	Common	Centrally stored sets of values for use in validation rules, bill determinants calculations, and other processes	None
L1	Market	Communications	Defines jurisdictions or regulatory environments in which a service point participates	None
L1	Measurement Cycle	Device Installation	Defines the schedule for manual meter reading of devices at service points in that cycle	None
L1	Measurement Cycle Schedule	Device Installation	Define the dates on which devices are scheduled to be read for a given measurement cycle	None
L1	Service Provider	Communications	External entities that serve various roles relative to the application (head-end systems, billing systems, market participants, outage management systems, etc.)	None
L1	Service Quantity Identifier	Common	Used to further distinguish between measured quantities that have identical UOM/TOU combinations	None

Seq	Object	Functional Menu	Description	Prerequisites
L1	Service Type	Common	Defines specific types of service for which usage can be recorded and captured (electric, gas, steam, etc.)	None
L1	VEE Group	VEE Rules	Collections of VEE rules that are applied to initial measurement data	None
L2	Manufacturer	Device	Individual companies that makes devices. Manufacturers also reference models.	Service Type
L2	Unit of Measure	Common	Quantities measured and recorded by the system (CCF, KWH, KW, etc.)	Service Type
L2	VEE Rule	VEE Rules	Standard and custom VEE rules that perform checking and/or manipulation of initial measurement data	VEE Group, Exception Type
L3	Measuring Component Type*	Device	Defines the most important properties of a measuring component	Factor (Profile), Service Quantity Identifier, Service Type, Time of Use, Unit of Measure, VEE Group
L4	Device Configuration Type	Device	Defines properties of device configurations of a given type	Service Type, Measuring Component Type
L4	Device Type	Device	Defines information about a class of devices	Service Type, Device Configuration Type
L5	Service Point Type	Device Installation	Defines specific types of points at which service is delivered	Service Type, Device Type

* Measuring component types also reference other measuring component types, TOU maps, and extendable lookups.

Chapter 4

Devices, Measuring Components, and Device Configurations

This chapter provides descriptions of devices, device configurations, and measuring components, including:

- **Understanding Devices, Measuring Components, and Device Configurations**
- **Devices In Detail**
- **Device Configurations In Detail**
- **Measuring Components In Detail**
- **Measuring Component Types In Detail**
- **Configuring Devices, Measuring Components, and Device Configurations**

Note: References to objects and options pre-fixed with “D2” are available only with Oracle Utilities Meter Data Management, and included for illustrative purposes only.

Understanding Devices, Measuring Components, and Device Configurations

This section provides an overview of devices, measuring components, and device configurations, including how they are used in the meter data framework and related products including Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway.

Devices

Devices are physical or virtual objects that hold one or more measuring components that can produce data to be handled by the system. While most devices are meters, an implementation might set up devices for every asset that measures or monitors resource usage. For example, a device could be set up to record average daily temperature (if temperature plays a part in usage calculations). Examples of devices include meters, substations, transformers, demand response devices, weather stations, etc.

Measuring Components

Measuring components are single points for which data will be received and stored in the system.

Types of Measuring Components

A measuring component can be associated to a physical device, which can have one or more measuring components, or it can be "virtual" or "stand-alone," meaning that it is not associated to a physical device. The meter data framework supports the following types of measuring components:

- **Physical:** Physical measuring components are those that physically exist, and that are linked to a device that can be configured differently over time. Interval channels and scalar registers are examples of physical measuring components.

Note: The terms register and channel are synonyms for measuring component.

- **Standalone:** A standalone measuring component is used to record measurements for something that does not have a physical presence. For example, you might create a standalone measuring component to record the average daily temperature supplied by a weather station.
- **Scratchpad:** User create scratchpad measuring components to experiment with measurement manipulation functions before applying the functions to a physical or standalone measuring component. Examples of measurement manipulation might include experimenting with the impact of executing the "spike smooth" function on an initial measurement, or adding or removing intervals to or from the measurement. Scratchpad measuring components provide users with a means to manipulate "scratchpad" measurement data without affecting existing "live" measurement data.
- **Aggregator:** An aggregator measuring component holds summarized usage from other measuring components. For example, aggregator measuring components could be configured to hold total consumption for each postal code within a service territory.

Head-end system processing statistics used by Oracle Utilities Smart Grid Gateway are stored as aggregated measurements for aggregator measuring components.

Scalar vs. Interval

Beyond the four types described above, measuring components generally fall into one of two primary classes of: scalar measuring components, and interval measuring components.

- **Scalar** measuring components are measured at unpredictable intervals. For example, "once-a-month" is not a predictable interval as the amount of time between reads is unpredictable and inconsistent.

Scalar measuring components are typically read manually

- **Interval** measuring components are measured at predictable intervals, such as every 15 minutes, every 30 minutes, every hour, etc.

The term Seconds-per-interval (SPI) is used to define the size of an interval measuring component's intervals.

Note: A device may have any combination of interval and/or scalar measuring components

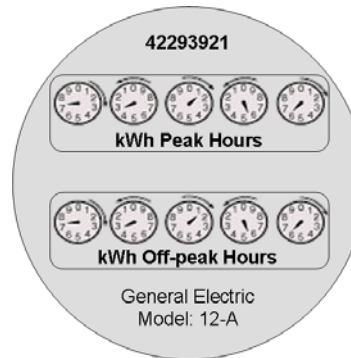
Measuring Component Measurements

Measuring components are configured to measure specific types of quantities. These include:

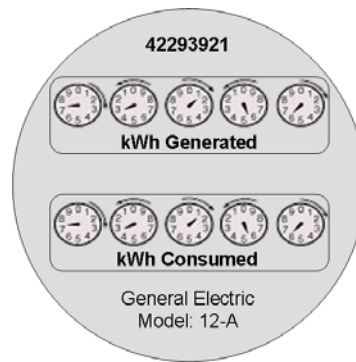
- **Unit of Measure:** The unit of measure for the quantity being recorded. Examples include kilo-watt hours (kWh), kilo-watts (kW), therms, cubic feet (CCF), temperature (Fahrenheit or Celsius), etc.
- **Time of Use:** Modifiers for a given unit of measure that indicate a period of time during which a quantity has been used, such as On-Peak (meaning during a time when the greatest quantity of some consumable is being used), Off-Peak (meaning during a time when the least amount of some consumable is being used), etc.
- **Service Quantity Identifiers:** Used to further distinguish between measured quantities that have identical UOM/TOU combinations, including situations in which the distinguishing identifier of a UOM is not accurately described as a TOU. Generally, SQI is only used when multiple measuring components measure the same thing, but in different ways. A meter that measures both generation KWH and consumption KWH could use SQIs to differentiate between the two.

The combination of UOM, TOU and SQI define what a measuring component measures. TOU and SQI are optional, but UOM must be defined for all measuring components.

For example, consider a meter (as illustrated in the image below) with two measuring components, both measuring the same unit of measure (kWh), but each measuring component measures consumption in different time of use (TOU) periods (peak and off-peak).



Another example might be a meter that records both generated KWH and consumed KWH. This meter would be configured to measure both UOM and SQI.



A measurement is recorded each time a measuring component is measured. This means that for a meter with two measuring components that is read once a month, two measurements, one for each measuring component, would be recorded each month.

Subtractive vs. Consumptive Measurements

Another attribute that defines how measuring components measure quantities is the distinction between subtractive and consumptive measuring components.

A subtractive measuring component's usage is equal to the current measurement (also known as the Stop or End Measurement or Reading) minus the previous measurement (also known as the Start Measurement or Reading). To put this more simply:

$$\text{Usage} = \text{End Measurement} - \text{Start Measurement}$$

Most residential scalar KWH meters are subtractive. The table below lists a series of measurements for a subtractive measuring component.

Date	KWH (Measurement)	Usage
01/01/2010	0	
01/31/2010	1000	1000 KWH
03/02/2010	1789	789 KWH
04/01/2010	2700	911 KWH

A consumptive measuring component's usage is equal to its current measurement. Consumptive measuring components are often used to measure demand, such as KW. The table below lists a series of measurements for a consumptive measuring component.

Date	KW (Measurement)	Usage
01/01/2010	0	
01/31/2010	10	10 KW
03/02/2010	6	6 KW
04/01/2010	5	5 KW

Interval measuring components can also be considered consumptive, in that the consumption value of each individual interval is equal to its measurement.

Interval vs. Scalar Measurements

As noted above, interval measuring components record measurements every interval, defined by the measuring component's SPI (seconds per interval). For interval measuring components,

measurements are only allowed on these time boundaries. For example, measurements for an interval measuring component with an SPI of 900 (15 minutes) on January 1, 2010 might be as follows:

Date	Time	KWH
01/01/2010	10:00 AM	0
01/01/2010	10:15 AM	10
01/01/2010	10:30 AM	6
01/01/2010	10:45 AM	5

In contrast to interval measuring components, scalar measuring components are read at unpredictable (and often inconsistent) intervals and are allowed at any point in time. In practice, scalar measuring components are read monthly, bimonthly, quarterly, etc. For example, measurements for an scalar measuring component from January 1, 2010 through April 1, 2010 might be as follows

Date	KWH (Measurement)
01/01/2010	0
01/31/2010	1000
03/02/2010	1789
04/01/2010	2700

Note that interval and scalar measuring components can exist on the single meter. For these types of meters, the scalar measuring component is typically used to verify and validate the interval measurements. For example, the sum of all the interval measurements within a measurement period should equal the scalar measurement for the same period.

Device vs. Measuring Component Attributes

The distinction between attributes used to define devices and measuring components is important when creating devices and measuring components as part of an implementation. For example, if you identify additional attributes you wish to capture, it's import to store those attributes in the most appropriate place.

Devices have attributes that are applicable to the physical object, and that are the same regardless of the number of measuring components on the device. For example:

- The type of device
- Manufacturer and Model
- Serial Number
- Badge Number
- Head-End System (for smart meters)

Measuring components have attributes that may differ for each measuring component on a device, for example:

- The type of measuring component (which in turn defines the measuring component's UOM, TOU, SQI, whether it is scalar or interval (and its SPI), and others
- Channel ID (for interval channel measuring components)

- Channel (or Register) multiplier (a value by which the measured consumption is multiplied to derive usage)
- Validation, Editing, and Estimation groups used when validating initial measurement data for the measuring component.

Device Configurations

A measuring component's attributes can change over time. Device configurations record how a device's measuring components look at an instant in time. A new device configuration is required whenever a device's measuring components are reconfigured. For example, if the register multiplier on a measuring component changes as of June 1, 2010, the device would require a new device configurations dated 1-Jun-2010 to reflect the change.

Note that device configurations don't typically capture the changed information, but instead indicate that changes of some sort have taken place on one or more of the device's measuring components.

Devices In Detail

This section provides details concerning the device objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom devices you create as part of your implementation. This section includes:

- A description of the D1-DEVICE maintenance object
- Lists of the base package device business objects, including “lite” business objects
- Details concerning device-specific configuration options
- A sample device business object (D1-SmartMeter)

Maintenance Object - D1-DEVICE

Device business objects use the D1-DEVICE maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-DEVICE
Description	Device
Service Name	D1-DEVICE (Device Maintenance)
Tables	<ul style="list-style-type: none"> • D1_DVC (Device) - Primary • D1_DVC_CHAR (Device Characteristics) - Child • D1_DVC_IDENTIFIER (Device Identifier) - Child • D1_DVC_LOG (Device Log) - Child • D1_DVC_LOG_PARM (Device Log Parameter) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Device Business Objects

The meter data framework base package includes the following device business objects:

Business Object Name	Description
D1-ManualMeter	Manual Meter Instances of this business object represent individual manual meters defined in the system.
D1-SmartMeter	Smart Meter Instances of this business object represent individual smart meters defined in the system.

The meter data framework base package includes the following “lite” device business objects:

Business Object Name	Description
D1-DeviceIDsLite	Lite BO to Get AMI ID Type dynamically

Business Object Name	Description
D1-DeviceLite	Device LITE
D1-DeviceLiteAMI	BO to Get AMI related details for Deive
D1-DeviceIdentifierLite	Device Identifiers LITE
D1-DeviceParentLite	Device Parent LITE

The meter data framework base package includes the following additional device business objects:

Business Object Name	Description
D1-SynchronizationAddDevice	Device Synchronization Add (used when adding a new device as a result of a data synchronization request)

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by device business objects.

Business Object Options

Device business objects can make use of the following business object options:

- **Install Event BO:** This option identifies the install event business object to use when installing device configurations for devices defined by this business object.

For example, for the D1-SmartMeter device, this option is set to D1-SmartMeterInstallEvent, meaning that any time a device configuration for the D1-SmartMeter device is installed, the install event business object used will be D1-SmartMeterInstallEvent.
- **Synchronization Add BO:** This option identifies the business object to use when adding new devices as a result of a data synchronization request.
- **Valid Command Request BO:** This option defines the valid commands available for the device, and the activity business object to use for each command. This option is used with Oracle Utilities Smart Grid Gateway and can be defined multiple times for the same device, once for each command supported by the device.

For example, for the D1-SmartMeter device, this option is defined seven times, once for each of the following commands: On-Demand Read - Interval, On-Demand Read - Scalar, Device Status Check, Remove Connect, Remote Disconnect, Device Commission, and Device Decommission.

Example Device - D1-SmartMeter

This section lists some of the details of the D1-SmartMeter device business object.

Option/Field	Description
Business Object	D1-SmartMeter
Description	Smart Meter
Maintenance Object	D1-DEVICE (Device)
Application Service	D1-SMARTMTRBOAS (Smart Meter BO)
Instance Control	Allow New Instances

Option/Field	Description
Options	<ul style="list-style-type: none"> • Install Event BO: D1-SmartMeterInstallEvent (Smart Meter Installation Event) • Synchronization Add BO: D1-SynchronizationAddBO (Device Synchroniation Add) • Valid Command Request BOs: <ul style="list-style-type: none"> • D1-OnDemandReadInterval (On-Demand Read Interval) • D1-OnDemandReadScalar (On-Demand Read Scalar) • D1-DeviceStatusCheck (Device Status Check) • D1-RemoteConnect (Remote Connect) • D1-RemoteDisconnect (Remote Disconnect) • D1-DeviceCommission (Device Commission) • D1-DeviceDecommission (Device Decommission) • Display UI Map: D1-SmartMeterDisplay (Smart Meter - Display) • Portal Navigation Option: d1dvcTabMenu (Device) • Display Map Service Script: D1-RtSmMtrDs (Smart Meter - Retrieve Details for Display) • Maintenance UI Map: D1-SmartMeterMaint (Smart Meter - Maintenance)
Algorithms	<ul style="list-style-type: none"> • Information: D1-SMARTINFO (Smart Meter Information) • Validation: D1-VALRETDT (Validate Device Retirement Date)
Lifecycle	<ul style="list-style-type: none"> • Active (Initial) • Retired (Final)

Use the Business Object portal to view additional details concerning this business object.

Device Configurations In Detail

This section provides details concerning the device configurations supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom device configurations you create as part of your implementation. This section includes:

- A description of the D1-DVCCONFIG maintenance object
- Lists of the base package device configuration business objects, including “lite” business objects
- A sample device configuration business object (D1-DeviceConfiguration)

Maintenance Object - D1-DVCCONFIG

Device configuration business objects use the D1-DVCCONFIG maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-DVCCONFIG
Description	Device Configuration
Service Name	D1-DVCCONFIG (Device Configuration Maintenance)
Tables	<ul style="list-style-type: none">• D1_DVC_CFG (Device Configuration) - Primary• D1_DVC_CFG_CHAR (Device Configuration Characteristics - Child• D1_DVC_CFG_LOG (Device Configuration Log) - Child• D1_DVC_CFG_LOG_PARM (Device Configuration Log Parameter) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Device Configuration Business Objects

The meter data framework base package includes the following device configuration business objects:

Business Object Name	Description
D1-DeviceConfiguration	Device Configuration Instances of the business object represent individual device configurations defined in the system.

The meter data framework base package includes the following “lite” device configuration business objects:

Business Object Name	Description
D1-DeviceConfigParentLITE	Device Configuration Parent LITE
D1-DeviceConfigurationLite	Device Configuration LITE

The meter data framework base package includes the following additional device business objects:

Business Object Name	Description
D1-SynchronizationAddDC	Device Configuration Synchronization Add (used when adding a new device configuration as a result of a data synchronization request)

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by device configuration business objects.

Business Object Options

Device configuration business objects can make use of the following business object options:

- **Synchronization Add BO:** This option identifies the business object to use when adding new device configurations as a result of a data synchronization request.

Example Device Configuration - D1-DeviceConfiguration

The table below lists the details of the D1-DeviceConfiguration device configuration business object.

Option	Description
Business Object	D1-DeviceConfiguration
Description	Device Configuration
Maintenance Object	D1-DVCCONFIG (Device Configuration)
Application Service	D1-DVCCONFIGBOAS (Device Configuration BO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Synchronization Add BO: D1-SynchronizationAddDC (Device Configuration Synchronization Add) • Display UI Map: D1-DeviceConfigDisplay (Device Configuration- Display) • Portal Navigation Option: d1dvcfgTabMenu (Device Configuration) • Display Map Service Script: D1-D1-RtDvcCfgD (Device Configuration - Retrieve Details for Display) • Maintenance UI Map: D1-DeviceConfigMaint (Device Configuration- Maintenance)

Option	Description
Algorithms	<ul style="list-style-type: none">• Information: D1-DVCOINFO (Device Configuration Information)• Pre-Processing: D1-DELMC (Delete Associated Measuring Components)• Pre-Processing: D1-DEFTIMZON (Default Time Zone value based on Installation Option)• Validation: D1-INSTDCVAL (Earlier Installed Device Configuration)• Validation: D1-VALTIMZON (Validates BO Time Zone value against Installation Option)
Lifecycle	<ul style="list-style-type: none">• Pending (Initial)• Active (Final)

Use the Business Object portal to view additional details concerning this business object.

Measuring Component Types In Detail

This section provides details concerning the measuring component type objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom measuring component type objects you create as part of your implementation. This section includes:

- A description of the D1-MCTYPE maintenance object
- Lists of the base package measuring component type business objects, including “lite” business objects
- Details concerning measuring component type-specific configuration options
- A sample measuring component type business object (D1-IntervalChannelTypePhysical)

Maintenance Object - D1-MCTYPE

Measuring component type business objects use the D1-MCTYPE maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-MCTYPE
Description	Measuring Component Type
Service Name	D1-MCTYPE (Measuring Component Type Maintenance)
Tables	<ul style="list-style-type: none"> • D1_MEASR_COMP_TYPE (Measuring Component Type) - Primary • D1_MC_TYPE_VALUE_IDENTIFIER (Measuring Component Type Value Identifier) - Child • D1_MC_TYPE_VALUE_IDENTIFIER_L (MC Type Value Identifier Language) - Child • D1_MEASR_COMP_TYPE_CHAR (Measuring Component Type Characteristics) - Child • D1_MEASR_COMP_TYPE_L (Measuring Component Type Language) - Child • D1_MEASR_COMP_TYPE_VEE_GRP (Measuring Component Type VEE Group) - Child • D1_MEASR_COMP_TYPE_FBK_VEE_GRP (Measuring Component Type Fallback VEE Grp) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Measuring Component Type Business Objects

The meter data framework base package includes the following measuring component type business objects:

Business Object Name	Description
D1-ActivityStatsAggType	Activity Statistics Aggregator Type Instances of this business object represent individual activity statistics aggregator measuring component types defined in the system. Activity Statistics Aggregators are used to track statistics related to device event and measurement uploads.
D1-AutoReadRegisterType	Auto-Read Register Type Instances of this business object represent individual auto-read register measuring component types defined in the system. Auto-Read Registers are automatically read on a regular basis but the system receives only one measurement at a time, meaning one measurement per Initial Measurement. These types of measuring components are categorized as scalar rather than interval, even though Initial Measurements are received on a regular basis.
D1-GenericMCType	Generic MC Type (used as a parent measuring component type)
D1-IntervalChannelTypePhysical	Interval Channel Type - Physical Instances of this business object represent individual physical interval channel measuring component types defined in the system.
D1-IntervalChannelTypeScratchp	Interval Channel Type - Scratchpad Instances of this business object represent individual scratchpad interval channel measuring component types defined in the system.
D1-RegisterTypePhysical	Register Type Instances of this business object represent individual physical register measuring component types defined in the system.

The meter data framework base package includes the following “lite” measuring component type business objects:

Business Object Name	Description
D1-MCTypeEstimateParmsLiteBO	MC Type Periodic Estimation LITE
D1-MCTypeLite	MC Type Lite

Business Object Name	Description
D1-MCTypeMainLite	Measuring Component Type LITE
D1-MCTypeParentLITE	Measuring Component Type Parent LITE
D1-MeasuringCompTypeLite	Measuring Component Type LITE

The meter data framework base package includes the following additional measuring component type business objects:

Business Object Name	Description
D1-MCTypeValueIdentifiers	Measuring Component Type Value Identifiers (used to retrieve the value identifiers of a measuring component type)
D1-MeasuringCompTypeBundlingBO	Bundling BO for Measuring Component Type
D1-MeasuringCompTypePhysicalBO	Physical BO for Measuring Component Type

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by measuring component type business objects.

System Events

Measuring component type business objects can make use of the following system events:

- **Calculate Interval Consumption:** This system event defines the algorithm used to calculate interval consumption for measuring components based on this measuring component type.
- **Calculate Scalar Consumption:** This system event defines the algorithm used to calculate scalar consumption for measuring components based of this type.

Other Options

Measuring component types define many attributes of the measuring components of that type. These options are specified when creating measuring component types based on a measuring component type business object, and include the following:

Value Identifiers

Value identifiers store the values of UOM, TOU, and SQI that identify the measured amounts for measuring components of this type. Value identifiers specify the quantities stored on the measurement records for measuring components of this type.

Value identifiers must also specify a Value Derivation Algorithm based on the D1-DERIVAQTY Algorithm Type (Derive a quantity using a formula). This algorithm is used to derive measurement elements by applying a formula to calculate a value.

VEE Groups (Valid and Fallback)

VEE Groups define the validation, editing, and estimation rules to be applied to initial measurement data for measuring components of this type.

- **Valid VEE Groups:** These define the VEE groups considered valid for measuring components of this type.

- **Fallback VEE Groups:** These define default VEE groups that can be used with all measuring components of this type. This alleviates the need to specify the same VEE groups on multiple measuring components of the same type. Each VEE group is designated a VEE group role that indicates when and how the VEE group is used (for initial load, manual override, or estimation).

Profile Factors

Profile factors are factors of type profile used when displaying initial measurement data on the Measuring Component portal of the 360 Degree View. Measuring component types reference the following types of profile factors:

- **Eligible Profile Factors (interval only):** These define the profile factors that are considered to be eligible for interval measuring components of this type. You can also specify one or more profile factors as a default.
- **Valid Profile Factors for Conversion from Scalar to Interval (scalar only):** These define the profile factors that are considered to be eligible for scalar measuring components of this type when converting scalar measurements to interval measurements. You can also specify one or more profile factors as a default.

Valid Scratchpad Measuring Component Types

These define the scratchpad measuring component types considered valid for measuring components of this type.

Display Properties

Measuring component types reference the following display properties:

- **Event Bar Profiles:** used when displaying measurement data for measuring components of this type. Event bar profiles are business objects defined as values for the “360 View Event Bar Profile” extendable lookup (D1-360EventBarProfile).
- **Final Values Overlay Profiles:** This display option is used when displaying final measurement data for measuring components of this type. Final values overlay profiles are business objects defined as values for the “Final Values Overlay Profile” extendable lookup (D1-FinalValuesOverlayProfile).

Example Measuring Component Type - D1-IntervalChannelTypePhysical

The table below lists the details of the D1-IntervalChannelTypePhysical measuring component type business object.

Option	Description
Business Object	D1-IntervalChannelTypePhysical
Description	Interval Channel Type - Physical
Maintenance Object	D1-MCTYPE (Measuring Component Type)
Application Service	D1-MCTYPE (Measuring Component Type MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Display UI Map: D1-IntervalChannelTypeDisplay (Interval Channel Type - Display) • Portal Navigation Option: d1mctypeTabMenu (Measuring Component Type) • Maintenance UI Map: D1-IntervalChannelTypeMaint (Interval Channel Type - Maintenance)

Option	Description
Algorithms	<ul style="list-style-type: none">• Calculate Interval Consumption: D1-IN-CNSUMP (Calculate Interval Consumption)• Validation: D1-INTSCRVAL (Validate List of Scratchpad Measuring Component Types)• Validation: D1-DEFTOUVAL (Validate Interval Size of TOU Map for Display)• Validation: D1-ELPRFTVAL (Validate Factors for Eligible Profile Factors)• Validation: D1-FNVALOVAL (Validate Final Values Overlay Profiles)• Validation: D1-EVTBARVAL (Validate Event Bar Profiles)• Validation: D1-HRDATAVAL (Hours of Data to Display Validation)

Use the Business Object portal to view additional details concerning this business object.

Measuring Components In Detail

This section provides details concerning the measuring components supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom measuring components you create as part of your implementation. This section includes:

- A description of the D1-MEASRCOMP maintenance object
- Lists of the base package measuring component business objects, including “lite” business objects
- Details concerning measuring component-specific configuration options
- A sample measuring component business object (D1-IntervalChannel)
- Lists of base package measurement functions including the BPA Script, Service Script, and a brief description of each

Maintenance Object - D1-MEASRCOMP

Measuring component business objects use the D1-MEASRCOMP maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-MEASRCOMP
Description	Measuring Component
Service Name	D1-MEASRCOMP (Measuring Component Maintenance)
Tables	<ul style="list-style-type: none">• D1_MEASR_COMP (Measuring Component) - Primary• D1_MEASR_COMP_CHAR (Measuring Component Characteristics) - Child• D1_MEASR_COMP_IDENTIFIER (Measuring Component Identifier) - Child• D1_MEASR_COMP_LOG (Measuring Component Log) - Child• D1_MEASR_COMP_LOG_PARM (Measuring Component Log Parameter) - Child• D1_MEASR_COMP_REL (Related Measuring Component) - Child• D1_MEASR_COMP_VEE_GROUP (Measuring Component VEE Group) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Measuring Component Business Objects

The meter data framework base package includes the following measuring component business objects:

Business Object Name	Description
D1-IntervalChannel	Interval Channel Instance of this business object represent individual interval channel measuring components defined in the system.
D1-IntervalScratchpad	Interval Scratchpad Instance of this business object represent individual interval scratchpad measuring components defined in the system.
D1-PayloadStatsAggEvent	Payload Statistics Aggregator Event Instance of this business object represent individual event payload statistics aggregator measuring components defined in the system.
D1-PayloadStatsAggIMD	Payload Statistics Aggregator - IMD Instance of this business object represent individual measurement payload statistics aggregator measuring components defined in the system.
D1-Register	Register Instance of this business object represent individual register measuring components defined in the system.
D1-RegisterAutoRead	Register - Auto-Read Instance of this business object represent individual auto-read register measuring components defined in the system.
D1-StatsAggregator	Statistics Aggregator (used as a parent to payload statistics aggregators)..

The meter data framework base package includes the following “lite” measuring component business objects:

Business Object Name	Description
D1-MCLatestMeasrDttm	Measuring Component Lite BO for Latest Measurement Date/Time
D1-MCLite	Measuring Component LITE
D1-MCScratchpadLite	MC Lite Scratchpad
D1-MCStandAlone	Measuring Component Standalone Lite
D1-MeasuringCompParentLITE	Measuring Component Parent LITE

The meter data framework base package includes the following additional measuring component business objects:

Business Object Name	Description
D1-SynchronizationAddMC	Measuring Component Synchronization Add (used when adding a new measuring component as a result of a data synchronization request)

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by measuring component business objects.

Business Object Options

Measuring component business objects can make use of the following business object options:

- **Estimation Initial Measurement Data BO:** This option identifies the initial measurement data business object to use when creating new “estimation” type initial measurement data for the measuring component.

For example, for the D1-IntervalChannel measuring component business object, this option is set to D1-EstimationIMDInterval, meaning that new “estimation” type initial measurement data for measuring components based on the D1-IntervalChannel business object would use the D1-EstimationIMDInterval business object.

- **Interval Initial Measurement Function:** This option defines the BPA Script used to apply a function to an interval initial measurement curve on the Initial Measurement Lens zone of the Initial Measurement portal. This option can be defined multiple times for the same measuring component.
- **Manual Override IMD BO:** This option identifies the initial measurement data business object to use when creating new “manual” type initial measurement data for the measuring component.

For example, for the D1-IntervalChannel measuring component business object, this option is set to D1-ManualIMDInterval, meaning that new “manual” type initial measurement data for measuring components based on the D1-IntervalChannel business object would use the D1-ManualIMDInterval business object.

- **Measuring Component Consumption Function:** This option defines the BPA Script used to apply a function to the measuring component's consumption on the zones of the Measuring Component portal in the 360 Degree View. This option can be defined multiple times for the same measuring component.
- **Synchronization Add BO:** This option identifies the business object to use when adding new measuring components as a result of a data synchronization request.

System Events

Measuring component business objects can make use of the following system events:

- **Find Constituent Measuring Components:** This system event defines the algorithm to use to identify constituent measuring components related to an aggregator measuring component. (An aggregator's constituent measuring components are the individual measuring components whose measurement data is aggregated when creating aggregation measurement data).
- **Periodic Estimation:** This system event defines the algorithm to use when performing periodic estimation for the measuring component.

Example Measuring Component - D1-IntervalChannel

The table below lists the details of the D1-IntervalChannel measuring component business object.

Option	Description
Business Object	D1-IntervalChannel
Description	Interval Channel
Maintenance Object	D1-MEASRCOMP (Measuring Component)
Application Service	D1-MEASURINGCOMP (Measuring Component MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Estimation Initial Measurement Data BO: D1-EstimationIMDInterval (Estimation IMD (Interval)) • Manual Override IMD BO: D1-ManualIMDInterval (Manual IMD (Interval)) • Synchronization Add BO: D1-SynchronizationAddMC (Measuring Component Synchronization Add) • Measuring Component Consumption Functions: <ul style="list-style-type: none"> • D2-CScIntFnc (Create Scratchpad) • D2-IETExc (Export to Excel) • D2-NewIntFnc (New IMD via XML) • D2-OvrIntFnc (Create/Override) • D2-RedValFnc (Rederive Values) • D2-SAsIntFnc (Save As) • Interval Initial Measurement Functions: <ul style="list-style-type: none"> • D2-AdjMathF (Adjust Intervals Using Math) • D2-AdjScalF (Adjust Intervals to Scalar Quantity) • D2-IIMDSAsF (Save As) • D2-InsIntF (Insert Intervals) • D2-RemIntF (Remove Intervals) • D2-SetCondsF (Set Conditions) • D2-ShiftIntF (Shift Intervals) • D2-SmoSpikeF (Smooth Spikes) • Display UI Map: D1-IntervalChannelDisplay (Interval Channel - Display) • Portal Navigation Option: d1mcTabMenu (Measuring Component) • Display Map Service Script: D1-RtMcIcChs (Interval Channel - Retrieve Details for Display) • Maintenance UI Map: D1-IntervalChannelMaint (Interval Channel - Maintenance)

Option	Description
Algorithms	<ul style="list-style-type: none">• Periodic Estimation: D1-CRIMTODO (Create Initial Measurement and To Do Entry)• Information: D1-SMTMCINFO (Measuring Component for Smart Devices - Information)• Pre-Processing: D1-DIMDCPRE (Delete Initial Measurement Data of Measuring Component.• Validation: D1-RELMCCVAL (Validate Consumption Reference Measuring Component)• Validation: D1-INTMCVAL (Check if Measuring Component's Type is Interval)

Use the Business Object portal to view additional details concerning this business object.

Configuring Devices, Measuring Components, and Device Configurations

This section provides high-level overviews of the steps involved in configuring custom devices, device configurations, measuring component types, and measuring components. See **Configuration Process Overview** in **Chapter One** for a high-level overview of the overall configuration process.

Note: The procedures below focus on specific configuration tasks and options related to each of the objects described in this chapter, and do not address all the steps involved in creating business objects and other configuration objects. For more information about these subjects, refer to the Oracle Utilities Application Framework documentation.

Configuring Custom Devices

Configuring custom devices involves the following steps:

1. Design the device business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom device-related configuration objects required for your business objects, including:

Business Object Options: Create business objects for the following business object options:

- Install Event BO
 - Synchronization Add BO
 - Valid Command Request BO (if using Oracle Utilities Smart Grid Gateway)
3. Create your device business objects, referencing the configuration objects created above as appropriate.
 4. Set up admin records that define the device types you will use in your implementation.

Configuring Custom Device Configurations

Configuring custom device configurations involves the following steps:

1. Design the device configuration business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom device configuration-related configuration objects required for your business objects, including:

Business Object Options: Create business objects for the following business object options:

- Synchronization Add BO
3. Create your device configuration business objects, referencing the configuration objects created above as appropriate.
 4. Set up admin records that define the device configuration types you will use in your implementation.

Configuring Custom Measuring Component Types

Configuring custom measuring component types involves the following steps:

1. Design the measuring component type business objects you will need to create for your implementation, including the data and processing required for each.

2. Create the custom measuring component type-related configuration objects required for your business objects, including:
System Events: Create algorithms for the following system events:
 - Calculate Interval Consumption (for interval and scalar measuring component types)
 - Calculate Scalar Consumption (for scalar measuring component types)**Options:** Create data as appropriate for the following options used when creating measuring component types:
 - Value Identifiers: Value Derivation Algorithms (based on the D1-DERIVAQTY Algorithm Type)
 - VEE Groups
 - Profile Factors
 - Scratchpad Measuring Component Types
 - Display Properties:
 - Event Bar Profiles: Values for the D1-360EventBarProfile extendable lookup
 - Final Values Overlay Profiles: Values for the D1-FinalValuesOverlayProfile extendable lookup.
3. Create your measuring component type business objects, referencing the configuration objects created above as appropriate.
4. Set up admin records that define the measuring component types you will use in your implementation.

Configuring Custom Measuring Components

Configuring custom measuring components involves the following steps:

1. Design the measuring component business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom measuring component-related configuration objects required for your business objects, including:
Business Object Options: Create business objects and algorithms for the following business object options:
 - Estimation Initial Measurement Data BO
 - Interval Initial Measurement Function
 - Manual Override IMD BO
 - Measuring Component Consumption Function
 - Synchronization Add BO
3. Create your measuring component business objects, referencing the configuration objects created above as appropriate.

Chapter 5

Service Points and Device Installation

This chapter provides descriptions of entities related to installation of meters, including service points, contacts, install events, activities, and other entities. This chapter includes:

- **Understanding Service Points and Device Installation**
- **Service Points In Detail**
- **Contacts In Detail**
- **Install Events In Detail**
- **Service Providers In Detail**
- **Processing Methods In Detail**
- **Configuring Service Point and Device Installation Objects**

Note: References to objects and options pre-fixed with “D2” are available only with Oracle Utilities Meter Data Management, and included for illustrative purposes only.

Understanding Service Points and Device Installation

This section provides an overview of entities related to the installation of devices (service points, contacts, install events, service providers, and others) how they are used in the meter data framework and related products, including Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway.

Service Points

Service points are physical locations at which a company supplies service. Devices are installed at service points. The relationship between individual service points and devices can change over time. For example, at any point in time:

- A service point may have a single device installed (or no device may be installed)
- A device may be installed at a single service point (or it may not be installed at a service points)

Over time:

- Different devices may be installed at a service point
- A device may be installed at different service points

An Aside: No Premise Object Exists

Oracle Utilities Meter Data Management (and other related meter data products) is not the system of record for premises or service points. The customer information system (or some other system) is considered the system of record for this type of information. In order to minimize the amount of data that needs to be synchronized, premise-oriented attributes used by meter data products are held on service points. This is an important distinction to keep in mind when creating custom service points for your implementation.

Contacts

Contacts are individuals or business entities with which a company has contact. Service points can have associated contacts which define the individual or business entity that uses the service (electricity, gas, water, etc.) delivered at the service point. Note that while contacts are optional for service points, usage subscriptions must reference contacts.

Note: The base-package name search on the 360° Search looks for usage subscription-related contacts. Use the Service Point Query portal to find a service point using a service point-related contact.

Installation Events

Whenever a device is installed at a service point, an installation event is created. Installation events capture the history of the devices that have been installed at a service point. This allows consumption for a service point to be calculated over time. In technical terms, installation events (or install events) link a specific device configuration to a service point.

While a device is installed at a service point, it may be turned off (and back on again). The installation event that records the original installation date and time also records the dates and times when the device has been turned on and off. When a device is removed, the original installation event is updated with the removal date and time.

Service Providers

Service providers are external entities that serve various roles relative to the application. These can include head-end systems, billing systems to which the application sends bill determinant data, market participants in a deregulated environment, outage management systems that receive meter event data from the application, or other parties that require or provide information to the system.

Service providers can have one or more associated processing methods that define the format or means by which a service provider receives data from the application, such as bill determinants, interval data, or meter events. Processing methods are also used to define how to create information internal to the application such as initial measurement data and usage transactions. Processing methods can also be used to define the information an external system wishes to subscribe to receive from our application.

Service Providers as Head-End Systems

Head-end systems are systems that collect measurement data and meter events for eventual submission to the application. Many devices can communicate to the application through a single head-end system, but a utility may have numerous head-end systems through which they communicate with devices.

As noted above, head-end systems are defined as service providers. Head-end systems utilize a processing method that specifies the type of initial measurement data to create for devices (and their related measuring components) based on measuring component type.

Service Providers in Deregulated Markets

Some utilities operate in deregulated markets. In implementations in deregulated markets, the system can send information to and receive information from a variety of market entities. These entities are defined as service providers.

For example, a service point's distribution company and/or energy supply company may subscribe to its consumption, or a service point's meter service provider may send requests to ping the meter that's installed at the service point to verify connectivity between the meter and its head-end system.

Different Relationship Types In Different Markets

Each market can define different relationship types between its service providers. A single instance of Oracle Utilities Meter Data Management or Oracle Utilities Smart Grid Gateway may have service points in different markets where each market has different relationship types and service providers. For example:

- In a regulated market the distribution company is the de facto energy supplier and meter service provider.
- Another market might have two relationship types and a single service provider for each relationship:
 1. There is a single energy supply company for the entire market
 2. There is a single meter service provider for the entire market
- Yet another market might have two relationship types (energy supply and meter service). In this market, there might be multiple service providers for each relationship type. Each service point can choose any of the relationship type's service providers. If a service point does not declare a specific service provider for a given relationship type, the relationship type's "fallback" service provider is assumed.

Measurement Cycles

Measurement cycles define the schedule for manual meter reading of devices at service points. More specifically:

- A **measurement cycle** defines WHEN the service point is visited
- A **route** within a cycle defines a group of service points in a cycle that are visited by a meter reader
- A **sequence** within a route defines the physical position of the service point within a route
- A **schedule** specifies the dates on which service points are visited.

Manually read service point reference a measurement cycle, route and sequence within the route. A batch process creates SP/Measurement Cycle Schedule Routes for each service point, which link the dates on a measurement cycle schedule to a measurement cycle route defined for the service point. These define the specific date on which a meter reader will visit service points associated with a specific measurement cycle route, and the sequence in which the service points on that route are visited.

Measurement Cycles can also be used to periodically push bill determinants to subscribing systems. See **Measurement Cycle And Creating Bill Determinants** below for more information.

Measurement Cycle Batch Processing

Measurement cycle processing is managed by the following three batch processes:

- Create Pending Measurement Cycle Schedule Routes (D1-CMCS)

This batch process creates Schedule Routes for Measurement Cycle Schedules whose schedule selection date is on or before the batch business date. This process is used if routes have the same schedule each month, quarter, etc. This process simply copies the routes from the Measurement Cycle to the Measurement Cycle Schedule on/after the scheduled selection date.

- Create Pending SP / Measurement Cycle Schedule Route Records (D1-CSPSR)

This batch process creates a “SP/Measurement Cycle Schedule Route” transaction for every service point in the Measurement Cycle Schedule Route that is ready for processing.

- Process Pending SP / Measurement Cycle Schedule Route Records (D1-PSPSR)

This batch process transitions the Pending “SP/Measurement Cycle Schedule Route” transactions to their Complete state. Custom algorithms can be configured to do any additional necessary work, such as creating a “Meter Read Download” activity. This custom algorithm would be configured as an Enter algorithm on the “Complete” state of the SP/Measurement Cycle Schedule Route business object.

Measurement Cycle And Creating Bill Determinants

The system can be configured to periodically push bill determinants to subscribing systems. In this case, measurement cycles can be configured to define when to create usage transactions for usage subscriptions associated to service points in the cycle. In this case, even service points whose meters are read automatically may reference measurement cycles.

Creating bill determinants (by creating a usage transaction) is performed by an algorithm on the “Complete” state of the SP/Measurement Cycle Schedule Route business object (similar to creating activities as described above).

When the Pending SP/Measurement Cycle Schedule Route records are processed by the “Process Pending SP / Measurement Cycle Schedule Route Records” process (D1-PSPSR), rather than create a handheld download activity, the algorithm can create a usage transaction (usage

transactions are transactions that cause bill determinants to be calculated for the service point's usage subscription(s)).

If the implementation needs to both manually read the meter and push bill determinants, both algorithms would be plugged in on the SP/Measurement Cycle Schedule Route business object.

Service Points In Detail

This section provides details concerning the service point objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom service point objects you create as part of your implementation. This section includes:

- A description of the D1-SP maintenance object
- Lists of the base package service point business objects, including “lite” business objects
- Details concerning service point-specific configuration options
- A sample service point business object (D1-ServicePoint)

Maintenance Object - D1-SP

Service point business objects use the D1-SP maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-SP
Description	Service Point
Service Name	D1-SP (Service Point Maintenance)
Tables	<ul style="list-style-type: none">• D1_SP (Service Point) - Primary• D1_SP_CHAR (Service Point Characteristics) - Child• D1_SP_CONTACT (Service Point Contact) - Child• D1_SP_IDENTIFIER (Service Point Identifier) - Child• D1_SP_LOG (Service Point Log) - Child• D1_SP_LOG_PARM (Service Point Log Parameter) - Child• D1_SP_MKT_PARTICIPANT (Service Point Market Participant) - Child• D1_SP_REL (Service Point Relationship) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Service Point Business Objects

The meter data framework base package includes the following service point business objects:

Business Object Name	Description
D1-ServicePoint	Service Point Instances of this business object represent individual service points defined in the system.

The meter data framework base package includes the following “lite” service point business objects:

Business Object Name	Description
D1-ServicePointParentLITE	Service Point Parent LITE
D1-SPLite	Service Point Lite

The meter data framework base package includes the following additional service point business objects:

Business Object Name	Description
D1-SynchronizationAddSP	Service Point Synchronization Add (used when adding a new service point as a result of a data synchronization request)

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by service point business objects.

Business Object Options

Service point business objects can make use of the following business object options:

- **Synchronization Add BO:** This option identifies the business object to use when adding new service points as a result of a data synchronization request.

System Events

Service point business objects can make use of the following system events:

- **Process Measurement Cycle:** This system event specifies the algorithm invoked by the Process Measurement Cycle Schedule batch process. This batch process looks for measurement cycle routes that are scheduled to be processed and populates them on the measurement cycle schedule. Once the schedule is prepared, the batch process invokes this algorithm for each device in every route that is ready for processing. The Algorithm Entity for these algorithms is “Service Point (BO) - Process Measurement Cycle.” The base package provides the following algorithm types/algorithms for use with this system event:

Algorithm Type / Algorithm	Description
D1-CRMRDACT	Create Meter Read Download Activity (Creates a meter read download activity for a service point. The service point must have a measurement cycle and route whose schedule indicates a schedule type of “Meter Read Download”)
D2-CRUSGTRN	Create Usage Transaction (Creates a usage transaction for every active usage subscription linked to the service point. The business object used to create the usage transaction is determined from the processing method of the usage subscription's service provider.)

Use the Algorithm Type / Algorithm portal and the Application Viewer to view more details about these algorithms and algorithm types.

Example Service Point - D1-ServicePoint

The table below lists the details of the D1-ServicePoint service point business object.

Option	Description
Business Object	D1-ServicePoint
Description	Service Point
Maintenance Object	D1-SP (Service Point)
Application Service	D1-SPBOAS (Service Point BO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Synchronization Add BO: D1-SynchronizationAddSP (Service Point Synchronization Add) • Display UI Map: D1-SPDisplay (Service Point - Display) • Portal Navigation Option: d1spTabMenu (Service Point) • Pre-Processing Service Script: D1-SPMtnPre (Service Point Maintenance Pre-Processing) • Display Map Service Script: D1-SPRetDts (Service Point - Retrieve Details for Display) • Maintenance UI Map: D1-SPMaint (Service Point - Maintenance)
Algorithms	<ul style="list-style-type: none"> • Process Measurement Cycle: D1-CRMRDACT (Create Meter Read Download Activity) • Information: D1-SPINFO (Service Point Information) • Pre-Processing: D1-DEF*TIMZON (Default Time Zone Value based on Installation Option) • Validation: D1-SPADDRVAL (Validate Service Point's Address Information) • Validation: D1-MSCYCVAL (Validate Service Point's Measurement Cycle Information) • Validation: D1-VALTIMZON (Validate BO Time Zone Against Installation Option)
Lifecycle	<ul style="list-style-type: none"> • Active (Initial) • Inactive (Final)

Use the Business Object portal to view additional details concerning this business object.

Contacts In Detail

This section provides details concerning the contact objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom contact objects you create as part of your implementation. This section includes:

- A description of the D1-CONTACT maintenance object
- Lists of the base package contact business objects
- A sample contact business object (D1-Business)

Maintenance Object - D1-CONTACT

Contact business objects use the D1-CONTACT maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-CONTACT
Description	Contact
Service Name	D1-CONTACT (Contact Maintenance)
Tables	<ul style="list-style-type: none">• D1_CONTACT (Contact) - Primary• D1_CONTACT_CHAR (Contact Characteristics) - Child• D1_CONTACT_EMAIL (Contact Email) - Child• D1_CONTACT_IDENTIFIER (Contact Identifier) - Child• D1_CONTACT_NAME (Contact Name) - Child• D1_CONTACT_PHONE (Contact Phone) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Contact Business Objects

The meter data framework base package includes the following contact business objects:

Business Object Name	Description
D1-Business	Business Instances of this business object represent individual business contacts defined in the system.
D1-Person	Person Instances of this business object represent individual person contacts defined in the system.

The meter data framework base package includes the following additional contact business objects:

Business Object Name	Description
D1-SynchronizationAddContact	Contact Synchronization Add (used when adding a new contact as a result of a data synchronization request)

Example Contact - D1-Business

The table below lists the details of the D1-Business contact business object.

Option	Description
Business Object	D1-Business
Description	Business
Maintenance Object	D1-CONTACT (Contact)
Application Service	D1-CONTACT (Contact MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> Synchronization Add BO: D1-SynchronizationAddContact (Contact Synchronization Add) Display UI Map: D1-BusinessDisplay (Business - Display) Portal Navigation Option: d1contctTabMenu (Contact) Display Map Service Script: D1-RtBusDtl (Contact - Retrieve Details for Display) Maintenance UI Map: D1-BusinessMaint (Business - Maintenance)
Algorithms	<ul style="list-style-type: none"> Information: D1-BUS-INFO (Business Contact - Information)

Use the Business Object portal to view additional details concerning this business object.

Install Events In Detail

This section provides details concerning the install event objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom install event objects you create as part of your implementation. This section includes:

- A description of the D1-INSTLEVT maintenance object
- Lists of the base package install event business objects, including “lite” business objects
- A sample install event business object (D1-SmartMeterInstallEvent)

Maintenance Object - D1-INSTLEVT

Install event business objects use the D1-INSTLEVT maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-INSTLEVT
Description	Install Event
Service Name	D1-INSTLEVT (Install Event Maintenance)
Tables	<ul style="list-style-type: none"> • D1_INSTLEVT (Install Event) - Primary • D1_INSTLEVT_CHAR (Install Event Characteristics) - Child • D1_INSTLEVT_LOG (Install Event Log) - Child • D1_INSTLEVT_LOG_PARM (Install Event Log Parameter) - Child • D1_ON_OFF_HIST (On/Off History) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Install Event Business Objects

The meter data framework base package includes the following install event business objects:

Business Object Name	Description
D1-ManualMeterInstallEvent	Manual Meter Installation Event Instances of this business object represent individual manual meter installation events defined in the system.
D1-SmartMeterInstallEvent	Smart Meter Installation Event Instances of this business object represent individual smart meter installation events defined in the system.

The meter data framework base package includes the following “lite” install event business objects:

Business Object Name	Description
D1-InstallEventLite	Install Event Lite
D1-InstallEventMainLite	Install Event Main LITE
D1-InstallEventParentLITE	Install Event Parent LITE
D1-SmartMeterInstallEventLite	Smart Meter Install Event LITE

The meter data framework base package includes the following additional install event business objects:

Business Object Name	Description
D1-SynchronizationAddIE	Install Event Synchronization Add (used when adding a new installation event as a result of a data synchronization request)

Example Install Event - D1-SmartMeterInstallEvent

The table below lists the details of the D1-SmartMeterInstallEvent install event business object.

Option	Description
Business Object	D1-SmartMeterInstallEvent
Description	Smart Meter Installation Event
Maintenance Object	D1-INSTLEVT (Install Event)
Application Service	D1-SMTMTRINSEVTBOAS (Smart Event Installation Event BO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none">• Synchronization Add BO: D1-SynchronizationAddIE (Install Event Synchronization Add)• Display UI Map: D1-SmartMeterInstallEventDisplay (Smart Meter Install Event - Display)• Portal Navigation Option: d1inevtmTabMenu (Install Event)• Display Map Service Script: D1-SmtIERtDt (Smart Meter Install Event - Retrieve Details for Display)• Maintenance UI Map: D1-SmartMeterInstallEventMaint (Smart Meter Install Event - Maintenance)

Option	Description
Algorithms	<ul style="list-style-type: none"> • Information: D1-INEVTINFO (Install Event Information) • Pre-Processing: D1-DFLTINSTC (Default the Install Event's Installation Constant) • Pre-Processing: D1-POPARMSTS (Default the Arming Status) • Validation: D1-DEVCFGVAL (Validate Device the Configuration) • Validation: D1-ONHISTVAL (Validate the On/Off History based on the Previous Install Event) • Validation: D1-CHKHISEVT (Validate the On/Off History based on the Install Event's Status) • Validation: D1-CKIFOVLEX (Validate Overlapping Install Events) • Validation: D1-VALIERMDT (Validate Removal Info)
Lifecycle	<ul style="list-style-type: none"> • Pending (Initial) • Connected/Pre-Commissioned (Interim) • Pre-Connected/Commissioned (Interim) • Connected/Commissioned (Interim) • Connected/Decommissioned (Interim) • Disconnected/Commissioned (Interim) • Disconnected/Decommissioned (Interim) • Remove (Final)

Use the Business Object portal to view additional details concerning this business object.

Service Providers In Detail

This section provides details concerning the service provider objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom service provider objects you create as part of your implementation. This section includes:

- A description of the D1-SVCPROVDR maintenance object
- Lists of the base package service provider business objects, including “lite” business objects
- Details concerning service provider-specific configuration options
- A sample service provider business object (D1-HeadEndSystem)

Maintenance Object - D1-SVCPROVDR

Service provider business objects use the D1-SVCPROVDR maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-SVCPROVDR
Description	Service Provider
Service Name	D1-SVCPROVDR (Service Provider Maintenance)
Tables	<ul style="list-style-type: none">• D1_SPR (Service Provider) - Primary• D1_SPR_CHAR (Service Provider Characteristics) - Child• D1_SPR_L (Service Provider Language) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Service Provider Business Objects

The meter data framework base package includes the following service provider business objects:

Business Object Name	Description
D1-ExternalApplication	External Application Instances of this business object represent individual external applications defined in the system.
D1-HeadEndSystem	Head-End System Instances of this business object represent individual head-end systems defined in the system.
D1-MarketParticipant	Market Participant Instances of this business object represent individual market participants defined in the system.

Business Object Name	Description
D1-ServiceProvider	Service Provider (generic service provider business object used as a parent business object for all other service provider business objects)

The meter data framework base package includes the following “lite” service provider business objects:

Business Object Name	Description
D1-MarketParticipantLite	Market Participant Lite

The meter data framework base package includes the following additional service provider business objects:

Business Object Name	Description
D1-ServiceProviderBundlingAdBO	Bundling Add BO for Service Provider
D1-ServiceProviderPhysicalBO	Physical BO for Service Provider

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by service provider business objects.

Business Object Options

Service provider business objects can make use of the following business object options:

- **Service Provider Type:** This option defines the type of service provider. Valid values are defined as values for the SPR_TYPE_FLG lookup field. The base package includes the following options:

Lookup Field Value	Description
D1EA	Edge Application
D1HE	Head-End System

Example Service Provider - D1-HeadEndSystem

The table below lists the details of the D1-HeadEndSystem service provider business object.

Option	Description
Business Object	D1-HeadEndSystem
Description	Head-End System
Maintenance Object	D1-SVCPROVDR (Service Provider)
Parent Business Object	D1-ServiceProvider
Application Service	D1-SVCPROVIDER (Service Provider MO)
Instance Control	Allow New Instances

Option	Description
Options	<ul style="list-style-type: none">• Service Provider Type: D1HE (Head-End)• Display UI Map: D1-HeadEndSystemDisplay (Head-End System - Display)• Portal Navigation Option: d1svcproTabMenu (Service Provider)• Maintenance UI Map: D1-HeadEndSystemMaint (Head-End System - Maintenance)

Use the Business Object portal to view additional details concerning this business object.

Processing Methods In Detail

This section provides details concerning the processing method objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom processing method objects you create as part of your implementation. This section includes:

- A description of the D1-PROCMETHOD maintenance object
- Lists of the base package processing method business objects, including “lite” business objects
- Details concerning processing method-specific configuration options
- A sample service processing method object (D1-HowToCreateMCInformation)

Maintenance Object - D1-PROCMETHOD

Processing method business objects use the D1-PROCMETHOD maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-PROCMETHOD
Description	Processing Method
Service Name	D1-PROCMETHOD (Processing Method Maintenance)
Tables	<ul style="list-style-type: none"> • D1_PROC_METH (Processing Method) - Primary • D1_PROC_METH_CHAR (Processing Method Characteristics) - Child • D1_PROC_METH_L (Processing Method Language) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Business Objects

The meter data framework base package includes the following processing method business objects:

Business Object Name	Description
D1-AbstractProcessingMethod	Generic Processing Method (generic processing method business object used as a parent business object for all other processing method business objects)
D1-HowToCreateActivityOBComm	How to Create Outbound Communication / Send OB Message (used to determine the specific type of outbound communication/message to send)
D1-HowToCreateMCInformation	How To Create MC Related Information (used to define how measuring component-related information is created for the service provider, including initial measurement data)

Business Object Name	Description
D1-HowToProcDvceEvtsInformation	How to Process Device Event Related Information (used to send device events to a service provider)
D1-HowToProcessDeviceInfo	How to Process Device Related Information (used to define how UOM codes are mapped for devices of the head-end system service provide)
D1-HowToSendActInformation	How to Send Activity Related Informatoin (used to define how activity-related information is sent to the service provider)
D1-HowToSendActivityResponse	How to Send Activity Related Outbound Messages (used to define the outbound message type sent to a service provider in response to an activity)

The meter data framework base package includes the following additional processing method business objects:

Business Object Name	Description
D1-ProcessingMethodBundlingABO	Bundling Add BO for Processing Method
D1-ProcessingMethodPhysicalBO	Physical BO for Processing Method

Configuration Options

This section outlines specific types of BO Options and System Events used by service provider business objects.

Business Object Options

Service provider business objects can make use of the following business object options:

- **Applicable Processing Role:** This option defines the processing role for the processing method. Valid values are defined as values for the PROC_ROLE_FLG lookup field. The base package includes the following options:

Lookup Field Value	Description
D1AM	Obtain AMI Device Identifier
D1DA	Activity Notification
D1DC	Device Commission
D1DD	Device Decommission
D1DM	Device Event Mapping
D1DS	Device Status Check
D1EP	Event Processing Default Configuration
D1FR	Response - Fail
D1IM	Initial Measurement Creation

Lookup Field Value	Description
D1IN	On-Demand Read (Interval)
D1LC	Load Check
D1RC	Remote Connect
D1RD	Remote Disconnect
D1RM	Retrieve Meter
D1RR	Response - Received
D1SC	On-Demand Read (Scalar)
D1SD	Send Device Event
D1SR	Response - Success
D1UM	UOM Mapping

System Events

Service provider business objects can make use of the following system events:

- Determine Processing Method(s):** This system event defines the algorithm used to determine the processing methods (business object or batch code) to use, based on the related entity. The Algorithm Entity for available algorithms is “Proc Method (BO) - Determine Proc Method.” The base package includes the following algorithm types and algorithms for use with this system event:

Algorithm Type / Algorithm	Description
D1-BODIFFAT / D1-BODIFFAT	BO / Batch Differs By Activity Type (returns the BO code or batch code used to send activity-related information to a service provider for a processing role)
D1-BODIFFAT / D1-BOBDIFFAT	BO / Batch Differs By Activity Type (returns the BO code or batch code used to send activity-related information to a service provider for a processing role)
D1-BODIFFDT / D1-BODIFFDT	BO Differs by Device Type (returns the BO code used in the creation of device-oriented information for a service provider and processing role)
D1-BODIFFMCT / D1-BODIFFMCT	BO Differs By Measuring Component Type (returns the BO code used to create measuring component-related information for a service provider and processing role)
D1-DIFDVETC / D1-DIFDVETC	Method Differs By Device Event Category (returns the BO or outbound message type or batch process used to send device events to a service provider)
D1-OCDDT / D1-OCDDT	Outbound Communication Differs by Device Type (returns the BO used to send messages to a service provider)

Algorithm Type / Algorithm	Description
D1-OMTDAT / D1-OMTDAT	Outbound Message Type Differs by Activity Type (returns the outbound message type used to create outbound messages for a service provider and processing role or message category/number if outbound message creation is not supported)
D1-OMTDCT / D1-OMTDCT	Outbound Message Type Differs By Communication Type (returns the outbound message type used to create outbound messages for a service provider and processing role or message category/number if outbound message creation is not supported))

Example Processing Method - D1-HowToCreateMCInformation

The table below lists the details of the D1-HowToCreateMCInformation business object.

Option	Description
Business Object	D1-HowToCreateMCInformation
Description	How to Create Measuring Component Related Information
Maintenance Object	D1-PROC METHD (Processing Method)
Parent Business Object	D1-AbstractProcessingMethod
Lifecycle Business Object	Generic Processing Method
Application Service	D1-PROC METHD (Processing Method MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> Applicable Processing Role: D1IM Portal Navigation Option: d1svcproTabMenu (Service Provider) Maintenance UI Map: D1-HowToCreateMCInfoMaint (How To Crt MC Related Info - Maintenance)
Algorithms	<ul style="list-style-type: none"> Determine Processing Method(s): D1-BODIFFMCT (BO Differs by Measuring Component Type) Validation: D1-PROMTHVAL (Validate Processing Method)

Use the Business Object portal to view additional details concerning this business object.

Configuring Service Point and Device Installation Objects

This section provides high-level overviews of the steps involved in configuring custom service points, contacts, install events, service providers, and activities. See **Configuration Process Overview** in **Chapter One** for a high-level overview of the overall configuration process.

Note: The procedures below focus on specific configuration tasks and options related to each of the objects described in this chapter, and do not address all the steps involved in creating business objects, UI maps, algorithms, etc. For more information about these subjects, refer to the Oracle Utilities Application Framework documentation.

Configuring Custom Service Points

Configuring custom service points involves the following steps:

1. Design the service point business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom service point-related configuration objects required for your business objects, including:
Business Object Options: Create algorithms for the following business object options:
 - Process Measurement Cycle
3. Create your service point business objects, referencing the configuration objects created above as appropriate.
4. Set up admin records that define the service point types you will use in your implementation.

Configuring Custom Contacts

Configuring custom contacts involves the following steps:

1. Design the contact business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom contact-related configuration objects required for your business objects.
3. Set up admin records that define the contact types you will use in your implementation.

Configuring Custom Install Events

Configuring custom install events involves the following steps:

1. Design the install event business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom install event-related configuration objects required for your business objects.

Configuring Custom Service Providers

Configuring custom service providers involves the following steps:

1. Design the service provider business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom service provider-related configuration objects required for your business objects, including:

Processing Methods: Create processing method business objects for use with each service provider.

3. Create your service provider business objects, referencing the configuration objects created above as appropriate.

Note: Service provider business objects should reference D1-ServiceProvider as their Parent Business Object.

Configuring Custom Processing Methods

Configuring custom processing methods involves the following steps:

1. Design the processing method business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom processing method-related configuration objects required for your business objects, including:

System Events: Create algorithms for the following system events:

- Determine Processing Method(s)

3. Create your processing method provider business objects, referencing the configuration objects created above as appropriate.

Note: Processing method business objects should reference D1-AbstractProcessingMethod as their Parent Business Object.

Chapter 6

Measurement Data

This chapter provides descriptions of initial and final measurement data, including:

- **Understanding Initial Measurement Data and Final Measurements**
- **Initial Measurement Data In Detail**
- **Measurements In Detail**
- **Configuring Initial Measurements and Measurements**

Understanding Initial Measurement Data and Final Measurements

This section provides an overview of initial measurements and final measurements and how they are used in Oracle Utilities meter data framework and Oracle Utilities Meter Data Management, including:

- **Initial Measurement Data**
- **Final Measurements**
- **Daylight Saving Time Support**

Initial Measurement Data

Measurements read from a measuring component are referred to as “initial measurement data” (or initial measurements) and are used to record how much of the quantity (defined by UOM, TOU, and SQI) measured by the measuring component was consumed.

Initial measurement data for scalar measuring components contain a single “reading” or value, while initial measurement data for interval measuring components can contain multiple readings, one for each interval that falls between the start time and stop time of the measurement.

At a simple level, initial measurement data goes through the following process:

1. Initial measurements are loaded into the system.
2. Initial measurement data is validated, edited, and estimated.
3. Initial measurements are converted into final measurements.
4. Final measurements are used to calculate usage (bill determinants, etc.).

Creating Initial Measurements

The IMD Seeder business object (D1-IMDSeeder) is used to create initial measurements (via instantiating initial measurement business objects), based on the head-end system sending the measurement. The “Initial Measurement Creation” processing method defined for the head-end system service provider defines the specific type of initial measurement to create. If for some reason an initial measurement can’t be created, an instance of the IMD Seeder business object is created to allow tracking of the error (and once the error is resolved, the IMD Seeder instance can be reprocessed).

The IMD Seeder business object determines the service provider and measuring component for the measurement (based on attributes supplied in the incoming reading, such as device ID, external reference ID, etc.). This is performed by the “Derive Service Provider and Measuring Component” pre-processing algorithm (D1-DER-SPRMC).

The IMD Seeder also ensures that the Start and End Date/Time fields on the initial measurement are populated, deriving them from the incoming reading if necessary. This step is performed by the “Derive IMD Date/Time Values” pre-processing algorithm (D1-VALDR-INP).

Critical Validations

The IMD Seeder also performs a series of critical validations to ensure that the incoming reading contains valid data. These critical validations include validating that the device identifier supplied is valid and exists in the system, and performing Undercount and Overcount checks for interval readings.

- **Device Identifier:** Device identifier checks validate that the device identifier provided with the measurement is valid. Device identifiers can include serial number, badge number, channel ID, etc.
- **Undercount:** An undercount occurs when an interval initial measurement contains fewer interval values than appropriate based on the interval size (or seconds-per-interval, or SPI) and the Start and End Date/Time values. For example, if an initial measurement has an

interval size of one hour (or an SPI of 3600), and a Start Date/Time is October 27, 2011 and End Date/Time of October 28, 2011 (a total duration of 1 day), it should contain 24 interval values. If it contained less than 24 interval values, it could constitute an undercount.

- **Overcount:** An overcount occurs when an interval initial measurement contains more interval values than appropriate based on the interval size (or seconds-per-interval, or SPI) and the Start and End Date/Time values. For example, if an initial measurement has an interval size of one hour (or an SPI of 3600), and a Start Date/Time is October 27, 2011 and End Date/Time of October 28, 2011 (a total duration of 1 day), it should contain 24 interval values. If it contained more than 24 interval values, it could constitute an overcount.

If an incoming interval initial measurement contains either an invalid device identifier, an undercount, or an overcount, the measurement is rejected, and an instance of the IMD Seeder business object is created. Undercount and Overcount checks are performed by the “Perform Date/Time Adjustments and Undercount/Overcount Check” pre-processing algorithm (D1-DODTTMADJ).

Validation, Editing, and Estimation

Once received into the system, initial measurements are subject to validation, editing, and estimation. This process involves the following:

- **Validation:** Validates that the initial measurement data is within expected tolerances, and is correct
- **Editing:** If the initial measurement data is wrong in some way, the data can be automatically changed.
- **Estimation:** If initial measurement data is incomplete (for example, if one or more interval values within an interval measurement are missing), missing values can be automatically estimated.

As noted above, the values recorded in an initial measurement can change during the validation, editing, and estimation processing, and exceptions can be raised if initial measurements are wrong in some way (such as out-of-tolerance quantities, incorrect values, etc.).

Note: The validation, editing, and estimation process is referred to as VEE, and is described in more detail in a later chapter.

Skipping VEE Processing - High Quality Check for Interval Initial Measurements

Execution of VEE rules is performed by an Enter algorithm on the “VEE Complete” state of the initial measurement business object. For interval initial measurements, the “High Quality Check - Vector Band Based” (D1-HIGHQUALV) algorithm can be used to skip VEE processing and transition the initial measurement directly to the “Finalized” state.

This algorithm checks whether the intervals within an initial measurement are considered to be of “high quality” (defined below), and if they are considered “high quality”, the algorithm transitions the current initial measurement directly to the “Finalized” state.

An interval initial measurement is considered to be of “high quality” if the following conditions are met:

- There are no missing intervals in the initial measurement
- All interval values are within a range that extends above and below final interval values for the corresponding intervals from the previous reading. The range is defined by the “Low Tolerance” and “High Tolerance” algorithm parameters. These parameters define multipliers that are applied to the corresponding interval from the previous reading. The “Low Tolerance” is subtracted from the previous reading’s final measurement, and the “High Tolerance” is added to the previous reading’s final measurement to define the range within which each interval must fall.

For example, if the previous reading's final measurement value at 1:00 AM was 20, and the "Low Tolerance" parameter is set to 0.2 and the "High Tolerance" parameter is set to 0.25, the initial measurement's value at 1:00 AM must be between 16 ($20 - 0.2 \times 20$) and 25 ($20 + 0.25 \times 20$).

- All intervals have a condition code that falls between the range defined by the "Default Bottom Range Condition Value" and "Default Top Range Condition Value" algorithm parameters.

If the initial measurement fails any part of the high quality assessment, the routine simply exits.

Pre VEE and Post VEE Quantities

Initial measurement data contains both the original and final versions of the quantities recorded by the measuring component.

- **Pre VEE** quantities are consumption values derived from the measurements recorded by the head-end system or meter reader.
- **Post VEE** quantities are the "final" values, after VEE processing.

Pre VEE and Post VEE quantities in an initial measurement often differ based on a number of conditions, including:

1. The measuring component has a multiplier other than 1.

In this case, the Post VEE value is equal to the Pre VEE value times the multiplier.

2. The installation event has a constant other than 1.

In this case, the Post VEE value is equal to the Pre VEE value times the installation constant.

3. VEE rules have changed the quantities because they are missing or obviously wrong

In this, the Pre VEE values are adjusted based on the specifics of the VEE rules applied to the initial measurement to create the Post VEE values

4. Manual changes by a user.

Condition Codes

In addition to recorded consumption values, measurements also have condition codes, used to indicate the source and quality of a measurement. For example:

- Regularly recorded measurements might have a condition code of "Regular"
- Missing measurements might have a condition code of "Missing"
- Estimated measurements might have a condition code of "External Estimated" or "System Estimated" based on where the estimation was performed.

Both Pre VEE and Post VEE values have their own condition code, which can also change during VEE processing. For example, consider the following sample measurements from an interval measuring component:

Date / Time	Pre VEE Value	Pre VEE Condition	Post VEE Value	Post VEE Condition
01/01/2010 3:00 PM	14.678	Regular	15.1	Regular
01/01/2010 4:00 PM		Missing	20	Estimated
01/01/2010 5:00 PM	13.12	Regular	13.41	Regular
01/01/2010 6:00 PM	150.12	Regular	14.12	Estimated

For the 4:00 PM interval, note the Pre VEE condition indicates the interval is missing and the Post VEE condition highlights that it was estimated.

For the 6:00 PM interval (containing a spike, or an interval with conspicuously high usage relative to surrounding intervals), note that the system head-end (the system that recorded the measurement) indicated the interval value was fine (Pre VEE is regular), but the VEE process smoothed it, and set the Post VEE condition to “Estimated.”

Subtractive Measuring Components

Initial measurement data for subtractive measuring components (such as most scalar measuring components) also typically contain start and stop readings in addition to Pre and Post VEE usage. For example, a set of initial measurements for a subtractive scalar measuring component might look like the following:

Date / Time	Start Reading	Stop Reading	Pre VEE Usage	Pre VEE Condition	Post VEE Value	Post VEE Usage
01/01/2010 3:00 PM	0	1500	1500	Regular	1515	Regular
02/2/2010 4:11 PM	1500	2100	600	Regular	606	Regular
03/03/2010 5:22 PM	2100	2900	800	Regular	808	Regular
04/01/2010 1:00 PM	2900	3500	600	Regular	606	Regular

Rollover Calculations

Subtractive measuring components can “rollover” when the reading exceeds the maximum value based on the number of dials. For example, a register with a 4 dials can record values up to 9999 before rolling over to 0000. When this occurs, consumption is calculated based on the following attributes and calculated values.

- **Rollover Threshold** is the percentage of the measuring component's dial capacity at which measurements for measuring components of this type are considered to have rolled over. Dial capacity is the largest value that can be recorded for the measuring component, based on the measuring component's number of dials. For example, a measuring component with 5 dials has a dial capacity of 99999.
- **MaxDialCapacity** is the maximum value for the number of dials, rounded up to the next whole multiple of 10 (or 10 raised to the power of the number of dials). For example, for a register with 4 dials, the MaxDialValue is 10000.
- **MaxAcceptableDifference** is the maximum acceptable consumption that can be recorded for the register. This is equal to the MaxDialCapacity multiplied by the rollover threshold. For example, the MaxAcceptableDifference for a register with 4 dials and a rollover threshold of 90% would be 9000. If the consumption is greater than this value, the initial measurement is transitioned to the Error state.
- **Difference:** The difference between the Stop Reading and Start Reading, obtained by subtracting the Start Reading from the Stop Reading. If the Difference is less than zero (<0), then add the MaxDialCapacity to calculate Rollover.
- **Rollover:** The adjusted consumption for a reading on a register that has rolled over. Only applicable if the Difference (Stop Reading - Start Reading) is less than zero (<0).
- **Consumption:** The calculated consumption for the reading, equal to either the Difference or Rollover. If the Difference is greater than or equal to zero, consumption is equal to the Difference. If the Difference is less than zero (<0), and the Rollover is less than or equal to the MaxAcceptableDifference, the consumption is equal to the Rollover.

Example: Consider an initial measurement with the following attributes:

- Number of Dials: 4

- Rollover Threshold: 90 (%)
- Start Reading: 8900
- Stop Reading: 0500

For this reading,

MaxDialCapacity = 10000

MaxAcceptableDifference = 9000 (10000 * .90)

Difference = 0500 (Stop Reading) - 8900 (Start Reading) or **-8400**

Rollover = 10000 (MaxDialCapacity) + -8400 (Difference) or **1600**

Consumption is equal to **1600** (Rollover).

Estimated Initial Measurements

Over the course of time, it may happen that the system will not receive usage for a device for some period of time. When the system detects that a measuring component is missing final measurements, it can create an initial measurement via estimation. This type of initial measurement is referred to as an estimated initial measurement (as opposed to an Initial Load or Manual initial measurement).

At a high-level, the estimation process is as follows:

- Missing Measurements are detected by a “Period Estimation” system event algorithm on the measuring component business object
- Estimated initial measurements are created for the “missing” time period by the “Period Estimation” system event algorithm on the measuring component business object
- Values and consumption for the estimated initial measurements are calculated by “estimation” VEE rules.

It's important to note that the processes that detect missing measurements do NOT themselves estimate consumption. Rather, these detection processes simply create an initial measurement and let the estimation VEE rules estimate the consumption for the initial measurement.

Detecting Missing Measurements

The detection of missing measurements occurs at different points in time for interval and scalar measuring components:

- For interval measuring components, a dedicated batch process exists to initiate creation of estimated initial measurements
- For manually read scalar measuring components, a usage calculation rule creates the initial measurement if it cannot find a measurement and it has been given permission to estimate. (Note: the information that follows applies primarily to interval measuring components).

The batch process that detects missing consumption for interval measuring components uses two elements on interval measuring component type:

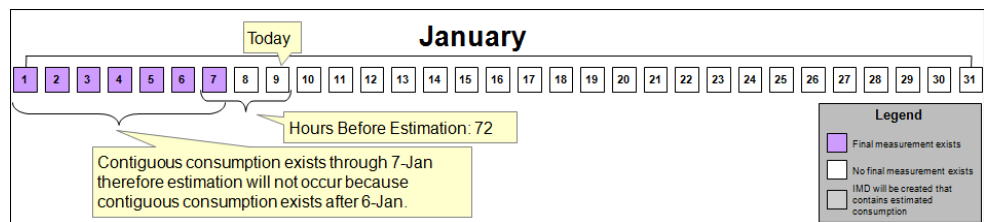
- **Hours Before Estimation:** the number of hours after the End Date and Time of the most recent measurement that must pass before the measuring component is considered due for estimation. For example, if set to 72 hours, estimation will only take place 72 hours after the End Date and Time of the latest measurement.
- **Number of Hours to Estimate:** the number of hours of measurement data that are estimated when estimation is performed for the measuring component.

If a measuring component has contiguous final measurements on/after a date and time equal to the current date/time minus the “Hours Before Estimate” value, the measuring component is not estimated.

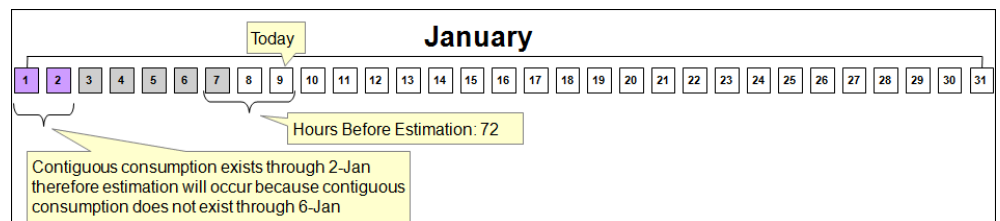
If the measuring component does NOT have contiguous final measurements on/after a date and time equal to the current date/time minus the “Hours Before Estimate” value, the measuring component is estimated. If the measuring component is subject to estimation, measurements will be estimated through the a date and time equal to the current date/time minus the (Hours Before Estimation - Number of Hours to Estimate).

The following examples illustrate how the system determines if estimation should take place. These examples are all assume a measuring component whose measuring component type specifies an “Hours Before Estimation” of 72 (3 days) and a “Number of Hours to Estimate” of 24 (1 day).

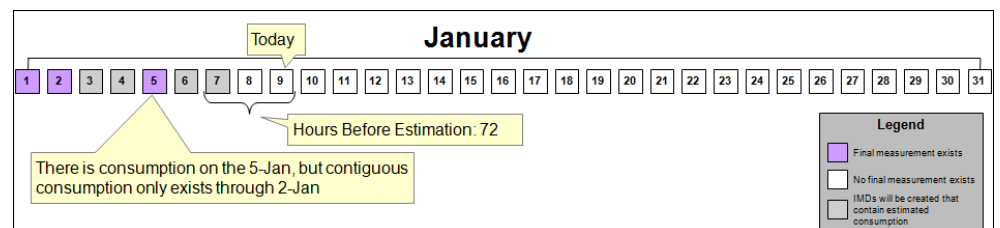
Example 1: On January 9, if there is contiguous consumption through January 7, estimation would not occur because consumption exists after January 6 (3 days prior to January 9). The diagram below illustrates this example:



Example 2: On January 9 if contiguous consumption existed through January 2, estimation would take place because consumption does not exist through January 6 (3 days prior to January 9). In this example, consumption would be estimated for missing intervals from January 2 through January 7 (January 9 minus 48 hours (Hours Before Estimation minus Number of Hours to Estimate)). Note that this assumes that period estimation had NOT been run after January 5. The diagram below illustrates this example:



Example 3: On January 9 if contiguous consumption existed through January 2 and for January 5 (but NOT for January 3-4 or 6-7), two initial measurements would be created, one for January 3-4, and another for January 6-7. The diagram below illustrates this example:



Estimation Algorithms

Estimation for interval measuring components is detected and initiated by two algorithms, one on the device business object, and the other on the measuring component business object.

Devices whose measuring components are subject to periodic estimation (in most cases, interval measuring components) have a “Period Estimation” Monitor algorithm (D1-PERESTM) on their Active state. This monitoring algorithm retrieves the device's measuring component and invokes the algorithm on the “Periodic Estimation” system event on the measuring component business object. Note - the device monitoring algorithm can be configured to look at device configurations

from a number of days in the past defined by an algorithm parameter. The base package includes this algorithm on the Active state of the Smart Meter (D1-SmartMeter) device business object. The “Periodic Estimation” monitor algorithm can be triggered by the “Smart Meter State Monitor Process” batch process (D1-SMMTR).

The “Periodic Estimation” System Event algorithm determines if a measuring component is missing final measurements (as described above), and if final measurements are missing, estimates initial measurements and/or creates a To Do Entry. In the base package, this algorithm is the “Create Initial Measurement and To Do Entry” algorithm (D1-CRIMTODO). Algorithm parameters can specify if an initial measurement is to be created, if an To Do Entry is to be created, and the To Do Type and Role (if applicable) for any To Do entries created. The base package includes this algorithm on the Period Estimation system event state of the Interval Channel (D1-IntervalChannel) measuring component business object.

Estimation Calculations

When new estimated initial measurements are created (for either interval or scalar measuring components), the Pre-VEE and Post-VEE values are initially set to zero. When the initial measurement enters the “VEE Complete” state, the VEE rules within the VEE group defined for the “Estimation” VEE role (defined by the “VEE Group For Estimation” field on the measuring component) calculate values and consumption for the initial measurement.

Note that the VEE rules used in this process can also include validation rules that perform validation on the estimated values and consumption. For example, the “estimation” VEE group might contain rules to estimate interval values from a profile, and then perform a sum check to validate the resulting interval measurement against measurements from a consumption reference measuring component. In this example, the sum check validation would be configured to be applied after the interval profile estimation.

Estimated initial measurements are created using “estimation” initial measurement business objects. The base package includes the Estimation IMD (Interval) and Estimation IMD (Scalar) business objects (D1-EstimationIMDInterval or D1-EstimationIMDScalar, respectively).

Final Measurements

When an initial measurement is considered “final,” that is, it has pass all VEE processing and no additional modifications or changes need to be made, it is transformed into a Final Measurement, or simply a Measurement (the terms measurement, final measurement, and final consumption all reference this same “final” measurement data).

When creating final measurements from initial measurement data:

- Final measurements are created using Post VEE quantities
- Each final measurement's condition is copied from the Post VEE condition
- Initial measurements are normalized into final measurements where each final measurement is for a specific date and time.
- Because a single initial measurement may contain many “readings,” a separate final measurement is created for each interval in the initial measurement. For example, if an initial measurement contains 24 hours of 15 minute readings, 96 measurements will be created, each with a specific date and time.

Final measurements are periodically transformed into more concise and accessible usage (also known as bill determinants) for the subscribing systems. In this example, a time-of-use map is applied to the final measurements for an entire month. The usage calculation process is described in more detail in a later chapter.

Derived Values

Final measurements can record up to 10 derived values in addition to the "as measured" value. The derivation formula for each value on a final measurement is held in an algorithm and therefore can derive anything. For example, a set of measurements can adjusted or converted into other units of measure:

Date / Time	As Measured UOM: CCF SQL: n/a	Loss Adjusted UOM: CCF SQL: n/a	Thermal Units UOM: BTU SQL: n/a
01/01/2010 3:00 PM	10	10.1	10.11
01/01/2010 4:00 PM	15	15.15	15.165
01/01/2010 5:00 PM	10	10.1	10.11

Derived values are not reliant on consumption values, but can also come from factors, historical data, or another source. For example, measurements might be compared to “normal” usage for the usage period:

Date / Time	As Measured UOM: CCF SQL: n/a	Loss Adjusted UOM: CCF SQL: n/a	Thermal Units UOM: BTU SQL: n/a	Normal CCF UOM: CCF SQL: Normal	Percent (%) of Normal UOM: CCF SQL: n/a
01/01/2010 3:00 PM	10	10.1	10.11	10	100%
01/01/2010 4:00 PM	15	15.15	15.165	10	150%
01/01/2010 5:00 PM	10	10.1	10.11	10	100%

Updating Final Measurements

Final measurements can be updated if necessary. If final measurements are discovered to incorrect (for whatever reason), **a new initial measurement is created to correct them**. This new initial measurement contains the corrected consumption, and after the initial measurement has completed VEE processing, the existing final measurements are updated with the newly calculated consumption.

Note: The primary key on the table used to store final measurements (the Measurement table) is a combination of the measuring component ID and the date/time of the measurement. This means that it is impossible for more than one final measurement to exist for a measuring component for a given date/time.

The reason a new initial measurement must be created and completed to add or update measurements because there no user interface that allows users to directly edit final measurements.

Daylight Saving Time Support

This section describes how the Oracle Utilities meter data framework and its related products support Daylight Saving Time (DST) for measurement data, including:

- **Types of Devices**
- **Date/Time Storage and Display**
- **Oracle Utilities Application Framework**
- **Typical Daylight Saving Time Scenarios**

Types of Devices

In Oracle Utilities meter data framework initial measurement data processing, the application understands a device that is either:

- a. Aware of the fact that Local time in the device's time zone has been shifted from "Standard", or
- b. Unaware of any such shifting.

Devices in the "unaware" category ("b") will always send Oracle Utilities meter data framework initial measurement data with measurements in Standard time. Devices in the "aware" category ("a") will always send the application initial measurement data in Local time.

Whether a device falls into category "a" (Aware) or "b" (Unaware) is configured via the **Incoming Data Shift** flag on the device type (which can be overridden on the device). The values of the flag are:

- "Always in Local Time" (used with "aware" devices, or category "a")
- "Always in Standard Time" (used with "unaware" devices, or category "b")

This flag is used by pre-processing algorithms (Perform Date/Time Adjustments and Undercount/Overcount Check) in the IMD Seeder business object to convert any date/times on the initial measurement into standard time. Note that this conversion is only done if the device falls into category "a."

Date/Time Storage and Display

Within the database, measurements are stored with two (2) date/times: Standard and Local. The meter data framework uses the date/time in Standard as part of the prime key of the measurement table. The presence of the Local date/time field facilitates querying measurement data using local time.

When displaying dates and times for initial measurement data:

- Display of the data on the Oracle Utilities Meter Data Management **360 View** is in Local time.
- The **IMD Lens** zone (in the Oracle Utilities Meter Data Management version of the Initial Measurement portal) also displays data in Local time.
- The **Raw Data, Pre-VEE and Post-VEE XML Data** zone on the Initial Measurement portal does not shift the data into Local time, so if that the pre-processing algorithm has shifted the data into standard time, the date/times displayed will be in Standard time. Please note that the only two date/times visible in that zone will typically be the Start date/time and End date/time of the initial measurement; the meter data framework strips off the date/times from the individual intervals of the initial measurement at pre-processing time.
- The **Measurement** zone shows both the local and standard date/times as-is.

Oracle Utilities Application Framework

When during initial measurement data processing, it is determined that time shifting is required, the meter data management looks at the time zone metadata in the application. Oracle Utilities Application Framework utilizes the configuration of an Olson DB time zone code on the time zone metadata. This Olson DB contains the shift date/times for every time zone across the globe.

In North America for example, the available Olson DB time zone codes are much more specific than "Eastern/Central/Mountain/Pacific", and include details for areas places such as Arizona and Indiana where there may or may not be shifting for daylight saving time.

Oracle Utilities Application Framework provides business services that wrap the application services that perform time shifting. These services use the time zone metadata to retrieve shift date/times using the Olson DB.

Typical Daylight Saving Time Scenarios

The following table illustrates typical daylight saving time scenarios.

Time Springs Forward		Other Days		Time Falls Back	
DST Shifted Meter in Local Time	Shift & Store time as standard in IMD	DST Shifted Meter in Local Time	Shift & Store time as standard in IMD	DST Shifted Meter in Local Time	Shift & Store time as standard in IMD
3/14/2011	3/14/2011	7/18/2011	7/18/2011	11/7/2011	11/7/2011
1:00	1:00	1:00	0:00	1:00	0:00
3:00	2:00	2:00	1:00	2:00	1:00
4:00	3:00	3:00	2:00	2:00	2:00
5:00	4:00	4:00	3:00	3:00	3:00
6:00	5:00	5:00	4:00	4:00	4:00
7:00	6:00	6:00	5:00	5:00	5:00
8:00	7:00	7:00	6:00	6:00	6:00
9:00	8:00	8:00	7:00	7:00	7:00
10:00	9:00	9:00	8:00	8:00	8:00
11:00	10:00	10:00	9:00	9:00	9:00
12:00	11:00	11:00	10:00	10:00	10:00

Time Springs Forward		Other Days		Time Falls Back	
DST Shifted Meter in Local Time	Shift & Store time as standard in IMD	DST Shifted Meter in Local Time	Shift & Store time as standard in IMD	DST Shifted Meter in Local Time	Shift & Store time as standard in IMD
13:00	12:00	12:00	11:00	11:00	11:00
14:00	13:00	13:00	12:00	12:00	12:00
15:00	14:00	14:00	13:00	13:00	13:00
16:00	15:00	15:00	14:00	14:00	14:00
17:00	16:00	16:00	15:00	15:00	15:00
18:00	17:00	17:00	16:00	16:00	16:00
19:00	18:00	18:00	17:00	17:00	17:00
20:00	19:00	19:00	18:00	18:00	18:00
21:00	20:00	20:00	19:00	19:00	19:00
22:00	21:00	21:00	20:00	20:00	20:00
23:00	22:00	22:00	21:00	21:00	21:00
0:00	23:00	23:00	22:00	22:00	22:00
		0:00	23:00	23:00	23:00
				0:00	0:00
23 hours	23 hours	24 hours	24 hours	25 hours	25 hours

Bold-faced entries indicate times that are impacted by daylight saving time conversion.

Initial Measurement Data In Detail

This section provides details concerning the initial measurement data objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom initial measurement data objects you create as part of your implementation. This section includes:

- A description of the D1-IMD maintenance object
- Lists of the base package initial measurement data business objects, including “lite” business objects
- Details concerning initial measurement data specific configuration options
- A sample initial measurement data business object (D1-InitialLoadIMDInterval)

Maintenance Object - D1-IMD

Initial measurement business objects use the D1-IMD maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-IMD
Description	Initial Measurement
Service Name	D1-INITMSRMTDATA (Initial Measurement Maintenance)
Tables	<ul style="list-style-type: none"> • D1_INT_MSRMT_DATA (Initial Measurement) - Primary • D1_INT_MSRMT_DATA_CHAR (Initial Measurement Characteristics) - Child • D1_INT_MSRMT_DATA_LOG (Initial Measurement Log) - Child • D1_INT_MSRMT_DATA_LOG_PARM (Initial Measurement Log Parameters) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Base Package Business Objects

The base package includes the following initial measurement data business objects:

Business Object Name	Description
D1-EstimationIMDInterval	Estimation IMD (Interval) Instances of this business object represent individual estimated interval initial measurements in the system.
D1-EstimationIMDScalar	Estimation IMD (Scalar) Instances of this business object represent individual estimated scalar initial measurements in the system.

Business Object Name	Description
D1-InitialLoadIMDInterval	Initial Load IMD (Interval) Instances of this business object represent individual interval initial measurements in the system.
D1-InitialLoadIMDScalar	Initial Load IMD (Scalar) Instances of this business object represent individual scalar initial measurements in the system.
D1-ManualIMDInterval	Manual IMD (Interval) Instances of this business object represent individual manually-created interval initial measurements in the system.
D1-ManualIMDScalar	Manual IMD (Scalar) Instances of this business object represent individual manually-created scalar initial measurements in the system.

The base package includes the following “lite” initial measurement data business objects:

Business Object Name	Description
D1-AuditList - IMD	Audit List Section Only (Lite)
D1-GenericIMDMain - IMD	IMD - Main Section Only LITE
D1-IMDLite	IMD LITE
D1-IMDParentLITE	IMD Parent LITE
D1-IMDPeriod	IMD Lite for High Quality Check
D1-IMDPostVEE	IMD Main and Post VEE LITE
D1-IMDPostVEERaw	IMD Main and Post VEE Raw LITE
D1-IMDPreAndPostVEE	IMD Pre and Post VEE - Lite
D1-IMDPreVEE	IMD Main and Pre VEE LITE
D1-IMDRawData	Initial Measurement Data Raw Lite
D1-IMDSeederLite	IMD Seeder Lite
D1-IMDTraceAndPostVEE	IMD Trace and Post VEE - Lite
D1-IMDTraceZone	IMD - Trace List Section Only LITE
D1-InitialLoadIMDIntervalRaw	Initial Load IMD (Interval) - Raw LITE
D1-InitialLoadIMDMain	Initial Load IMD - Main Section Only LITE
D1-PostVEE	Post VEE Only LITE
D1-PreVEE	Pre VEE Only LITE

The base package includes the following additional initial measurement data business objects:

Business Object Name	Description
D1-IMDRetry	IMD Retry BO
D1-IMDSeeder	IMD Seeder (used to instantiate initial measurement business objects, based on the head-end system sending the measurement)

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by initial measurement data business objects.

Business Object Options

Initial measurement business objects can make use of the following business object options:

- **IMD Status Extendable Lookup:** This option defines an extendable lookup that can be used to define different statuses which can apply to initial measurement data created from this business object.
- **Initial Measurement Data Type:** This option defines the data type for initial measurement data created from this business object. Valid values are based on the D1_IMD_TYPE_FLG lookup, and include the following:

Field Value	Description
D1GA	IMD Seeder
D1IL	Initial Load
D1MO	Manual
D1ES	Estimation

- **Interval Status Mapping to Condition with Priority:** This option defines the Interval Status Code extendable lookup that can be used to map status codes for initial measurement data created from this business object.

Example Initial Measurement - D1-InitialLoadIMDInterval

The table below lists the details of the D1-InitialLoadIMDInterval initial measurement data business object.

Option/Field	Description
Business Object	D1-InitialLoadIMDInterval
Description	Initial Load IMD (Interval)
Maintenance Object	D1-IMD (Initial Measurement Data)
Application Service	D1-INITLOADIMDBOAS (Initial Load IMD Interval BO)
Instance Control	Allow New Instances

Option/Field	Description
Options	<ul style="list-style-type: none"> Initial Measurement Data Type: D1IL (Initial Load) Display UI Map: D1-IMDDisplay (Initial Measurement Data - Display) Portal Navigation Option: d1imdsaTabMenu (Initial Measurement) Display Map Service Script: D1-IMDDisp (Initial Measurement Data Display) Maintenance UI Map: D1-IMDMaint (Initial Load IMD (Interval) - Maintenance)
Algorithms	<ul style="list-style-type: none"> Post-Processing: D1-AUD-QTYUE (Audit Changes made to Quantity and set User-Edited Flag) Validation: D1-IMD-VAL (Validates Status Condition for Delete) Validation: D1-IMD-COMM (Validate Initial Measurement Data Common Input) Validation: D1-INT-SPEC (Validate Interval Initial Measurement Data Input)
Lifecycle	<ul style="list-style-type: none"> Pending (Initial) Additional Mapping (Interim) Mapping Error (Interim) VEE Ready (Interim) Error (Interim) Removed from Processing (Final) VEE Complete (Interim) Exception (Interim) Discarded (Final) Force Complete (Interim) Finalized (Final)

Use the Business Object portal to view additional details concerning this business object.

Measurements In Detail

This section provides details concerning the measurement objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom measurement objects you create as part of your implementation. This section includes:

- A description of the D1-MSRMT maintenance object
- Lists of the base package measurement business objects, including “lite” business objects
- Details concerning measurement-specific configuration options
- A sample measurement business object (D1-Measurement)

Maintenance Object - D1-MSRMT

Measurement business objects use the D1-MSRMT maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-MSRMT
Description	Measurement
Service Name	D1-MEASUREMENT (Measurement Maintenance)
Tables	<ul style="list-style-type: none"> • D1-MSRMT (Measurement) - Primary • D1-MSRMT_CHAR (Measurement Characteristics - Child)

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Base Package Device Business Objects

The base package includes the following measurement business objects:

Business Object Name	Description
D1-Measurement	Measurement Instances of this business object represent individual final measurements stored in the system.

The base package includes the following “lite” device business objects:

Business Object Name	Description
D1-MeasurementParentLITE	Measurement Parent LITE
D1-MSRMTLite	Measurement LITE

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by measurement business objects.

Business Object Options

Measurement business objects can make use of the following business object options:

- **Measurement Log Business Object:** This option defines the business object that will be used to log changes that have occurred to the measurement.

Example Device - D1-Measurement

The table below lists the details of the D1-Measurement device business object.

Option/Field	Description
Business Object	D1-Measurement
Description	Measurement
Maintenance Object	D1-MSRMT (Measurement)
Application Service	D1-MEASUREMENT (Measurement MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none">• Measurement Log Business Object: D1-MeasurementLog (Measurement Log)• Display UI Map: D1-MeasurementDisplay (Measurement - Display)• Portal Navigation Option: d1meassaTabMenu (Measurement)• Display Map Service Script: D2-MsrmtDisp (Measurement Data Display)
Algorithms	<ul style="list-style-type: none">• Audit: D1-AMSRMTLOG (Formats the standard description of a Measurement)• Information: D1-MSRMTINFO (Measurement Information)

Use the Business Object portal to view additional details concerning this business object.

Configuring Initial Measurements and Measurements

This section provides high-level overviews of the steps involved in configuring custom objects to define initial measurements and (final) measurements. See **Configuration Process Overview** in **Chapter One** for a high-level overview of the overall configuration process.

Note: The procedures below focus on specific configuration tasks and options related to each of the objects described in this chapter, and do not address all the steps involved in creating business objects, UI maps, algorithms, etc. For more information about these subjects, refer to the Oracle Utilities Application Framework documentation.

Configuring Custom Initial Measurements

Configuring custom initial measurement objects involves the following steps:

1. Design the initial measurement business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom initial measurement-related configuration objects required for your business objects, including:

Business Object Options: Create business objects for the following business object options:

- IMD Status Extendable Lookup
3. Create your initial measurement business objects, referencing the configuration objects created above as appropriate.

Configuring Custom Measurements

Configuring custom measurement objects involves the following steps:

1. Design the measurement business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom measurement-related configuration objects required for your business objects, including:

Business Object Options: Create business objects for the following business object options:

- Measurement Log Business Object
3. Create your initial measurement business objects, referencing the configuration objects created above as appropriate.

Chapter 7

Validation, Editing, and Estimation

This chapter provides descriptions of validation, editing, and estimation groups and rules, including:

- **Understanding Validation, Editing, and Estimation**
- **VEE Groups In Detail**
- **VEE Rules In Detail**
- **VEE Eligibility Criteria In Detail**
- **VEE Exceptions In Detail**
- **Configuring VEE Groups, Rules, Eligibility Criteria, and Exceptions**

Note: References to objects and options pre-fixed with “D2” are available only with Oracle Utilities Meter Data Management, and included for illustrative purposes only.

Understanding Validation, Editing, and Estimation

This section describes the validation, editing, and estimation (or VEE) process used by the meter data framework and related products, including Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway. This section includes:

- **Overview of the Validation, Editing, and Estimation Process**
- **VEE Rules and VEE Groups**
- **VEE Exceptions**

Overview of the Validation, Editing, and Estimation Process

As noted in **Chapter 6: Measurement Data**, once received into the system, initial measurements are subject to validation, editing, and estimation. This process involves the following:

- **Validation:** Validates that the initial measurement data is within expected tolerances, and is correct
- **Editing:** If the initial measurement data is wrong in some way, the data can be automatically changed.
- **Estimation:** If initial measurement data is incomplete (for example, if one or more interval values within an interval measurement are missing), missing values can be automatically estimated.

The values recorded in an initial measurement can change during this process, and exceptions can be raised if initial measurements are wrong in some way (such as out-of-tolerance quantities, incorrect values, etc.).

Beyond raising an exception, the VEE process can also transition an initial measurement to an Exception state if it detects a problem that it is not able or allowed to correct.

VEE Rules and VEE Groups

The specific validation, editing, and estimation processing performed on initial measurement data is defined in individual VEE rules, each performing a specific set of validation logic. Examples of VEE rules include:

- **Interval Size Validation:** Checks to ensure that the interval size of the incoming initial measurement data matches the value defined in the measuring component type
- **Multiplier Check:** Checks to ensure that the device multiplier value of the incoming initial measurement data matches the multiplier value stored on the measuring component
- **Unit of Measure Check:** Checks to ensure that the Unit of Measure (UOM) of the incoming initial measurement data matches the UOM specified on the measuring component's type

The base package contains many VEE rules you can use in your implementation, and you can also create your own custom VEE rules.

Some VEE rules create exceptions if the initial measurement data doesn't fall within parameters specified by the rule. Other rules override measurements, changing measurement values as dictated by the rule's parameters. Some rules can both create exceptions and override the measurement as part of a single process. By convention, VEE rules change the Post VEE quantities of initial measurement data, but VEE rules can change ANYTHING on an initial measurement.

VEE Groups

VEE groups are collections of VEE rules that are applied to initial measurement data. During the VEE process, the system executes the VEE rules defined in each VEE group. The rules within a

VEE groups are defined in a specific sequence, allowing control over the order in which the rules are executed.

VEE groups can be associated to a specific measuring component, or to a measuring component type (or both). VEE groups associated with a measuring component type are applied to all measuring components of that type, while those associated to a specific measuring component are applied only to that measuring component. VEE groups associated to a measuring component override those assigned to a measuring component type.

Effective Dates

Every VEE rule has an effective period. Rules will only be applied if the initial measurement's start date is within the rule's effective period. For example, an Interval Spike Check rule with a Start Date of 11/15/2010 will only be applied if the start date of the initial measurement is on or after 11/15/2010.

This allows you to update the specifics of a rule without removing the previous version of the rule. For example, you might change the tolerance of an Interval Spike Check rule from 1.2 to 1.5 as of a certain date. However, for initial measurement data for the period prior to the change, you would want to use the tolerance for the original version of the rule (1.2) instead of the new tolerance (1.5).

Eligibility Criteria

Each VEE rule may optionally have eligibility criteria that controls if the rule is applied. This feature can greatly reduce the number of VEE groups you need to create, because it allows a single VEE group to have conditional VEE rules based on eligibility criteria (rather than requiring a distinct VEE group for every combination of VEE rules).

For example, you might create a rule that compares interval consumption against related scalar consumption (such as might be the case with a device with both interval and scalar measuring components). This rule might use eligibility criteria that specifies that the rule is only applied if the initial measurement's measuring component has a corresponding scalar register measuring component.

Another example might be a rule that compares the current consumption against standard consumption for measuring components of a certain type for the first six months after installation. You might create eligibility criteria that specifies that the rule is only applied if the measuring component's device configuration has been installed at a service point for less than six months.

Generic Utility Rules - Tools For Creating VEE Groups

While most VEE rules are used to validate the usage (or some other attribute) in an initial measurement, Oracle Utilities meter data framework also provides a number of "generic utility" VEE rules that can be used when configuring VEE groups. These include:

- Referred VEE Group Rules
- VEE Group Matrix (Factor) Rules
- Exception Handler Rules
- Successful Termination Rules

Execute VEE Group Rules - Reusing Groups Of Rules

A common situation in many implementations is in which several rules are to be applied to multiple different types of measuring components. For example, you might want to perform device identifier validations, multiplier checks, and UOM checks on all measuring components.

One way to meet this requirement would be to repeat these three rules in multiple VEE groups. However, this solution becomes hard to maintain if changes to the rules are required (or if new "global rules" are introduced) as each group would have to be updated.

Instead of this, you can create a VEE rule that executes the rules in a referenced VEE group. Rules of this type are called Execute VEE Group rules. Rules of this type can have effective dates and eligibility criteria, just like all VEE rules.

Using the example above, you could create a group called “Rules for All MCs” that contains a device identifier validations rule, a multiplier check rule, and a UOM check rule, and then reference the “Rules for All MCs” group in a Execute VEE Group rule.

Execute VEE Group rules can be “nested.” That is, a group executed by a Execute VEE Group rule can, in turn, execute the rules in another group, and so on.

VEE Group Matrix (Factor) Rules - Using Factors To Implement Dynamic VEE Groups

Another situation likely to occur in many implementations is where specific rules may need to be applied to measurement data based on specific criteria, such as geography. For example, some geographic territories may have unique VEE rules in addition to rules that are applied to all geographic territories.

This requirement could be implemented using eligibility criteria, such as only applying a rule (or a group of rules) if the service point for an initial measurement is located in a hot summer area. If there are limited number of these unique rules, this solution is suitable, but if there are many territories and each territory has several unique rules, an implementation of this sort would become hard to maintain (and slow to execute) as the VEE group would have many rules with varying eligibility criteria.

Instead of this, you can create a VEE rule that dynamically executes another group's rules based on specific conditions. For example, the VEE process can be configured to execute different VEE rules based on where the service point is located, or the number of tamper events in the last 6 months, or the type of customer, or the meter's head-end system, etc. Rules of this type are called VEE Group Matrix (Factor) rules. These rule and can have effective dates and eligibility criteria, just like all VEE rules.

Factors are used to implement the dynamic selection of VEE group (note the term factor is intentionally generic as factors can be used for other purposes). Factors used for these rules have a Factor Class of “VEE group,” and use some unique rules:

- VEE group factors reference a characteristic type (with pre-defined values).
- VEE group factors reference an algorithm that retrieves or derives the value of the characteristic type at runtime.
- Factor values for a VEE group factor are effective-dated pairings of a characteristic value and a corresponding VEE group.

At run time, the rule retrieves / derives the characteristic value for the factor's characteristic type and then finds the VEE group associated with the respective characteristic value.

Factors can be related to any real or dynamic attribute, so rules of this type are very flexible. For example:

- **Real Attribute:** you could create a rule that executes a VEE group based on the head-end system of the device.
- **Dynamic Attribute:** you could create a rule that executes a VEE group based on the number of tamper events linked to the measuring component in the last 180 days, executing one group if there are 6-10 events (a characteristic value of 6-10), and another if there are more than 10 events (a characteristic value of 10+). The number of tamper events is dynamically calculated at execution time and is compared to the characteristic values defined for the factor, and executes the appropriate VEE group. In this example, if the count of tamper events was anything less than six, no VEE group would be executed.

Exception Handler Rules

Exception Handler rules are described in the section below on VEE exceptions.

Successful Termination Rules

Successful Termination rules are described in the section below on VEE exceptions.

VEE Roles

Initial measurement data can come from different sources, such as a head-end system or estimation processes, or it can be manually created by a user (to override or estimate consumption). Measurement data from these different sources might use different VEE rules. For example:

- Initial measurements sent a head-end system might use strict VEE rules
- Initial measurements created by a user (to override or estimate consumption) may use less strict rules
- Initial measurements created by the system to estimate consumption have very few (if any) VEE rules

Applying different numbers and types of VEE rules based on the source of the initial measurement data could be implemented using eligibility criteria (e.g., only apply a rule if the initial measurement data's source is X) or factor-based rules (e.g., the factor's characteristic is the initial measurement data's source), but both of these techniques are potentially difficult to maintain if there are many source-dependent rules.

Instead of that approach, you can define different VEE groups for different source, or roles. The three base package roles are:

- **Estimation:** Used for initial measurement data estimated by the system
- **Initial Load:** Used for initial measurement data received from a head-end system or import process
- **Manual Override:** Used for initial measurement data manually created by a user

A measuring component's Measuring Component Type can define "fallback" VEE groups for each of these roles. In addition, an individual measuring component can specify a VEE group for each role. If the measuring component doesn't have a VEE group specified for a role, the "fallback" VEE group defined for the measuring component type is used.

VEE Exceptions

Each VEE rule defines an exception type and severity that specify how exceptions are tracked by the system. When an initial measurement fails a validation, an exception of the type specified for the failed VEE rule is created. A single initial measurement can have multiple exceptions, one (or more) for each rule the measurement fails. This allows users to see all of the problems detected during the VEE process.

Exception Types

Each exception has an exception type. Exception types allow you to distinguish between different exceptions based on the rule that triggered them. Exception types can be created each VEE rule, at a more general level exception types, such as "Insufficient Data" to be used to signify that a measurement didn't have sufficient data for the VEE rule to execute.

Exception Categories

There are three categories, or severities of exceptions:

- **Info:** Used to highlight something interesting, but not sufficient to cause the initial measurement to be put into the Exception state. Exceptions of this category can be used to report on the frequency of interesting, but not fatal issues.
- **Issue:** Used to report a problem that will prevent the initial measurement from being finalized. Multiple "issue exceptions" can be created during VEE processing. If at least one issue exists after all rules have been applied, the initial measurement is transitioned to the Exception state.

- **Terminate:** Used to report a severe issue that will cause the VEE process to stop and the initial measurement to be transitioned immediately to the Exception state. Only one terminate exception can be issued (as the first one causes VEE to stop on an initial measurement).

Exceptions and To Do Entries

In addition to exceptions, VEE processing can also trigger the creation of To Do Entries related to failed validations.

If Issue or terminate exceptions exist for an initial measurement, a To Do Entry is created when the initial measurement is transitioned to the Exception state. The To Do Type and default To Do Role of this To Do Entry are defined on the Enter system event for the Exception state of the business object used to define the initial measurement.

To Do Entries created in this way can be routed to different roles depending on the exception's message category and number (using the To Do Type's Message Overrides tab).

Exception Handler Rules - Aborting When There Are Too Many Issues

There may be times when an implementation's requirements are to terminate processing for any initial measurement that contains a pre-defined number of exceptions. Exception Handler VEE rules can issue a "terminate exception" if they detect too many exceptions. This is useful when individual exceptions are not sufficient to stop VEE processing.

The criteria used by this rule can simply reference a number of exceptions of a given exception type, or can specify more complex AND/OR criteria that must be satisfied before VEE processing is terminated. For example, processing might terminate when 3 exceptions of one type AND 2 exceptions of another type have been issued, or if 2 exceptions of one type OR 2 exceptions of a different type have been issued.

The terminate exception created by Exception Handler rules can be of a specific exception type. In addition, Exception Handler rules can also create a different type of To Do Type and To Do Role than the default.

Exception Handler rules can be placed at any point throughout a VEE group where each rule can reference different exception types.

Successful Termination Rules

There may be times when an implementation's requirements are to successfully terminate processing for any initial measurement that passes a pre-defined set of validations before accumulating a pre-defined number of exceptions. For example, a set of validation rules can be executed early in the overall sequence of rules that proves that the data is good enough to use, such that no further rules need to be executed. In this case, implementations might want to terminate the VEE process to save on execution time rather than execute further rules that won't ultimately affect the data. This is accomplished through Successful Termination rules.

The criteria used by Successful Termination rules can simply reference a number of exceptions of a given exception type, or can specify more complex AND/OR criteria that must be satisfied before VEE processing is terminated. For example, processing might terminate when less than 3 exceptions of one type AND less than 2 exceptions of another type have been issued, or if less than 2 exceptions of one type OR less than 2 exceptions of a different type have been issued.

Successful termination rules can be placed at any point throughout a VEE group where each rule can reference different exception types.

Available Actions for Initial Measurements with Exceptions

Users have a number of options for dealing with initial measurements with exceptions.

- After correcting the cause of the issues that triggered the exceptions, a user can re-VEE the initial measurement.

- A user can discard the initial measurement.
- A user can edit the Post VEE quantities (if necessary) and manually complete the initial measurement. This will cause final measurements to be created using the contents of the Post VEE quantities.

Note: No VEE processing is performed on manually completed initial measurement data.

Regardless of the action taken by the user, the system will complete any open To Do Entries that created when the initial measurement entered the Exception state.

Exceptions Are Not Deleted

Note that exceptions are not deleted when an initial measurement is adjusted or corrected. After any issues are corrected or the initial measurement is overridden (or manually completed), the exceptions persist (in the Closed state) for reporting purposes.

VEE Groups In Detail

This section provides details concerning the VEE group objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom VEE group objects you create as part of your implementation. This section includes:

- A description of the D1-VEEGROUP maintenance object
- Lists of the base package VEE group business objects, including “lite” business objects
- A sample VEE group business object (D1-VEEGroup)

Maintenance Object - D1-VEEGROUP

Device business objects use the D1-VEEGROUP maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-VEEGROUP
Description	VEE Group
Service Name	D1-VEEGROUP (VEE Group Maintenance)
Tables	<ul style="list-style-type: none"> • D1_VEE_GRP (VEE Group) - Primary • D1_VEE_GRP_CHAR (VEE Group Characteristics - Child • D1_VEE_GRP_L (VEE Group Language) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package VEE Group Business Objects

The meter data framework base package includes the following VEE group business objects:

Business Object Name	Description
D1-VEEGroup	VEE Group

The meter data framework base package includes the following additional VEE group business objects:

Business Object Name	Description
D1-VEEGroupBundlingAddBO	Bundling Add BO for VEE Group
D1-VEEGroupPhysicalBO	Physical BO for VEE Group

Example VEE Group - D1-VEEGroup

The table below lists the details of the D1-VEEGroup device business object.

Option/Field	Description
Business Object	D1-VEEGroup
Description	VEE Group
Maintenance Object	D1-VEEGROUP (VEE Group)
Application Service	D1-VEEGROUP (VEE Group MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none">• Display UI Map: D1-VEEGroupDisplay (VEE Group - Display)• Portal Navigation Option: d1veegrpTabMenu (VEE Group)• Maintenance UI Map: D1-VEEGroupMaint (VEE Group - Maintenance)

Use the Business Object portal to view additional details concerning this business object.

VEE Rules In Detail

This section provides details concerning the VEE rule objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom VEE rule objects you create as part of your implementation. This section includes:

- A description of the D1-VEERULE maintenance object
- Lists of the base package VEE rule business objects, including “lite” business objects
- Details concerning VEE rule-specific configuration options
- A sample VEE rule business object (D2-IntervalSpikeCheck)
- A list of base package VEE rules, including the algorithm / algorithm type and a brief description of each

Maintenance Object - D1-VEERULE

Device business objects use the D1-VEERULE maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-VEERULE
Description	VEE Rule
Service Name	D1-VEERULE (VEE Rule Maintenance)
Tables	<ul style="list-style-type: none"> • D1_VEE_RULE (VEE Rule) - Primary • D1_VEE_RULE_CHAR (VEE Rule Characteristics) - Child • D1_VEE_RULE_L (VEE Rule Language) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package VEE Rule Business Objects

The meter data framework base package includes the following VEE rule business objects:

Business Object Name	Description
D1-GenericVEERule	Generic VEE Rule (generic VEE rule business object used as a parent business object for all other VEE rule business objects)
D1-VEERuleExceptionHandler	VEE Rule Exception Handler Instances of this business object represent individual exception handler VEE rules defined in the system.
D1-VEERuleGroupFactor	VEE Group Matrix (Factor) Instances of this business object represent individual VEE Group Matrix (Factor) VEE rules defined in the system.

Business Object Name	Description
D1-VEERuleReferredVEEGroup	Execute VEE Group Instances of this business object represent individual Execute VEE Group VEE rules defined in the system.
D1-VEERuleSuccessTermination	Successful Termination Instances of this business object represent individual Successful Termination VEE rules defined in the system.

The meter data framework base package includes the following “lite” VEE rule business objects:

Business Object Name	Description
D1-VEERuleGroupFactorLite	VEE Rule Group Factor LITE
D1-VEERuleParentLITE	VEE Rule Parent LITE

The meter data framework base package includes the following additional VEE rule business objects:

Business Object Name	Description
D1-VEERuleBundlingAddBO	Bundling Add BO for VEE Rule
D1-VEERuleExecReSequence	VEE Rule Execution Resequencing (used when resequencing VEE rules in a VEE group)
D1-VEERulePhysicalBO	Physical BO for VEE Rule

The Oracle Utilities Meter Data Management base package includes the following VEE rule business objects:

Business Object Name	Description
D2-EnsureIMDExistsforSibling	Ensure IMD Exists for Sibling MCs
D2-IntervalAdjustmentFrmScalar	Interval Adjustment from Scalar
D2-IntervalAveragingEstimation	Interval Averaging Estimation
D2-IntervalInterpolationEst	Interval Interpolation Est
D2-IntervalProfileEstimation	Interval Profile Estimation
D2-IntervalReplacementRule	Interval Replacement Rule
D2-IntervalSizeValidation	Interval Size Validation
D2-IntervalSpikeCheck	Interval Spike Check
D2-NegativeConsumptionCheck	Negative Consumption Check
D2-RaiseMissingQuantityExcp	Raise Missing Quantity Exception
D2-RegisterMultiplierCheck	Multiplier Check
D2-ScalarCalcFromInterval	Scalar Calculation From Interval
D2-ScalarEstimation	Scalar Estimation

Business Object Name	Description
D2-ScalarProfileEstimation	Scalar Profile Estimation
D2-ScalarReplacementRule	Scalar Replacement Rule
D2-SumCheck	Sum Check
D2-UOMCheck	Unit of Measure Check
D2-VEERuleHighLowCheck	High/Low Check
D2-ZeroConsumptionCheck	Zero Consumption Check

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by VEE rule business objects.

System Events

VEE rule business objects can make use of the following system events:

- **Apply VEE Rule:** This system event defines the algorithm to use when executing the VEE rule.

Other Options

VEE rules use various parameters and properties. These options are specified when creating VEE rules based on a VEE rule business object, and include the following:

Exception Types

Exception types define the properties common to exceptions. When creating VEE rules, you might create an exception type for each rule. You might also create more general exception types, such as "Insufficient Data" to be used to signify that a measurement didn't have sufficient data for the VEE rule to execute.

Generic Utility VEE Rules

Oracle Utilities meter data framework includes three "generic utility" base package VEE rule types that can be used when configuring VEE groups and rules. This section outlines the configuration options you need to configure before you can create rules of these types.

Execute VEE Group: Referred VEE Group rules reference a VEE group. You must create the VEE group to reference before can create rules of this type.

Exception Handler: Exception Handler rules are used to define options and logic to terminate the VEE process when a set of user configured criteria are met. VEE rules of this type can be included in a group to specify how exceptions are handled for that group, and allow for creation of a single "parent" exception for the group. Exception Handler rules use the following options:

- **To Do Type:** An override To Do Type for To Do Entries created as a result of the rule. This To Do Type is used instead of the default (specified in the Enter algorithm on the Exception state of the initial measurement data business object's lifecycle).
- **To Do Role:** An override To Do Role for To Do Entries created as a result of the rule. This To Do Role is used instead of the default (specified in the Enter algorithm on the Exception state of the initial measurement data business object's lifecycle).
- **Exception Type:** The Exception Type for exceptions created by this rule.

VEE Group Matrix (Factor): VEE Group Factor rules are used to define business logic to allow reference to a factor (of type VEE group) where the values of the factor are a list of VEE groups.

This allows creating a VEE rule that can select from a list of VEE groups (referred to as a matrix) whose rules to execute next. VEE Group Matrix (Factor) rules use the following options:

- **Factor:** The factor referenced by the rule. The factor must have a Factor Class of VEE Group (i.e. it must be based on the VEE group factor business object).
- **Characteristic Type:** The characteristic type referenced by the factor. This characteristic type must be one with pre-defined values.
- **Characteristic Type Values:** Specific values for the characteristic type. These are the values retrieved and evaluated to determine the VEE group whose rules should be executed. These must be values that can be retrieved from some object (device, service point, etc.) related to the measuring component whose initial measurement data is being validated by this rule.
- **Characteristic Source Algorithm:** The algorithm used to retrieve the characteristic value (which in turn determines the VEE group whose rules should be executed). The base package includes the following algorithm types that can be used when creating this algorithm:
 - Factor Characteristic Source - Device (D1-FCSDEVICE)
 - Factor Characteristic Source Measuring Component (D1-FCSMC)
 - Factor Characteristic Source - Service Point (D1-FCSSP)
 - Factor Characteristic Source - Usage Subscription (D1-FCSUS)
- **VEE Groups:** The VEE groups associated with each characteristic value.

Example VEE Rule - D2-IntervalSpikeCheck

The table below lists the details of the D2-IntervalSpikeCheck VEE rule business object.

Option/Field	Description
Business Object	D2-IntervalSpikeCheck
Description	Interval Spike Check
Maintenance Object	D1-VEERULE (VEE Rule)
Parent Business Object	D1-GenericVEERule (Generic VEE Rule)
Lifecycle Business Object	Generic VEE Rule
Application Service	D1-VEERULE (VEE Rule MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Display UI Map: D2-IntervalSpikeCheckDisplay (Interval Spike Check - Display) • Portal Navigation Option: d1veerleTabMenu (VEE Rule) • Maintenance UI Map: D2-IntervalSpikeCheckMaint (Interval Spike Check - Maintenance)
Algorithms	<ul style="list-style-type: none"> • Apply VEE Rule: D2-INTSPKCHK (Interval Spike Check) • Validation: D2-INTSPKVLD (Interval Spike Check Minimum Number of Intervals Validation)

Use the Business Object portal to view additional details concerning this business object.

VEE Eligibility Criteria In Detail

This section provides details concerning the VEE eligibility criteria objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom VEE eligibility criteria objects you create as part of your implementation. This section includes:

- A description of the D1-VEEELIGCR maintenance object
- Lists of the base package VEE eligibility criteria business objects, including “lite” business objects
- Details concerning VEE eligibility criteria-specific configuration options
- A sample VEE eligibility criteria business object (D1-VEEEligibilityCriteria)

Maintenance Object - D1-VEEELIGCR

VEE eligibility criteria business objects use the D1-VEEELIGCR maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-VEEELIGCR
Description	VEE Eligibility Criteria
Service Name	D1-VEEELIGCR (VEE Eligibility Criteria)
Tables	<ul style="list-style-type: none"> • D1_VEE_ELIG_CRIT (VEE Eligibility Criteria) - Primary • D1_VEE_ELIG_CRIT_CHAR (VEE Eligibility Criteria Characteristics) - Child • D1_VEE_ELIG_CRIT_L (VEE Eligibility Criteria Language) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package VEE Eligibility Criteria Business Objects

The meter data framework base package includes the following VEE eligibility criteria business objects:

Business Object Name	Description
D1-VEEEligibilityCriteria	VEE Eligibility Criteria Instances of this business object represent individual eligibility criteria defined in the system.

The meter data framework base package includes the following additional VEE eligibility criteria business objects:

Business Object Name	Description
D1-VEEEligCritBundlingAddBO	Bundling Add BO for VEE Eligibility Criteria
D1-VEEEligCritPhysicalBO	Physical BO for VEE Eligibility Criteria

Configuration Options

This section outlines specific configuration options, such as business object options, system events, and other options used by VEE eligibility criteria business objects.

System Events

VEE eligibility criteria business objects can make use of the following system events:

- **Apply VEE Rule Eligibility Criteria:** This system event defines the algorithm to use to apply eligibility criteria to a VEE rule.

Example VEE Eligibility Criteria - D1-VEEEligibilityCriteria

The table below lists the details of the D1-SmartMeter device business object.

Option/Field	Description
Business Object	D1-VEEEligibilityCriteria
Description	VEE Eligibility Criteria
Maintenance Object	D1-VEEELIGCR (VEE Eligibility Criteria)
Application Service	D1-VEEELIGCR(VEE Eligibility Criteria MO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none">• Portal Navigation Option: d1veerleTabMenu (VEE Rule)• Maintenance UI Map: D1-VEEEligibilityCritMaint (VEE Eligibility Criteria - Maintenance)
Algorithms	<ul style="list-style-type: none">• Apply VEE Rule Eligibility Criteria: D1-ECF-APECT (Evaluate Criteria Field - Apply Eligibility Criteria)

Use the Business Object portal to view additional details concerning this business object.

VEE Exceptions In Detail

This section provides details concerning the VEE exception objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom VEE exception objects you create as part of your implementation. This section includes:

- A description of the D1-VEEEXCP maintenance object
- Lists of the base package VEE exception business objects, including “lite” business objects
- A sample VEE exception business object (D1-VEEException)

Maintenance Object - D1-VEEEXCP

VEE exceptions business objects use the D1-VEEEXCP maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-VEEEXCP
Description	VEE Exception
Service Name	D1-VEEEXCP (VEE Exception Maintenance)
Tables	<ul style="list-style-type: none"> • D1_VEE_EXCP (VEE Exception) - Primary • D1_VEE_EXCP_CHAR (VEE Exception Characteristics) - Child • D1_VEE_EXCP_PARM (VEE Exception Parameter) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package VEE Exception Business Objects

The meter data framework base package includes the following VEE exception business objects:

Business Object Name	Description
D1-VEEException	VEE Exception

Example VEE Exception - D2-VEEException

The table below lists the details of the D1-SmartMeter device business object.

Option/Field	Description
Business Object	D1-VEEException
Description	VEE Exception
Maintenance Object	D1-D1-VEEEXCP (VEE Exception)
Application Service	D1-D1-VEEEXCP (VEE Exception MO)
Instance Control	Allow New Instances

Option/Field	Description
Options	<ul style="list-style-type: none">• Display UI Map: D1-VEEExceptionDisplay (VEE Exception - Display)• Display Map Service Script: D1-VEEExcptn (Display VEE Exception Message Details)
Algorithms	<ul style="list-style-type: none">• Information: D1-VEXCPINFO (VEE Exception Information)

Use the Business Object portal to view additional details concerning this business object.

Configuring VEE Groups, Rules, Eligibility Criteria, and Exceptions

This section provides high-level overviews of the steps involved in configuring custom VEE groups and rules. See **Configuration Process Overview** in **Chapter One** for a high-level overview of the overall configuration process.

This section also provides information related to creating instances of the “generic utility” VEE rules described earlier in this chapter.

Note: The procedures below focus on specific configuration tasks and options related to each of the objects described in this chapter, and do not address all the steps involved in creating business objects, UI maps, algorithms, etc. For more information about these subjects, refer to the Oracle Utilities Application Framework documentation.

Configuring Custom VEE Groups

Configuring custom VEE groups involves the following steps:

1. Design the VEE group business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom VEE group-related configuration objects required for your business objects.
3. Create your VEE group business objects, referencing the configuration objects created above as appropriate.

Configuring Custom VEE Rules

Configuring custom VEE rules involves the following steps:

1. Design the VEE rule business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom VEE rule-related configuration objects required for your business objects, including:

System Events: Create algorithms for the following system events:

- Apply VEE Rule

Options: Create data as appropriate for the following options used when creating VEE rules:

- Exception Types
- Generic Utility VEE Rules: Define the following options used with “generic utility” VEE rules
 - Referred VEE Group: VEE groups to be referenced by these rules.
 - Exception Handler: To Do Type, To Do Role, and Exception Type used by these rules.
 - VEE Group Matrix (Factor): Factor, Characteristic Type and Values, Characteristic Source Algorithm, VEE groups

3. Create your VEE rule business objects, referencing the configuration objects created above as appropriate.

Note: VEE rule business objects should reference D1-GenericVEERule as their Parent Business Object.

Configuring Custom VEE Eligibility Criteria

Configuring custom VEE eligibility criteria involves the following steps:

1. Design the VEE eligibility criteria business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom VEE eligibility criteria-related configuration objects required for your business objects, including:

System Events: Create algorithms for the following system events:

- Apply VEE Rule Eligibility Criteria

3. Create your VEE eligibility criteria business objects, referencing the configuration objects created above as appropriate.

Configuring Custom VEE Exceptions

Configuring custom VEE exceptions involves the following steps:

1. Design the VEE exception business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom VEE exception-related configuration objects required for your business objects.
3. Create your VEE exception business objects, referencing the configuration objects created above as appropriate.

Creating Generic Utility VEE Rules

This section outlines the steps involved in creating instances of the “generic utility” VEE rules. Refer to the Oracle Utilities Meter Data Framework online help and user’s guide for general procedures used in creating VEE rules.

Creating Execute VEE Group VEE Rules

Use the following procedure to create Execute VEE Group VEE rules:

1. Create the VEE group to which the rule will belong.
2. Create the VEE group that the rule will reference
3. Create the VEE rule referencing the group created in the previous step

Creating VEE Group Matrix (Factor) VEE Rules

Use the following procedure to create VEE Group Matrix (Factor) VEE rules:

1. Create the Characteristic Type and Values to be used by the factor that will be referenced by the rule.
2. Create the Characteristic Source Algorithm to be used by the factor that will be referenced by the rule.
3. Create the VEE Groups to be associated to the characteristic values.
4. Create the Factor that will be referenced by the rule.
5. Create the Factor Values for the factor, each referencing an effective-dated characteristic value/VEE group pairings.
6. Create the rule, referencing the factor

Creating Exception Handler VEE Rules

Use the following procedure to create Exception Handler VEE rules:

1. Create the To Do Type to be used by the rule.
2. Create the To Do Role to be used by the rule.
3. Create the Exception Type to be used by the rule.
4. Create the rule, including:
 - To Do Type, To Do Role, and Exception Type created in the previous steps.
 - Criteria comparison for the rule.

Chapter 8

Device Communication and Device Events

This chapter provides descriptions of entities related to device communications, including activities, communications, and completion events. This chapter also describes entities related to device events. This chapter includes:

- **Understanding Device Communication**
- **Understanding Device Events**
- **Inbound Communications In Detail**
- **Outbound Communications In Detail**
- **Completion Events In Detail**
- **Device Events In Detail**
- **Configuring Device Communication and Device Event Objects**

Understanding Device Communication

This section provides an overview of entities related to device communications, including activities, communications, and completion events and how they are used in the meter data framework and related products, including Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway.

Activities

Activities are records of a communication related to a device, measuring component, or other entity in the system. Examples of activities include meter read downloads (for manually read meters) or “last gasp” messages sent by devices when they detect they are about to power down. Activities are used extensively in Oracle Utilities Smart Grid Gateway to record meter commands such as device commissioning/decommissioning, remote connect/disconnect, device status check, or on-demand reading commands. Activities are also used in Oracle Utilities Smart Grid Gateway to track statistics related to uploading initial measurements and device events sent from head-end systems.

Commands

Commands issued to devices, such as remote connect, remote disconnect, and others are defined as activities. These represent commands sent via Oracle Utilities Smart Grid Gateway to devices to invoke a specific type of action.

The table below outlines the commands supported by Oracle Utilities Smart Grid Gateway.

Command	Description
Commission Device	A command issued to establish communication between a device and the head-end system. The goal is to ensure connectivity has been established with the device, that any information needed to communicate with the device has been defined in both Oracle Utilities Smart Grid Gateway and the head-end system, and that the device will begin capturing usage and events.
Decommission Device	A command issued to inform the head-end system when a device needs to be removed from a service point, so that no further reads or events will arrive from the device. Decommissioning is invoked when a device must be removed or deactivated. The goal is to stop any communication between the device and the head-end system.
Device Status Check	A command used to test whether the device is communicating with the network, determine the connection status of the device, and when possible, and check if there are any known malfunctions.
On-Demand Read	A request for the most up-to-date reading from a particular device. These commands are not guaranteed to return immediately. In some cases, completing the command might require a person to manually read the device. The purposes are to check the operational status of the device and/or obtain a more recent reading than is currently available.
Remote Connect	A command issued when a device needs to be connected at a service point.
Remote Disconnect	A command issued when a device needs to be disconnected or shut off at a service point.

Attributes used to define commands include the following:

- **Parent Activity:** the parent activity (if any) for the command.
- **Command Effective Date/Time:** the date and time on which the command takes effect. Commands issued prior to this date and time remain in the "Waiting for Effective Date" status until this time, at which time the command is executed.
- **Command Expiration Date/Time:** the date time when the command expires. The command cannot be executed after this date and time.
- **Priority:** the priority for the command.
- **Requester:** the application sending the command.
- **Requester User:** the user who initiated the command.
- **Requester Transaction ID:** an ID for the command, defined by the requester.
- **Utility Device ID:** ID of the device used by the utility. Used to derive the device ID if the device ID is not provided.

When a command is initiated, it in turn creates an outbound communication which sends the command request to the head-end system.

Upload Statistics

Upload statistics are statistics related to the uploading of initial measurement data and device events sent from a head-end system, and are defined as activities in the system. This section describes upload statistics and how they are captured and maintained in Oracle Utilities Smart Grid Gateway, including:

- **Upload Statistics Activities**
- **Upload Statistics - XAI Inbound Services**
- **Head-End System Processing Statistics**

Upload Statistics Activities

There are three types of upload statistics activities:

- **Payload Statistics:** Contains statistics related to a specific payload (file) containing one or more initial measurements or device events. Payload Statistics activities contain:
 - Basic information about the payload (head-end system, file name, and status)
 - Middleware statistics including specifics about the file, the total number of initial measurements or device events processed, the number of initial measurement or device events errors, and total processing time
 - Initial measurement statistics including the number of initial measurements processed
 - Device event statistics including the number of device events processed
- **Payload Error Notification:** Contains details concerning processing errors encountered in an individual payload (file) containing one or more initial measurements or device events. Payload Error Notification activities are related to Payload Statistics activities.
- **Payload Summary:** Contains processing summary statistics for an individual payload (file) containing one or more initial measurements or device events. Payload Summary activities are related to Payload Statistics activities, and are used to update related payload statistics upon the completion of payload processing.

Upload statistics activities are created during processing of payload files as follows:

- When processing begins for a payload, a Payload Statistics activity is created to record the process.

- If an error occurs during processing, a Payload Error Notification activity is created.
- When payload processing is complete, a Payload Summary activity is created, which in turn, updates the Payload Statistics activity with details concerning the processing of the payload, including the start and end time of the processing, the total processing time, the number of initial measurements or device events processed, and the number of initial measurement or device event errors (if any).

Upload Statistics - XAI Inbound Services

Upload statistics activities are created by the middleware components used by Smart Grid Gateway adapters via XAI inbound services. XAI inbound services define the details of how messages are received from an external system, including the activity business object to be invoked when the response message is received. Smart Grid Gateway adapters use a set of XAI inbound services to create upload statistics activities, outlined in the table below:

This type of Upload Statistics Activity...	Is created by this type of XAI Inbound Service
D1-PayloadStatistics	D1-PayloadStatistics
D1-PayloadSummary	D1-PayloadSummary
D1-PayloadErrorNotif	D1-PayloadErrorNotif

Refer to the Oracle Utilities Application Framework documentation for more information about XAI Inbound Services.

Head-End System Processing Statistics

As Oracle Utilities Smart Grid Gateway processes payloads containing initial measurements or device events, statistics for each payload are captured in Payload Statistics activities. Over time, payload statistics for each head-end system are summarized to allow administrators to view summary statistics for the head-end system. These summarized statistics are referred to as head-end system processing statistics.

Head-end system processing statistics are stored as aggregated measurements for aggregator measuring components. A separate aggregator measuring component must be set up for each head-end system for which processing statistics will be aggregated.

Communications

Communications are records of messages sent between Oracle Utilities Smart Grid Gateway and an external system, such as a head-end system or edge application as a result of initiating a command for a device. Communications can flow both outbound and inbound. Communications are most often created as a result of a command activity.

Attributes used to define communications include the following:

- **Device ID:** the ID of the device related to the communication. All communications (and their related commands) are related to a device.
- **AMI Device Identifier Number:** the identifier for the device used by the head-end system.
- **Event Date/Time:** the date and time of the message.
- **Command Information:** details concerning the command that created the communication, including:
 - **Recipient:** the recipient of the command (recipients are defined as service providers)
 - **Transaction ID:** an ID for the command that created the communication.
 - **External Transaction ID:** ID for the command that created the communication in the external system that sent or received the communication.

- **Event Date/Time:** the date and time of the command that created the communication.

See **Understanding the Command Communication Process** below for more information about the role of communications in the device communication process.

Outbound Communications

Outbound Communications represent messages sent from Oracle Utilities Smart Grid Gateway to an external system, such as a head-end system or edge application (such as Oracle Customer Care and Billing). Outbound communications use the following types of objects:

- **Outbound Communication Business Objects**
- **Outbond Message Types**
- **XAI Senders**
- **External Systems**

Outbound Communication Business Objects

An outbound communication business object must be created for each type of message to be sent to an external system. For head-end systems, this is based on the types of messages the system is designed to accept. For example, suppose a head-end system supports the types of commands outlined above (device commission, device decommission, device status check, on-demand readings, remote connect, and remote disconnect), and that this head-end system accepts a separate type of message for each command. For this example, you would need to create outbound communication business objects for each command, as follows:

Command	Outbound Communication Business Object
Commission Device	Commission Device Outbound Communication
Decommission Device	Decommission Device Outbound Communication
Device Status Check	Device Status Check Outbound Communication
On-Demand Read	On-Demand Read Outbound Communication
Remote Connect	Remote Connect Outbound Communication
Remote Disconnect	Remote Disconnect Outbound Communication

Outbond Message Types

A outbound message type must also be created for each type of message to be sent to an external system. Again, this is based on the types of messages the system is designed to accept. To continue the example above, you might create the following outbound message types:

Command	Outbound Message Type
Commission Device	Commission Device
Decommission Device	Decommission Device
Device Status Check	Device Status Check
On-Demand Read	On-Demand Read
Remote Connect	Connect Device
Remote Disconnect	Disconnect Device

Refer to the Oracle Utilities Application Framework documentation for more information about outbound message types.

XAI Senders

You must also create an XAI Sender for each type of message to be sent to an external system. XAI senders define the details of how messages are sent to an external system. As in the case of outbound communication business objects and outbound message types, the set of XAI senders you need to create is based on the types of messages the system is designed to accept. To continue the example above, you might create the following XAI senders:

Command	XAI Sender
Commission Device	Commission Device
Decommission Device	Decommission Device
Device Status Check	Device Status Check
On-Demand Read	On-Demand Read
Remote Connect	Connect Device
Remote Disconnect	Disconnect Device

Refer to the Oracle Utilities Application Framework documentation for more information about XAI senders.

External Systems

You must also create an External System for each external system to which Oracle Utilities Smart Grid Gateway will send messages. Each external system defines a set of outbound message types that will be sent to that system. Each external system outbound message type also specifies the following:

- The processing method used to send the message (Batch, XAI, or Real-time)
- The corresponding XAI senders
- Batch Control (if Processing Method is set to Batch)
- Message XSL, W3C Schema, and Response XSL (as applicable)

To continue the example above, you might create the following external system:

Head-End System			
Outbound Message Type	Processing Method	XAI Sender	Message XSL / Response XSL
Commission Device	Real-time	Commission Device	HES-Request.xml / HES-Response.xml
Decommission Device	Real-time	Decommission Device	HES-Request.xml / HES-Response.xml
Device Status Check	Real-time	Device Status Check	HES-Request.xml / HES-Response.xml
On-Demand Read	Real-time	On-Demand Read	HES-Request.xml / HES-Response.xml
Remote Connect	Real-time	Connect Device	HES-Request.xml / HES-Response.xml
Remote Disconnect	Real-time	Disconnect Device	HES-Request.xml / HES-Response.xml

Refer to the Oracle Utilities Application Framework documentation for more information about external systems.

Inbound Communications

Inbound Communications represent messages sent from a head-end system or edge application (such as Oracle Customer Care and Billing) to Oracle Utilities Smart Grid Gateway. Inbound communications are typically sent in response to a command. Inbound communications use the following types of objects:

- **Inbound Communication Business Objects**
- **XAI Inbound Service**

Inbound Communication Business Objects

An inbound communication business object must be created for each type of message to be received from an external system. For head-end systems, this is based on the types of messages the system is designed to send. To continue the above example, a head-end system supports the types of commands outlined above (device commission, device decommission, device status check, on-demand readings, remote connect, and remote disconnect), and sends a separate type of message in response to each command. For this example, you would need to create the following inbound communication business objects:

Command Being Responded To	Inbound Communication Business Object
Commission Device	Commission Device Response
Decommission Device	Decommission Device Response
Device Status Check	Device Status Check Response
On-Demand Read	On-Demand Read Response
Remote Connect	Remote Connect Response
Remote Disconnect	Remote Disconnect Response

XAI Inbound Service

You must also create an XAI Inbound Service for each type of message to be received from an external system. XAI inbound services define the details of how messages are received from an external system, including the inbound communication business object (or business service or service script) to be invoked when the response message is received. As in the case of inbound communication business objects, the set of XAI inbound services you need to create is based on the types of messages the system is designed to send. To continue the example above, you might create the following XAI inbound services:

XAI Inbound Service	Schema (Inbound Communication Business Object)
Commission Device Response	Commission Device Response
Decommission Device Response	Decommission Device Response
Device Status Check Response	Device Status Check Response
On-Demand Read Response	On-Demand Read Response
Connect Device Response	Remote Connect Response
Disconnect Device Response	Remote Disconnect Response

Refer to the Oracle Utilities Application Framework documentation for more information about XAI Inbound Services.

Completion Events

Completion events are used to create or update data to reflect the effect of an activity or command. Completion events are created upon successful receipt of inbound communications related to an activity or command. For example, a commission device command could result in the creation or update of an install event, while a on-demand read command could result in the creation of an initial measurement.

Attributes used to define completion device events include the following:

- **Activity:** the activity (command) that initiated the completion event.
- **Sequence:** defines the relative order by which completion events for the activity are executed (in the event that more than one completion event is created for an activity).
- **Inbound Communication:** the inbound communication that triggered the completion event.
- **Event Date/Time:** the date and time of the completion event.

Several of the commands supported by Oracle Utilities Smart Grid Gateway have related completion events. The table below describes the completion events for each command.

Command	Completion Event
Commission Device	Device Commissioning Completion Event <ul style="list-style-type: none"> • Creates an install event for the device (if one doesn't exist) • Updates the device's install event status to reflect that the device has been commissioned
Decommission Device	Device Decommissioning Completion Event <ul style="list-style-type: none"> • Updates the device's install event status to reflect that the device has been decommissioned
On-Demand Read	Create IMD Completion Event <ul style="list-style-type: none"> • Creates an initial measurement for the device
Remote Connect	Connect Device Completion Event <ul style="list-style-type: none"> • Updates the device's install event status to reflect that the device has been connected
Remote Disconnect	Disconnect Device <ul style="list-style-type: none"> • Updates the device's install event status to reflect that the device has been disconnected

See **Understanding the Command Communication Process** below for more information about the role of completion events in the device communication process.

Understanding the Command Communication Process

This section provides an overview of the communication process that takes place when a command is initiated for a device. For each step in the process, the table below provides a brief description of the processing that takes place, and lists the specific objects used by the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr. Refer to the *Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide* for more details about the configuration objects used by this adapter.

Note that the process outlined below has been simplified for illustrative purposes, and does not reference every step performed in this process.

Step	Process	Landis+Gyr Adapter Objects
1.	<p>A user initiates a remote connect command for a device.s</p> <p>A remote connect activity business object is instantiated for the command.</p>	Activity BO: Remote Connect (D1-RemoteConnect)
2.	<p>The remote connect command activity business object creates an outbound communication.</p> <p>The specific type of outbound communication business object created is determined by the head-end system service provider (based on the processing role defined in the Enter algorithm of the “Awaiting Response” status of the outbound communication business object’s lifecycle).</p>	Outbound Communication BO: Initiate Connect Disconnect (D3-InitiateConnectDisconnect)
3.	<p>The outbound communication creates an outbound message.</p> <p>The specific type of outbound message created is determined by the head-end system service provider. (based on the processing role defined in the Enter algorithm of the “Awaiting Response” status of the outbound communication business object’s lifecycle).</p>	Outbound Message Type: Connect Device (D3-CONNECT)
4.	<p>The outbound message is sent to middleware components via an External System and XAI Sender..</p> <p>Middleware components utilize Business Process Execution Language (BPEL).</p>	<p>External System: Landis+Gyr Head End System</p> <p>XAI Sender: Connect Device (D3-Connect)</p>
5.	The middleware converts the outbound message from SGG format into the format used by the head-end system, and sends the message to the head-end system.	
6.	When the head-end system sends a response, the middleware receives the response message from the head-end system, and converts it from the format used by the head-end system to SGG format and invokes an XAI Inbound Service.	XAI Inbound Service: D3-ConDisconStChgNotification (D361040925)

Step	Process	Landis+Gyr Adapter Objects
7.	<p>The XAI Inbound Service picks up the message, and creates a corresponding inbound communication.</p> <p>The specific type of inbound communication business object created is determined by the XAI Inbound Service..</p>	<p>XAI Inbound Service: D3-ConDisconStChgNotification (D361040925)</p> <p>Inbound Communication BO: Connect Disconnect State Changed Notification (Multispeak) (D3-ConnectDisconStateChgNtf)</p>
8.	<p>The inbound communication identifies the parent outbound communication.</p>	<p>Outbound Communication BO: Initiate Connect Disconnect (D3-InitiateConnectDisconnect)</p>
9.	<p>The inbound communication creates a completion event to update the status of the device to indicate it has been connected.</p> <p>The specific type of completion event business object created is specified in an Enter algorithm on the “Create Completion Event” Status of the inbound communication business object’s lifecycle.</p>	<p>Completion Event BO: Connect Device (D1-ConnectDevice)</p> <p>Algorithm: D3-CCE (Create Completion Event)</p>
10.	<p>The inbound communication updates the outbound communication.</p> <p>This update is performed by an Enter algorithm on the “Completed” Status of the inbound communication business object’s lifecycle.</p>	<p>Outbound Communication BO: Initiate Connect Disconnect (D3-InitiateConnectDisconnect)</p>
11.	<p>The outbound communication updates the “Connect/Disconnect Completion Flag” and the original activity business object.</p> <p>This update is performed by an Enter algorithm on the “Completed” Status of the outbound communication business object’s lifecycle.</p>	<p>Outbound Communication BO: Initiate Connect Disconnect (D3-InitiateConnectDisconnect)</p>

Understanding Device Events

This section provides an overview of device events and how they are used in the meter data framework and related products, including Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway.

Device Events

Device events are events of some sort that have taken place relative to a device, and can include power outages, power restorations, tampering alerts, command completions, and other events.

Attributes used to define device events include the following:

- **Device Event Date/Time:** the date and time of the event. For events with a duration, such as a power outage, this is the start date and time of the duration.
- **Device Event End Date/Time:** the end date and time of events with durations (such as power outages). Not applicable to events with no duration, such as a tampering alert or power restoration.

In addition, device events also reference details specific to the head-end system that sent the event, including the following:

- **Sender:** the head-end system (defined as a service provider in SGG) from which the event was sent.
- **External Sender ID:** the external ID for the head-end system that sent the event.
- **External Event Name:** the external, head-end-specific name for the event. This name is translated into a "standard" event name within SGG.
- **External Source Identifier:** an identifier for the source of the event.

Standard vs. Paired Device Events

Some device events represent events that occur without a duration, such as a tampering alert, while other device events represent events with a duration, such as an outage (the duration being the time between the start and end of the outage period).

Device events with no duration are defined using “standard” device event business objects. The meter data framework base package provides a sample standard device event business object that can be extended as needed to support implementation-specific requirements. Instances of the standard event business object represent individual device events received into the system.

Device events with a duration are defined using “paired event” business objects, with the first of the pair representing the start of the event, and the last of the pair representing the end of the event. The meter data framework base package provides a sample set of paired device event business objects that can be extended as needed to support implementation-specific requirements. Events of this type can be configured to create or complete activities that represent the event. For example, an outage event might create an outage activity that is completed when power is restored. In this example, the outage event would be the first of the pair, while the power restoration event would be the last of the pair.

When pairs of events arrive in rapid succession (such as a last gasp followed quickly by a power restoration), these "paired event" business objects are designed to prevent them from being sent to subscribing applications.

Sending Device Events to Subscribing Systems

When device events are received, they are typically passed onto to another subscribing system, such as Oracle Utilities Meter Data Management, a customer information system (such as Oracle Utilities Customer Care and Billing), an outage system (such as Oracle Utilities Network Management System), or some other application.

The means of sending device event information to subscribing system is defined in the “How to Send Device Event Related Information” processing method for the service provider representing the subscribing system. Device event information can be sent via outbound communication business object, outbound communication, or batch process.

The method for sending device events to subscribing systems (business object, outbound message, or batch process) can be defined for each device event category, and can be overridden for individual device event types (including the ability to exclude specific device event types within a category. In addition, a default event processing method can be also be configured that applies when methods of transmission aren't specified at the individual category level.

The “Subscribe to Device Event” business service is used to process device event subscription requests, and allow external applications to manage the categories of events they receive.

Understanding Device Event Processing

This section provides an overview of the process that takes place when device events are received. For each step in the process, the table below provides a brief description of the processing that takes place, and lists the specific objects used by the meter data framework

Note that the process outlined below has been simplified for illustrative purposes, and does not reference every step performed in this process.

Step	Process	Meter Data Framework Objects
1.	<p>The head-end system sends a payload of device events to middleware components. The payload contains both standard and paried device events.</p> <p>Oracle Utilities Smart Grid Gateway adapters use Oracle Service Bus. (OSB) as the middleware components for import of usage readings and device events,</p>	
2.	The middleware parses the payload into individual device events, and then transforms each from the format used by the head-end system into the SGG format.	
3.	The middleware then invokes the Device Event Seeder XAI Inbound Service for each device event. The Device Event Seeder XAI Inbound Service is mapped to the Device Event Seeder business object.	XAI Inbound Service: D1-DeviceEventSeeder (D103879193)
4.	<p>The Device Event Seeder business object determines the type of device event business object to create based on the Device Event Mapping business object (extendable lookup) defined for the head-end system service provider.</p> <p>Different head-end systems may use different names for the same type of event. This process maps device event names sent by the head-end system to standard device event names.</p>	<p>Processing Method: D1-HowToProcessDeviceInfo (How to Process Device Related Information)</p> <p>Device Event Mapping BO: D1-DeviceEventMappingLookup</p>

Step	Process	Meter Data Framework Objects
5.	<p>An instance of the appropriate device event business object is created in the system for each device event received.</p> <p>For paired events, the first event business objects may create activities to represent the event.</p>	<p>Standard Device Event BO: D1-StandardDeviceEvent</p> <p>Paired (First) Device Event BO: D1-PairedEventFirstDeviceEvent (Device Event - Paired Event (First))</p> <p>Paired (Last) Device Event BO: D1-PairedEventLastDeviceEvent (Device Event - Paired Event (Last))</p>
6.	<p>If there are systems that subscribe to device events (defined in a processing method for the system service provider), the device event is sent to those systems.</p> <p>Sending device event information is triggered by an Enter algorithm on the “Sent to Subscribers” Status of the device event business object’s lifecycle.</p> <p>Note: Sending device event information may involve configuration of XAI senders, outbound message types, and external systems. Refer to the Oracle Utilities Application Framework for more information about using XAI.</p>	<p>Processing Method: D1-HowToProcDveEvtsInformation (How to Process Device Event Related Info)</p>

Activities In Detail

This section provides details concerning the activity objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom activity objects you create as part of your implementation. This section includes:

- A description of the D1-ACTIVITY maintenance object
- Lists of the base package activity business objects
- Details concerning activity-specific configuration options
- A sample activity business object (D1-RemoteConnect)

Maintenance Object - D1-ACTIVITY

Activity business objects use the D1-ACTIVITY maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-ACTIVITY
Description	Activity
Service Name	D1-ACTIVITY (Activity Maintenance)
Tables	<ul style="list-style-type: none"> • D1_ACTIVITY (Activity) - Primary • D1_ACTIVITY_CHAR (Activity Characteristics) - Child • D1_ACTIVITY_IDENTIFIER (Activity Identifier) - Child • D1_ACTIVITY_LOG (Activity Log) - Child • D1_ACTIVITY_LOG_PARM (Activity Log Parameter) - Child • D1_ACTIVITY_REL (Activity Relationship) - Child • D1_ACTIVITY_REL_OBJ (Activity Related Object) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Activity Business Objects

The meter data framework base package includes the following activity business objects:

Business Object Name	Description
D1-CancelCommand	Cancel Command Instances of this business object represent individual command cancellations in the system.
D1-DeviceCommission	Device Commission Instances of this business object represent individual device commission commands initiated in the system.

Business Object Name	Description
D1-DeviceDecommission	Device Decommission Instances of this business object represent individual device decommission commands initiated in the system.
D1-DeviceStatusCheck	Device Status Check Instances of this business object represent individual device status check (ping) commands initiated in the system.
D1-DeviceWithDurationActivity	Outage Activity with Duration Instances of this business object represent individual outage activities (triggered by outage device events) in the system.
D1-DeviceWithDurationActParent	Device Activity with Duration Parent (used a parent for device activity with duration business objects)
D1-EnableService	Enable Service
D1-GenericConnect	Generic Connect
D1-GenericDisconnect	Generic Disconnect
D1-GenericReadDevice	Generic Read Device
D1-MeterReadDownloadActivity	Meter Read Download Activity Instances of this business object represent individual meter read downloads in the system.
D1-OnDemandReadAbstract	On-Demand Read Abstract Parent (used a a parent for on-demand read activity business objects)
D1-OnDemandReadInterval	On-Demand Read Interval Instances of this business object represent individual interval on-demand read commands initiated in the system.
D1-OnDemandReadScalar	On-Demand Read Scalar Instances of this business object represent individual scalar on-demand read commands initiated in the system.
D1-PayloadErrorNotif	Payload Error Notification Instances of this business object represent individual payload error notifications in the system.
D1-PayloadExtractScheduler	Payload Extract Scheduler (used a a parent for vendor-specific event and usage extract activity business objects)
D1-PayloadNotification	Payload Notification Instances of this business object represent individual payload notifications in the system.

Business Object Name	Description
D1-PayloadStatistics	Payload Statistics Instances of this business object represent individual payload statistics in the system.
D1-PayloadSummary	Payload Summary Instances of this business object represent individual payload statistics summaries in the system.
D1-RemoteConnect	Remote Connect Instances of this business object represent individual remote connect commands initiated in the system.
D1-RemoteDisconnect	Remote Disconnect Instances of this business object represent individual remote disconnect commands initiated in the system.
D1-SPActivityOrchestration	SP Activity Orchestration

The meter data framework base package includes the following “lite” activity business objects:

Business Object Name	Description
D1-ActivityLite	Activity LITE

Configuration Options

This section outlines specific types of BO Options used by activity business objects.

Business Object Options

Activity business objects can make use of the following business object options:

- **Initiate Command Processing Role:** This option defines the processing role used to initiate the command for the activity. Valid values are defined as values for the PROC_ROLE_FLG lookup field. The base package includes the following options:

Lookup Field Value	Description
D1AM	Obtain AMI Device Identifier
D1DA	Activity Notification
D1DC	Device Commission
D1DD	Device Decommission
D1DM	Device Event Mapping
D1DS	Device Status Check
D1EP	Event Processing Default Configuration
D1FR	Response - Fail
D1IM	Initial Measurement Creation
D1IN	On-Demand Read (Interval)

Lookup Field Value	Description
D1LC	Load Check
D1RC	Remote Connect
D1RD	Remote Disconnect
D1RM	Retrieve Meter
D1RR	Response - Received
D1SC	On-Demand Read (Scalar)
D1SD	Send Device Event
D1SR	Response - Success
D1UM	UOM Mapping

Example Activity - D1-RemoteConnect

The table below lists the details of the D1-RemoteConnect activity business object.

Option	Description
Business Object	D1-RemoteConnect
Description	Remote Connect
Maintenance Object	D1-ACTIVITY (Activity)
Application Service	D1-REMOTECONNECTBOAS (Remote Connect BO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> Applicable Processing Role: D1RC (Remote Connect) Applicable Processing Role: D1SC (On-Demand Read - Scalar) Applicable Processing Role: D1LC (Load Check) Related Administration BO: D1-RemoteConnectType (Remote Connect Type) Display UI Map: D1-RemoteConnectDisplay (Remote Connect - Display) Portal Navigation Option: d1ActivityNavOpt (Activity Navigation Option) Display Map Service Script: D1-RtRCDtls (Remote Connect - Retrieve Details for Display) Maintenance UI Map: D1-RemoteConnectMaint (Remote Connect - Maintenance)

Option	Description
Algorithms	<ul style="list-style-type: none"> • Information: D1-CRAINFO (Command Request Activity Information) • Pre-Processing: D1-DEFACTTYP (Determine Activity Type) • Pre-Processing: D1-DMRO (Default Measurement Requested Override) • Pre-Processing: D1-DDR (Determine Device and Recipient) • Validation: D1GINPVAL (Common Input Validation) • Validation: D1-VALMDEST (Validate Measurement Destination) • Validation: D1-VALMREQO (Validate Measurement Requested Override)
Lifecycle	<ul style="list-style-type: none"> • Pending (Initial) • Validation (Interim) • Validation Error (Interim) • Waiting for Effective Date (Interim) • Connection Ready (Interim) • Communication in Progress (Interim) • Communication Error (Interim) • Retry (Interim) • Execute Completion Events (Interim) • Completion Events Error (Interim) • Waiting for Measurement (Interim) • Wait Expired (Interim) • Completed (Final) • Discarded (Final)

Use the Business Object portal to view additional details concerning this business object.

Inbound Communications In Detail

This section provides details concerning the inbound communication objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom inbound communication objects you create as part of your implementation. This section includes:

- A description of the D1-COMMIN maintenance object
- Lists of the base package inbound communication business objects, including “lite” business objects
- A sample inbound communication business object (D1-CommInLite)

Maintenance Object - D1-COMMIN

Inbound communication business objects use the D1-COMMIN maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-COMMIN
Description	Communication In
Service Name	D1-COMMIN (Communication In Maintenance)
Tables	<ul style="list-style-type: none">• D1_COMM_IN (Communication In) - Primary• D1_COMM_IN_CHAR (Communication In Characteristics) - Child• D1_COMM_IN_IDENTIFIER (Communication In Identifier) - Child• D1_COMM_IN_LOG (Communication In Log) - Child• D1_COMM_IN_LOG_PARM (Communication In Log Parameter) - Child• D1_COMM_IN_REL_OBJ (Communication In Related Object) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Inbound Communication Business Objects

The meter data framework base package includes the following “lite” inbound communication business objects:

Business Object Name	Description
D1-CommInLite	Inbound Communication Lite

Oracle Utilities Smart Grid Gateway adapters include vendor-specific inbound communication business objects.

Example Inbound Communication - D1-CommInLite

The table below lists the details of the D1-CommInLite inbound communication business object.

Option	Description
Business Object	D1-CommInLite
Description	Inbound Communication Lite
Maintenance Object	D1-COMMIN (Communication In)
Application Service	F1-DFTAPS (Default Execution Application Service)
Instance Control	Allow New Instances

Use the Business Object portal to view additional details concerning this business object.

Outbound Communications In Detail

This section provides details concerning the outbound communication objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom outbound communication objects you create as part of your implementation. This section includes:

- A description of the D1-COMMOUT maintenance object
- Lists of the base package outbound communication business objects, including “lite” business objects
- A sample inbound communication business object (D1-CommOutLite)

Maintenance Object - D1-COMMOUT

Inbound communication business objects use the D1-COMMOUT maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-COMMOUT
Description	Communication Out
Service Name	D1-COMMOUT (Communication Out Maintenance)
Tables	<ul style="list-style-type: none"> • D1_COMM_OUT (Communication Out) - Primary • D1_COMM_OUT_CHAR (Communication Out Characteristics) - Child • D1_COMM_OUT_IDENTIFIER (Communication Out Identifier) - Child • D1_COMM_OUT_LOG (Communication Out Log) - Child • D1_COMM_OUT_LOG_PARM (Communication Out Log Parameter) - Child • D1_COMM_OUT_REL_OBJ (Communication Out Related Object) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Outbound Communication Business Objects

The meter data framework base package includes the following “lite” outbound communication business objects:

Business Object Name	Description
D1-CommOutLite	Outbound Communication Lite
D1-CommOutPlacerBO	Communication Out Placer BO
D1-CommunicationLite	Communication Lite

Oracle Utilities Smart Grid Gateway adapters include vendor-specific outbound communication business objects.

Example Outbound Communication - D1-CommOutLite

The table below lists the details of the D1-CommOutLite outbound communication business object.

Option	Description
Business Object	D1-CommOutLite
Description	Outbound Communication Lite
Maintenance Object	D1-COMMOUT (Communication Out)
Application Service	F1-DFTAPS (Default Execution Application Service)
Instance Control	Allow New Instances

Use the Business Object portal to view additional details concerning this business object.

Completion Events In Detail

This section provides details concerning the completion event objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom completion event objects you create as part of your implementation. This section includes:

- A description of the D1-CEVT maintenance object
- Lists of the base package completion event business objects
- A sample completion event business object (D1-ConnectDevice)

Maintenance Object - D1-CEVT

Completion event business objects use the D1-CEVT maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-CEVT
Description	Completion Event
Service Name	D1-CEVT (Completion Event Maintenance)
Tables	<ul style="list-style-type: none">• D1_COMPL_EVT (Completion Event) - Primary• D1_COMPL_EVT_CHAR (Completion Event Characteristics) - Child• D1_COMPL_EVT_LOG (Completion Event Log) - Child• D1_COMPL_EVT_LOG_PARM (Completion Event Log Parameters) - Child• D1_COMPL_EVT_REL_OBJ (Completion Event Related Objects) - Child

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Completion Event Business Objects

The meter data framework base package includes the following completion event business objects:

Business Object Name	Description
D1-CommissionDevice	Commission Device Completion Event Instances of this business object represent individual commission device completion events in the system.
D1-CompletionEvent	Completion Event (generic completion event business object used as a parent business object for all other completion event business objects)
D1-ConnectDevice	Connect Device Completion Event Instances of this business object represent individual connect device completion events in the system.

Business Object Name	Description
D1-CreateIMD	Create IMD Completion Event Instances of this business object represent individual create IMD completion events in the system.
D1-DecommissionDevice	Decommission Device Completion Event Instances of this business object represent individual decommission device completion events in the system.
D1-DisconnectDevice	Disconnect Device Completion Event Instances of this business object represent individual disconnect device completion events in the system.

Example Completion Event - D1-ConnectDevice

The table below lists the details of the D1-ConnectDevice completion event business object.

Option	Description
Business Object	D1-ConnectDevice
Description	Connect Device
Maintenance Object	D1-CEVT (Completion Event)
Parent Business Object	D1-CompletionEvent (Completion Event)
Lifecycle Business Object	Completion Event
Application Service	D1-CEVTBOAS (Completion Event BO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> Display UI Map: D1-ConnectDeviceCEvtDisp (Connect Device Completion Event - Display) Portal Navigation Option: d1cevtTabMenu (Completion Event) Display Map Service Script: D1-D1-CEvtRetDt (Completion Event - Retrieve Details for Display) Maintenance UI Map: D1-ConnectDeviceCEvtMaint (Connect Dvc Completion Event - Maintenance)
Algorithms	<ul style="list-style-type: none"> Validation: D1-VALTRCEVT (Validate Transition Completion Events)

Use the Business Object portal to view additional details concerning this business object.

Device Events In Detail

This section provides details concerning the device event objects supplied as part of the base package. This information illustrates how the base package objects were designed, and can serve as the basis for any custom device event objects you create as part of your implementation. This section includes:

- A description of the D1-DVCEVENT maintenance object
- Lists of the base package device event business objects, including “lite” business objects
- A sample device event business object (D1-SmartMeterdeviceEvent)

Maintenance Object - D1-DVCEVENT

Device event business objects use the D1-DVCEVENT maintenance object. The table below outlines some of the details of this maintenance object.

Option/Field	Description
Maintenance Object	D1-DVCEVENT
Description	Device Event
Service Name	D1-DVCEVENT (Device Event Maintenance)
Tables	<ul style="list-style-type: none"> • D1_DVC_EVT (Device Event) - Primary • D1_DVC_EVT_CHAR (Device Event Characteristic) - Child • D1_DVC_EVT_IDENTIFIER (Device Event Identifier) • D1_DVC_EVT_LOG (Device Event Log) - Child • D1_DVC_EVT_LOG_PARM (Device Event Log Parameter) - Child • D1_DVC_EVT_REL_OBJ (Device Event Related Object) - Child
Algorithms	<ul style="list-style-type: none"> • Transition Error: D1-GEN-MOERR (MO Transition Error)

Use the Maintenance Object portal and the Application Viewer to view more details about this maintenance object.

Meter Data Framework Base Package Device Event Business Objects

The meter data framework base package includes the following device event business objects:

Business Object Name	Description
D1-DeviceEvent	Device Event (generic device event business object used as a parent business object for all other device event business objects.
D1-DeviceEventSeeder	Device Event Seeder (used to to determine the device event business object to use when creating new device events)

Business Object Name	Description
D1-DeviceEvtComResp	Device Event Communication Response Instances of this business object represent communications created in response to device events
D1-PairedEventFirstDeviceEvent	Device Event - Paired Event (First) Instances of this business object represent individual paired event (first) device events in the system.
D1-PairedEventLastDeviceEvent	Device Event - Paired Event (Last) Instances of this business object represent individual paired event (last) device events in the system.
D1-StandardDeviceEvent	Standard Device Event Instances of this business object represent individual standard device events in the system.

Example Device Event - D1-DeviceEvent

The table below lists the details of the D1-DeviceEvent device event business object. Note that this business object is used as a parent business object for other device events business objects.

Option	Description
Business Object	D1-DeviceEvent
Description	Device Event
Maintenance Object	D1-DVCEVENT (Device Event)
Application Service	D1-DEVICEEVENTBOAS (Device Event BO)
Instance Control	Allow New Instances
Options	None (options are defined for child business objects)
Algorithms (apply to all child business objects)	<ul style="list-style-type: none"> Information: D1-DEVTINFO (Device Event Info) Validation: D1-VALDVCEVT (Validate Device Event) Validation: D1-VALDEXEVT (Validate External Event Name)
Lifecycle (apply to all child business objects)	<ul style="list-style-type: none"> Pending (Initial) Additional Processing (Interim) Help (Interim) Discarded (Final) Sent to Subscribers (Final)

Use the Business Object portal to view additional details concerning this business object.

Configuring Device Communication and Device Event Objects

This section provides high-level overviews of the steps involved in configuring custom activities, communications (inbound and outbound), completion events, and device events. See **Configuration Process Overview** in **Chapter One** for a high-level overview of the overall configuration process.

Note: The procedures below focus on specific configuration tasks and options related to each of the objects described in this chapter, and do not address all the steps involved in creating business objects, UI maps, algorithms, etc. For more information about these subjects, refer to the Oracle Utilities Application Framework documentation.

Configuring Custom Activities

Configuring custom activities involves the following steps:

1. Design the activity business objects you will need to create for your implementation, including the data and processing required for each.
2. Create the custom activity-related configuration objects required for your business objects, including:

System Events: Create algorithms for the following system events:

- Applicable Processing Role(s)

3. Set up admin records that define the activity types you will use in your implementation.

Configuring Custom Inbound Communications

Configuring custom inbound communications involves the following steps:

1. Design the inbound communication business objects you will need to create for your implementation, including the data and processing required for each.
2. Create your inbound communication business objects..
3. Set up admin records that define the inbound communication types you will use in your implementation.

Configuring Custom Outbound Communications

Configuring custom outbound communications involves the following steps:

1. Design the outbound communication business objects you will need to create for your implementation, including the data and processing required for each.
2. Create your outbound communication business objects.
3. Set up admin records that define the outbound communication types you will use in your implementation.

Configuring Custom Completion Events

Configuring custom completion events involves the following steps:

1. Design the completion event business objects you will need to create for your implementation, including the data and processing required for each.
2. Create your completion event business objects.

Note: Completion event business objects should reference D1-CompletionEvent as their Parent Business Object.

Configuring Custom Device Events

Configuring custom device events involves the following steps:

1. Design the device event business objects you will need to create for your implementation, including the data and processing required for each.
2. Create your device event business objects.

Note: Device event business objects should reference D1-DeviceEvent as their Parent Business Object.

Chapter 9

Batch Processing

This chapter describes the base package batch processes provided with Oracle Utilities meter data framework.

Refer to the following documentation for more information about batch processing:

- Oracle Utilities Smart Grid Gateway Batch Server Administration Guide
- Oracle Utilities Application Framework Business Process Guide (Batch Jobs)
- Oracle Utilities Application Framework Administration Guide (Defining Background Processes)

Meter Data Framework Base Package Batch Controls

The table below lists the batch controls provided in the meter data framework base package.

Batch Control	Name / Description
D1-ACTVY	Activity Monitor Invokes the generic Application Framework auto-transition batch process that can do automatic/scheduled transition for activities
D1-CMCS	Create Measurement Cycle Schedule Routes
D1-COMM	Batch Control for Communications
D1-CPSR	Complete Pending Msrmt Cycle Schd Routes - DEPRECATED
D1-CRERR	Command Request Error - Retry
D1-CRWT	Command Request Wait - Monitor
D1-CSPSR	Create SP Msrmt Cycle Schedule Rte Records
D1-DVEVS	Device Event Seeder Monitor Process
D1-DVEVT	Device Event MO Periodic Monitor Process
D1-EXTSC	Usage/Event Extract Scheduler Monitor
D1-GNIMD	Generic IMD Monitor
D1-ICERR	Inbound Communication Error - Retry
D1-IMD	IMD Monitor Invokes monitoring rules associated with the current state of an initial measurement. All monitoring rules throughout the initial measurement's business object's inheritance chain are considered. Batch parameters govern whether the processing is further restricted by batch code, business object, status, etc.
D1-IMDMC	IMD Monitor Unattached MCs
D1-INEVT	Install Event MO Periodic Monitor Process Invokes monitoring rules associated with the current state of an install event. All monitoring rules throughout the install event's business object's inheritance chain are considered. By default, the process periodically monitors install events whose current state is not associated with a batch code. Batch parameters govern whether the processing is further restricted by batch code, business object and status.
D1-MC	MC MO Periodic Monitor Process Invokes monitoring rules associated with the current state of a measuring component. All monitoring rules throughout the measuring component's business object's inheritance chain are considered. By default, the process periodically monitors measuring components whose current state is not associated with a batch code. Batch parameters govern whether the processing is further restricted by batch code, business object and status.

Batch Control	Name / Description
D1-MSRMT	<p>Measurement - Derive Other Values</p> <p>Initiates a reprocessing of Derive Other Values for Measurements. To do this, it moves the status of the Measurement from OK to REDERIVEVAL. The REDERIVEVAL status will have a monitor algorithm that will execute all "Derive Other Values" algorithms that have been attached to the business object's Plug-in spot.</p> <p>Batch parameters are required and are used to select the specific business object as well as the starting date and ending date for the records to be processed.</p>
D1-OCERR	Outbound Communication Error - Retry
D1-OCWT	Outbound Communication Wait - Monitor
D1-PMCS	<p>Process Measurement Cycle Schedule</p> <p>Looks for measurement cycle routes that are scheduled to be processed and populates them on the measurement cycle schedule. Once the schedule is prepared, the Process Measurement Cycle algorithm (specified on the service point business object) is then invoked for each device in every route that is ready for processing.</p>
D1-PSPSR	Process SP / MC Schedule Route Records
D1-SMMTR	Smart Meter State Monitor Process
D1-SP	<p>Service Point MO Periodic Monitor Process</p> <p>Invokes monitoring rules associated with the current state of a service point. All monitoring rules throughout the service point's business object's inheritance chain are considered.</p> <p>By default, the process periodically monitors service points whose current state is not associated with a batch code. Batch parameters govern whether the processing is further restricted by batch code, business object and status.</p>
D1-SYNC	Data Sync MO Periodic Monitor Process
D1-TOUTR	<p>Time of Use Map Data Generation Monitor</p> <p>Invokes monitor algorithms for Time Of Use Map instances whose current state does not reference a monitor process.</p>
D1-UT	Usage Transaction Monitor Process - NOT USED
D1-UTCD	Usage Transaction Calculate Defer Monitor - NOT USED
D1-UTID	Usage Transaction Issue Detected Monitor - NOT USED
D1-UTSED	Usage Transaction Seeder - Error - NOT USED

Synchronization Request Batch Controls

The table below lists the batch controls used by the meter data framework for data synchronization. “Initial Sync Request - Load Data <batch control>” batch controls load data (created new instances of business objects) for requests of the appropriate type (device, measuring component, etc.). “Initial Sync Request - Resolve Keys <batch control>” batch controls invoke a generic maintenance object transition process to invoke the “Resolve Keys - Initial Sync” algorithm for synchronization requests of the appropriate type.

Batch Control	Description
D1-CMSYN	Composite Sync Request Transition Composite Sync Request BO instances in Pending state to the next state.
D1-SIHER	Initial Sync Request - Error Transition Initial Inbound Sync Request BO instances in Error state to the next state.
D1-SILCN	Initial Sync Request - Load Data Contact
D1-SILDC	Initial Sync Request - Load Data DC
D1-SILDV	Initial Sync Request - Load Data Device
D1-SILIE	Initial Sync Request - Load Data IE
D1-SILMC	Initial Sync Request - Load Data MC
D1-SILSP	Initial Sync Request - Load Data SP
D1-SILUS	Initial Sync Request - Load Data US
D1-SIKCN	Initial Sync Request - Resolve Keys Contact
D1-SIKDC	Initial Sync Request - Resolve Keys DC
D1-SIKDC	Initial Sync Request - Resolve Keys Device
D1-SIKIE	Initial Sync Request - Resolve Keys IE
D1-SIKMC	Initial Sync Request - Resolve Keys MC
D1-SIKSP	Initial Sync Request - Resolve Keys SP
D1-SIKUS	Initial Sync Request - Resolve Keys US
D1-SIOER	Ongoing Sync Request -Error Transition Ongoing Inbound Sync Request BO instances in Error state to the next state.
D1-SIOPE	Ongoing Sync Request - Pending Transition Ongoing Inbound Sync Request BO instances in Pending state to the next state.
D1-SRSDE	Sync Request Seeder - Error Transition Sync Request Seeder BO instances in Error state to the next state.
D2-SIKUS	Initial Sync Request - Resolve Keys US
D2-SILUS	Initial Sync Request - Load Data US

Meter Data Framework Batch Processing Guidelines

This section provides some guidelines around how to schedule batch processing for the batch controls used with Meter Data Framework (including those used by Oracle Utilities Meter Data Management and Oracle Utilities Smart Grid Gateway). Note that there are no strict dependencies between batch controls, meaning there are no situations in which you must run a specific batch process before running any other. Dependencies are based on the way a utility wants to run their business.

Ongoing Processes

Throughout the course of the day, utilities will likely want to run jobs to bring measurement and event data in from their various metering / head-end systems and other external systems. To accompany this, they might want to run the following batch processes:

Ongoing Master Data Sync Processing

If the system is integrated with a customer information system such as Oracle Utilities Customer Care and Billing and synchronizing master data such as service points, install events, usage subscriptions, etc. on an ongoing basis, this data should be as up-to-date as possible when loading and processing data. This can include:

- Processing composite sync requests that will create individual sync requests for multiple objects (D1-CMSYN)
- "Cleaning out" any ongoing sync requests in error from previous runs (D1-SIOER) and those that have more severe issues and remain as sync request "seeders" (D1-SRSDE)
- Processing sync records that are pending (D1-SIOPE), and again those in error (D1-SIOER).

Note: Processing records in error after processing the batch of "pending" ongoing sync requests is a recommended practice as it serves to clean up dependent sync requests that may have been picked up in an incorrect order. For example, attempting to process a Usage Subscription sync request referencing a Service Point before processing that service point's sync request will put the usage subscription sync request in error. Running D1-SIOER will pick up that usage subscription sync request that had been left behind, and can process it successfully now that the service point it references has been synced in.

Ongoing master data sync could be part of a nightly schedule rather than running it throughout the day, depending on whether the attributes being synced impact VEE and/or usage processing. If nothing synced from a source system impacts VEE, then sync processing need not be run before VEE.

Command Processing

If using Oracle Utilities Smart Grid Gateway to process device commands, it's important to keep communications flowing to and from smart meters to provide the most accurate picture of the state of a given meter. This would include:

- Retrying inbound communications in error (D1-ICERR)
- Retrying outbound communications in error (D1-OCERR)
- Processing outbound communications waiting for a response (D1-OCWT) to see if they should be timed out.
- Processing command request activities in error (D1-CRERR) and those that are waiting (D1-CRWT)

Note that base package business objects for communications and command activities are designed to trap any processing errors encountered and transition the object into an Error state. To deal with unexpected errors that can't be trapped, which could leave communications / command activities in unmonitored states, implementations can choose to configure their own batch controls based on the delivered D1-ACTVY and D1-COMM batch controls, restricting records processed by business object or maintenance object as needed.

Initial Measurement Processing

For initial measurement processing, the following batch processes should be scheduled on an ongoing basis:

- Processing initial measurements in the pending state (D1-IMD)
- Processing initial measurements in the exception state (D1-GNIMD)

In addition, processing initial measurement seeders (IMD Seeders) in error should be performed. This can be accomplished through creation of a custom batch control that references the following Java program used by the “Generic IMD Monitor” (D1-GNIMD):

```
com.splwg.base.domain.common.businessObject.batch.AutoTransitionBatchProcess
```

This custom batch process could be configured to restrict processing to the IMD Seeder business object (D1-IMDSeeder).

Event Processing

The base package is configured such that device events are processed immediately upon receipt, since they might need to be sent to some other application such as an outage system. This can be changed by configuring a monitor process on the device event business object to stop records in a specified state, and then use a batch process to process the events all at once. Beyond this type batch-oriented processing for events, other even processing could include:

- Re-processing device event "seeders" in error (D1-DVEVS),
- Picking up device events for processing if they've stopped in any state (D1-DVEVT).

If device events are configured to be held from being sent onto downstream applications, such as to prevent "flicker" outage events (an outage event and a restore event received in rapid succession) from being sent, device event monitoring (D1-DVEVT) should be set up to be run periodically to ensure timely transmission of events.

Daily/Nightly

In addition to the above ongoing processes, the following daily or nightly processes can also be scheduled.

Periodic Interval/Smart Meter Estimation

Periodic estimation is driven by monitoring devices via the “Smart Meter State Monitor Process” batch control (D1-SMMTR) which executes a monitor algorithm to execute the estimation algorithm on the measuring component business object, which in turn can be configured to create Estimated initial measurements.

Measurement Cycle Processing (Optional)

If using a billing system that doesn't request for bill determinants and requires that bill determinants be pushed to it, the following processes can be used:

- Creating meter read cycle schedule routes (D1-CMCS)
- Creating SP / Measurement Cycle Schedule Routes (D1-CSPSR)
- Processing SP / Measurement Cycle Schedule Routes (D1-PSPSR)

The above processes can also be used to drive the download of meter read cycle & route information for manually-read meters.

Periodic / Ad Hoc

In addition to the above ongoing and daily/nightly processes, the following periodic processes can be run on an as needed basis.

TOU Map Data Generation

TOU map data must be in place for all TOU maps used in usage calculations. Generation of this data is performed using the “Time of Use Map Data Generation Monitor” batch control (D1-TOU-TR). This process can be performed for long time periods, such as a year, generating data for all time-of-use maps for the entire following year, or it could be done more frequently, such as whenever schedules are updated via the TOU map templates.

Re-Derive Measurement Values

If certain data such as a factor value was found to be incorrect, derived measurement values might need to be re-calculated across all final measurements for a given date range. This can be performed using the “Measurement - Derive Other Values” batch control (D1-MSRMT). Given the possible volume of data impacted by this process, careful consideration should be given before performing this process.

Master Data Monitoring

The base package also provides batch processes intended to monitor service points (D1-SP), install events (D1-INEVT), and measuring components (D1-MC) on an ad-hoc basis if processing driven from the lifecycles of these objects is needed. Note that the base package contains no such processing.

Chapter 10

Integrating Oracle Utilities Smart Grid Gateway with Other Systems

This chapter provides an overview of how Oracle Utilities Smart Grid Gateway can be integrated with other systems, including:

- **Understanding Communications Processing**
- **Invoking Meter Commands From an External System**
- **Creating Custom Smart Grid Gateway Adapters**
- **Initial Measurement and Device Event XML Formats**

Understanding Communications Processing

This section describes the manner in which Oracle Utilities Smart Grid Gateway communicates with head-end systems, including:

- **One-Way Communications: Import of Usage and Events**
- **Two-Way Communications: Meter Commands**

One-Way Communications: Import of Usage and Events

Importing of usage readings and device events is supported through one-way communications with a head-end system: the head-end system sends a payload of usage readings or events, and Oracle Utilities Smart Grid Gateway picks up the payload and imports the readings and/or events into the system.

The table below provides an overview of the process that takes place when usage or device events are received. For each step in the process, the table below provides a brief description of the processing that takes place, and lists the specific meter data framework objects used in processing that step or operation.

Note that the process outlined below has been simplified for illustrative purposes, and does not reference every step performed in this process.

Step	Process	Meter Data Framework Objects
1.	<p>The head-end system sends a payload of usage readings or device events to middleware components. The device event payload contains both standard and paried device events.</p> <p>Oracle Utilities Smart Grid Gateway adapters use Oracle Service Bus. (OSB) as the middleware components for import of usage readings and device events,</p>	
2.	The middleware parses the payload into individual usage readings or device events and then transforms each from the format used by the head-end system into the SGG format.	
3.	<p>The middleware then invokes an XAI Inbound Service for each reading or device event.</p> <p>For usage readings, the middleware invokes the Initial Load IMD inbound service. This service is mapped to the IMD Seeder business object.</p> <p>For device events, the middleware invokes he Device Event Seeder inbound service. This service is mapped to the Device Event Seeder business object.</p>	<p>Usage Reading XAI Inbound Service: D1-InitialLoadIMD (D107437309)</p> <p>Device Event XAI Inbound Service: D1-DeviceEventSeeder (D103879193)</p>

Step	Process	Meter Data Framework Objects
4.	<p>The IMD Seeder business object determines the type of initial measurement business object to create for each reading based on the processing method defined for the “Initial Measurement Creation” processing role on the head-end system service provider.</p> <p>Different head-end systems may use different names for units of measure (UOMs). The IMD Seeder business object maps vendor-specific UOMs to “standard” UOMs based on the processing method defined for the “UOM Mapping” processing role on the head-end system service provider.</p> <p>The Device Event Seeder business object determines the type of device event business object to create based on the Device Event Mapping business object (extendable lookup) defined for the head-end system service provider.</p> <p>Different head-end systems may use different names for the same type of event. This process maps device event names sent by the head-end system to standard device event names.</p>	<p>IMD Seeder Processing Method: D1-HowToCreateMCInformation (How to Create MD Related Information)</p> <p>UOM MappingBO: D1-HowToProcessDeviceInfo (How to Process Device Related Information)</p> <p>Device Event Seeder Processing Method: D1-HowToProcessDeviceInfo (How to Process Device Related Information)</p> <p>Device Event Mapping BO: D1-DeviceEventMappingLookup</p>
5.	<p>An instance of the appropriate initial measurement data business object is created in the system for each usage reading received.</p> <p>An instance of the appropriate device event business object is created in the system for each device event received.</p> <p>For paired events, the first event business objects may create activities to represent the event.</p>	<p>Initial Measurement Data BOs:</p> <ul style="list-style-type: none"> D1-InitialLoadIMDInterval D1-InitialLoadIMDScalar <p>Standard Device Event BO: D1-StandardDeviceEvent</p> <p>Paired (First) Device Event BO: D1-PairedEventFirstDeviceEvent (Device Event - Paired Event (First))</p> <p>Paired (Last) Device Event BO: D1-PairedEventLastDeviceEvent (Device Event - Paired Event (Last))</p>
6.	<p>Device Events Only:</p> <p>If there are systems that subscribe to device events (defined in a processing method for the system service provider), the device event is sent to those systems.</p> <p>Sending device event information is triggered by an Enter algorithm on the “Sent to Subscribers” Status of the device event business object’s lifecycle.</p> <p>Note: Sending device event information may involve configuration of XAI senders, outbound message types, and external systems. Refer to the Oracle Utilities Application Framework for more information about using XAI.</p>	<p>Processing Method: D1-HowToProcDveEvtsInformation (How to Process Device Event Related Info)</p>

Refer to the following documentation for more details concerning the specific configurations of Oracle Service Bus and the specific business objects used by Oracle Utilities Smart Grid Gateway adapters for importing usage reading and device events:

- Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide
- Oracle Utilities Smart Grid Gateway MV90 Adapter for Itron Configuration Guide

Two-Way Communications: Meter Commands

Issuing meters commands and receiving a response from the head-end system is supported through two-way communications with a head-end system: Oracle Utilities Smart Grid Gateway sends a message for the command, the head-end system receives and performs the command, and then sends an acknowledgement back to Oracle Utilities Smart Grid Gateway.

The table below provides an overview of the communication process that takes place when a command is initiated for a device. For each step in the process, the table below provides a brief description of the processing that takes place, and lists the specific objects used by the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr. Refer to the *Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide* for more details about the configuration objects used by this adapter.

Note that the process outlined below has been simplified for illustrative purposes, and does not reference every step performed in this process.

Step	Process	Landis+Gyr Adapter Objects
1.	<p>A user initiates a remote connect command for a device.</p> <p>A remote connect activity business object is instantiated for the command.</p>	Activity BO: D1-RemoteConnect (Remote Connect)
2.	<p>The remote connect command activity business object creates an outbound communication.</p> <p>The specific type of outbound communication business object created is determined by the head-end system service provider (based on the processing role defined in the Enter algorithm of the “Awaiting Response” status of the outbound communication business object’s lifecycle).</p>	Outbound Communication BO: D3-InitiateConnectDisconnect (Initiate Connect Disconnect)
3.	<p>The outbound communication creates an outbound message.</p> <p>The specific type of outbound message created is determined by the head-end system service provider. (based on the processing role defined in the Enter algorithm of the “Awaiting Response” status of the outbound communication business object’s lifecycle).</p>	Outbound Message Type: D3-CONNECT (Connect Device)

Step	Process	Landis+Gyr Adapter Objects
4.	<p>The outbound message is sent to middleware components via an External System and XAI Sender.</p> <p>Oracle Utilities Smart Grid Gateway adapters use Oracle Business Process Execution Language (BPEL) as the middleware components for communication with head-end systems.</p>	<p>External System: Landis+Gyr Head End System</p> <p>XAI Sender: D3-Connect (Connect Device)</p>
5.	The middleware converts the outbound message from SGG format into the format used by the head-end system, and sends the message to the head-end system.	
6.	When the head-end system sends a response, the middleware receives the response message from the head-end system, and converts it from the format used by the head-end system to SGG format and invokes an XAI Inbound Service.	XAI Inbound Service: D3-ConDisconStChgNotification (D361040925)
7.	<p>The XAI Inbound Service picks up the message, and creates a corresponding inbound communication.</p> <p>The specific type of inbound communication business object created is determined by the XAI Inbound Service.</p>	<p>XAI Inbound Service: D3-ConDisconStChgNotification (D361040925)</p> <p>Inbound Communication BO: D3-ConnectDisconStateChgNtf (Connect Disconnect State Changed Notification (Multispeak))</p>
8.	The inbound communication identifies the parent outbound communication.	Outbound Communication BO: D3-InitiateConnectDisconnect (Initiate Connect Disconnect)
9.	<p>The inbound communication creates a completion event to update the status of the device to indicate it has been connected.</p> <p>The specific type of completion event business object created is specified in an Enter algorithm on the “Create Completion Event” Status of the inbound communication business object’s lifecycle.</p>	<p>Completion Event BO: D1-ConnectDevice (Connect Device)</p> <p>Algorithm: D3-CCE (Create Completion Event)</p>
10.	<p>The inbound communication updates the outbound communication.</p> <p>This update is performed by an Enter algorithm on the “Completed” Status of the inbound communication business object’s lifecycle.</p>	Outbound Communication BO: D3-InitiateConnectDisconnect (Initiate Connect Disconnect)
11.	<p>The outbound communication updates the “Connect/Disconnect Completion Flag” and the original activity business object.</p> <p>This update is performed by an Enter algorithm on the “Completed” Status of the outbound communication business object’s lifecycle.</p>	Outbound Communication BO: D3-InitiateConnectDisconnect (Initiate Connect Disconnect)

Refer to the following documentation for more details concerning the specific configurations of Oracle Business Process Execution and the specific business objects used by Oracle Utilities Smart Grid Gateway adapters for supporting meter commands via two-way communication:

- Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide
- Oracle Utilities Smart Grid Gateway Adapter for Echelon Configuration Guide

Invoking Meter Commands From an External System

This section provides an overview of how to issue or invoke commands through Oracle Utilities Smart Grid Gateway from an external system. This is useful if implementing Oracle Utilities Smart Grid Gateway alongside external systems for which no productized integration is available, such as an outage system or customer information system.

This section includes:

- **Defining External System as a Service Provider**
- **Exposing Command Activity Business Objects**
- **Sending Command Responses to the External System**
- **Configuring A Middleware Communication Layer**

Defining External System as a Service Provider

The first step in enabling the issuing of commands through Oracle Utilities Smart Grid Gateway from an external system is to define the external system as a service provider. Several processes within Oracle Utilities Smart Grid Gateway use the service provider and its associated processing methods to determine how to process specific types of operations.

Service Providers are defined by the following:

- **Service Provider Business Object:** The business object used to create the service provider. For external systems, this should be “External Application (D1-ExternalApplication).”
- **Service Provider:** A code that designates the external system
- **Description:** A description of the external system
- **External Reference ID:** A reference ID for the external system
- **External System:** A related external system (if applicable)
- **Our Name/ID in Their System:** The name of Oracle Utilities Smart Grid Gateway as defined in the external system
- **Utility Device ID Type:** The type of ID used by the external system to identify devices (badge number, external ID, internal meter number, serial number, or pallet number)
- **Utility Measuring Component ID Type:** The type of ID used by the external system to identify measuring components (channel ID or external ID)

Refer to the Oracle Utilities Meter Data Framework user documentation for more information about creating service providers.

The table below contains an example service provider.

Attribute/Field	Value
Service Provider Business Object	External Application (D1-ExternalApplication)
Service Provider	CM_OUTAGE_SYSTEM
Description	Outage System
External Reference ID	12345
External System	N/A
Our Name/ID in Their System	OUSGG
Utility Device ID Type	Serial Number

Attribute/Field	Value
Utility Measuring Component ID Type	Channel ID

Processing Methods

You must also define a number of processing methods for the service provider. These designate the method by which command (activity) related information is sent to the external system.

In this case, you want to define processing methods that designate how outbound messages related to commands (activities) are sent. The table below outlines the processing roles and methods that can be used for this.

Processing Role	Processing Method
Response - Received (D1RR)	How to Send Activity Related Outbound Messages (D1-HowToSendActivityResponses)
Response - Success (D1SR)	How to Send Activity Related Outbound Messages (D1-HowToSendActivityResponses)
Response - Failure (D1FR)	How to Send Activity Related Outbound Messages (D1-HowToSendActivityResponses)

For each of the above processing methods, you would define the outbound message type, or message category and message number to use when sending messages. The type of message to send can be a default type, or can be specified by activity (command) type.

Exposing Command Activity Business Objects

The next step in enabling the issuing of commands through Oracle Utilities Smart Grid Gateway from an external system is to expose the activity business objects for the commands you wish to invoke as web services. This is done by creating an XAI Inbound Service for each command.

XAI Inbound Services are defined by the following:

- **XAI Inbound Service Name:** A code for the inbound service.
- **Adapter:** The type of adapter used by the inbound service. This should be “BusinessAdapter (Business Adapter)”.
- **Schema Type:** The type of object to be invoked via the inbound service (business object, business service, or service script). This should be Business Object.
- **Schema Name:** The activity business object to invoke for the command
- **Description:** A description of the inbound service.

When you create an XAI Inbound Service, the system creates the Web Service Definition Language (WSDL) schema for the service. This can be used by the external system to invoke the web services associated with each command.

The table below lists a set of sample XAI Inbound Services that could be created to support the commands supported through Oracle Utilities Smart Grid Gateway.

XAI Inbound Service	Schema Name
CM-DeviceCommissionService	D1-DeviceCommission (Device Commissioning)
CM-DeviceDecommissionService	D1-DeviceDecommission (Device Decommissioning)
CM-DeviceStatusCheckService	D1-DeviceStatusCheck (Device Status Check)

XAI Inbound Service	Schema Name
CM-OnDemandReadIntervalService	D1-OnDemandReadInterval (On-Demand Read Interval)
CM-OnDemandReadScalarService	D1-OnDemandReadScalar (On-Demand Read Scalar)
CM-RemoteConnectService	D1-RemoteConnect (Remote Connect)
CM-RemoteDisconnectService	D1-RemoteDisconnect (Remote Disconnect)

Refer to the Oracle Utilities Application Framework documentation for more information about XAI Inbound Services.

Sending Command Responses to the External System

The next step in enabling the issuing of commands through Oracle Utilities Smart Grid Gateway from an external system is to set up the configuration necessary to send outbound messages from Oracle Utilities Smart Grid Gateway to the external system. This includes configuring the following:

- **Outbond Message Types**
- **XAI Senders**
- **External Systems**

Outbond Message Types

A outbound message type must also be created for each type of message to be sent to an external system, based on the types of messages the system is designed to accept. To continue the example above, you might create the following outbound message types:

Command	Outbound Message Type
Commission Device	Commission Device
Decommission Device	Decommission Device
Device Status Check	Device Status Check
On-Demand Read Interval	On-Demand Read Interval
On-Demand Read Scalar	On-Demand Read Scalar
Remote Connect	Connect Device
Remote Disconnect	Disconnect Device

Refer to the Oracle Utilities Application Framework documentation for more information about outbound message types.

XAI Senders

You must also create an XAI Sender for each type of message to be sent to an external system. XAI senders define the details of how messages are sent to an external system. The set of XAI senders you need to create is based on the types of messages the system is designed to accept. To continue the example above, you might create the following XAI senders:

Command	XAI Sender
Commission Device	Commission Device
Decommission Device	Decommission Device
Device Status Check	Device Status Check
On-Demand Read	On-Demand Read
Remote Connect	Connect Device
Remote Disconnect	Disconnect Device

Refer to the Oracle Utilities Application Framework documentation for more information about XAI senders.

External Systems

You must also create an External System for each external system to which Oracle Utilities Smart Grid Gateway will send messages. Each external system defines a set of outbound message types that will be sent to that system. Each external system outbound message type also specifies the following:

- The processing method used to send the message (Batch, XAI, or Real-time)
- The corresponding XAI senders
- Batch Control (if Processing Method is set to Batch)
- Message XSL, W3C Schema, and Response XSL (as applicable)

To continue the example above, you might create the following external system:

Head-End System			
Outbound Message Type	Processing Method	XAI Sender	Message XSL / Response XSL
Commission Device	Real-time	Commission Device	OS-Request.xml / OS-Response.xml
Decommission Device	Real-time	Decommission Device	OS-Request.xml / OS-Response.xml
Device Status Check	Real-time	Device Status Check	OS-Request.xml / OS-Response.xml
On-Demand Read	Real-time	On-Demand Read	OS-Request.xml / OS-Response.xml
Remote Connect	Real-time	Connect Device	OS-Request.xml / OS-Response.xml
Remote Disconnect	Real-time	Disconnect Device	OS-Request.xml / OS-Response.xml

Refer to the Oracle Utilities Application Framework documentation for more information about external systems.

Configuring A Middleware Communication Layer

The last step in enabling the issuing of commands through Oracle Utilities Smart Grid Gateway from an external system is to configure a middleware component that sends web service requests from the external system to the XAI Inbound Services, and receives messages from the XAI Senders. This can be via queue-based messaging such as JMS queues, or via tools such as Oracle SOA Suite and Business Process Execution Language (BPEL).

Creating Custom Smart Grid Gateway Adapters

This section provides an overview of the steps involved in creating custom adapters for use with Oracle Utilities Smart Grid Gateway. Adapters are configurations designed to support the specific communication requirements of a specific head-end system currently not supported by Oracle Utilities Smart Grid Gateway, and include:

- **Adapters for Importing Usage Readings and Device Events**
- **Adapters for Issuing and Initiating Commands**
- **Creating Custom Commands**

The Oracle Utilities Smart Grid Gateway demonstration data includes a “generic” adapter that can be used as the basis for creating custom adapters. See **Appendix D: The Oracle Utilities Smart Grid Gateway “Generic” Adapter** for more information about the Oracle Utilities Smart Grid Gateway generic adapter.

Adapters for Importing Usage Readings and Device Events

This section outlines the steps involved in creating an adapter to support import of usage readings and device events from a head-end system. These steps include:

- **Configure Middleware**
- **Create Business Objects**
- **Create Service Provider for Head-End System**

Configure Middleware

Middleware is used to facilitate communication between the head-end system and Oracle Utilities Smart Grid Gateway. In the case of importing usage readings and device events, the middleware performs the following steps:

- Accepts payloads from head-end system

The middleware layer must be configured to accept payloads from the head-end system. This is often an queue-based process utilizing JMS queues configured within an application server.

- Parses payloads into individual readings/events

Usage readings and device events are typically send data in payloads comprising hundreds or even thousands of readings or events. The middleware layer parses these payloads into individual readings or event, each of which is then processed separately.

- Transforms readings/events from head-end system format into SGG format

Usage readings and device events are sent in a format used by the head-end system. The middleware layer transforms each reading or event from the head-end system format into the XML format used by Oracle Utilities Smart Grid Gateway. See **Initial Measurement and Device Event XML Formats** on page 10-31 for details concerning this XML format.

- Sends each reading or event to Oracle Utilities Smart Grid Gateway by invoking an XAI Inbound Service

After parsing and transforming the incoming data, the middleware then sends each reading or event to the application. This can be done via web service call to an XAI Inbound Service or by depositing the reading or event in a JMS queue.

- Creates upload statistics activities to record the processing of each payload

As it processes each payload (including all the steps above), the middleware creates upload statistics activities as follows:

- When processing begins for a payload, a Payload Statistics activity is created to record the process.
- If an error occurs during processing, a Payload Error Notification activity is created.
- When payload processing is complete, a Payload Summary activity is created, which in turn, updates the Payload Statistics activity with details concerning the processing of the payload.

See **Upload Statistics** on page 8-3 for more information about upload statistics activities.

Oracle Utilities Smart Grid Gateway adapters use Oracle Service Bus (OSB) to perform these operations. Refer to the following documentation for more details concerning the specific configurations of Oracle Service Bus used by Oracle Utilities Smart Grid Gateway adapters for importing usage reading and device events:

- Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide
- Oracle Utilities Smart Grid Gateway MV90 Adapter for Itron Configuration Guide
- Oracle Utilities Smart Grid Gateway Adapter for Echelon NES Configuration Guide

Pre-configured OSB processes can be extended to support custom requirements during processing.

Create Business Objects

For each type of usage reading and/or device event to be imported, you need to create business object that defines the initial measurement or device event. If the head-end system uses specific device event names, units of measure, or condition codes, you may also need to create extendable lookup business objects that can be used to perform mapping between the head-end system codes and “standard” codes.

Initial Measurement Data Business Objects

Initial measurement data business objects defines the types of initial measurement data to be received from the head-end system. You must create an initial measurement data business object for each type of initial measurement data you expect to store. For example, the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr uses the following initial measurement data business objects:

Initial Measurement Data Business Object	Description
D3-InitialLoadIMDInterval (Landis+Gyr Initial Load IMD - Interval)	Used for interval initial measurement data received from Landis+Gyr.
D3-InitialLoadIMDScalar (Landis+Gyr Initial Load IMD - Scalar)	Used for scalar initial measurement data received from Landis+Gyr.

Notes:

- All head-end system-specific interval initial measurement data business objects should be child business objects of the D1-InitialLoadIMDInterval business object.
- All head-end system-specific scalar initial measurement data business objects should be child business objects of the D1-InitialLoadIMDScalar business object.

See **Initial Measurement Data In Detail** on page 6-13 for more information about initial measurement data business objects.

Measurement Business Objects

Measurement business objects represent the different types of final measurements to be stored in the system that originate from the head-end system. If final measurements need to capture head-end system-specific data elements, you should create a measurement business object to

accommodate those data elements. See **Measurements In Detail** on page 6-17 for more information about measurement business objects.

Device Event Business Objects

Device event business objects represent the different types of device events that can be sent by the head-end system. Some device events represent events that occur without a duration, such as a tampering alert, while other device events represent events with a duration, such as an outage (the duration being the time between the start and end of the outage period).

Oracle Utilities Smart Grid Gateway provides a set of basic device event business objects that can be extended to support head-end system-specific device events.

- Device events with no duration are defined using “standard” device event business objects.
- Device events with a duration are defined using “paired event” business objects, with the first of the pair representing the start of the event, and the last of the pair representing the end of the event.

Notes:

- All head-end system-specific device event business objects should be child business objects of the D1-DeviceEvent business object.

See **Device Events In Detail** on page 8-26 for more information about device event business objects.

Extendable Lookup Business Objects

If the head-end system uses its own set of units of measure, or device event names, or condition codes, you can create extendable lookup business objects to perform mappings between the head-end system codes and the standard codes used by Oracle Utilities Smart Grid Gateway. You reference these business objects in processing methods defined for the service provider that represents the head-end system.

Refer to the Oracle Utilities Application Framework documentation for more information about extendable lookups.

Create Service Provider for Head-End System

Each head-end system that communicates with Oracle Utilities Smart Grid Gateway should be defined as a service provider. The service provider’s processing roles and methods are used by many processes to determine the specific manner in which data is to be sent or received from the service provider. For example, the “Initial Measurement Creation” processing role defines the specific type of initial measurement data business object to use when creating initial measurement received from the head-end system.

For example, the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr could use the following service provider to represent the Landis+Gyr Command Center head-end system.

Attribute/Field	Value
Service Provider Business Object	Head-End System (D1-HeadEndSystem)
Service Provider	LG
Description	Landis+Gyr
External Reference ID	LG
External System	Landis+Gyr
Our Name/ID in Their System	
AMI Device ID Type	Internal Meter Number

Attribute/Field	Value
AMI Measuring Component ID Type	Channel ID

This service provider uses the following processing methods:

Processing Role	Processing Method
Initial Measurement Creation / How to create Initial Load IMD	<p>D1-HowToCreateMCInformation (How To Create MC Related Information)</p> <p>This processing method specifies the type of initial measurement data business object to use when creating initial measurements received from Landis+Gyr. (D3-InitialLoadIMDScalar)</p> <p>Note: The processing methods can define a default initial measurement data business object, or can specify an initial measurement data business object for each Measuring Component Type.</p>
UOM Mapping / How to map UOMs from L+G when creating IMDs	<p>D1-HowToProcessDeviceInfo (How to Process Device Related Information)</p> <p>This processing method specifies the extendable lookup business object to use when mapping Landis+Gyr UOM codes to SGG UOM codes. (D3-HeadendUOMLookup)</p> <p>Note: The processing methods can define a default extendable lookup business object, or can specify an extendable lookup business object for each Device Type.</p>

Note: The table above only lists processing methods used in processing readings and device events.

Another processing role and method used in processing device events is described below

Processing Role	Processing Method
Device Event Mapping / How to Map Device Events	<p>D1-HowToProcessDeviceInfo (How to Process Device Related Information)</p> <p>This processing method specifies the extendable lookup business object to use when mapping Landis+Gyr device event names to “standard” device event names.</p> <p>Note: The processing methods can define a default extendable lookup business object, or can specify an extendable lookup business object for each Device Type.</p>

See **Service Providers In Detail** on page 5-14 and **Processing Methods In Detail** on page 5-17 for more information about service providers and processing methods.

Adapters for Issuing and Initiating Commands

This section outlines the steps involved in creating an adapter to support initiating commands and communications between Oracle Utilities Smart Grid Gateway and a head-end system. These steps include:

- **Configure Outbound Communication**
- **Configure Middleware**
- **Configure Inbound Communication**
- **Create Service Provider for Head-End System**
- **Creating Custom Commands**

Configure Outbound Communication

Outbound Communications represent messages sent from Oracle Utilities Smart Grid Gateway to the head-end system or edge application (such as Oracle Customer Care and Billing). Outbound communications use the following types of objects:

- **Outbound Communication Business Objects**
- **Outbound Message Types**
- **XAI Senders**
- **External Systems**

Outbound Communication Business Objects

Outbound communication business object must be created for each type of message to be sent to the head-end system. This is based on the types of messages the system is designed to accept. For example, suppose a head-end system supports the following types of commands: device commission, device decommission, device status check, on-demand readings, remote connect, and remote disconnect, and that this head-end system accepts a separate type of message for each command. For this example, you would need to create outbound communication business objects for each command, as follows:

Command	Outbound Communication Business Object
Commission Device	Commission Device Outbound Communication
Decommission Device	Decommission Device Outbound Communication
Device Status Check	Device Status Check Outbound Communication
On-Demand Read	On-Demand Read Outbound Communication
Remote Connect	Remote Connect Outbound Communication
Remote Disconnect	Remote Disconnect Outbound Communication

The specifics of each business object are based on the type of message to be sent to the head-end system. The table below provides an example of the “Initiate Connect Disconnect” outbound communication business object used by the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr.

Option	Description
Business Object	D3-InitiateConnectDisconnect
Description	Initiate Connect Disconnect
Maintenance Object	D1-COMMOUT (Communication Out)
Application Service	D3-INITIATECONDISCON (Initiate Connect Disconnect)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Related Administration BO: D3-InitiateConnectDisconType (Initiate Connect Disconnect Type) • Display UI Map: D3-InitConDisconMltSpkDisplay (Initiate Connect Disconnect Display) • Portal Navigation Option: d1OBCommunicationNavOpt (Outbound Communication Navigation Option) • Display Map Service Script: D3-RtICDDtls (Connect Disconnect - Retrieve Details for Display) • Maintenance Map UI Map: D3-InitConDisconMltSpkMaint (Connect Disconnect Maintenance)
Algorithms	<ul style="list-style-type: none"> • Information: D1-COMMINFO (Communication Information) • Pre-Processing: D3-GDRFFPA (Default required fields from Parent Activity) • Pre-Processing: D3-DLAC (Default Load Action Check)

Option	Description
Lifecycle / Algorithms	<ul style="list-style-type: none"> Pending
Note: Does not include “Transition to Default Next Status” monitor algorithms.	<ul style="list-style-type: none"> Validate <ul style="list-style-type: none"> Enter: D1-VALCOMTP (Validate Communication Type) Validation Error <ul style="list-style-type: none"> Monitor: D1-RBOE (Rety BO in Error) Enter: D1-CTDEBOE (Create To Do Entry for BO in Error) Exit: D1-GTDCBO (Generic To Do Completion for BOs) Awaiting Response <ul style="list-style-type: none"> Monitor: D1-TIMEOUT (Time Out) Enter: D3-PICDSD (Populate Initial Connect Disconnect Send Detail) Enter: D3-CCDOBTMSG (Create Connect/Disconnect Out Bound Message) Retry Response Error <ul style="list-style-type: none"> Same as Validation Error (above) Discarded <ul style="list-style-type: none"> Enter: D1-TPATOF (Transition Parent Activity to Failed) Completed <ul style="list-style-type: none"> Enter: D3-UCCDFTPA (Update Connect Disconnect Completion Flag And Transition Parent Activity)

Outbound Message Types

A outbound message type must also be created for each type of message to be sent to the head-end system. Again, this is based on the types of messages the head-end system is designed to accept. To continue the example above, you might create the following outbound message types:

Command	Outbound Message Type
Commission Device	Commission Device
Decommission Device	Decommission Device
Device Status Check	Device Status Check
On-Demand Read	On-Demand Read
Remote Connect	Connect Device
Remote Disconnect	Disconnect Device

Refer to the Oracle Utilities Application Framework documentation for more information about outbound message types.

XAI Senders

You must also create an XAI Sender for each type of message to be sent to an external system. XAI senders define the details of how messages are sent to an external system. As in the case of outbound communication business objects and outbound message types, the set of XAI senders you need to create is based on the types of messages the head-end system is designed to accept. To continue the example above, you might create the following XAI senders:

Command	XAI Sender
Commission Device	Commission Device
Decommission Device	Decommission Device
Device Status Check	Device Status Check
On-Demand Read	On-Demand Read
Remote Connect	Connect Device
Remote Disconnect	Disconnect Device

Refer to the Oracle Utilities Application Framework documentation for more information about XAI senders.

External Systems

You must also create an External System for each external system to which Oracle Utilities Smart Grid Gateway will send messages. Each external system defines a set of outbound message types that will be sent to that system. The external system also specifies the following for each outbound message type:

- The outbound message type
- The processing method used to send the message (Batch, XAI, or Real-time)
- The corresponding XAI senders
- Batch Control (if Processing Method is set to Batch)
- Message XSL, W3C Schema, and Response XSL (as applicable)

To continue the example above, you might create the following external system:

Head-End System			
Outbound Message Type	Processing Method	XAI Sender	Message XSL / Response XSL
Commission Device	Real-time	Commission Device	HES-Request.xml / HES-Response.xml
Decommission Device	Real-time	Decommission Device	HES-Request.xml / HES-Response.xml
Device Status Check	Real-time	Device Status Check	HES-Request.xml / HES-Response.xml
On-Demand Read	Real-time	On-Demand Read	HES-Request.xml / HES-Response.xml
Remote Connect	Real-time	Connect Device	HES-Request.xml / HES-Response.xml

Head-End System			
Outbound Message Type	Processing Method	XAI Sender	Message XSL / Response XSL
Remote Disconnect	Real-time	Disconnect Device	HES-Request.xsl / HES-Response.xsl

Refer to the Oracle Utilities Application Framework documentation for more information about external systems.

Configure Middleware

Middleware is used to facilitate communication between the head-end system and Oracle Utilities Smart Grid Gateway. In the case of processing commands, the middleware performs the following steps:

- Accepts outbound messages from XAI Senders
The middleware must monitor the XAI senders configured to send messages from Oracle Utilities Smart Grid Gateway to the head-end system.
- Transforms communications from SGG format into head-end system format
Messages are sent in a standard format used by Oracle Utilities Smart Grid Gateway. The middleware must transform each message from this standard format into the format used by the head-end system.
- Sends messages to the head-end system
The middleware must then send the transformed message to the head-end system. In Oracle Utilities Smart Grid Gateway adapters, this is done via a web service call to web services defined for the head-end system.
- Receives responses from the head-end system
The middleware also receives responses from the head-end system that indicate if the command was successfully carried out.
- Transforms communications from head-end system format into SGG format
Response messages are sent in a format used by the head-end system. The middleware must transform each message from this format into the standard format used by Oracle Utilities Smart Grid Gateway.
- Sends response message to Oracle Utilities Smart Grid Gateway by invoking an XAI Inbound Service
The middleware then sends the response message to the application. This can be done via web service call to an XAI Inbound Service or by depositing the message in a JMS queue.

Oracle Utilities Smart Grid Gateway adapters use Oracle Business Process Execution Language (BPEL) to perform these operations. Refer to the following documentation for more details concerning the specific configurations of BPEL used by Oracle Utilities Smart Grid Gateway adapters for processing commands:

- Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr Configuration Guide

Pre-configured BPEL processes can be extended to support custom requirements during message processing.

Configure Inbound Communication

Inbound Communications represent messages sent from a head-end system or edge application (such as Oracle Customer Care and Billing) to Oracle Utilities Smart Grid Gateway. Inbound communications are typically sent in response to a command. Inbound communications use the following types of objects:

- **Inbound Communication Business Objects**
- **XAI Inbound Services**

Inbound Communication Business Objects

An inbound communication business object must be created for each type of message to be received from the head-end systems, based on the types of messages the system is designed to send. To continue the above example, a head-end system that supports the types of commands outlined above (device commission, device decommission, device status check, on-demand readings, remote connect, and remote disconnect) might send a separate type of message in response to each command. For this example, you would need to create the following inbound communication business objects:

Command Being Responded To	Inbound Communication Business Object
Commission Device	Commission Device Response
Decommission Device	Decommission Device Response
Device Status Check	Device Status Check Response
On-Demand Read	On-Demand Read Response
Remote Connect	Remote Connect Response
Remote Disconnect	Remote Disconnect Response

The specifics of each business object are based on the type of message to be sent to the head-end system. The table below provides an example of the “Connect Disconnect State Changed Notification (Multispeak)” inbound communication business object used by the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr.

Option	Description
Business Object	D3-ConnectDisconStateChgNtf
Description	Connect Disconnect State Changed Notification (Multispeak)
Maintenance Object	D1-COMMIN (Communication In)
Application Service	D3-CONDISSTCHGNTBOAS (State Change Notification BO)
Instance Control	Allow New Instances

Option	Description
Options	<ul style="list-style-type: none"> Related Administration BO: D3-ConnectDisconStateChgNtfTyp (Connect Disconnect State Change Notify Type) Display UI Map: D3-ConDisconStChgMltSpkDisplay (State Change Notification - Display) Portal Navigation Option: d1IBCommunicationNavOpt (Inbound Communication Navigation Option) Display Map Service Script: D3-RtSCNDtls (State Change - Retrieve Details for Display) Maintenance Map UI Map: D3-ConDisconStChgMltSpkMaint (State Change Notification - Maintenance)
Algorithms	<ul style="list-style-type: none"> Information: D1-COMMINFO (Communication Information) Pre-Processing: D3-SETDFBOEC (Set Default BO elements - Connect/Disconnect Notification)
Lifecycle / Algorithms	<ul style="list-style-type: none"> Pending Validate <ul style="list-style-type: none"> Enter: D3-FPCBORC (Find Parent Communication BO Remote Connect) Enter: D1-VALCOMTP (Validate Communication Type) Enter: D3-VSCC (Validate State Change Code) Validation Error <ul style="list-style-type: none"> Monitor: D1-RBOE (Rety BO in Error) Enter: D1-CTDEBOE (Create To Do Entry for BO in Error) Exit: D1-GTDCBO (Generic To Do Completion for BOs) Discarded <ul style="list-style-type: none"> Enter: D3-FAILPCOUT (Fail Parent Outbound Communication) Create Completion Event <ul style="list-style-type: none"> D3-CCE (Create Completion Event) Completed <ul style="list-style-type: none"> Enter: D3-UPCMEVDTR (Update Parent Communication's Event Date Time Connect) Enter: D3-TRANPRBO (Transition Parent Outbound Communication BO)

XAI Inbound Services

You must also create an XAI Inbound Service for each type of message to be received from an external system. XAI inbound systems define the details of how messages are received from an external system, including the inbound communication business object (or business service or service script) to be invoked when the response message is received. As in the case of inbound communication business objects, the set of XAI inbound services you need to create is based on the types of messages the system is designed to send. To continue the example above, you might create the following XAI inbound services:

XAI Inbound Service	Schema (Inbound Communication Business Object)
Commission Device Response	Commission Device Response
Decommission Device Response	Decommission Device Response
Device Status Check Response	Device Status Check Response
On-Demand Read Response	On-Demand Read Response
Connect Device Response	Remote Connect Response
Disconnect Device Response	Remote Disconnect Response

Refer to the Oracle Utilities Application Framework documentation for more information about XAI Inbound Services.

Create Service Provider for Head-End System

Each head-end system that communicates with Oracle Utilities Smart Grid Gateway should be defined as a service provider. The service provider's processing roles and methods are used by many processes to determine the specific manner in which data is to be sent or received from the service provider. For example, the "Remote Connect" processing role defines the specific type of outbound communication business object to use when sending outbound communications to the head-end system as a result of a remote connect command.

The Landis+Gyr service provider provided with the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr uses the following processing methods:

Processing Role	Processing Method
On-Demand Read (Scalar) (D1SC)	D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message) This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to Landis+Gyr as a result of an on-demand read (scalar) command. (D3-InitiateMRByMtrNbr / Initiate Meter Read By Meter Number (CMD3))

Processing Role	Processing Method
Remote Connect (D1RC)	<p>D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message)</p> <p>This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to Landis+Gyr as a result of a remote connect command. (D3-InitiateConnectDisconnect / Connect Device (D3-CONNECT))</p>
Remote Disconnect (D1RD)	<p>D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message)</p> <p>This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to Landis+Gyr as a result of a remote connect command. (D3-InitiateConnectDisconnect / Disconnect Device (D3-DISCONNEC))</p>

Note: The table above only lists processing methods used in processing commands.

Command processing methods can define a default outbound communication business object/outbound message type, or can specify an outbound communication business object/outbound message type for each Device Type.

Other processing roles and methods used in processing commands are described below

Processing Role	Processing Method
Device Commission (D1DC)	<p>D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message)</p> <p>This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to a head-end system as a result of a commission device command.</p>
Device Decommission (D1DD)	<p>D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message)</p> <p>This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to a head-end system as a result of a decommission device command.</p>

Processing Role	Processing Method
Device Status Check (D1DS)	<p>D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message)</p> <p>This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to a head-end system as a result of a device status check command.</p>
On-Demand Read (Interval) (D1IN)	<p>D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message)</p> <p>This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to a head-end system as a result of an on-demand read (interval) command.</p>

See **Service Providers In Detail** on page 5-14 and **Processing Methods In Detail** on page 5-17 for more information about service providers and processing methods.

Creating Custom Commands

Oracle Utilities Smart Grid Gateway supports a number of commands, including:

- Commission Device
- Decommission Device
- Device Status Check
- On-Demand Read
- Remote Connect
- Remote Disconnect

However, most head-end systems provide additional commands beyond these. This section outlines the steps involved in creating custom commands, including:

- **Create Command (Activity) Business Objects**
- **Create Processing Roles and Configure Processing Methods**
- **Create Outbound and Inbound Communications**
- **Configure Middleware to Support New Messages**

Create Command (Activity) Business Objects

Each command available through Oracle Utilities Smart Grid Gateway is defined as an activity business object. For each custom command you wish to create, you must create an activity business object. The specifics of each business object are based on the type of command to be issued to the head-end system. The table below provides an example of the “Remote Connect” activity business object used by the Oracle Utilities Smart Grid Gateway.

Option	Description
Business Object	D1-RemoteConnect
Description	Remote Connect
Maintenance Object	D1-ACTIVITY (Activity)
Application Service	D1-REMOTECONNECTBOAS (Remote Connect BO)
Instance Control	Allow New Instances
Options	<ul style="list-style-type: none"> • Applicable Processing Role: D1RC (Remote Connect) • Applicable Processing Role: D1SC (On-Demand Read - Scalar) • Applicable Processing Role: D1LC (Load Check) • Related Administration BO: D1-RemoteConnectType (Remote Connect Type) • Display UI Map: D1-RemoteConnectDisplay (Remote Connect - Display) • Portal Navigation Option: d1ActivityNavOpt (Activity Navigation Option) • Display Map Service Script: D1-RtRCDtls (Remote Connect - Retrieve Details for Display) • Maintenance UI Map: D1-RemoteConnectMaint (Remote Connect - Maintenance)

Option	Description
Algorithms	<ul style="list-style-type: none"> Information: D1-CRAINFO (Command Request Activity Information) Pre-Processing: D1-DEACTTYP (Determine Activity Type) Pre-Processing: D1-DMRO (Default Measurement Requested Override) Pre-Processing: D1-DDR (Determine Device and Recipient) Validation: D1GINPVAL (Common Input Validation) Validation: D1-VALMDEST (Validate Measurement Destination) Validation: D1-VALMREQO (Validate Measurement Requested Override)
Lifecycle Note: Does not include “Transition to Default Next Status” monitor algorithms.	<ul style="list-style-type: none"> Pending Validation <ul style="list-style-type: none"> Enter: D1-VALACTTDI (Validate Activity Type and Transition to Error State If Invalid) Enter: D1-VALDVCNCD (Validate Device is not already Connected) Enter: D1-VHCPODR (Validate Head-end’s Capability to perform On-Demand Read) Enter: D1-VHCPCD (Validate Head-end’s Capability to perform Connect Disconnect) Enter: D1-CECD (Check for existing future disconnects) Validation Error <ul style="list-style-type: none"> Monitor: D1-WTTMOUT (Wait Time Out) Monitor: D1-RBOE (Rety BO in Error) Enter: D1-CTDEBOE (Create To Do Entry for BO in Error) Exit: D1-GTDCBO (Generic To Do Completion for BOs) Waiting for Effective Date <ul style="list-style-type: none"> Monitor: D1-WAITFFDT (Wait for Effective Date) Enter: D1-RRER (Send Received Response to External Requester) Connection Ready <ul style="list-style-type: none"> Enter: D1-CLCOC (Create Load Check Outbound Communication) Enter: D1-CODROC (Create On-Demand Read for Start Measurement) Enter: D1-CRCOC (Create Remote Connect Outbound Communication)

Option	Description
Lifecycle (continued)	<ul style="list-style-type: none"> Communication in Progress
Note: Does not include “Transition to Default Next Status” monitor algorithms.	<ul style="list-style-type: none"> Communication Error <ul style="list-style-type: none"> Same as Validation Error (above) Retry <ul style="list-style-type: none"> Enter: D1-COOC (Cancel Outstanding Outbound Communication) Execute Completion Events <ul style="list-style-type: none"> Enter: D1-EXCMPEVTS (Execute Completion Events) Completion Events Error <ul style="list-style-type: none"> Same as Validation Error (above) Waiting for Measurement <ul style="list-style-type: none"> Monitor: D1-RMVCE (Retrieve Measurements via Completion Events) Monitor: D1-RFINSC (Retrieve Scalar Final Measurements) Monitor: D1-RSINIMS (Retrieve Scalar Initial Measurements) Monitor: D1-WFMTO (Wait for Measurement Time Out) Wait Expired <ul style="list-style-type: none"> Same as Validation Error (above) Completed <ul style="list-style-type: none"> Enter: D1-SRCNTEA (Send Remote Connect Notification to Edge Application) Enter: D1-SODRTEA (Send Start Measurement to Edge Application) Discarded <ul style="list-style-type: none"> Enter: D1-COOC (Cancel Outstanding Outbound Communication) Enter: D1-COCE (Cancel Outstanding Completion Events) Enter: D1-FAILPA (Fail Parent Activity) Enter: D1-FRER (Send Fail Response to External Requester)

Create Processing Roles and Configure Processing Methods

After the command activity business object has been created, you must also create a processing role for the command, and associate to your activity business object and the appropriate processing method business object, and then add the processing role/method for the command to the service provider that represents the head-end system.

Create Processing Role For The Command

To create a processing role for your new command, add a value that designates your command to the PROC_ROLE_FLG lookup.

For example, suppose you were creating a custom command to support retrieving outage events for a device. Your lookup value might look like this:

Lookup Field Value	Description (Java Value Name)
CMRO	Retrieve Outage Events (retrieveOutageEvents)

Refer to the Oracle Utilities Application Framework documentation for more information about creating values for lookup fields.

Add Processing Role to Processing Method Business Object

Once the processing role lookup has been created, you have to add it as an Applicable Processing Role on the processing method business object that will be used to determine how to send out outbound communications. For most Oracle Utilities Smart Grid Gateway commands, this is the D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message) business object.

To continue the example of the “Retrieve Outage Events” command, you would add the following option to the D1-HowToCreateActivityOBComm business object on the Main tab of the Business Object portal:

Option Type	Option Value
Applicable Processing Role	CMRO

Add Processing Role to Command Business Object

You must also add your processing role lookup as an Initiate Command Processing Role on the command (activity) business object you just created.

To continue the example of the “Retrieve Outage Events” command, you would add the following option to the your activity business object on the Main tab of the Business Object portal:

Option Type	Option Value
Initiate Command Processing Role	CMRO

Add Processing Role/Method to Service Provider

You must also add your new processing role and corresponding processing method to the service provider that represents the head-end system.

To continue the example of the “Retrieve Outage Events” command, you would add the following processing method to your head-end system service provider.

Processing Role	Processing Method
Retrieve Outage Events (CMRO)	D1-HowToCreateActivityOBComm (How To Create OB Communication/Send OB Message) This processing method specifies the type of outbound communication business object and outbound message type to use when sending outbound communications to a head-end system as a result of a retrieve outage events command.

Create Outbound and Inbound Communications

The next step in enabling your new command is to create any necessary outbound and inbound communications to allow the messages related to the command to be sent/received to and from the head-end system. See **Adapters for Issuing and Initiating Commands** on page 10-16 for more information about creating outbound and inbound communications.

Configure Middleware to Support New Messages

Lastly, you need to configure your middleware to support transformation of the communications used by your new command. For example, if your middleware uses XSLT to perform transformation, you need to add specifics about how to transform messages related to “Retrieve Outage Events” command.

Initial Measurement and Device Event XML Formats

This section provides details concerning the XML formats used when importing initial measurements and device events, including:

- **Initial Measurement Data XML Format**
- **Device Event XML Format**

Initial Measurement Data XML Format

This section describes the XML format used for inbound initial measurement data. This includes interval and scalar examples, descriptions of the individual XML elements, and the initial measurement data XML schema based on the D1-IMDSeeder business object.

Example - Interval Initial Measurement Data

```
<IMD-IMPORT>
  <serviceProvider>HEADEND-1</serviceProvider>
  <serviceProviderExternalId>MDCS-1</serviceProviderExternalId>
  <preVEE>
    <dvcIdN>037090184721</dvcIdN>
    <mcId>135914144111</mcId>
    <mcIdN>123</mcIdN>
    <externalId>IMD1234567</externalId>
    <uom>KWH</uom>
    <stDt>2009-01-02-00.00.00</stDt>
  </stQty>
  <enDt>2009-01-03-00.00.00</enDt>
</enQty>
  <imdType>D1IL</imdType>
  <inShift>N</inShift>
  <mcm>1.0</mcm>
</nd>
  <tz>USPACIFIC</tz>
  <spi>3600</spi>
</ccond>
  <sts>
    <stsL>
      <s>1</s>
      <st>REGULAR</st>
    </stsL>
  </sts>
  <msrs>
    <mL>
      <s>1</s>
      <q>1.6</q>
      <sts>
        <stsL>
          <s>1</s>
          <st>REGULAR</st>
        </stsL>
      </sts>
    </mL>
    <mL>
      <s>2</s>
      <q>1.57</q>
      <sts>
        <stsL>
          <s>1</s>
          <st>REGULAR</st>
        </stsL>
      </sts>
    </mL>
    <mL>
      <s>3</s>
      <q>0.0</q>
```

```
        <sts>
          <stsL>
            <s>1</s>
            <st>MISSING</st>
            <s>2</s>
            <st>OUTAGE</st>
          </stsL>
        </sts>
      </mL>
    <mL>
      <s>4</s>
      <q>0.0</q>
      <sts>
        <stsL>
          <s>1</s>
          <st>MISSING</st>
          <s>2</s>
          <st>OUTAGE</st>
        </stsL>
      </sts>
    </mL>
    <mL>
      <s>5</s>
      <q>1.0</q>
      <sts>
        <stsL>
          <s>1</s>
          <st>REGULAR</st>
        </stsL>
      </sts>
    </mL>
    <mL>
      <s>6</s>
      <q>1.45</q>
      <sts>
        <stsL>
          <s>1</s>
          <st>REGULAR</st>
        </stsL>
      </sts>
    </mL>
    ...
  </msrs>
</preVEE>
</IMD-IMPORT>
```

Example - Scalar Initial Measurement Data

```
<IMD-IMPORT>
  <serviceProvider>HEADEND-2</serviceProvider>
  <serviceProviderExternalId>MDCS-2</serviceProviderExternalId>
  <preVEE>
    <dvcIdN>037090184721</dvcIdN>
    <mcId>327604570580</mcId>
    <mcIdN>123</mcIdN>
    <externalId>IMD7654321</externalId>
    <uom>KWH</uom>
    <stDt>2009-01-31-11.25.00</stDt>
  </stQty>
  <enDt>2009-02-28-13.13.00</enDt>
  <enQty>110.00</enQty>
  <imdType>D1IL</imdType>
  <inShift>N</inShift>
  <mcm>1.0</mcm>
  <nd>5</nd>
  <tz>USPACIFIC</tz>
</ccond>
<sts>
```

```

    <stsL>
      <s>1</s>
      <st>REGULAR</st>
    </stsL>
  </sts>
</preVEE>
</IMD-IMPORT>

```

Element Descriptions - Initial Measurement Data

The table below provides descriptions of the elements used in the initial measurement data XML format.

Element	Description
<{SERVICE_NAME}>	Root element containing an initial measurement. This element should match the name of the inbound service used to import the usage.
<serviceProvider>	Name of the head-end system (defined as a service provider) in MDM
<serviceProviderExternalId>	External ID of the head-end system.
<preVEE>	Element containing Pre VEE measurement data
<dvcIdN>	Device Identifier Number, e.g. a Serial Number. The "type" of device identifier that the head-end system understands is configured within MDM - and so can differ per head-end.
<mcId>	Measuring Component ID
<mcIdN>	Measuring Component Identifier Number. Populated with channel ID. The "type" of measuring component identifier that the head-end system understands is configured within MDM
<externalId>	File name of ID of the XML document containing the measurement data
<uom>	Unit of Measure (Optional)
<stDt>	Start Date/Time. Required for interval measurement data. Optional for scalar measurement data. Must be in the following format: YYYY-MM-DD-HH.MM.SS Example:2008-12-31-00.30.00
<stQty>	Start Reading. Optional.
<enDt>	End Date/Time. Required. Must be in the following format: YYYY-MM-DD-HH.MM.SS Example:2008-12-31-00.30.00

Element	Description
<enQty>	End Reading. Required for scalar measurements.
<imdType>	Initial measurement data type. Valid values include: <ul style="list-style-type: none"> D1IL (Initial Load) D1MO (Manual) D1ES (Estimation) Defaults to D1IL (Initial Load) if not supplied.
<inShift>	Incoming Data Shift flag. Indicates if the device is DST aware.
<mcm>	Meter multiplier.
<nd>	Number of Dials
<tz>	Time Zone
<spi>	Seconds-per-interval
<ccond>	Measurement condition
<sts>	Element containing status code information for entire measurement.
<stsL>	List of head-end system status codes for the entire measurement
<s>	Sequence
<st>	Head-end status codes applicable to the entire set of data.
<msrs>	Element containing measurement data
<mL>	Element containing an individual interval measurements. Used for interval measurement data only.
<s>	Sequence of the interval measurement
<dt>	Date and time of the interval measurement. Optional.
<q>	Quantity of the interval measurement
<ue>	Used-Edited flag. Indicates if the interval measurement has been manually edited
<fc>	Final Condition code for the interval measurement. Optional.
<sts>	Element containing lists of status codes for each interval measurement

Element	Description
<stSL>	Element containing a sequence/status pairing for each interval measurement
<S>	Sequence of the status code for this interval measurement.
<st>	Head-end status code of the interval measurement.

Schema - IMD Seeder (D1-IMDSeeder) Business Object

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ouaf="http://
ouaf.oracle.com/" targetNamespace="http://oracle.com/D1-InitialLoadIMD.xsd"
elementFormDefault="qualified">
  <xsd:import namespace="http://ouaf.oracle.com/" />
  <xsd:element name="D1-InitialLoadIMD">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="initialMeasurementDataId" type="xsd:string"
minOccurs="0" />
        <xsd:element name="preVEE" minOccurs="0">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="simdId" type="xsd:string" minOccurs="0" />
              <xsd:element name="dvcIdN" type="xsd:string" minOccurs="0" />
              <xsd:element name="mcId" type="xsd:string" minOccurs="0" />
              <xsd:element name="mcIdN" type="xsd:string" minOccurs="0" />
              <xsd:element name="externalId" type="xsd:string" minOccurs="0" /
            >
            <xsd:element name="uom" type="xsd:string" minOccurs="0" />
            <xsd:element name="externalUOM" type="xsd:string" minOccurs="0"
          />
          <xsd:element name="stDt" type="xsd:dateTime" minOccurs="0" />
          <xsd:element name="stQty" type="xsd:decimal" minOccurs="0" />
          <xsd:element name="enDt" type="xsd:dateTime" minOccurs="0" />
          <xsd:element name="enQty" type="xsd:decimal" minOccurs="0" />
          <xsd:element name="imdType" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:enumeration value="estimation" />
                <xsd:enumeration value="imdSeeder" />
                <xsd:enumeration value="initialLoad" />
                <xsd:enumeration value="manual" />
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="inShift" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:enumeration value="notShifted" />
                <xsd:enumeration value="shifted" />
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="mcm" type="xsd:decimal" minOccurs="0" />
          <xsd:element name="nd" type="xsd:decimal" minOccurs="0" />
          <xsd:element name="tz" type="xsd:string" minOccurs="0" />
          <xsd:element name="spi" type="xsd:int" minOccurs="0" />
          <xsd:element name="ccond" minOccurs="0">
            <xsd:simpleType>
              <xsd:restriction base="xsd:string">
                <xsd:enumeration value="301000" />
                <xsd:enumeration value="901000" />
                <xsd:enumeration value="501000" />
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:schema>

```

```

        <xsd:enumeration value="101000" />
        <xsd:enumeration value="100000" />
        <xsd:enumeration value="201000" />
        <xsd:enumeration value="401000" />
        <xsd:enumeration value="402000" />
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="sts" minOccurs="0">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="stsL" minOccurs="0" maxOccurs
="unbounded">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="s" type="xsd:decimal" minOccurs="0"
/>
                        <xsd:element name="st" type="xsd:string" minOccurs="0"
/>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
<xsd:element name="msrs" minOccurs="0">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="mL" minOccurs="0" maxOccurs="unbounded">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="s" type="xsd:decimal" minOccurs="0"
/>
                        <xsd:element name="dt" type="xsd:dateTime"
minOccurs="0" />
                        <xsd:element name="q" type="xsd:decimal" minOccurs="0"
/>
                        <xsd:element name="ue" minOccurs="0">
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:string">
                                    <xsd:enumeration value="userEdited" />
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:element>
                        <xsd:element name="fc" minOccurs="0">
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:string">
                                    <xsd:enumeration value="301000" />
                                    <xsd:enumeration value="901000" />
                                    <xsd:enumeration value="501000" />
                                    <xsd:enumeration value="101000" />
                                    <xsd:enumeration value="100000" />
                                    <xsd:enumeration value="201000" />
                                    <xsd:enumeration value="401000" />
                                    <xsd:enumeration value="402000" />
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:element>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
<xsd:element name="sts" minOccurs="0">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="stsL" minOccurs="0"
maxOccurs="unbounded">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="s" type="xsd:decimal"
minOccurs="0" />

```

```

minOccurs="0" />
        <xsd:element name="st" type="xsd:string"
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
        <xsd:element name="rawData" type="xsd:anyType" minOccurs="0"
maxOccurs="unbounded" />
        <xsd:element name="processData" minOccurs="0">
        <xsd:complexType>
        <xsd:sequence>
        <xsd:element name="isShiftedStartEnd" minOccurs="0">
        <xsd:simpleType>
        <xsd:restriction base="xsd:string">
        <xsd:enumeration value="no" />
        <xsd:enumeration value="yes" />
        </xsd:restriction>
        </xsd:simpleType>
        </xsd:element>
        <xsd:element name="isShiftedIntervals" minOccurs="0">
        <xsd:simpleType>
        <xsd:restriction base="xsd:string">
        <xsd:enumeration value="no" />
        <xsd:enumeration value="yes" />
        </xsd:restriction>
        </xsd:simpleType>
        </xsd:element>
        <xsd:element name="isErrorEncountered" minOccurs="0">
        <xsd:simpleType>
        <xsd:restriction base="xsd:string">
        <xsd:enumeration value="no" />
        <xsd:enumeration value="yes" />
        </xsd:restriction>
        </xsd:simpleType>
        </xsd:element>
        <xsd:element name="servicePointId" type="xsd:string"
minOccurs="0" />
        <xsd:element name="installationConstant" type="xsd:decimal"
minOccurs="0" />
        <xsd:element name="deviceId" type="xsd:string" minOccurs="0" />
        <xsd:element name="logs" minOccurs="0">
        <xsd:complexType>
        <xsd:sequence>
        <xsd:element name="logsList" minOccurs="0"
maxOccurs="unbounded">
        <xsd:complexType>
        <xsd:sequence>
        <xsd:element name="logsEntry" minOccurs="0">
        <xsd:complexType>
        <xsd:sequence>
        <xsd:element name="sequence" type="xsd:decimal"
minOccurs="0" />
        <xsd:element name="mo" type="xsd:string"
minOccurs="0" />
        <xsd:element name="pkValue1" type="xsd:string"
minOccurs="0" />

```

minOccurs="0" />	<xsd:element name="pkValue2" type="xsd:string"
minOccurs="0" />	<xsd:element name="pkValue3" type="xsd:string"
minOccurs="0" />	<xsd:element name="pkValue4" type="xsd:string"
minOccurs="0" />	<xsd:element name="pkValue5" type="xsd:string"
minOccurs="0" />	<xsd:element name="logEntryType" minOccurs="0"> <xsd:simpleType> <xsd:restriction base="xsd:string"> <xsd:enumeration value="todos" /> <xsd:enumeration value="created" /> <xsd:enumeration
value="statusTransitionError" />	<xsd:enumeration value="exception" /> <xsd:enumeration value="statusTransition" />
>	<xsd:enumeration value="system" /> <xsd:enumeration value="todo" /> <xsd:enumeration value="userDetails" /> </xsd:restriction> </xsd:simpleType> </xsd:element> <xsd:element name="logDateTime"
type="xsd:dateTime" minOccurs="0" />	<xsd:element name="boStatus" type="xsd:string"
minOccurs="0" />	<xsd:element name="description"
type="xsd:string" minOccurs="0" />	<xsd:element name="user" type="xsd:string"
minOccurs="0" />	<xsd:element name="logMessage" type="xsd:string"
minOccurs="0" />	<xsd:element name="characteristicType"
type="xsd:string" minOccurs="0" />	<xsd:element name="characteristicValue"
type="xsd:string" minOccurs="0" />	<xsd:element name="adhocValue" type="xsd:string"
minOccurs="0" />	<xsd:element name="fkValue1" type="xsd:string"
minOccurs="0" />	<xsd:element name="fkValue2" type="xsd:string"
minOccurs="0" />	<xsd:element name="fkValue3" type="xsd:string"
minOccurs="0" />	<xsd:element name="fkValue4" type="xsd:string"
minOccurs="0" />	<xsd:element name="fkValue5" type="xsd:string"
minOccurs="0" />	<xsd:element name="messageCategory"
type="xsd:decimal" minOccurs="0" />	<xsd:element name="messageNumber"
type="xsd:decimal" minOccurs="0" />	<xsd:element name="messageParm1"
type="xsd:string" minOccurs="0" />	<xsd:element name="messageParm2"
type="xsd:string" minOccurs="0" />	<xsd:element name="messageParm3"
type="xsd:string" minOccurs="0" />	<xsd:element name="messageParm4"
type="xsd:string" minOccurs="0" />	<xsd:element name="messageParm5"
type="xsd:string" minOccurs="0" />	<xsd:element name="messageParm6"
type="xsd:string" minOccurs="0" />	<xsd:element name="messageParm7"
type="xsd:string" minOccurs="0" />	

```

        <xsd:element name="messageParm8"
type="xsd:string" minOccurs="0" />
        <xsd:element name="messageParm9"
type="xsd:string" minOccurs="0" />
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:element name="boStatus" type="xsd:string" minOccurs="0" />
<xsd:element name="statusReason" type="xsd:string" minOccurs="0" />
<xsd:element name="bo" type="xsd:string" />
<xsd:element name="creationDateTime" type="xsd:dateTime" minOccurs="0"
/>
<xsd:element name="boStatusDateTime" type="xsd:dateTime" minOccurs="0"
/>
<xsd:element name="isTraceOn" minOccurs="0">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="no" />
            <xsd:enumeration value="yes" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="isIntervalDateTimePopulated">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="no" />
            <xsd:enumeration value="yes" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="isReprocessPerformed" minOccurs="0">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="no" />
            <xsd:enumeration value="yes" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="serviceProvider" type="xsd:string" minOccurs="0" />
<xsd:element name="serviceProviderExternalId" type="xsd:string"
minOccurs="0" />
<xsd:element name="fromDateTime" type="xsd:dateTime" minOccurs="0" />
<xsd:element name="toDateTime" type="xsd:dateTime" minOccurs="0" />
<xsd:element name="timeZone" type="xsd:string" minOccurs="0" />
<xsd:element name="isAutomatedRetry" minOccurs="0">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="no" />
            <xsd:enumeration value="yes" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="retryUntilDateTime" type="xsd:dateTime"
minOccurs="0" />
<xsd:element name="version" type="xsd:decimal" minOccurs="0" />
</xsd:sequence>
<xsd:attribute name="dateTimeTagFormat" type="xsd:string" fixed="xsd"
use="required" />
<xsd:attribute name="transactionType">

```

```

        <xsd:simpleType>
          <xsd:restriction base="xsd:token">
            <xsd:enumeration value="RWOV" />
            <xsd:enumeration value="FADD" />
            <xsd:enumeration value="FUPD" />
            <xsd:enumeration value="DEL" />
            <xsd:enumeration value="UPD" />
            <xsd:enumeration value="ADD" />
            <xsd:enumeration value="READ" />
            <xsd:enumeration value="REPL" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>

```

Device Event XML Format

This section describes the XML format used for inbound device events. This includes an example, descriptions of the individual XML elements, and the device event XML schema based on the D1-DeviceEventSeeder business object.

Example - Device Event

```

<D1-DeviceEventSeeder>
  <externalSenderId>L+G</externalSenderId>
  <deviceIdentifierNumber>GD_LL_SN100</deviceIdentifierNumber>
  <externalEventName>GD_LL_TAMPER_INDICATION</externalEventName>

  <rawEventInformation>D-Meter~GD_LL_SN100~GD_LL_ELECTRIC~1342395718~3~GD_HeadEnd_Power_Off~2010-09-09T13:11:41.0000000-05:00~Alert~Tamper indication on serial number GD_LL_SN100.~2010-09-09T13:11:41.0000000-05:00</rawEventInformation>
  <eventDateTime>2010-09-09-13.11.41</eventDateTime>
  <externalSourceIdentifier>EVENT_test.lg</externalSourceIdentifier>
  <eventInformation>
    <externalEventCategory>3</externalEventCategory>
    <externalEventSeverity>Alert</externalEventSeverity>
    <externalDeviceType>Meter</externalDeviceType>
    <externalServiceLocationId>GD_LL_ELECTRIC</externalServiceLocationId>
    <externalCommunicationModuleIdentifier>1342395718</externalCommunicationModuleIdentifier>
    <externalStatusValue>Tamper indication on serial number GD_LL_SN100.</externalStatusValue>
    <externalStatusDateTime>2010-09-09-13.11.41</externalStatusDateTime>
  </eventInformation>
</D1-DeviceEventSeeder>

```

Element Descriptions - Device Events

The table below provides descriptions of the elements used in the device event XML format.

Element	Description
<{SERVICE_NAME}>	Root element containing a device event. This element should match the name of the inbound service used to import device events.
<externalSenderId>	Id of the external system. Used to identify the head-end system from which the event is being sent.
<deviceIdentifierNumber>	The identifier number of the device that experienced the event.

Element	Description
<externalEventName>	The name of the event as defined by the external system. This will be mapped to a standard name via a device mapping business object.
<rawEventInformation>	A string containing information about the event from the external system.
<eventDateTime>	The date and time of the event.
<externalSourceIdentifier>	The name of the file containing the device event.
<eventInformation>	Element containing specific information about the event
<externalEventCategory>	The event category as defined by the external system.
<externalEventSeverity>	The event severity as defined by the external system.
<externalDeviceType>	The device type as defined by the external system. Can be one of the following: <ul style="list-style-type: none"> • Meter • Collector • Router
<externalServiceLocationId>	The service location for the event defined by the external system.
<externalCommunicationModuleIdentifier>	The identifier for the communication module associated with the device.
<externalStatusValue>	Optional information related to the event.
<externalStatusDateTime>	Date and time at which optional information (specified in the <externalStatusValue> element was recorded.

Schema - Device Event Seeder (D1-DeviceEventSeeder) Business Object

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ouaf="http://ouaf.oracle.com/" targetNamespace="http://oracle.com/D1-DeviceEventSeeder.xsd" elementFormDefault="qualified">
  <xsd:import namespace="http://ouaf.oracle.com/" />
  <xsd:element name="D1-DeviceEventSeeder">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="deviceEventId" type="xsd:string" minOccurs="0" />
        <xsd:element name="bo" type="xsd:string" minOccurs="0" />
        <xsd:element name="boStatus" type="xsd:string" minOccurs="0" />
        <xsd:element name="sender" type="xsd:string" minOccurs="0" />
        <xsd:element name="externalSenderId" type="xsd:string" minOccurs="0" />
        <xsd:element name="deviceEventType" type="xsd:string" minOccurs="0" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>

```

```

        <xsd:element name="externalEventName" type="xsd:string" minOccurs="0" /
    >
        <xsd:element name="eventDateTime" type="xsd:dateTime" />
        <xsd:element name="eventEndDateTime" type="xsd:dateTime" minOccurs="0"
    />
        <xsd:element name="deviceId" type="xsd:string" minOccurs="0" />
        <xsd:element name="creationDateTime" type="xsd:dateTime" minOccurs="0"
    />
        <xsd:element name="statusUpdateDateTime" type="xsd:dateTime"
minOccurs="0" />
        <xsd:element name="statusReason" type="xsd:string" minOccurs="0" />
        <xsd:element name="rawEventInformation" type="xsd:anyType"
minOccurs="0" maxOccurs="unbounded" />
        <xsd:element name="externalSourceIdentifier" type="xsd:string"
minOccurs="0" />
        <xsd:element name="eventInformation" minOccurs="0">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="externalEventIdentifier" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalEventCategory" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalEventSeverity" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalDeviceType" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalServiceLocationId" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalCommunicationModuleIdentifier"
type="xsd:string" minOccurs="0" />
                    <xsd:element name="externalGatewayIdentifier" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalStatusValue" type="xsd:string"
minOccurs="0" />
                    <xsd:element name="externalStatusDateTime" type="xsd:dateTime"
minOccurs="0" />
                    <xsd:element name="externalCommandId" type="xsd:string"
minOccurs="0" />
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="version" type="xsd:decimal" minOccurs="0" />
        <xsd:element name="deviceIdentifierNumber" type="xsd:string"
minOccurs="0" />
        <xsd:element name="newDeviceEvent" type="xsd:string" minOccurs="0" />
        <xsd:element name="processData" minOccurs="0">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="errorEncountered" minOccurs="0">
                        <xsd:simpleType>
                            <xsd:restriction base="xsd:string">
                                <xsd:enumeration value="no" />
                                <xsd:enumeration value="yes" />
                            </xsd:restriction>
                        </xsd:simpleType>
                    </xsd:element>
                    <xsd:element name="dateTimesInStandard" minOccurs="0">
                        <xsd:simpleType>
                            <xsd:restriction base="xsd:string">
                                <xsd:enumeration value="no" />
                                <xsd:enumeration value="yes" />
                            </xsd:restriction>
                        </xsd:simpleType>
                    </xsd:element>
                    <xsd:element name="logs" minOccurs="0">
                        <xsd:complexType>
                            <xsd:sequence>

```



```

        <xsd:element name="logsList" minOccurs="0"
maxOccurs="unbounded">
        <xsd:complexType>

            <xsd:sequence>
                <xsd:element name="logsEntry" minOccurs="0">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="sequence" type="xsd:decimal"
minOccurs="0" />
                            <xsd:element name="mo" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="pkValue1" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="pkValue2" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="pkValue3" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="pkValue4" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="pkValue5" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="logEntryType" minOccurs="0">
                                <xsd:simpleType>
                                    <xsd:restriction base="xsd:string">
                                        <xsd:enumeration value="toDos" />
                                        <xsd:enumeration value="created" />
                                        <xsd:enumeration
value="statusTransitionError" />
                                        <xsd:enumeration value="exception" />
                                        <xsd:enumeration value="statusTransition" /
>
                                        <xsd:enumeration value="system" />
                                        <xsd:enumeration value="todo" />
                                        <xsd:enumeration value="userDetails" />
                                    </xsd:restriction>
                                </xsd:simpleType>
                            </xsd:element>
                            <xsd:element name="logDateTime"
type="xsd:dateTime" minOccurs="0" />
                            <xsd:element name="boStatus" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="description"
type="xsd:string" minOccurs="0" />
                            <xsd:element name="user" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="logMessage" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="characteristicType"
type="xsd:string" minOccurs="0" />
                            <xsd:element name="characteristicValue"
type="xsd:string" minOccurs="0" />
                            <xsd:element name="adhocValue" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="fkValue1" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="fkValue2" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="fkValue3" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="fkValue4" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="fkValue5" type="xsd:string"
minOccurs="0" />
                            <xsd:element name="messageCategory"
type="xsd:decimal" minOccurs="0" />
                            <xsd:element name="messageNumber"
type="xsd:decimal" minOccurs="0" />

```

```

                                <xsd:element name="messageParm1"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm2"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm3"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm4"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm5"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm6"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm7"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm8"
type="xsd:string" minOccurs="0" />
                                <xsd:element name="messageParm9"
type="xsd:string" minOccurs="0" />
                                </xsd:sequence>
                                </xsd:complexType>
                                </xsd:element>
                                </xsd:sequence>
                                </xsd:complexType>
                                </xsd:element>
                                </xsd:sequence>
                                </xsd:complexType>
                                </xsd:element>
                                </xsd:sequence>
                                </xsd:complexType>
                                </xsd:element>
                                </xsd:sequence>
                                <xsd:attribute name="dateTimeTagFormat" type="xsd:string" fixed="xsd"
use="required" />
                                <xsd:attribute name="transactionType">
                                <xsd:simpleType>
                                <xsd:restriction base="xsd:token">
                                <xsd:enumeration value="RWOV" />
                                <xsd:enumeration value="FADD" />
                                <xsd:enumeration value="FUPD" />
                                <xsd:enumeration value="DEL" />
                                <xsd:enumeration value="UPD" />
                                <xsd:enumeration value="ADD" />
                                <xsd:enumeration value="READ" />
                                <xsd:enumeration value="REPL" />
                                </xsd:restriction>
                                </xsd:simpleType>
                                </xsd:attribute>
                                </xsd:complexType>
                                </xsd:element>
                                </xsd:schema>
```

Chapter 11

Sample Implementation

This chapter describes the steps involved in configuring Oracle Utilities Smart Grid Gateway in a simple example implementation, including the following:

- **Implementation Description and Requirements**
- **Implementation Steps**

Note: The implementation described in this chapter is intended for example purposes only, and is intentionally simple, and as such does not involve configuration of every type of object described in this book. Also, this example assumes that the base package admin business objects (device type, measuring component type, etc.) meet the requirements of the implementation.

Note: References to objects and options pre-fixed with “D2” are available only with Oracle Utilities Meter Data Management, and included for illustrative purposes only.

Implementation Description and Requirements

The sample implementation described in this chapter will be for a small electric utility providing service to a small town that includes residential, commercial, and industrial customers. The details and requirements of this implementation are summarized as follows

Requirement	Description
Types of Customers	<ul style="list-style-type: none"> Residential (approximately 50,000) Commercial (approximately 1,000) Industrial (approximately 100)
Types of Meters	<ul style="list-style-type: none"> Residential: Scalar, single register Commercial: Interval channel/scalar register Industrial: Two interval channels
Meter Manufacturers / Head-End Systems	<ul style="list-style-type: none"> MetersRUs: Scalar (used for residential customers) MeterTech, Inc.: Interval (used for commercial and industrial customer)
Usage Metered	<ul style="list-style-type: none"> Residential: KWH (scalar) Commercial: KWH (interval and scalar) Industrial: KWH and KVARH (interval)
Readings/Measurement Data	<ul style="list-style-type: none"> Scalar (for residential and commercial) Interval (for commercial and industrial, 15 minutes) <p>Both head-end systems use their own UOM codes.</p>
Device Events	Both head-end systems listed above send standard and “paired” device events, and use their own device event names.
Meter Commands	<p>Both head-end systems support the following commands:</p> <ul style="list-style-type: none"> Commission Device Decommission Device Device Status Check On-Demand Read (interval and scalar) Remote Connect Remote Disconnect <p>Each command sends and received a single message when invoking the command.</p>
Validation, Editing, and Estimation Rules	Validation, editing, and estimation is performed by Oracle Utilities Meter Data Management.*
Bill Determinants	Bill determinant calculations are performed by Oracle Utilities Meter Data Management.*
Billing System	Oracle Utilities Customer Care and Billing

*See **Chapter 16: Sample Implementation** in the Oracle Utilities Meter Data Management Configuration Guide for examples of implementing VEE and bill determinant calculations.

Implementation Steps

This section outlines the steps in configuring Oracle Utilities Smart Grid Gateway to meet the requirements outlined above. These steps include:

1. **Design and Create Business Objects**

In this first step, we'll outline the specific business objects and other configuration data required to address the sample requirements.

2. **Create Admin Data**

In this step, we'll outline the admin data that would need to be created to address the sample requirements.

3. **Create Master Data**

In this step, we'll outline the master data (individual devices, service points, etc.) that would need to be created to address the sample requirements.

Design and Create Business Objects

The first step in implementing and configuring the system is to identify the business objects (and related configuration objects) needed to meet the requirements of the implementation. This section outlines the business objects and other significant configuration objects that could be created to meet the requirements of our sample implementation. This does NOT include listings of all configuration objects needed (such as individual display and maintenance UI maps, portal navigation options, etc.).

Service Points and Device Installation

For service point and device installation data, we would need to create the following:

Service Points: Service point business objects for each type of customer, as follows:

- Residential: CM-ResidentialSP
- Commercial: CM-CommercialSP
- Industrial: CM-IndustrialSP

Contacts: Contact business objects for each type of customer, as follows:

- Residential: CM-ResPerson
- Commercial: CM-ComBusiness
- Industrial: CM-IndBusiness

Install Events: Install event business objects for each type of customer, as follows:

- Residential: CM-ResidentialMeterInstallEvent
- Commercial: CM-CommercialMeterInstallEvent
- Industrial: CM-IndustrialMeterInstallEvent

Service Providers: Service provider business objects for each head-end system, and for the billing system, as follows:

- Head-End System: CM-HeadEndMRU (Meters R Us)
- Head-End System: CM-HeadEndMT (Meter Tech, Inc.)
- Billing System: CM-ExternalAppCCB

Devices and Measuring Components

For devices and measuring components, we would need to create the following:

Devices: Device business objects for each type of meter, as follows:

- Residential: CM-ScalarRegister
 - Install Event BO: CM-ResidentialInstallEvent (see above)
 - Valid Command Request BO: D1-DeviceCommission, etc.* (see below)
- Commercial: CM-IntChanScalarReg
 - Install Event BO: CM-CommercialInstallEvent (see above)
 - Valid Command Request BO: D1-DeviceCommission, etc.* (see below)
- Industrial: CM-Interval2Channels
 - Install Event BO: CM-IndustrialInstallEvent (see above)
 - Valid Command Request BO: D1-DeviceCommission, etc.* (see below)

***Note:** Each device business object should specify the specific commands available for devices based on that business object

Measuring Components: Measuring component business objects for scalar registers and/or interval channels, as follows:

- Residential - Scalar Register: CM-ResScalarRegister
- Commercial/Industrial - Interval Channel: CM-IntervalChannel (used for both commercial and industrial meters)
- Commercial - Scalar Register: CM-ScalarValRegister

Measurement Data

For measurement data, we would need to create the following:

Initial Measurement Data: Initial load, estimation, and manual initial measurement business objects for each reading/measurement type, as follows:

- Initial Load - Interval: CM-InitialLoadIMDInterval
- Initial Load - Scalar: CM-InitialLoadIMDScalar
- Estimation - Interval: CM-EstimationIMDInterval
- Estimation - Scalar: CM-EstimationIMDScalar
- Manual - Interval: CM-ManualIMDInterval
- Manual - Scalar: CM-ManualIMDScalar

Measurements: A single measurement business object for all final measurements, as follows:

- Final Measurement: CM-FinalMeasurement

UOM Mapping: UOM mapping extendable lookup business objects for each head-end system, as follows:

- MetersRUs: CM-MRUUOMMapping
- MeterTech, Inc.: CM-MTUOMMapping

Device Events

For device events, we would need to create the following:

Device Events: Device event business objects for “standard” and “paired” device events, as follows:

- Standard Device Event: CM-StandardDeviceEvent
- Paired Device Event - First: CM-PairedDeviceEventFirst
- Paired Device Event - Last: CM-PairedDeviceEventLast

Device Event Mapping: Device event mapping extendable lookup business objects for each head-end system, as follows:

- MetersRUs: CM-MRUDeviceEventMapping
- MeterTech, Inc.: CM-MTDeviceEventMapping

Meter Commands

For meter commands data, we would need to create the following:

Activities: For activities, this implementation can use the base package business objects, including the activity command business objects.

Outbound Communication: Outbound communication business objects for each command, as follows:

- Commission Device: CM-InitiateCommissionDevice
- Decommission Device: CM-InitiateDecommissionDevice
- Device Status Check: CM-InitiateDeviceStatusCheck
- On-Demand Read Interval: CM-InitiateOnDemandReadInterval
- On-Demand Read Scalar: CM-InitiateOnDemandReadScalar
- Remote Connect: CM-InitiateRemoteConnect
- Remote Disconnect: CM-InitiateRemoteDisconnect

Inbound Communication: Inbound communication business objects for each command response, as follows:

- Commission Device: CM-CommissionDeviceResponse
- Decommission Device: CM-DecommissionDeviceResponse
- Device Status Check: Cm-DeviceStatusCheckResponse
- On-Demand Read Interval: CM-OnDemandReadIntervalResponse
- On-Demand Read Scalar: CM-OnDemandReadScalarResponse
- Remote Connect: CM-RemoteConnectResponse
- Remote Disconnect: CM-RemoteDisconnectResponse

Completion Events: This implementation can use the base package completion event business objects.

Note: Because this implementation supports two head-end systems, you would need business objects for each communication for each head-end system. This list is for only one head-end system.

Create Admin Data

With all of the custom business objects needed for the implementation in place, the next step would be to create admin data. This section outlines the admin data that would need to be created to meet the requirements of our sample implementation. In general these listings list only the name (or code) and description of each record to be created, and do not include details for every attribute of each record created. Where listing additional attributes is important to understanding how the data would be created, it is noted, and additional details are provided in a separate section.

The table below summarizes the common admin data needed for our implementation.

Admin Data Type	Data to Create
Activity Type	N/A (will use base package activity types)
Communication Type	<p>One for each type of communication (outbound or inbound) as follows:</p> <ul style="list-style-type: none"> Commission Device / Response <p>Business Object (Initiate): CM-InitiateCommissionDeviceType</p> <p>Business Object (Response): CM-CommissionDeviceResponseType</p> Decommission Device / Response <p>Business Object (Initiate): CM-InitiateDecommissionDeviceType</p> <p>Business Object (Response): CM-DecommissionDeviceResponseType</p> Device Status Check / Response <p>Business Object (Initiate): CM-InitiateDeviceStatusCheckType</p> <p>Business Object (Response): CM-DeviceStatusCheckResponseType</p> On-Demand Read Interval / Response <p>Business Object (Initiate): CM-InitiateOnDemandReadIntervalType</p> <p>Business Object (Response): CM-OnDemandReadIntervalResponseType</p> On-Demand Read Scalar / Response <p>Business Object (Initiate): CM-InitiateOnDemandReadScalarType</p> <p>Business Object (Response): CM-OnDemandReadScalarResponseType</p> Remote Connect / Response <p>Business Object (Initiate): CM-InitiateRemoteConnectType</p> <p>Business Object (Response): CM-RemoteConnectResponseType</p> Remote Disconnect / Response <p>Business Object (Initiate): CM-InitiateRemoteDisconnectType</p> <p>Business Object (Response): CM-RemoteDisconnectResponseType</p> <p>Note: The communication business objects described under “Create Business Objects” serve as the Related Transaction BO for each of the above.</p>

Contact Type	<p>One for each type of customer, as follows:</p> <ul style="list-style-type: none"> RESIDENTIAL (Residential - Person) Business Object: CM-ResPerson COMMERCIAL (Commercial - Business) Business Object: CM-ComBusiness INDUSTRIAL (Industrial - Business) Business Object: CM-IndBusiness
Device Event Type	<p>One for each type of device event sent from the head-end systems. Examples might include:</p> <ul style="list-style-type: none"> BROKEN SEAL Business Object: CM-StandardDeviceEvent OUTAGE Business Object: CM-PairedDeviceEventFirst POWER RESTORATION Business Object: CM-PairedDeviceEventLast TAMPER Business Object: CM-StandardDeviceEvent
Exception Type	One for each VEE rule, as appropriate.
Factor	N/A
Market	<p>Single market:</p> <ul style="list-style-type: none"> SMALLTOWNUSA
Measurement Cycle	<p>Twenty cycles for each type of customer, as follows:</p> <ul style="list-style-type: none"> RESMC01, RESMC02, ..., RESMC20 COMMC01, COMMC02, ..., COMMC20 INDMC01, INDMC02, ..., INDMC20
Measurement Cycle Schedule	<p>One per measurement cycle per month, as follows:</p> <ul style="list-style-type: none"> RESMC01: <ul style="list-style-type: none"> Scheduled Selection Date: 08/02/2010 Expected Work Date: 08/03/2010 RESMC02: <ul style="list-style-type: none"> Scheduled Selection Date: 08/03/2010 Expected Work Date: 08/04/2010 RESMC03: <ul style="list-style-type: none"> ...

Service Provider	<p>One for each head-end system, and one for the billing system, as follows:</p> <ul style="list-style-type: none"> METERSRUS (Meters R Us) Business Object: CM-HeadEndMRU METERTECH (MeterTech, Inc.) Business Object: CM-HeadEndMT OUCCB (Oracle Utilities Customer Care and Billing) Business Object: CM-ExternalAppCCB <p>* See Service Providers on page 11-11 for additional details.</p>
Service Quantity Identifier	N/A
Service Type	<p>Single service type:</p> <ul style="list-style-type: none"> ELECTRIC (Electric)
Manufacturer	<p>One for each manufacturer, as follows:</p> <ul style="list-style-type: none"> METERSRUS (Meters R Us) <ul style="list-style-type: none"> Models: RES2010 METERTECH (MeterTech, Inc.) <ul style="list-style-type: none"> Models: COM2010, IND2010
Unit of Measure	<p>One for each type of metered usage and for calculated usage, as follows:</p> <ul style="list-style-type: none"> KVA (calculated) KVAR (used in KVA calculation) KVARH (measured) KW (derived) KWH (measured) <p>* All UOMs use the ELECTRIC service type.</p>
Measuring Component Type	<p>Five measuring component types, as follows:</p> <ul style="list-style-type: none"> RESSCALAR (Residential Scalar Register) Business Object: CM-ResScalarRegister COMINTERVAL (Commercial Interval Channel) Business Object: CM-IntervalChannel COMSCALAR (Commercial Scalar Register) Business Object: CM-ScalarValRegister INDINTERVALKVARH (Industrial KVARH Interval Channel) Business Object: CM-IntervalChannel INDINTERVALKWH (Industrial KWH Interval Channel) Business Object: CM-IntervalChannel <p>* All measuring component types use the ELECTRIC service type. See Measuring Component Types on page 11-13 for additional details.</p>

Device Configuration Type	<p>One device configuration type for each type of meter installed, as follows:</p> <ul style="list-style-type: none"> • RESDVCCFG (Residential Device Configuration) <p>Valid Measuring Component Types:</p> <ul style="list-style-type: none"> • RESSCALAR (Residential Scalar Register) <ul style="list-style-type: none"> • COMDVCCFG (Commercial Device Configuration) <p>Valid Measuring Component Types:</p> <ul style="list-style-type: none"> • COMSCALAR (Commercial Scalar Register) • COMINTERVAL (Commercial Interval Channel) <ul style="list-style-type: none"> • INDDVCCFG (Industrial Device Configuration) <p>Valid Measuring Component Types:</p> <ul style="list-style-type: none"> • INDINTERVALKVARH (Industrial KVARH Interval Channel) • INDINTERVALKWH (Industrial KWH Interval Channel) <p>*All device configuration types use the ELECTRIC service type.</p>
Device Type	<p>One device type for each type of meter installed, as follows:</p> <ul style="list-style-type: none"> • RESDVC (Residential Device) <p>Business Object: CM-ScalarRegister</p> <p>Valid Device Configuration Types:</p> <ul style="list-style-type: none"> • RESDVCCFG (Residential Device Configurations) <ul style="list-style-type: none"> • COMDVC (Commercial Device) <p>Business Object: CM-IntChanScalarReg</p> <p>Valid Device Configuration Types:</p> <ul style="list-style-type: none"> • COMDVCCFG (Commercial Device Configurations) <ul style="list-style-type: none"> • INDDVC (Industrial Device) <p>Business Object: CM-Interval2Channels</p> <p>Valid Device Configuration Types:</p> <ul style="list-style-type: none"> • INDDVCCFG (Industrial Device Configurations) <p>*All device types use the ELECTRIC service type.</p>

Service Point Type	<p>One for each type of customer rule, as follows:</p> <ul style="list-style-type: none">• RESIDENTIAL (Residential) Business Object: CM-ResidentialSP Valid Device Types:<ul style="list-style-type: none">• RESDVC (Residential Devices)• COMMERCIAL (Commercial) Business Object: CM-CommercialSP Valid Device Types:<ul style="list-style-type: none">• COMDVC (Commercial Devices)• INDUSTRIAL (Industrial) Business Object: CM-IndustrialSP Valid Device Types:<ul style="list-style-type: none">• INDDVC (Industrial Devices) <p>* All service point types use ELECTRIC service type</p>
--------------------	---

Additional Details

This section provides additional details related to the admin data described above. Not all attributes are listed for all types of data.

Service Providers

This section provides additional details for each of the service providers listed above.

Service Provider: METERSRUS

- Business Object: CM-HeadEndMRU
- Description: Meters R Us
- External Reference ID: HE-MRU
- Our Name/ID in Their System: HE-MRU-11
- Processing Methods List:

Processing Role	Processing Method	Default Processing Method	Override Processing Methods
Initial Measurement Creation	How To Create MC Related Information	CM-InitialLoadIMDScalar (Business Object)	MC Type: Residential Scalar Register (RESSCALAR) Business Object: CM- InitialLoadIMDScalar
Device Commission	How to Create OB Communication / Send OB Message	CM-InitiateCommissionDevice (Business Object)	
Device Decommission	How to Create OB Communication / Send OB Message	CM-InitiateDecommissionDevice (Business Object)	
Device Event Mapping	How to Process Device Related Information	CM-MRSDDeviceEventMapping (Business Object)	
Device Status Check	How to Create OB Communication / Send OB Message	CM-InitiateDeviceStatusCheck (Business Object)	
On-Demand Read (Interval)	How to Create OB Communication / Send OB Message	CM-InitiateOnDemandReadInterval (Business Object)	
On-Demand Read (Scalar)	How to Create OB Communication / Send OB Message	CM-InitiateOnDemandReadScalar (Business Object)	
Remote Connect	How to Create OB Communication / Send OB Message	CM-InitiateRemoteConnect (Business Object)	
Remote Disconnect	How to Create OB Communication / Send OB Message	CM-InitiateRemoteDisconnect (Business Object)	
UOM Mapping	How to Process Device Related Information	CM-MRUUOMMapping (Business Object)	

Service Provider: METERTECH

- Business Object: CM-HeadEndMT
- Description: MeterTech, Inc.
- External Reference ID: HE-MTECH

- Our Name/ID in Their System: HE-MTECH-11
- Processing Methods List:

Processing Role	Processing Method	Default Processing Method	Override Processing Methods
Initial Measurement Creation	How To Create MC Related Information	CM-InitialLoadIMDInterval (Business Object)	MC Type: Residential Scalar Register (RESSCALAR) / Business Object: CM-InitialLoadIMDScalar MC Type: Commercial Scalar Register (COMSCALAR) / Business Object: CM-InitialLoadIMDScalar
Device Commission	How to Create OB Communication / Send OB Message	CM-InitiateCommissionDevice (Business Object)	
Device Decommission	How to Create OB Communication / Send OB Message	CM-InitiateDecommissionDevice (Business Object)	
Device Event Mapping	How to Process Device Related Information	CM-MTDeviceEventMapping (Business Object)	
Device Status Check	How to Create OB Communication / Send OB Message	CM-InitiateDeviceStatusCheck (Business Object)	
On-Demand Read (Interval)	How to Create OB Communication / Send OB Message	CM-InitiateOnDemandReadInterval (Business Object)	
On-Demand Read (Scalar)	How to Create OB Communication / Send OB Message	CM-InitiateOnDemandReadScalar (Business Object)	
Remote Connect	How to Create OB Communication / Send OB Message	CM-InitiateRemoteConnect (Business Object)	
Remote Disconnect	How to Create OB Communication / Send OB Message	CM-InitiateRemoteDisconnect (Business Object)	
UOM Mapping	How to Process Device Related Information	CM-MTUOMMapping (Business Object)	

Service Provider: OUCCB

- Business Object: CM-ExternalAppCCB
- Description: Oracle Utilities Customer Care and Billing
- External Reference ID: EXT-CCB
- Our Name/ID in Their System: EXT-CCB-11
- Processing Methods List:

Measuring Component Types

This section provides additional details for each of the measuring component types listed above.

Measuring Component Type: RESSCALAR

- Description: Residential Scalar Register
- Measuring Component Business Object: CM-ResScalarRegister
- Measurement Business Object: CM-FinalMeasurement
- Service Type: Electric
- Value Identifiers:

Value Identifier Type	Short-Hand Description	UOM
Measurement	KWH	KWH

- Valid VEE Groups:
 - Initial Load VEE Group - Scalar (SCALAR_MCS)
- Fallback VEE Groups:
 - Initial Load: Initial Load VEE Group - Scalar (SCALAR_MCS)

Measuring Component Type: COMINTERVAL

- Description: Commercial Interval Channel
- Measuring Component Business Object: CM-IntervalChannel
- Measurement Business Object: CM-FinalMeasurement
- Service Type: Electric
- Value Identifiers:

Value Identifier Type	Short-Hand Description	UOM
Measurement	KWH	KWH

- Valid VEE Groups:
 - Initial Load VEE Group - Commercial Interval (COM_INTD_MCS)
- Fallback VEE Groups:
 - Initial Load: Initial Load VEE Group - Commercial Interval (COM_INTD_MCS)

Measuring Component Type: COMSCALAR

- Description: Commercial Scalar Register
- Measuring Component Business Object: CM-ScalarValRegister
- Measurement Business Object: CM-FinalMeasurement
- Service Type: Electric
- Value Identifiers:

Value Identifier Type	Short-Hand Description	UOM
Measurement	KWH	KWH

- Valid VEE Groups:
 - Initial Load VEE Group - Scalar (SCALAR_MCS)
- Fallback VEE Groups:
 - Initial Load: Initial Load VEE Group - Scalar (SCALAR_MCS)

Measuring Component Type: INDINTERVALKVARH

- Description: Industrial KVARH Interval Channel
- Business Object: CM-IntervalChannel
- Measurement Business Object: CM-FinalMeasurement
- Service Type: Electric
- Value Identifiers:

Value Identifier Type	Short-Hand Description	UOM
Measurement	KVARH	KVARH

- Valid VEE Groups:
 - Initial Load VEE Group - Industrial Interval (IND_INTD_MCS)
- Fallback VEE Groups:
 - Initial Load: Initial Load VEE Group - Industrial Interval (IND_INTD_MCS)

Measuring Component Type: INDINTERVALKWH (Industrial KWH Interval Channel)

- Description: Industrial KWH Interval Channel
- Business Object: CM-IntervalChannel
- Measurement Business Object: CM-FinalMeasurement
- Service Type: Electric
- Value Identifiers:

Value Identifier Type	Short-Hand Description	UOM
Measurement	KWH	KWH

- Valid VEE Groups:
 - Initial Load VEE Group - Industrial Interval (IND_INTD_MCS)
- Fallback VEE Groups:
 - Initial Load: Initial Load VEE Group - Industrial Interval (IND_INTD_MCS)

Create Master Data

In this last step, we would create the actual master data (individual devices, service points, etc.) for the implementation. For purposes of this section, only a single example of each type of data is presented.

Contacts

A typical residential contact might look like this:

Residential - Person:

- Information: John Smith / 555-555-5555
- Contact Type: Residential - Person (RESIDENTIAL)
- Name: John Smith
- Home Phone: 555-555-5555

Service Points

A typical commercial service point might look like this:

Commercial Service Point:

- Information: 35 York Street, Burlington, MA, 01803, US / Commercial / Active
- Service Point Type: Commercial (COMMERCIAL)
- Status: Active
- Time Zone: US - Eastern Time
- Market: Small Town USA
- Main Contact: Phillip Jones
- Address:
 - Country: United States
 - Postal Code: 01803
 - Street Address: 35 York Street
 - City: Burlington
 - State: MA
- Measurement Cycle:
 - Measurement Cycle: Commercial Cycle 01 (COMMC01)
 - Route: Route 2
 - Sequence: 10

Devices

A typical industrial device might look like this:

Industrial Device:

- Information: 123456 / Industrial Device / Install Date/Time: 08-01-2010 12:00 AM / Pending / MeterTech, Inc. / Active
- Device Type: Industrial Device (INDDVC)
- Serial Number: 123456
- Internal Meter Number: 654321

- Pallet Number: 123456
- Manufacturer: MeterTech, Inc.
- Model: IND2010
- Incoming Data Shift: Shifted
- Arming Required: Arming Required
- Head-End System: MeterTech, Inc. (METERTECH)
- Status: Active

Device Configurations

A typical industrial device configuration might look like this:

Industrial Device Configuration:

- Information: Industrial Device / Effective Date/Time: 08-01-2010 12:00 AM / Industrial Device Configuration / 2 Measuring Component(s) / Active
- Device Configuration Type: Industrial Device Configuration (INDDVCCFG)
- Device: 123456 / Industrial Device / Install Date/Time: 08-01-2010 12:00 AM / Pending / MeterTech, Inc. / Active
- Effective Date/Time: 08-01-2010 12:00 AM
- Time Zone: US - Eastern Time
- Status: Active

Measuring Components

Typical industrial measuring components might look like this:

KVARH Interval Channel:

- Information: 123456 / 2 / Industrial KVARH Interval Channel
- Measuring Component Type: Industrial KVARH Interval Channel (INDINTERVALKVARH)
- Device Configuration: Industrial Device / Effective Date/Time: 08-01-2010 12:00 AM / Industrial Device Configuration / 2 Measuring Component(s) / Active
- Consumption Reference Measuring Component: N/A
- How to Use: Consumptive
- Number of Digits Left: 5
- Number of Digits Right: 2
- Channel Multiplier: 1.000
- Latest Read Date/Time: N/A
- Channel ID: 2

KWH Interval Channel:

- Information: 123456 / 1 / Industrial KWH Interval Channel
- Measuring Component Type: Industrial KWH Interval Channel (INDINTERVALKWH)
- Device Configuration: Industrial Device / Effective Date/Time: 08-01-2010 12:00 AM / Industrial Device Configuration / 2 Measuring Component(s) / Active
- Consumption Reference Measuring Component: N/A

- How to Use: Consumptive
- Number of Digits Left: 5
- Number of Digits Right: 2
- Channel Multiplier: 1.000
- Latest Read Date/Time: N/A
- Channel ID: 1

Install Events

A typical industrial installation event might look like this:

Industrial Install Event:

- Information: Install Date/Time: 08-01-2010 / On
- Device Configuration: Industrial Device / Effective Date/Time: 08-01-2010 12:00 AM / Industrial Device Configuration / 2 Measuring Component(s) / Active
- Service Point: 47 North Street, Burlington, MA, 01803, US / Industrial / Active
- Status: On
- Installation Date/Time: 08-01-2010
- Installation Constant: 1.00000
- Device On/Off Status: On
- On/Off History: N/A

Other Components

Note that for each communication you create, you must also create corresponding components of the following types used to send/received messages from the middleware. These include:

Outbound Communications

- Outbound Message Type
- XAI Sender (XAI)
- External System

Inbound Communications

- XAI Inbound Service

See **Chapter 10: Integrating Oracle Utilities Smart Grid Gateway with Other Systems** for more information about creating these components when configuring communications with head-end systems and middleware.

Appendix A

Measurement Services

This appendix provides brief descriptions of the base package measurement services used by VEE rules and measurement functions. The measurement services described in this appendix are implemented as business services. These business services can be used by custom algorithms or BPA/Service scripts created for your implementation.

The table below lists the available back package measurement services, including the business service that implements each service and a brief description for each.

Measurement Service	Business Service	Description
Add Scalar Value To Intervals	D1-AddScalarValueToIntervals	Uses the Apply Formula measurement service to add a scalar value to the value of a specified set of interval data.
Adjust Intervals to Supplied Value	D1-AdjustIntervalsToSuppldVal	Uses the Apply Formula measurement service to adjust the total value of a specified set of interval data to a scalar value.
Apply Formula	D1-ApplyFormula	Used to apply a formula to a specified set of interval data, either by applying a summary function against all intervals of the set, or by manipulating each individual interval in series via a formula using declared constants, or within the context of other sets of input interval data.
Apply TOU Map To Interval Measuring Component	D1-ApplyTOUMapToIntervalMC	Used to apply a TOU map to a set of intervals for a specified measuring component and date/time range, thereby isolating and summarizing those intervals that occurred during a specific time of use.
Axis Conversion	D1-AxisConversion	Used to convert interval data between units of measure (UOMs) and interval sizes (SPIs), including the conversion between peak and consumption-oriented UOMs.
Convert Scalar Consumption To Interval	D1-IntervalizeScalarConsumptn	Used to convert a scalar consumption value to a set of interval measurements.

Measurement Service	Business Service	Description
Create Intervals	D1-CreateIntervals	Used to create interval data based on supplied parameters (UOM, SPI, number of intervals, value, etc.)
Divide Intervals By Scalar Value	D1-DivideIntervalsByScalarVal	Uses the Apply Formula measurement service to divide the values of a specified set of interval data by a scalar value.
Extract Subset of Intervals	D1-ExtractSubsetOfIntervals	Used to extract a subset of interval data from a specified set of intervals.
Identify Spikes	D1-IdentifySpikes	Used to identify spikes in a specified set of interval data based on a spike percentage tolerance.
Insert Intervals	D1-InsertIntervals	Used to insert one or more intervals into a set of interval measurements.
Merge Intervals	D1-MergeIntervals	Used to merge a subset of interval data with a specified set of intervals (where overlaps occur, the subset intervals replace the original intervals).
Multiply Intervals By Scalar Value	D1-MultiplyIntervalsByScalarVal	Uses the Apply Formula measurement service to multiply the values of a specified set of interval data by a scalar value.
Remove Intervals	D1-RemoveIntervals	Used to remove one or more intervals from a set of interval measurements.
Retrieve Interval Consumption	D1-IntvConsumptionRetriever	Used to retrieve one or more interval measurements.
Retrieve Scalar Consumption	D1-ScalarConsumptionRetriever	Used to retrieve one or more scalar measurements.
Set Condition	D1-SetCondition	Used to set the condition (status) code of a specified set of interval data.
Shift Intervals	D1-ShiftIntervals	Used to shift one or more intervals forward or backward in time.
Subtract Scalar Value From Intervals	D1-SubtractScalarValToIntervls	Uses the Apply Formula measurement service to subtract a scalar value from the value of a specified set of interval data.

Use the Business Service portal to view more details concerning these measurement services.

Appendix B

Glossary

This glossary provides definitions of commonly used terms.

Activity Type

Defines properties common to a specific type of activity.

Add Scalar Value To Intervals

Measurement service that uses the Apply Formula measurement service to add a scalar value to the value of a specified set of interval data.

Adjust Intervals to Supplied Value

Measurement service that uses the Apply Formula measurement service to adjust the total value of a specified set of interval data to a scalar value.

Advanced Metering Infrastructure (AMI)

Refers to systems that measure, collect and analyze energy usage, and interact with advanced devices such as electricity meters, gas meters, heat meters, and water meters, through various communication media either on request (on-demand) or on pre-defined schedules.

Apply Formula

Measurement service used to apply a formula to a specified set of interval data, either by applying a summary function against all intervals of the set, or by manipulating each individual interval in series via a formula using declared constants, or within the context of other sets of input interval data.

Apply TOU Map To Interval Measuring Component

Measurement service used to apply a TOU map to a set of intervals for a specified date/time range, thereby isolating and summarizing those intervals that occurred during a specific time of use.

Automatic Meter Reading (AMR)

The technology of automatically collecting consumption, diagnostic, and status data from water meter or energy metering devices (water, gas, electric) and transferring that data to a central database for billing, troubleshooting, and analyzing.

Axis Conversion

Measurement service used to convert interval data between units of measure (UOMs) and interval sizes (SPIs), including the conversion between peak and consumption-oriented UOMs.

Bill Determinants

Measurement data summarized for use by a billing application. Bill determinants can take the form of TOU-mapped interval consumption, scalar consumption, scalar readings, and/or interval consumption obtained via measurements. A common variety of bill determinant is TOU-mapped interval consumption, which reduces a full month's worth of interval data into several buckets of

consumption based on time of use.

Command

A communication sent to a device to perform some action on the device, such as Connect, Disconnect, Commission, Decommission, On-Demand Read, or Device Status Check (Ping)

Communication

A record of a message sent between Oracle Utilities Smart Grid Gateway and an external system, such as a head-end system or edge application. Communications can flow both inbound and outbound, and can be both one-way and two-way.

Completion Event

Records used to create or update transactions that reflect the effect of an activity. For example, issuing a commission device command could result in the creation or update of an install event while a read device command could result in the creation of initial measurement data.

Consumption

A measurement by a given device of the amount of energy, water, gas, etc. consumed over a given time period. Synonymous with the term "measurement".

Consumptive

Describes a measuring component for which readings are equivalent to the consumption. For example, if we receive a reading of 400 on January 15 and a reading of 600 on February 15, a consumptive measuring component's consumption between January 15 and February 15 would be 600 (not 200).

Contact

An individual or a business entity with which a company has contact. Each contact must reference a contact type.

Contact - Email

Email addresses related to a contact

Contact - Identifier

Identifiers related to a contact, such as social security number, driver's license number, or the contact's ID in a prior system.

Contact - Name

Names related to a contact

Contact - Phone

Phone numbers related to a contact

Contact Type

Defines the properties of a class of entities (businesses, persons).

Convert Scalar Consumption To Interval

Measurement service used to convert a scalar consumption value to a set of interval measurements.

Create Intervals

Measurement service used to create interval data based on supplied parameters (UOM, SPI, number of intervals, value, etc.)

Demand

The rate at which a commodity is delivered at a given instant or averaged over a designated time. For electricity, demand is often expressed in kilowatts (kW) or kilovolt-amperes (kVa).

Device

A physical or virtual object that holds one or more measuring components that can produce data to be handled by the system. Devices can include meters, substations, transformers, demand response devices, weather stations, etc.

Device Configuration

A specific configuration of a device. Over time, a device can have many configurations. Use of effective-dated device configuration allows the device to retain its identifier(s) even while the quantities it is measuring are changing.

Device Configuration Type

Defines the properties of device configurations of this type, including the valid types of measuring components that can be configured for the device.

Device Event

An event of some sort that has taken place relative to a device. Device events can include power outages, power restorations, tampering alerts, command completion, and other information

Device Status Check

A communication sent to a device to test whether the device is communicating with the network, determine the connection status of the meter, and when possible if there are any known malfunctions

Device Type

Information about a class of devices, including properties that apply to all devices of a type, but can be overridden for an individual device.

Distribution Company (DISCO)

A utility company that constructs and maintains the distribution network that delivers a commodity to customers. Depending upon the regulations within the territory, a distribution company may or may not be responsible for billing the customer.

Divide Intervals By Scalar Value

Measurement service that uses the Apply Formula measurement service to divide the values of a specified set of interval data by a scalar value.

Exception Type

Defines properties common to many exceptions, including the category of the exception.

Extract Subset of Intervals

Measurement service used to extract a subset of interval data from a specified set of intervals.

Factor

A centrally stored set of values for use in validation rules, bill determinants calculations, and other processes. A factor can have different values depending upon some definable attribute of a system object, such as customer size associated with a service point. The values are effective-dated so that changes over time are retained. Examples of factors can include minimum/ maximum thresholds, loss factors, etc. Classes of factors are defined that can have numeric values (as in the above examples), or values pointing to profile measuring components or VEE groups.

Factor Value

An effective-dated value - either a number, a profile measuring component, a VEE group, or some custom-defined value - assigned to a factor and associated to the value of some attribute of a system object. For example, let's assume that a service point can be classified as residential, commercial, or industrial. The tolerance percentage by which a customer's consumption can exceed last month's consumption can be tighter as the customer's SP increases in size. An example configuration of factor values for a single factor called "tolerance percentage" could be:

Residential - 20% Commercial - 10% Industrial - 5%

Final Measurement

Measurement data that has been validated, and if necessary, edited & estimated, and is ready for use in down-stream processing such as bill determinants calculations. Only one final measurement can exist for a given date/time for a given measuring component; one final measurement exists per interval, and likewise one final measurement exists for each scalar reading. In both cases, the final measurement value stored represents the amount consumed between its date/time and the prior final measurement's date/time.

Function

An online-initiated action applied to measurement data, comprising one or more measurement services.

Head-End System

A system that collects measurement data and meter events for eventual submission to the application. Many devices can communicate to the application through a single head-end system. A utility may have numerous head-end systems through which they communicate with devices.

Identify Spikes

Measurement service used to identify spikes in a specified set of interval data based on a spike percentage tolerance.

Identifiers

Names, numbers, or other values used to identify an entity within the system, including devices, measuring components, service points, etc.

Inbound Communication

Communication sent to Oracle Utilities Smart Grid Gateway from an external system, such as a head-end system or edge application

Inbound Communication

Communication sent to MDF (Meter Data Framework) from a head-end system or other external system. Each inbound communication has an associated communication type that defines common properties of the communication.

Independent System Operator (ISO)

The entity charged with reliable operation of the grid and provision of open transmission access to all market participants on a non-discriminatory basis.

Initial Measurement Data (IMD)

A set of one or more readings or measurements that have been loaded into the application, usually in a format that is standard for MDF (Meter Data Framework). Over its lifecycle (as pertains to MDM - Meter Data Management), any readings within the IMD are converted into consumption, which is then typically subject to VEE processing and then finalized - meaning stored as final measurements. Only initial measurements can be edited directly by end users of MDM. An IMD for a scalar measuring component will have a single measurement (along with a reading from which the measurement value is derived), while an IMD for an interval measuring component will usually contain multiple interval measurements.

Insert Intervals

Measurement service used to insert one or more intervals into a set of interval measurements.

Installation Constant

An installation constant is set to a value other than 1 as an indication that when calculating consumption, the installation requires that measurement data be multiplied by this value to get accurate results.

Installation Event

A device's installation information at a service point. The install event represents both the installation and removal of a device. It also records turning a device on or off while it is installed at a service point.

Installation On and Off History

A single installation event records each time the device is turned on and turned off while it is installed at a service point.

Interval Channel (Measuring Component)

A business object (BO) that represents channels associated to a device.

Interval Channel Type - Physical (Measuring Component Type)

A business object (BO) that maps properties of interval measuring component types for those Measuring Components that are part of physical devices.

Interval Channel Type - Scratchpad (Measuring Component Type)

A business object (BO) that maps properties relevant to stand-alone measuring components functioning as scratchpads for interval data manipulation.

Interval Data

Time-series data in which measurements are captured in pre-defined intervals (5 minutes, 15 minutes, 1 hour, etc.). A set of interval measurements for an interval measuring component composes an individual initial measurement data record.

Interval Data Services

Services used to access and manipulate interval measurements.

Interval Scratchpad (Measuring Component)

A stand-alone measuring component that provides the user with a means to manipulate measurement data without affecting existing measurements.

Interval Size

The "size" of an interval, representing the length of time between intervals. Interval size is typically measured in seconds-per-interval (SPI).

Manual Meter

A business object (BO) used to model a meter that does not accommodate two-way communications and must be read manually.

Manual Meter Installation Event

A business object (BO) that defines the lifecycle of the installation of a manual meter at a service point.

Manual Meter Type

A business object (BO) used to model properties for meters that are manually read.

Manufacturer

The company that makes devices, defined as an attribute of the device itself.

Market

The jurisdiction or regulatory environment in which a service point participates, defining the valid service providers and their roles. While each service point specifies only one market, different service points throughout the utility's service territory can be linked to different markets.

Market - Fallback Service Provider

For a given market relationship type, a fallback service provider may be defined at the market level, rather than storing the information redundantly on each service point. For example, an entire market might have only one ISO, and if the utility wants to store this information, they can

identify the ISO as a fallback service provider for the market and the market relationship type of ISO.

Market - Relationship Type

The valid roles within a market (ISO, Distribution Company, Retailer, etc.) that have some business significance in the application.

Market - Valid Service Provider

The valid service providers for each market relationship type relevant for a given market. The service providers referenced on a service point must be valid for the combination of the service point's market and the market relationship type.

Market Participant

A variety of service provider; a company with a role within a given market such as a retailer or a distribution company.

Measurement

A measurement in MDM is synonymous with consumption, which implies that constants or multipliers may have been applied to its value. This term can be used in the context of an IMD or in reference to Final Measurements.

Measurement Condition

Codes that indicate the circumstances (estimated, missing, etc.) of individual measurements. Conditions are assigned to both scalar and interval measurement data both for initial measurement data and final measurements.

Measurement Cycle

The measurement cycle can serve two purposes: it can define the schedule for manual meter reading of devices at service points in that cycle, and it can also be configured to define when to create usage transactions for usage subscriptions associated to service points in the cycle.

Measurement Cycle Route

The route used to collect measurements for a given measurement cycle.

Measurement Cycle Route Sequence

The sequence in which measurements are collected along a measurement route.

Measurement Cycle Schedule

Defines the dates on which devices are scheduled to be read.

Measurement Service

Java services that can be invoked to manipulate interval and scalar measurements. Measurement services are invoked by measurement functions (available through certain zones within MDM), and are also used within processing of usage and VEE rules.

Measuring Component Summary

A zone shown on the VEE Group portal that displays a list of measuring components that reference a given VEE group.

Measuring Component

A single point for which data will be received and stored in the system. A measuring component can be associated to a physical device, which can have one or more measuring components, or it can be stand-alone, meaning that it is not associated to a physical device (for example, an aggregator or interval scratchpad).

Measuring Component Type

The definition of the most important properties of a measuring component, including what it measures, how regularly it measures it, whether it should be connected to a physical device or if it's used as a scratchpad or an aggregator, how its final measurements should be stored and how its

user-defined values should be calculated, what rules govern VEE for Measuring Components of the type, as well as numerous display properties that are relevant within MDM. The measuring component type also defines sets of valid attribute values for groups of measuring components belonging to the type.

Measuring Component Types Referencing Group

A zone shown on the VEE Group portal that displays a list of Measuring Component types that reference the VEE group being viewed.

Merge Intervals

Measurement service used to merge a subset of interval data with a specified set of intervals (where overlaps occur, the subset intervals replace the original intervals).

Meter

A device used to measure a quantity of a service (electricity, gas, etc.) delivered to a service point.

Meter Read Download Activity Type

The structure and business rules applicable to downloading meter read information onto a handheld device.

Model

A specific model of a device produced by a manufacturer. Models for a single manufacturer can have diverse service types.

Multiplier

A value that may be applied to adjust the consumption values calculated for a device. Examples include meter/device multiplier, installation constant, loss factor, etc.

Multiply Intervals By Scalar Value

Measurement service that uses the Apply Formula measurement service to multiply the values of a specified set of interval data by a scalar value.

Normalized storage

Storing measurement data in a manner that allows for aggregation and reporting of data through database logic (SQL). Applies to both scalar and interval measurements.

Off-Peak Period

A time period during which the least amount of some consumable is being used. OR A period of relatively low system demand as specified by the supplier.

On-Peak Period

A time period during which the greatest quantity of some consumable is being used OR A period of relatively high system demand as specified by the supplier.

One-Way Communication

Communication from head-end system to Oracle Utilities Smart Grid Gateway that does not trigger a response. Examples of one-way communications include usage readings and device events.

Oracle Utilities Meter Data Management

Oracle Utilities application that provides functionality for handling large volumes of meter data to enable increased accuracy, flexibility, and scalability.

Oracle Utilities Smart Grid Gateway

Oracle Utilities application that provides functionality for orchestrating communication with head-end systems to support import of usage and events, and issuing of meter commands.

Outbound Communication

Communication sent from Oracle Utilities Smart Grid Gateway to a head-end system or other external system.

Peak

The maximum value for some measurable quantity recorded over a specified time period. A measuring component that measures peak quantities will record the highest value for the quantity over a period of time.

Peak Demand

The maximum rate of commodity consumption over a specific period of time.

Processing Method

Methods used to define the format or means by which a service provider receives data from the application, such as bill determinants, interval data, or meter events. Processing methods are also used to define how to create information internal to the application such as initial measurement data and usage transactions. Processing methods can also be used to define the information an external system wishes to subscribe to receive from our application. A BO or batch extract code are the typical processing methods defined for the transmission of data to a service provider.

Processing Role

Each processing method has a processing role, which defines the purpose of the processing method. Some examples of processing roles include: * Initial Measurement Creation (D1) * Device Activity Notification (D1) * Usage Transaction Notification (D2) * Usage Transaction Creation (D2)

Profiling of Scalar Data

The process of applying an interval consumption "shape" to a scalar measurement, using an existing interval measuring component. The individual interval values are adjusted such that when totaled, they equal the value of the scalar measurement.

Reading

The value recorded by a measuring component at a given point in time. A reading often needs to be interpreted in the context of an earlier reading in order to derive a consumption value that would be stored as a measurement. For example, a reading of 1000 for a subtractive measuring component taken on February 1 in the context of a prior reading of 600 taken on January 15 would result in a consumption (measurement) of 400. Readings can either be consumptive or subtractive.

Register (Measuring Component)

A business object (BO) that represents a scalar register found on a standard or smart meter. It does not have a lifecycle, and should be associated with a device configuration.

Register Type - Physical (Measuring Component Type)

Measuring component type business object (BO) that enumerates the properties used by scalar registers.

Remove Intervals

Measurement service used to remove one or more intervals from a set of interval measurements.

Retail Company

A company that is authorized to buy and re-sell a commodity (such as electricity or gas) directly to customers based on territorial regulations.

Retrieve Interval Consumption

Measurement service used to retrieve one or more interval measurements.

Retrieve Scalar Consumption

Measurement service used to retrieve one or more scalar measurements.

Scalar Usage

A measurement of the amount of energy, water, gas, etc. consumed for a given measuring component for a given time period.

Seconds Per Interval

Seconds Per Interval, a way of expressing the length of time between which measurements are taken.

Service Order Requests

Requests that orchestrate the field activities (FAs) and smart meter messages (commands) necessary to change the service point and its installation, to enable or disable service, cut service for non-payment, etc.

Service Point

A location at which a company supplies service. Used to store information describing the type of service and how it is measured.

Service Point Identifier Type

Specific types of service point identifiers.

Service Point Identifier

A collection of identifiers for a given service point.

Service Point Parent

The parent of one or more service points.

Service Point Type

A specific type of service point. Defines how the application manages many aspects of the service point's behavior.

Service Provider

External entities that serve various roles relative to the application. These can be a head-end system, a billing system to which the application sends bill determinant data, a market participant in a deregulated environment, an outage management system that receives meter event data from the application, or other parties that require or provide information to the system.

Service Quantity Identifier

Further distinguishes between measured quantities that have identical UOM/TOU combinations, including situations in which the distinguishing identifier of a UOM is not accurately described as a TOU. SQIs can also be used as a stand-alone representation of a service quantity that is not measured (i.e. one that is not properly described as a UOM) within a Usage SQ collection (e.g. a billing determinant).

Service Type

Specific types of service, such as electric, gas, steam, etc.

Set Condition

Measurement service used to set the condition (status) code of a specified set of interval data.

Shift Intervals

Measurement service used to shift one or more intervals forward or backward in time.

Smart Meter

A business object (BO) used to model smart meters of different service types.

Smart Meter Installation Event

A business object (BO) that defines the lifecycle and rules for installing a smart meter at a service point.

Smart Meter Type

A business object (BO) for device type that references a head-end system as well as a collection of head-ends that are valid for devices of the type, and indicates whether incoming data incorporates the daylight savings time shift. Additionally, the smart meter type includes a list of valid device configurations for its devices.

Substation

A subsidiary station of an electricity generation, transmission and distribution system where voltage is transformed from high to low or the reverse using transformers.

Subtract Scalar Value From Intervals

Measurement service that uses the Apply Formula measurement service to subtract a scalar value from the value of a specified set of interval data.

Subtractive

Describes a measuring component for which consecutive readings must be subtracted to derive a consumption value.

Time of Use

Time of Use - modifiers for a given unit of measure that indicate a period of time during which a quantity has been used, such as On-Peak (meaning during a time when the greatest quantity of some consumable is being used), Off-Peak (meaning during a time when the least amount of some consumable is being used), etc.

Transformer

A device that transfers electrical energy from one circuit to another.

Two-Way Communication

Communication sent from Oracle Utilities Smart Grid Gateway to an external system, such as a head-end system or edge application that triggers a response. Most commands are two-way communications, where Oracle Utilities Smart Grid Gateway issues a command, and the head-end system sends a response as to the success or failure of the command.

Unit of Measure

Identifies quantities measured, such as KWH, KW, cubic feet, degrees Celsius, etc.

Upload Statistics Activities

Activities used to track processing of initial measurement data and device event upload processing.

Usage

A generic term for the amount of energy, water, gas, etc. consumed at one or more service points, sometimes representing quantities that have been adjusted from the original calculated consumption.

User-Defined Measurement Values

Additional values optionally stored with a given measurement that can be used in various calculations. For example, a customer's gas consumption might be measured in cubic feet, but needs to be sent to a billing system in therms. A user-defined value to convert consumption in cubic feet into therms can be configured, and the therm value will then be stored with the measurement in cubic feet.

Validation, Estimation, and Editing (VEE)

The process by which initial measurement data is validated, estimated (if necessary) and edited (if necessary) based on a set of user-defined rules.

VEE Eligibility Criteria

User-definable conditions that could cause a given VEE rule to be applied or skipped. This could involve the evaluation of some attribute of the device or measuring component, or something else entirely.

VEE Exception

An exception generated during Validation, Estimation and Editing (VEE) processing of initial measurement data. Exceptions are assigned a severity that is used in determining whether or not the initial measurement data should be transitioned into an exception state.

VEE Group

A collection of VEE Rules.

VEE Group Matrix (Factor)

A VEE rule within a VEE group can be configured to select from a list of VEE groups (referred to as a matrix) whose associated rules are to be executed next. This list of VEE groups is configured as the values of a factor. One example of its use could be to call geographically-specific VEE groups from within a larger-purpose group. A residential VEE group might contain a rule that will pick the VEE group to execute based on service point location, where the VEE Group Matrix specifies: SP in the North - VEE Group N SP in the East - VEE Group E SP in the South - VEE Group S.

VEE Group Matrix (Factor) Referencing Group

A zone that displays a list of VEE group matrices (factors) that reference the VEE group being viewed in the VEE group portal.

VEE Rule

Standard and custom Validation, Estimation and Editing (VEE) Rules that perform checking and/or manipulation of initial measurement data.

VEE Rules Referencing Group

A zone that displays a list of VEE rules that reference the VEE group being viewed in the VEE group portal.

Appendix C

Base Package Configuration Objects

This appendix provides lists of some base package configuration objects that can aid when implementing Oracle Utilities Smart Grid Gateway. This includes:

- **Base Package Data Areas**
-

Base Package Data Areas

The table below lists the available meter data framework base package data areas.

Data Area	Description
D1-ActivityCommon	Activity Command Data Area
D1-ActivityTypeBasis	Common Schema of Activity Type
D1-AdminBOExceptionHandling	Admin BO Exception Handling Common
D1-AmountandCondition	Amount and Condition Data Area
D1-BOAuditChangedValues	BO Audit Changed Values
D1-CancelCommand Common	Cancel Command Data Area
D1-CommTypeBasicDA	Communication Type Basis
D1-CommandReqIBCommCommonDA	Common Inbound Commn for Command Request
D1-CommandReqOBCommCommonDA	Common Outbound Commn for Command Request
D1-CommandRequestCommon	Command Request Common Data Area
D1-CommandRequestTypeCommon	Command Request Type Common Data Area
D1-CommonActivityType	Activity Type
D1-CommonFactor	Common Factor
D1-CommonFactorValue	Common Factor Value
D1-CommonIMD	Common Initial Measurement Data
D1-CommonIMDAudit	Common IMD Audit
D1-CommonInstantiableBOOutput	Common Instantiable BO's Output
D1-CommonMeasurementSvcsOutput	Common Measurement Service Output
D1-CommonMeterData	Common Meter Data
D1-CommonMeterTypeData	Common Meter Type Data
D1-CommnoProcMethodDetails	Common Processing Method Details
D1-CommonSrvcProvDetails	Common Service Provider Details
D1-CommonSyncRequestContact	Common for Sync Request Contact
D1-CommonSyncRequestDC	Common for Sync Request Device Config
D1-CommonSyncRequestDvc	Common for Sync Request Device
D1-CommonSyncRequestIE	Common for Sync Request Install Event
D1-CommonSyncRequestInData	Sync Request Inbound Common Data
D1-CommonSyncRequestMC	Common for Sync Request Inbound MC
D1-CommonSyncRequestSP	Common for Sync Request Service Point

Data Area	Description
D1-CommonUomSpiIdentGroup	Common UOM/SPI Identification Group
D1-CommonVEEIMD	Common VEE Initial Measurement Data
D1-CompletionEvent	Completion Event Parent Data Data
D1-ConDisconNotification	ConnectDisconnect Notification
D1-DeviceConfigRelatedActivity	Device Config Related To Activity DA
D1-DeviceEvent	Device Event
D1-DeviceRelatedActivity	Device Related to Activity Data Area
D1-DeviceStatusCheckNotif	Device Status Check Notification
D1-EligibleProfileFactors	Eligible Profile Factors for Data Creation
D1-EventBarProfileData	Event Bar Profile Data
D1-EventBarProfileList	Event Bar Profile List
D1-EventInformation	Event Information
D1-ExtendableLookupColumn	Extendable Lookup Common Data Area
D1-ExtendedMsrmtList	Extended ML
D1-FVOverlayDefaultProfileList	Final Values Overlay Default Profile List
D1-FVOverlayProfileData	Final Values Overlay Profile Data
D1-FailReponse	Fail Response Data Area
D1-FallbackSPrValList	Fallback Service Provider Validation List
D1-FileHeaderInfoDA	Upload Statistics File Header Information
D1-GenSavePointDispDA	Generic Savepoint Dispatcher DA
D1-IBCommCommonDA	Inbound Communication Common
D1-IERelatedActivityDataArea	Common Schema of IE Related Activities
D1-InstallEventRelatedActivity	Activity Related Installation Event
D1-MCSnapShot	SnapShot group for Sync Request Inbound MC
D1-MRDownloadActivityCommon	Common schema Meter Read Dwnld Activity
D1-MeasurementDataList	Measurement Data List
D1-NumberOfIntervalsAndDateTm	Number of Intervals and Date Time
D1-OBCommCommonDA	Outbound Communication Common Data Area
D1-PayloadProcessErrorsDA	Payload Errors DA
D1-PayloadProcessInfoDA	Payload Processing Information
D1-ReadDeviceNotification	Read Device Notification Data Area
D1-ReceivedResponse	Received Response Data Area

Data Area	Description
D1-RetryDetails	Retry Details
D1-SOAPFaultDA	SOAP Fault
D1-SPMeterHistorySyncRequestIE	MDM SP/Meter History
D1-SPRelatedActivity	SP Related to Activity Data Area
D1-ScalToIntProfileFactor	Convert Scalar to Interval Profile Factors
D1-ScratchpadMCTypes	Scratchpad MC Types
D1-StandardDeviceEventType	Standard Device Event Type
D1-StandardLogFields	Standard Log Fields
D1-StatusCodeDescriptions	Status Code Descriptions Data
D1-SuccessResponse	Success Response Data Area
D1-SyncRequestInbound	Inbound Sync Request
D1-TraceData	Trace Data
D1-UsageCommon	Usage Common Data Area
D1-VEECommon	VEE Common Data Area
D1-VEEIMDSeeder	VEE IMD Seeder Initial Measurement Data

Use the Data Area portal to view more details concerning these data areas.

Base Package Extendable Lookups

The table below lists the available meter data framework base package extendable lookups.

Business Object	Description
D1-360EventBarProfile	360 View Event Bar Profile
D1-DaysSinceLastNormalMsrmntLkp	Days Since Last Normal Measurement
D1-DeviceEventMappingLookup	Device Event Mapping
D1-DeviceLocationLookup	Device Location
D1-DvcCommunicationStatLookup	Device Communication Status
D1-DvcConnectionStatLookup	Device Connection Status
D1-DvcEventCategoryLookup	Device Event Category
D1-DvcFunctionalStateLookup	Device Functional State
D1-ExecutionPriorityLookup	Execution Priority
D1-ExternalActTypeIdentifier	External Activity Type Identifier
D1-ExternalCommTypeLookup	External Communication Type
D1-FinalValuesOverlayProfile	Final Values Overlay Profile
D1-HeadendUOMLookup	Headend UOM Code to Standard UOM
D1-KeyLookup	Key
D1-LogicalStatusLookup	Logical Status
D1-MeasurementConditionLookup	Measurement Condition
D1-OKToEnterLookup	Ok to Enter
D1-SPInstructionLookup	SP Instruction
D1-SPWarningLookup	SP Warning
D1-StdEventNameLookup	Standard Event Name
D1-TransactionTypeLookup	Transaction Type

Use the Extenable Lookup portal to view more details concerning these extendable lookups.

Appendix D

The Oracle Utilities Smart Grid Gateway “Generic” Adapter

This document describes the “generic” adapter provided as part of the demonstration data for Oracle Utilities Smart Grid Gateway. This adapter is provided as a starting point for customers to create their own adapters for using Oracle Utilities Smart Grid Gateway with different head-end systems. This document includes:

- **Overview**
- **Oracle Utilities Smart Grid Gateway Generic Adapter Data**
- **Configuring and Customizing the Generic Adapter Oracle Service Bus (OSB) Processes**
- **Configuring and Customizing the Generic Adapter BPEL Processes**
- **Emulating a Head-End System**

The generic adapter supports the Remote Connect, Remote Disconnect, and On-Demand Read commands, which can be used as the basis for adding additional commands based on your specific requirements.

Important Note

Customization of the Oracle Utilities Smart Grid Gateway generic adapter requires and understanding of Java programming and of Oracle Service Bus (OSB) and Oracle Business Process Execution Language (BPEL). These tools are part of the Oracle SOA Suite. This guide does not provide specific information about the use of these tools. See the Oracle SOA Suite Documentation library (<http://www.oracle.com/technetwork/middleware/soasuite/documentation/index.html>) for more information about using these tools.

Overview

The Oracle Utilities Smart Grid Gateway generic adapter is provided as a sample adapter to illustrate the configuration required to support communication between head-end systems and Oracle Utilities Smart Grid Gateway. This generic adapter is intended to be used as a starting point for customers to create their own adapters for using Oracle Utilities Smart Grid Gateway with different head-end systems.

The Oracle Utilities Smart Grid Gateway generic adapter includes the following:

- **Demonstration Data:** A set of sample data provided in the demonstration database that can be used with the generic adapter processes as delivered (prior to any customization you might perform as part of your implementation).

See **Oracle Utilities Smart Grid Gateway Generic Adapter Data** on page D-3 for more information about the demonstration data provided with the generic adapter.

- **Sample Oracle Service Bus Process:** A simple file parsing and transformation process used to support import of usage readings, delivered as an Oracle Service Bus (OSB) configuration.

See **Configuring and Customizing the Generic Adapter Oracle Service Bus (OSB) Processes** on page D-11 for more information about the sample OSB process provided with the generic adapter.

- **Sample Oracle Business Process Execution Language Processes:** A set of sample communication processes that support Remote Connect, Remote Disconnect, and On-Demand Read commands delivered as an Oracle Business Process Execution Language (BPEL) project.

See **Configuring and Customizing the Generic Adapter BPEL Processes** on page D-27 for more information about the sample BPEL processes provided with the generic adapter.

- **Sample Two-Way Communication Test Harness:** A head-end system emulator delivered as a soapUI project.

See **Emulating a Head-End System** on page D-36 for more information about the sample test harness provided with the generic adapter.

Oracle Utilities Smart Grid Gateway Generic Adapter Data

This section outlines the data used by the Oracle Utilities Smart Grid Gateway by the “generic” adapter that is provided in the demonstration database, including:

- **Required Data**
- **Demonstration Data**

Required Data

In order to configure and test the generic adapter, the following data must be configured in Oracle Utilities Smart Grid Gateway:

- **Service Provider:** A Service Provider that defines the head-end system.
 - Business Object: D1-HeadEndSystem
 - Utility Device ID Type: Internal Meter Number
 - Utility Measuring Component ID Type: Channel ID
 - External System: external system (see below) for the head-end system
 - Processing Roles: Remote Connect, Remote Disconnect
- **Device Data:** Device and related data that will connect to the head-end system
 - Device
 - Business Object: D1-SmartMeter
 - Head-End System: the head-end system
 - Must have an Internal Meter Number (based on head-end system)
 - Device Configuration
 - Business Object: D1-DeviceConfiguration
 - Measuring Component (one or more)
 - Business Object: D1-IntervalChannel or D1-Register
 - Service Point
 - Business Object: DM_ElectricServicePoint
 - Install Event
 - Business Object: D1-SmartMeterInstallEvent
- **Commands and Communications:** Commands and related data for each type of command to be issues
 - Activity Types for each type of command

Note: These are provided as part of the Oracle Utilities Smart Grid Gateway base package, but confirm that they are “Active” via the Activity Type portal.
 - Outbound Communications
 - Outbound Communication Business Object for each communication type to be sent the head-end system
 - Outbound Message Type for each communication type to be sent to the head-end system (connect, disconnect, etc.)
 - XAI Sender for communication type to be sent to the head-end system (connect, disconnect, etc.)

- External System that represents the head-end system
 - Links Outbound Message Types to XAI Senders for each type of message
- Inbound Communication
 - Inbound Communication Business Object for each communication type to be received from the head-end system
 - XAI Inbound Service for each communication type to be received from the head-end system (linked to Inbound Communication business objects).
- Completion Events appropriate to each type of command

Note: These are provided as part of the Oracle Utilities Smart Grid Gateway base package.

Demonstration Data

This section provides descriptions of the Oracle Utilities Smart Grid Gateway demonstration data provided for use with the generic adapter, including:

- **Device and Service Point Data**
- **Service Provider and Communications**

Device and Service Point Data

The following device and service point related data has been configured to work with the Oracle Utilities Smart Grid Gateway generic adapter.

Device

The Oracle Utilities Smart Grid Gateway demonstration data contains the following device which can be used with the generic adapter:

- **Device Type:** Electric Smart Meter
- **Serial Number:** ER-SM-UNV-01
- **Internal Meter Number:** ER-SM-UNV-01
- **Pallet Number:** ER-SM-UNV-01
- **External ID:** ER-SM-UNV-01
- **Manufacturer:** General Electric
- **Model:** GE simple kWh
- **Incoming Data Shift:** Always in Local Time
- **Arming Required:** Arming Not Required
- **Head-End System:** Universal Head-End Provider
- **Status:** Active

Device Configuration

The Oracle Utilities Smart Grid Gateway demonstration data contains the following device configuration which can be used with the generic adapter:

- **Device Configuration Type:** Electric Residential Smart Meter - 60min kWh
- **Device ID:** ER-SM-UNV-01
- **Effective Date:** 01-01-2011 12:00 AM
- **Time Zone:** US Pacific Time

- **External ID:** ER-SM-UNV-01
- **Status:** Active

Measuring Component

The Oracle Utilities Smart Grid Gateway demonstration data contains the following measuring components which can be used with the generic adapter:

Interval Channel:

- **Measuring Component Type:** Electric kWh 60min
- **Device Configuration:** Electric Smart Meter / Effective Date: 01-01-2011 12:00 AM
- **How To Use:** Additive
- **Number of Digits Left:** 5
- **Number of Digits Right:** 4
- **Channel Multiplier:** 1.000000
- **Channel ID:** 1
- **External ID:** ER-SM-UNV-01

Register:

- **Measuring Component Type:** Electric kWh
- **Device Configuration:** Electric Smart Meter / Effective Date: 01-01-2011 12:00 AM
- **How To Use:** Additive
- **Number of Digits Left:** 5
- **Number of Digits Right:** 3
- **Register Multiplier:** 1.000000
- **Read Sequence:** 0
- **Read Out Type:** Automatic
- **Tolerance:**
- **Full Scale:** 0.000000
- **Channel ID:** 1

Service Point

The Oracle Utilities Smart Grid Gateway demonstration data contains the following service point which can be used with the generic adapter:

- **Service Point Type:** Electric Residential
- **Status:** Active
- **Address:** 4 Embarcadero Center, San Francisco, CA, 94111, USA
- **Time Zone:** US Pacific Time
- **Geographic Latitude:** 0.000000
- **Geographic Longitude:** 0.000000
- **Market:**
- **Source Status:** Connected
- **External ID:**
- **Main Contact:**

- **Loss Factor Classification:** Regular
- **Temperature Zone:** North Stark County
- **Customer Area:** San Francisco
- **Network Device:** NETWORK2
- **Consumption Profile:** Residential

Install Event

The Oracle Utilities Smart Grid Gateway demonstration data contains the following install event which can be used with the generic adapter:

- **Device Configuration:** Electric Smart Meter / Effective Date/Time: 01-01-2011 12:00 AM
- **Service Point:** 4 Embarcadero Center, Street Level, SF, San Francisco, CA, USA
- **Status:** Connected / Commissioned
- **Installation Date/Time:** 01-01-2011 12:00 AM
- **Installation Constant:** 1.000000
- **Arming Status:** Armed
- **External ID:**
- **Device On/Off Status:** On

Service Provider and Communications

The following service provider and communications-related data has been configured to work with the Oracle Utilities Smart Grid Gateway generic adapter.

Service Provider / Head-End System

The Oracle Utilities Smart Grid Gateway demonstration data includes the following service provider to represent the head-end system with the generic adapter.

Attribute/Field	Value
Service Provider Business Object	Head-End System (D1-HeadEndSystem)
Service Provider	UNVHD
Description	Universal Head-End Provider
External Reference ID	UNVHD
External System	Universal Head-End System
Our Name/ID in Their System	UNVHD
Utility Device ID Type	Internal Meter Number
Utility Measuring Component ID Type	Channel ID

The UNVHD service provider provided with the Oracle Utilities Smart Grid Gateway generic adapter uses the following processing methods:

Processing Role	Processing Method
Initial Measurement Creation	How To Create MC Related Information Default Processing Method - Business Object: DM-InitialLoadIMDScalar (Universal Head-End Initial Load IMD - Scalar)
On-Demand Read (Scalar)	How To Create Activity Outbound Communication Processing Method <ul style="list-style-type: none"> Default Business Object: DM-InitiateMRByMtrNbr (Initiate Meter Read by Meter (Multispeak)) Default Outbound Message Type: On-Demand Read
Remote Connect	How To Create Activity Outbound Communication Processing Method <ul style="list-style-type: none"> Default Business Object: DM-InitiateConnectDisconnect (Initiate Connect Disconnect) Default Outbound Message Type: Connect Device
Remote Disconnect	How To Create Activity Outbound Communication Processing Method <ul style="list-style-type: none"> Default Business Object: DM-InitiateConnectDisconnect (Initiate Connect Disconnect) Default Outbound Message Type: Connect Device

Command processing methods can define a default outbound communication business object/outbound message type, or can specify an outbound communication business object/outbound message type for each Device Type.

See **Service Providers in Detail** and **Processing Methods in Detail** in the *Oracle Utilities Smart Grid Gateway Configuration Guide* for more information about service providers and processing methods.

Outbound Communication Business Objects

An outbound communication business object must be created for each type of message to be sent to the head-end system, based on the types of messages the system is designed to accept. The Oracle Utilities Smart Grid Gateway demonstration data includes the following outbound communication business objects for use with the generic adapter:

Command	Outbound Communication Business Object
Remote Connect	DM-InitiateConnectDisconnect (Initiate Connect Disconnect)
Remote Disconnect	DM-InitiateConnectDisconnect (Initiate Connect Disconnect)
On-Demand Read	DM-InitiateMRByMtrNbr (Initiate Meter Read by Meter (Multispeak))

Outbound Message Types

An outbound message type must also be created for each type of message to be sent to the head-end system. Again, this is based on the types of messages the head-end system is designed to accept. The Oracle Utilities Smart Grid Gateway demonstration data includes the following outbound message types for use with the generic adapter:

Command	Outbound Message Type
Remote Connect	DM-CONNECT (Connect Device)
Remote Disconnect	DM-DISCONN (Disconnect Device)
On-Demand Read	DM-ONDMRD (On-Demand Read)

Refer to the Oracle Utilities Application Framework documentation for more information about outbound message types.

XAI Senders

You must also create an XAI Sender for each type of message to be sent to an external system. XAI senders define the details of how messages are sent to an external system. As in the case of outbound communication business objects and outbound message types, the set of XAI senders you need to create is based on the types of messages the head-end system is designed to accept. The Oracle Utilities Smart Grid Gateway demonstration data includes the following XAI senders for use with the generic adapter:

Command	XAI Sender
Remote Connect	DM-Connect (Connect Device)
Remote Disconnect	DM-Disconnec (Disconnect Device)
On-Demand Read	DM-ONDMRD (On-Demand Read)

Refer to the Oracle Utilities Application Framework documentation for more information about XAI senders.

External Systems

You must also create an External System for each external system to which Oracle Utilities Smart Grid Gateway will send messages. Each external system defines a set of outbound message types that will be sent to that system. The external system also specifies the following for each outbound message type:

- The outbound message type
- The processing method used to send the message (Batch, XAI, or Real-time)
- The corresponding XAI senders
- Batch Control (if Processing Method is set to Batch)
- Message XSL, W3C Schema, and Response XSL (as applicable)

The Oracle Utilities Smart Grid Gateway demonstration data includes the following external system for use with the generic adapter:

- **External System:** UNVHD

- **Description:** Universal Head-End System

Universal Head-End System			
Outbound Message Type	Processing Method	XAI Sender	Message XSL / Response XSL
DM-COMMS (Commission Device)	Real-time	DM-Comms (Commission Device)	DM-Request.xml / DM-Response.xml
DM-CONNECT (Connect Device)	Real-time	DM-Connect (Connect Device)	DM-Request.xml / DM-Response.xml
DM-DECOMMS (Decommission)	Real-time	DM-Decomms (Decommissioning Sender)	DM-Request.xml / DM-Response.xml
DM-DISCONN (Disconnect Device)	Real-time	DM-Disconnect (Disconnect Device)	DM-Request.xml / DM-Response.xml
DM-ONDMRD (On-Demand Read)	Real-time	DM-ONDMRD (On-Demand Read)	DM-Request.xml / DM-Response.xml

Note: This initial release of the generic adapter supports only Remote Connect and Remote Disconnect commands.

Refer to the Oracle Utilities Application Framework documentation for more information about external systems.

Inbound Communication Business Objects

An inbound communication business object must be created for each type of message to be received from the head-end systems, based on the types of messages the system is designed to send. The Oracle Utilities Smart Grid Gateway demonstration data includes the following inbound communication business objects for use with the generic adapter:

Command Being Responded To	Inbound Communication Business Object
Remote Connect	DM-ConnectDisconStateChgNtf (Connect Disconnect State Changed Notification (Multispeak))
Remote Disconnect	DM-ConnectDisconStateChgNtf (Connect Disconnect State Changed Notification (Multispeak))
On-Demand Read	DM-ReadingChgNotification (Reading Changed Notification (Multispeak))

XAI Inbound Services

You must also create an XAI Inbound Service for each type of message to be received from an external system. XAI inbound systems define the details of how messages are received from an external system, including the inbound communication business object (or business service or service script) to be invoked when the response message is received. As in the case of inbound communication business objects, the set of XAI inbound services you need to create is based on the types of messages the system is designed to send. In addition, you must also create an XAI Inbound Service to import usage readings and/or device events that come in from the head-end system. The Oracle Utilities Smart Grid Gateway demonstration data includes the following XAI inbound services for use with the generic adapter:

XAI Inbound Service	Schema (Inbound Communication Business Object)
D1-DeviceEventSeeder	D1-DeviceEventSeeder (Device Event Seeder)
D1-InitialLoadIMD	D1-IMDSeeder (IMD Seeder)
DM-ReadingChangedNotification	DM-ReadingChgNotification (Reading Changed Notification (Multispeak))
Initiate Connect Disconnect Response	DM-ConnectDisconStateChgNtf (Connect Disconnect State Changed Notification (Multispeak))

Refer to the Oracle Utilities Application Framework documentation for more information about XAI inbound services.

Configuring and Customizing the Generic Adapter Oracle Service Bus (OSB) Processes

This section describes the sample Oracle Service Bus (OSB) processes provided with the generic adapter, and how to deploy it to work with your implementation, including:

- **Understanding the Generic Adapter Oracle Service Bus (OSB) Processes**
- **Opening and Deploying the OSB Processes**
- **Customizing the Generic Adapter Oracle Service Bus (OSB) Processes**
- **FileParser Interface**

Understanding the Generic Adapter Oracle Service Bus (OSB) Processes

This section provides an overview of the sample Oracle Service Bus (OSB) processes provided with the Oracle Utilities Smart Grid Gateway generic adapter for use when importing usage readings.

The sample generic adapter OSB process is provided in the “sgg-osb-generic-adapter.zip” file. This is a sample OSB configuration that can be edited and changed to meet the requirements of your head-end system and implementation. The sample OSB process comprises two parts:

Java Package: This contains an implementation to parse and transform an incoming payload in the format used by the Initial Measurement Data Upload feature (see **Example: Processing An XML File** on page D-15 for an example of this XML format). This implementation leverages the JCA (Java EE Connector Architecture) Adapter functionality of OSB. See **Technical Notes Regarding the Use of the JCA Adapter** on page D-14 for more information.

The generic adapter Java package contains the following two classes:

- **FileParserGenGeneric:** a super class with members that are common for Usage and Event related functionality. This class implements the `com.splwg.d1.sgg.osb.common.FileParser` interface (see **FileParser Interface** on page D-25). It also implements the `com.splwg.d1.sgg.osb.common.FileParser2` interface to support IMD & Event Upload Statistics related functionality (see **FileParser2 Interface** on page D-26).
- **FileParserGenGenericUsage:** an implementation of the following logic:
 1. parsing of incoming payload
 2. extracting information about current device
 3. breaking incoming data to separate initial measurement data structures
 4. extending the structures with device information
 5. returning the structures one by one as a stream to caller

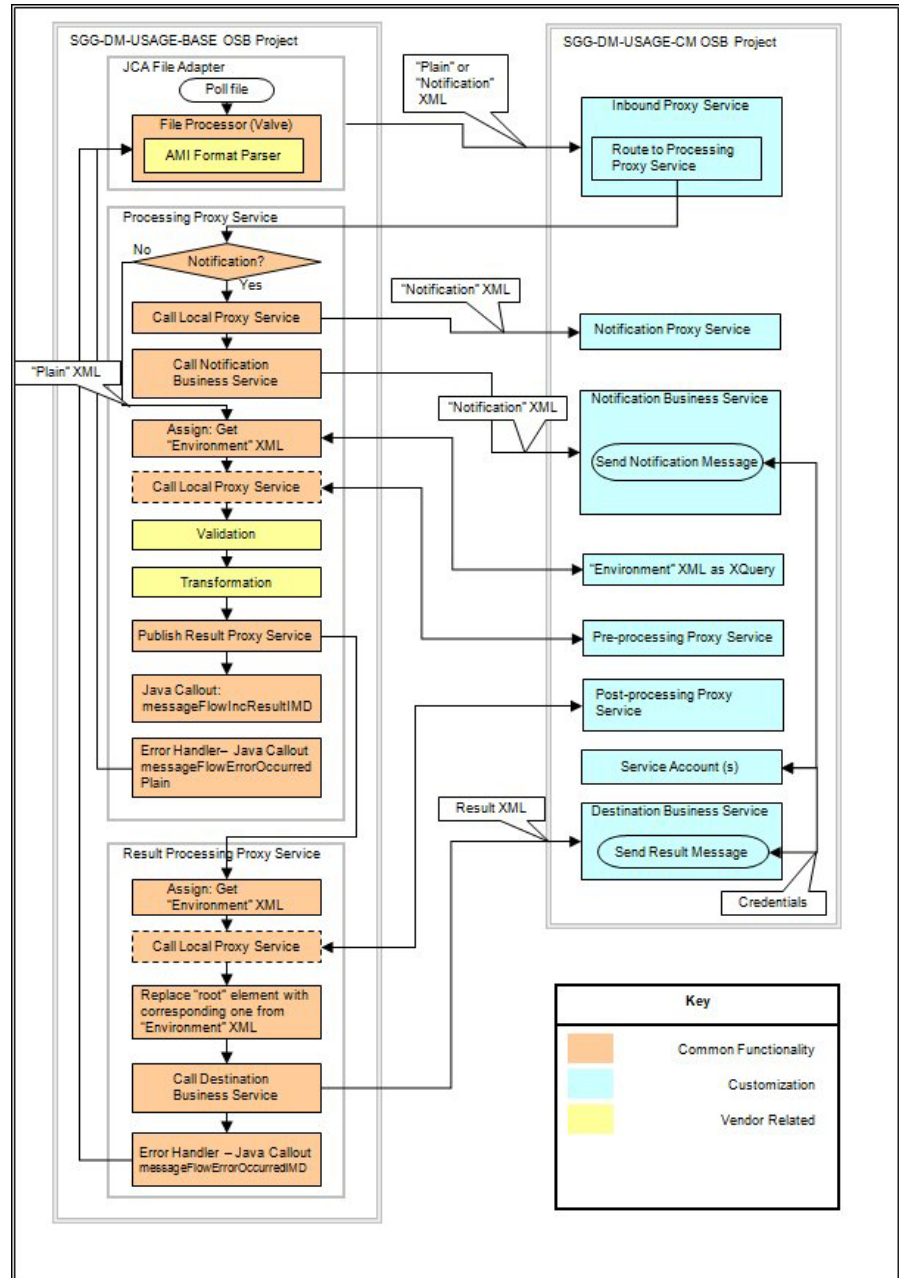
OSB Configuration: The OSB configuration consists of two Oracle Service Bus (OSB) projects:

- **SGG-DM-USAGE-BASE** contains the components responsible for “actual” processing of incoming data. This project can be upgraded without affecting the customization and environment settings done in the **SGG-DM-USAGE-CM** project. This project provides the following functional areas:
 - Processing of notification messages that are sent from the File Processing and File Parsing functionality implemented in Java.
 - Call the customizable local service proxies for pre- and post- processing of passed data
 - Validation of passed data

- Transformation of passed data. In the sample generic adapter, a simple XQuery code is used just to show the functionality available for transformation. This is used to replace the root element of the XML structure of the individual structures.
- Sending of resulting XML structures to destination JMS queue in a format acceptable by an XAI inbound service.
- Update of statistic information via calling MessageFlowIncResultIMD method.
- Error handling that provides information about occurred error to the File Processing component. Error handling contains separate handlers for two stages: before and after the XML generated in FileParserGenGenericUsage is transformed to the final XML structure. The separation is necessary to make Java calls to appropriate methods (messageFlowErrorOccurred or messageFlowErrorOccurredIMD) according to the current processing stage.
- **SGG-DM-USAGE-CM** allows the customization and simplifies the future upgrades.

OSB Process Overview

This section provides an overview of logic performed by the generic adapter OSB process. The diagram below illustrates the OSB process used by the generic adapter for incoming measurement data.



The sample OSB process performs the following steps:

1. Parses a sample payload containing multiple measurement readings for a device into individual records.
2. Inserts an IMD type element (<imdType>) into each individual record.
3. Inserts an external ID element (<externalId>) containing the file name into each individual record.
4. Inserts a Device ID element (<dvcIdN>) into each individual record.

5. Inserts a Service Provider element (<serviceProvider>) and Service Provider External ID element (<serviceProviderExternalId>) into each individual record.
6. Replaces the root element (<initialMeasurementData>) of each individual record with an “Environment” XML element (<D1-InitialLoadIMD>) defined in the SGG-DM-USAGE-CM project. This is the name of the XAI Inbound Service to which the record will be sent.
7. Passes the individual records to the XAI Inbound Service

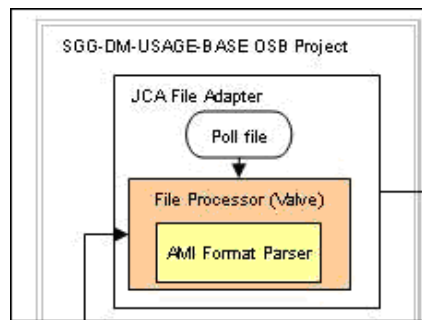
Steps 1 through 4 are performed by the Java Package described above.

Steps 5 and 6 are performed by the OSB configuration described above (containing the SGG-DM-USAGE-BASE and SGG-DM-USAGE-CM OSB projects).

Technical Notes Regarding the Use of the JCA Adapter

This section provides a high-level description of the relationship between the Java package provided with the generic adapter and the JCA adapter functionality provided by OSB.

The diagram below (excerpted from the diagram in the **OSB Process Overview** on page D-13) illustrates the relationship between the JCA Adapter for Files/FTP, the Valve interface, and the Java package used by the generic adapter OSB process.



- Oracle Service Bus (OSB) provides a J2EE Connector Architecture (JCA) transport that interacts with Enterprise Information Systems (EIS), allowing EIS applications and services participate in the service bus environment. OSB supports multiple JCA-compliant adapters, including:
 - Oracle Adapter for Oracle Applications
 - Oracle JCA Adapter for AQ
 - Oracle JCA Adapter for Database
 - Oracle JCA Adapter for Files/FTP

See section 25, **JCA Transport** (http://download.oracle.com/docs/cd/E17904_01/doc.1111/e15866/jca.htm#i1106345) in the *Oracle Fusion Middleware User's Guide for Technology Adapters* for more information about JCA transport supported by OSB.

- For one-way communication from head-end systems, Oracle Utilities Smart Grid Gateway adapters use the Oracle JCA Adapter for Files/FTP. See section 4, **Oracle JCA Adapter for Files/FTP** (http://download.oracle.com/docs/cd/E17904_01/integration.1111/e10231/adptr_file.htm#BABCJJCD) in the *Oracle Fusion Middleware User's Guide for Technology Adapters* for more information about this adapter.

More specifically, Smart Grid Gateway adapters utilize this JCA adapter's ability to inject pre-processing logic into its pipeline. see section 4.2.14, **Pre-Processing and Post-Processing of Files** (http://download.oracle.com/docs/cd/E17904_01/integration.1111/e10231/adptr_file.htm#BABCDFDGC) in the *Oracle Fusion Middleware User's Guide for Technology Adapters* for more information about pre-processing and post-processing of files using the JCA Adapter For Files/FTP.

The Oracle JCA Adapter for Files/FTP is depicted in by the “JCA File Adapter” block in the above diagram.

- The Valve interface required by the Oracle JCA Adapter for Files/FTP is implemented in Java code in the meter data framework. This code is invoked by the JCA adapter.

The Valve interface is depicted by the “File Processor” block in the above diagram.

- The “File Processor” code invokes the implementation of a vendor-specific “AMI File Parser”. In the case of the generic adapter, this implementation is done in the Java package provided in the sgg-osb-generic-adapter.zip file. Note that only implementation of file parsing is provided with the generic adapter, while the meter data framework related code (the “File Processor” block) is deployed as a part of the Oracle Utilities Smart Grid Gateway installation.

The Java package is depicted by the “AMI File Parser” block in the above diagram.

Refer to the *Oracle Fusion Middleware User's Guide for Technology Adapters* for more information about using JCA Adapters with OSB.

Example: Processing An XML File

This section provides an illustration of how an input XML file is processed and transformed by the sample OSB process.

Sample Initial File

This sample file contains 3 readings for a single sample device (ER-SM-UNV-01) connected to a sample head-end system (UNVHD).

File Name: dm-usage-1.xml

```
<deviceList>
  <device>
    <headEnd>UNVHD</headEnd>
    <headEndExternalId>UNVHD</headEndExternalId>
    <deviceId/>
    <deviceIdIdentifierNumber>ER-SM-UNV-01</deviceIdIdentifierNumber>
    <initialMeasurementDataList>
      <initialMeasurementData>
        <preVEE>
          <uom>KWH</uom>
          <stDt>2011-01-01-00.00.00</stDt>
          <enDt>2011-01-01-00.30.00</enDt>
          <spi>900</spi>
          <msrs>
            <mL>
              <s>1</s>
              <q>0.2316</q>
            </mL>
            <mL>
              <s>2</s>
              <q>0.1416</q>
            </mL>
          </msrs>
        </preVEE>
      </initialMeasurementData>
    </initialMeasurementDataList>
  </device>
</deviceList>
```

```

        <stDt>2011-01-01-00.30.00</stDt>
        <enDt>2011-01-01-01.00.00</enDt>
        <spi>900-error</spi>
        <msrs>
            <mL>
                <s>1</s>
                <q>0.2316</q>
            </mL>
            <mL>
                <s>2</s>
                <q>0.1416</q>
            </mL>
        </msrs>
    </preVEE>
</initialMeasurementData>
<initialMeasurementData>
    <preVEE>
        <uom>KWH</uom>
        <stDt>2011-01-01-01.00.00</stDt>
        <enDt>2011-01-01-01.30.00</enDt>
        <spi>900</spi>
        <msrs>
            <mL>
                <s>1</s>
                <q>0.2316</q>
            </mL>
            <mL>
                <s>2</s>
                <q>0.1416</q>
            </mL>
        </msrs>
    </preVEE>
</initialMeasurementData>
</initialMeasurementDataList>
</device>
</deviceList>

```

Interim Format

The Java package parses and transforms the XML structure of the payload into an interim format in which individual usage readings are sent from the Java parser to the SGG-DM-USAGE-BASE and SGG-DM-USAGE-CM OSB projects. Note the presence and location of the <imdType>, <externalId>, <dvcIdN>, <serviceProvider>, and <serviceProviderExternalId> elements in each record.

Record 1:

```

<initialMeasurementData>
    <preVEE>
        <imdType>D1IL</imdType>
        <externalId>dm-usage-1.xml</externalId>
        <dvcIdN>ER-SM-UNV-01</dvcIdN>
        <uom>KWH</uom>
        <stDt>2011-01-01-00.00.00</stDt>
        <enDt>2011-01-01-00.30.00</enDt>
        <spi>900</spi>
        <msrs>
            <mL>
                <s>1</s>
                <q>0.2316</q>
            </mL>
            <mL>
                <s>2</s>
                <q>0.1416</q>
            </mL>
        </msrs>
    </preVEE>
    <serviceProvider>UNVHD</serviceProvider>

```



```
<serviceProviderExternalId>UNVHD</serviceProviderExternalId>
</initialMeasurementData>
```

Record 2:

```
<initialMeasurementData>
  <preVEE>
    <imdType>D1IL</imdType>
    <externalId>dm-usage-1.xml</externalId>
    <dvcIdN>ER-SM-UNV-01</dvcIdN>
    <uom>KWH</uom>
    <stDt>2011-01-01-00.30.00</stDt>
    <enDt>2011-01-01-01.00.00</enDt>
    <spi>900-error</spi>
    <msrs>
      <mL>
        <s>1</s>
        <q>0.2316</q>
      </mL>
      <mL>
        <s>2</s>
        <q>0.1416</q>
      </mL>
    </msrs>
  </preVEE>
  <serviceProvider>UNVHD</serviceProvider>
  <serviceProviderExternalId>UNVHD</serviceProviderExternalId>
</initialMeasurementData>
```

Record 3:

```
<initialMeasurementData>
  <preVEE>
    <imdType>D1IL</imdType>
    <externalId>dm-usage-1.xml</externalId>
    <dvcIdN>ER-SM-UNV-01</dvcIdN>
    <uom>KWH</uom>
    <stDt>2011-01-01-01.00.00</stDt>
    <enDt>2011-01-01-01.30.00</enDt>
    <spi>900</spi>
    <msrs>
      <mL>
        <s>1</s>
        <q>0.2316</q>
      </mL>
      <mL>
        <s>2</s>
        <q>0.1416</q>
      </mL>
    </msrs>
  </preVEE>
  <serviceProvider>UNVHD</serviceProvider>
  <serviceProviderExternalId>UNVHD</serviceProviderExternalId>
</initialMeasurementData>
```

Final Format

The OSB configuration then transforms the data into a final format in which the records are sent from OSB to XAI Inbound Service via JMS Queue. Note that the <initialMeasurementData> root element has been replaced by the <D1-InitialLoadIMD> root element.

Record 1:

```
<D1-InitialLoadIMD>
  <preVEE>
    <imdType>D1IL</imdType>
    <externalId>dm-usage-1.xml</externalId>
    <dvcIdN>ER-SM-UNV-01</dvcIdN>
    <uom>KWH</uom>
    <stDt>2011-01-01-00.00.00</stDt>
```

```
<enDt>2011-01-01-00.30.00</enDt>
<spi>900</spi>
<msrs>
  <mL>
    <s>1</s>
    <q>0.2316</q>
  </mL>
  <mL>
    <s>2</s>
    <q>0.1416</q>
  </mL>
</msrs>
</preVEE>
<serviceProvider>UNVHD</serviceProvider>
<serviceProviderExternalId>UNVHD</serviceProviderExternalId>
</D1-InitialLoadIMD>
```

Record 2:

```
<D1-InitialLoadIMD>
  <preVEE>
    <imdType>D1IL</imdType>
    <externalId>dm-usage-1.xml</externalId>
    <dvcIdN>ER-SM-UNV-01</dvcIdN>
    <uom>KWH</uom>
    <stDt>2011-01-01-00.30.00</stDt>
    <enDt>2011-01-01-01.00.00</enDt>
    <spi>900-error</spi>
    <msrs>
      <mL>
        <s>1</s>
        <q>0.2316</q>
      </mL>
      <mL>
        <s>2</s>
        <q>0.1416</q>
      </mL>
    </msrs>
  </preVEE>
  <serviceProvider>UNVHD</serviceProvider>
  <serviceProviderExternalId>UNVHD</serviceProviderExternalId>
</D1-InitialLoadIMD>
```

Record 3:

```
<D1-InitialLoadIMD>
  <preVEE>
    <imdType>D1IL</imdType>
    <externalId>dm-usage-1.xml</externalId>
    <dvcIdN>ER-SM-UNV-01</dvcIdN>
    <uom>KWH</uom>
    <stDt>2011-01-01-01.00.00</stDt>
    <enDt>2011-01-01-01.30.00</enDt>
    <spi>900</spi>
    <msrs>
      <mL>
        <s>1</s>
        <q>0.2316</q>
      </mL>
      <mL>
        <s>2</s>
        <q>0.1416</q>
      </mL>
    </msrs>
  </preVEE>
  <serviceProvider>UNVHD</serviceProvider>
  <serviceProviderExternalId>UNVHD</serviceProviderExternalId>
</D1-InitialLoadIMD>
```

Upload Statistics

The OSB configuration also sends upload statistics information (including Payload Statistics, Payload Error Notification, and Payload Summary) to SGG via an XAI Inbound Service via JMS Queue.

Payload Statistics:

```
<D1-PayloadStatistics>
  <externalSenderId>DM</externalSenderId>
  <fileName>dm-usage-1.xml-2011-07-13-14-21-58-789</fileName>
  <middlewareStatistics>
    <processingInformation>
      <receivedDateTime>2011-07-13-14.21.58</receivedDateTime>
    </processingInformation>
  </middlewareStatistics>
</D1-PayloadStatistics>
```

Payload Error Notification: (note that the original file has an incorrect SPI in record 2)

```
<?xml version="1.0" encoding="UTF-8"?>
<D1-PayloadErrorNotif>
  <externalSenderId>DM</externalSenderId>
  <fileName>dm-usage-1.xml-2011-07-13-14-21-58-789</fileName>
  <errors>
    <error>
      <errorCode>12802</errorCode>
      <errorDateTime>2011-07-13-14.22.03</errorDateTime>
      <errorDetail>
        <![CDATA[<con:ValidationFailureDetail xmlns:con="http://www.bea.com/
wli/sb/stages/transform/config"><con:message>Invalid decimal value: unexpected
char '45'</con:message><con:xmlLocation><spi>900-broken</spi></
con:xmlLocation></con:ValidationFailureDetail>]]>
      </errorDetail>
      <messageParameterList>
        <pameterSequence>1</pameterSequence>
        <messageParameterType>MSG</messageParameterType>
        <messageParameterValue>BEA-382505</messageParameterValue>
        <pameterSequence>2</pameterSequence>
        <messageParameterType>MSG</messageParameterType>
        <messageParameterValue>OSB Validate action failed validation</
messageParameterValue>
        <pameterSequence>3</pameterSequence>
        <messageParameterType>MSG</messageParameterType>

        <messageParameterValue>PipelinePairNode11PipelinePairNode1_requeststage1request-pipeline</messageParameterValue>
      </messageParameterList>
    </error>
  </errors>
</D1-PayloadErrorNotif>
```

Payload Summary:

```
<?xml version="1.0" encoding="UTF-8"?>
<D1-PayloadSummary>
  <externalSenderId>DM</externalSenderId>
  <fileName>dm-usage-1.xml-2011-07-13-14-21-58-789</fileName>
  <middlewareStatistics>
    <fileHeaderInformation>
      <creationDateTime>2011-07-12-18.07.34</creationDateTime>
      <fileSize>3232</fileSize>
    </fileHeaderInformation>
    <processingInformation>
      <receivedDateTime>2011-07-13-14.21.58</receivedDateTime>
      <processedDateTime>2011-07-13-14.22.03</processedDateTime>
      <processTime>6</processTime>
      <totalPayloadTransactions>3</totalPayloadTransactions>
      <totalPayloadTransactionsProcessed>2</totalPayloadTransactionsProcessed>
      <totalPayloadTransactionsinError>1</totalPayloadTransactionsinError>
```

```
<totalTransactions>3</totalTransactions>
<totalIMDTransactionsProcessed>2</totalIMDTransactionsProcessed>
<totalIMDTransactionsinError>1</totalIMDTransactionsinError>
<totalEventTransactionsProcessed>0</totalEventTransactionsProcessed>
<totalEventTransactionsinError>0</totalEventTransactionsinError>
</processingInformation>
</middlewareStatistics>
</D1-PayloadSummary>
```

Opening and Deploying the OSB Processes

This section describes how to open and deploy the sample OSB process provided with the generic adapter. This process can serve as an example when creating OSB processes for your implementation.

Prerequisites

- The Oracle Utilities Smart Grid Gateway application must be installed on an Oracle WebLogic server.
- The Oracle Service Bus must be installed.
- An Oracle Service Bus domain must be created. Refer to Oracle Service Bus documentation for more information about creating OSB domains.
- JMS queues for Initial Measurement Data and Notification messages must be created. Refer to the Oracle WebLogic and Oracle Utilities Application Framework documentation for more information about setting up JMS queues.
- The **spl-d1-osb-2.0.1.jar** file containing the **com.splwg.d1.sgg.osb.common** Java package must be available. This is the jar file used by all vendor-specific adapters, and is installed with the Oracle Utilities Smart Grid Gateway application.
- The **sgg-osb-generic-adapter.zip** file must be downloaded as a part of the Generic Adapter.

Opening and Deploying the Java Package

Opening and deploying the Java package used by the Oracle Utilities Smart Grid Gateway generic adapter OSB processes involves the following steps:

1. Locate the “sgg-osb-generic-adapter.zip” sample project.
2. Extract the contents of the “sgg-osb-generic-adapter.zip” file that contains the sample processes. This file contains:
 - FileParser: folder that contains Java classes used to parse and transform measurement data
 - com.splwg.dm.sgg.osb.configuration.jar
3. Create a JDeveloper project and import the “FileParserGenGeneric.java” and “FileParserGenUsage.java” files (in the FileParser folder) using Oracle JDeveloper. Refer to the Oracle JDeveloper documentation for more information about using JDeveloper.
4. Add the **spl-d1-osb-2.0.1.jar** file to the project's Libraries and Classpath properties. This file is installed with Oracle Utilities Smart Grid Gateway.
5. Deploy the project as a jar file to the OSB domain's lib folder.

Opening and Deploying the OSB Configuration

Opening and deploying the OSB configuration used by the Oracle Utilities Smart Grid Gateway generic adapter OSB processes involves the following steps:

1. Locate the “sgg-osb-generic-adapter.zip” sample project.
2. Extract the contents of the “sgg-osb-generic-adapter.zip” file that contains the sample processes. This file contains:
 - FileParser: folder that contains Java classes used to parse and transform measurement data
 - com.splwg.dm.sgg.osb.configuration.jar
3. Ensure that Oracle Service Bus is running in the domain that is created for the tutorial.

- Using Oracle Service Bus Console import resources via System Administration > Import/Export > Import Resources from the extracted com.splwg.dm.sgg.osb.configuration.jar file. Note that conflicts may occurred during import.

These conflicts are addressed in the following steps.

- Check and change the **JCA Transport Configuration** properties of the **SGG-DM-USAGE-CM/Proxy Services/InboundProxyService** proxy service according to the deployment environment, where:
 - Endpoint Properties/SGG_INPUT_PARSER**: is a name of class that implements file parsing functionality. See a Java project mentioned early.
 - Endpoint Properties/SGG_ERROR_FOLDER**: is a folder where a rejected transaction from incoming file will be placed in case when the parsing or validation is failed.
 - Endpoint Properties/SGG_SP_EXT_REF_ID**: is a value corresponding to the target Service Provider's external reference id. It used in the XML structures related to the IMD & Event Upload Statistics functionality. Also, this value will be placed into the D1-DeviceEventSeeder/externalSenderId element.
 - Activation Spec Properties/IncludeFiles**: is the naming convention that the Oracle File Adapter uses to poll for inbound files.
 - Activation Spec Properties/PhysicalArchiveDirectory**: is a folder where to archive successfully processed files.
 - Activation Spec Properties/PhysicalDirectory**: is an input folder to be polled.

ORACLE Service Bus 11gR1

Welcome, system Connected to : OS

View a Proxy Service (SGG-DM-USAGE-CM/Proxy Services/InboundProxyService)

Last Modified By	system	Description
Last Modified On	7/12/11 5:14 PM	
References	3 Ref(s)	
Referenced By	0	

Configuration Details | Operational Settings | SLA Alert Rules | Policies | Security

Proxy Service Configuration (SGG-DM-USAGE-CM/Proxy Services/InboundProxyService)

General Configuration

Service Type	Web Service - SOAP 1.1 (WSDL/SGG-DM-USAGE-CM/Proxy Services/InboundProxyService)
--------------	--

Transport Configuration

Protocol	JCA
Endpoint URI	jax:file/Adapter
Get All Headers	Yes

JCA Transport Configuration

JCA File	SGG-DM-USAGE-BASE/JCA Bindings/GenericFileProcessor
Adapter Name	File Adapter
Adapter Type	FILE
Endpoint Properties	SGG_INPUT_PARSER = "com.splwg.dm.osb.usage.FileParser.GenericUsage" SGG_ERROR_FOLDER = "c:/Storage/genusage-error" SGG_SP_EXT_REF_ID = "DM"
Always use configuration from JCA file	False

Activation Spec Properties

Operation Name	Read
MaxBatchSize	"5"
PollingFrequency	"30"
UseHeaders	"false"
DeleteFile	"true"
PhysicalArchiveDirectory	"c:/Storage/genusage-arch"
Recursive	"false"
IncludeFiles	".*.xml"
ThreadCount	"10"
PhysicalDirectory	"c:/Storage/genusage"
PipelineValves	"com.splwg.d1.sgg.osb.common.GenericFileProcessor"
MinimumAge	"0"

Connection Mode

Connection Mode	Managed
-----------------	---------

Operation Selection Configuration

- Change the **Endpoint URI** in the **Transport Configuration** section of the **SGG-DM-USAGE-CM/Business Services/DestinationBusinessService** business service according to the deployment environment.

The screenshot shows the Oracle Service Bus 11gR1 console. On the left, the Project Explorer shows the hierarchy: SGG-DM-USAGE-BASE > SGG-DM-USAGE-CM > Business Services > DestinationBusinessService. The main pane displays the 'Edit a Business Service - Summary' page for 'DestinationBusinessService'. The 'General Configuration' section shows the Service Name as 'DestinationBusinessService' and the Service Type as 'Any XML Service'. The 'Transport Configuration' section shows the Protocol as 'jms', Load Balancing Algorithm as 'none', and the Endpoint URI as 'jms://localhost:8900/weblogic.jms.XAConnectionFactory/DestinationQueue-DM'. The 'JMS Transport Configuration' section shows the Destination Type as 'Queue', Use SSL as 'Disabled', Message Type as 'Text', Expiration as '0', Enable Message Persistence as 'True', JNDI Timeout as '0', and Response Queues as 'None'.

General Configuration	
Service Name	DestinationBusinessService
Description	JMS transport based business service containing all settings about destination queue where the initial load
Service Type	Any XML Service

Transport Configuration	
Protocol	jms
Load Balancing Algorithm	none
Endpoint URI	jms://localhost:8900/weblogic.jms.XAConnectionFactory/DestinationQueue-DM
Retry Count	0
Retry Iteration Interval	30
Retry Application Errors	Yes

JMS Transport Configuration	
Destination Type	Queue
Use SSL	Disabled
Message Type	Text
Expiration	0
Enable Message Persistence	True
JNDI Timeout	0
Response Queues	None

- Change the **Endpoint URI** in **Transport Configuration** section of the **SGG-DM-USAGE-CM/Business Services/NotificationBusinessService** business service according to the deployment environment.

The screenshot shows the Oracle Service Bus 11gR1 console. On the left, the Project Explorer shows the hierarchy: SGG-DM-USAGE-BASE > SGG-DM-USAGE-CM > Business Services > NotificationBusinessService. The main pane displays the 'Edit a Business Service - Summary' page for 'NotificationBusinessService'. The 'General Configuration' section shows the Service Name as 'NotificationBusinessService' and the Service Type as 'Any XML Service'. The 'Transport Configuration' section shows the Protocol as 'jms', Load Balancing Algorithm as 'none', and the Endpoint URI as 'jms://localhost:8900/weblogic.jms.XAConnectionFactory/ory/NotificationQueue-DM'. The 'JMS Transport Configuration' section shows the Destination Type as 'Queue', Use SSL as 'Disabled', Message Type as 'Text', Expiration as '0', Enable Message Persistence as 'True', and JNDI Timeout as '0'.

General Configuration	
Service Name	NotificationBusinessService
Description	JMS transport based business service containing all settings about destination queue where the
Service Type	Any XML Service

Transport Configuration	
Protocol	jms
Load Balancing Algorithm	none
Endpoint URI	jms://localhost:8900/weblogic.jms.XAConnectionFactory/ory/NotificationQueue-DM
Retry Count	0
Retry Iteration Interval	30
Retry Application Errors	Yes

JMS Transport Configuration	
Destination Type	Queue
Use SSL	Disabled
Message Type	Text
Expiration	0
Enable Message Persistence	True
JNDI Timeout	0

8. Update the credentials stored in the **SGG-DM-USAGE-CM/Service Accounts/DestinationServiceAccount** service account according to the deployment environment:

The screenshot shows the Oracle Service Bus 11gR1 Change Center interface. On the left is a Project Explorer showing a tree structure with 'SGG-DM-USAGE-CM' expanded, containing 'Business Services', 'Proxy Services', 'Service Accounts', and 'XQueries'. The main area displays the 'View Service Account - SGG-DM-USAGE-CM/Service Accounts/DestinationServiceAccount' configuration page. It includes a table with metadata (Last Modified By: system, Last Modified On: 2/14/11 1:51 PM, References: 0, Referenced By: 2 Ref(s)) and a 'General Configuration' section. The 'Static Credentials' section has fields for 'User Name' (ouaf_user) and 'Password' (masked with asterisks). Buttons for 'Back' and 'Edit' are at the bottom.

View Service Account - SGG-DM-USAGE-CM/Service Accounts/DestinationServiceAccount		
Last Modified By	system	Description The DestinationServiceAccount contains creden
Last Modified On	2/14/11 1:51 PM	
References	0	
Referenced By	2 Ref(s)	
General Configuration		
Type	Static	
Static Credentials		
User Name	ouaf_user	
Password	[Masked]	
Back		Edit

Customizing the Generic Adapter Oracle Service Bus (OSB) Processes

You can also customize the OSB processes provided with the generic adapter (or create your own processes) to work with a specific head-end system. At a high-level, this process:

- Polls a directory or queue for incoming payloads containing one or more usage readings (or device events)
- Parses the payloads into individual usage readings (or device events)
- Transforms the individual usage readings (or device events) into the XML format required to import into Oracle Utilities Smart Grid Gateway / Meter Data Management.
- Deposits the individual usage readings (or device events) into JMS queues being monitored by the OUAF and Oracle Utilities Smart Grid Gateway / Meter Data Management, which in turn invokes XAI Inbound Services to import the usage readings (or device events).

Specifics concerning how to implement this functionality using OSB depend largely on the requirements of the implementation. At a high-level:

- Oracle Service Bus provides facilities for polling for new files and inserting records into JMS queues.
- File parsing can be performed through the use of a Java implementation (as in the example provided with the generic adapter).
- Data transformation can be performed through the use of a Java implementation (as in the example provided with the generic adapter) or through data transformation facilities within OSB.
- Parsing and/or transforming payloads using Java involves creating a Java class that implements the FileParser interface and parses and transforms the payloads as appropriate. The Java classes provided with the generic adapter can be used as models for any new Java classes developed for a custom adapter.
- The OSB projects provided with the generic adapter (SGG-DM-USAGE-BASE and SGG-DM-USAGE-CM) can be used as models for any new OSB projects developed for a custom adapter.

Refer to the Oracle Service Bus documentation for more information about using OSB for file processing, parsing, and transformation.

FileParser Interface

This section provides the source code of the FileParser interface used by the “FileParserGenGeneric” Java class delivered with the Java package described earlier in this section.

```
package com.splwg.d1.sgg.osb.common;

import java.io.IOException;
import java.io.InputStream;

/** <p>
 * FileParser component is responsible for processing the input
 * stream and returning a "plain" XML containing all data from
 * incoming transaction in a shape expected by the following
 * Oracle Service Bus message flow.
 * </p>
 */
public interface FileParser {
    /**
     * Returns the next message in "plain" XML format
     * or null if there are no more messages
     * @return next message as InputStream
     */
    InputStream getNextStream() throws IOException;

    /**
     * Returns the current position in the currently
     * parsed source input stream
     * @return the position in source input stream after
     * last incoming row was read and parsed
     */
    long getPosition();

    /**
     * Sets source stream
     * @param input the Input Stream that should be parsed
     */
    void setInputStream(InputStream input);

    /**
     * Sets "owner" FileProcessor
     * @param owner the Instance of FileProcessor that will
     * be notified about current processing via call to
     * utility methods (saveInvalidRow and setParseFinishNote)
     */
    void setProcessor(FileProcessor owner);

    /**
     * Sets the position from which the parsing of source
     * stream should be started.
     * It is necessary if last time the processing of
     * current file has been interrupted
     * @param position the starting position
     */
    void setStartPosition(long position);

    /**
     * Returns the byte array that contains a portion of an incoming
     * file that most recently has been read and converted to "Plain" XML.
     * It should be in the same format as incoming file. The header and
     * tail records, if there are, should be included and updated according
     * to the content
     */
    byte[] getCurrentInputPortion();

    /**
     * Returns type of currently generated transaction - USAGE or EVENT
     */
}
```

```
FileProcessor.TransactionType getCurrentTransactionType();

/**
 * Returns the number of successfully generated transactions of given type
 */
int getTransactionCount(FileProcessor.TransactionType type);

/**
 * Returns the number of expected transactions in input stream
 * Returns -1 if information not available
 */
int getTotalExpected();
}
```

FileParser2 Interface

This section provides the source code of the FileParser2 interface used by the “FileParserGenGeneric” Java class delivered with the Java package described earlier in this section.

```
package com.splwg.d1.sgg.osb.common;

import java.util.Date;

public interface FileParser2 extends FileParser {
    /**
     * FileProcessor usage method
     * Returns the File creation date and time stamp stored
     * in the file
     * Returns null if information not available
     */
    Date getCreationDateTime();
}
```

Configuring and Customizing the Generic Adapter BPEL Processes

This section outlines how to configure and customize the sample Oracle Business Process Execution Language (BPEL) processes provided with the generic adapter to work with your implementation, including:

- **Configuration and Customization Process Overview**
- **Editing Configuration Files**
- **Customizing the BPEL Processes**
- **Packaging and Deploying the Processes**
- **Configuring BPEL Security**

Configuration and Customization Process Overview

A set of generic adapter BPEL processes are provided in the “SGG-CM-SOA.ZIP” file. This is a sample BPEL project that can be edited and changed to meet the requirements of your head-end system and implementation.

Customizing the Oracle Utilities Smart Grid Gateway generic adapter BPEL processes involves the following steps:

1. Locate the “SGG-CM-SOA.zip” sample project.
2. Extract the contents of the “SGG-CM-SOA.zip” file that contains the sample BPEL processes.
3. Open the “SGG-DM-SOA.jws” file (JDeveloper Workspace file), using Oracle JDeveloper. Refer to the Oracle JDeveloper documentation for more information about using JDeveloper.
4. Edit the configuration and build files to suite your implementation and application server environment using JDeveloper (or a text or XML editor). See **Editing Configuration Files** on page D-28 for more details about the configuration and build files that must be edited.
5. To change the XSL files used in transforming the messages from the Oracle Utilities Smart Grid Gateway standard format into the format used by your head-end system, change the XSL files referenced in the BPEL project using JDeveloper. See **Customizing the BPEL Processes** on page D-32 for more details about changing.

Note: This step is not needed when setting up the Oracle Utilities Smart Grid Gateway generic adapter for testing. This step is only required when customizing the adapter for use with a head-end system.

6. Make any other customizations to the BPEL processes using JDeveloper. This allows you to add steps in the BPEL process to address your specific business requirements. See **Customizing the BPEL Processes** on page D-32 for more details about changing.

Note: This step is not needed when setting up the Oracle Utilities Smart Grid Gateway generic adapter for testing. This step is only required when customizing the adapter for use with a head-end system.

7. Package the processes with your customizations. This builds a jar file that represents the BPEL process that will need to be deployed. See **Packaging and Deploying the Processes** on page D-33 for more details about packaging the BPEL project for deployment.
8. Deploy the jar file to the SOA Suite application server. See **Packaging and Deploying the Processes** on page D-33 for more details about deploying the project.

The sections below provide more information about these steps.

Editing Configuration Files

This section outlines the changes that need to be made in various configuration and build files to enable the application servers used in your implementation to communicate with each other, and to support the specifics of your head-end system. These files are also used in packaging and deploying the processes.

Server Definitions

In order for your adapter to work properly, you must set up the application servers that run the components used by the adapter communicate with each other as follows:

- The Oracle Utilities Smart Grid Gateway application server must be able to send and receive messages to and from the SOA Suite application server.
- The SOA Suite application server must be able to send and receive messages to and from the head-end system application server, and the Oracle Utilities Smart Grid Gateway application server
- The head-end system application server must be able to receive and send messages from and to the SOA Suite application server.

The types of servers and the tokens that need to be replaced for each are listed below. Further below are the specific configuration files that must be modified, as well as the locations in each file that must be modified. Note that a port is listed, but may not be necessary depending on the type of installation. Also note that the SOA Server will have to have the partition name defined that is used.

- **SOA Server:** This is the application server running the SOA Suite components, including Oracle Service Bus (OSB) and Oracle Business Process Execution Language (BPEL). The SOA server is referenced using the following tokens:
 - {SOA_Server}: the URL where the SOA server has been installed
 - {SOA_Port}: the port used by the SOA server
 - {SOA_Partition}: the partition used by the SOA server. There are different partitions for the different adapters used by Oracle Utilities Smart Grid Gateway.
- **XAI (OUAF) Server:** This is the application server running the Oracle Utilities Smart Grid Gateway (including the Oracle Utilities Application Framework, or OUAF) software, including the XML Application Integration (XAI) components. The XAI (OUAF) server is referenced using the following tokens:
 - {XAI_Server}: the URL where Oracle Utilities Smart Grid Gateway and the XAI components have been installed
 - {XAI_Port}: the port used by the Oracle Utilities Smart Grid Gateway application
- **AMI Head-end Server:** This is the application server running the head-end system software. The AMI head-end server is referenced using the following tokens:
 - {AMI_HeadEnd_Server}: the URL for either the actual head-end system, or the URL to an emulator test harness being used (if applicable). For example, the generic adapter includes a soapUI configuration that can be used to emulate a head-end system for testing purposes.
 - {AMI_HeadEnd_Port}: the port used by the head-end system or emulator test harness

In addition, credentials for the WebLogic server must be specified in the build.properties file in the ..\SGG-CM-SOA\deploy folder:

- {WebLogic_UserID}: the user ID used to connect to the WebLogic server
- {WebLogic_Password}: the password for the {WebLogic_UserID} used to connect to the WebLogic server

Deploy Folder Files

Edit the following files in the “deploy” folder as noted.

File: build.properties

Path: ..\SGG-CM-SOA\deploy

line 13: soa.serverURL=http://{SOA_Server}:{SOA_Port}

line 19: wls.user={WebLogic_UserID}

line 21: wls.password={WebLogic_Password}

ConnectDisconnect Folder Files

Edit the following files in the “ConnectDisconnect” folder as noted.

File: composite.xml

Path: ..\SGG-CM-SOA\ConnectDisconnect

line 63: <property name="bpel.preference.CB_CDCallbackEndpoint">http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/ConnectDisconnect/CB_CD</property>

line 76: <property name="endpointURI">http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/ProcessCallout/ProcessCallout</property>

line 83: <property name="endpointURI">http://{AMI_HeadEnd_Server}:{AMI_HeadEnd_Port}/mockCD_CBSOap</property>

line 96: <property name="endpointURI" type="xs:string" many="false">http://{XAI_Server}:{XAI_Port}/ouaf{XAI_Server}:{XAI_Port}/ouafNotification</property>

File: ConnectDisconnect_cfgplan.xml

Path: ..\SGG-CM-SOA\ConnectDisconnect

line 80: <replace>http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/ConnectDisconnect/CB_CD</replace>

line 102: <replace>http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/ProcessCallout/ProcessCallout</replace>

line 116: <replace>http://{AMI_HeadEnd_Server}:{AMI_HeadEnd_Port}/mockCD_CBSOap</replace>

line 133: <replace> http://{XAI_Server}:{XAI_Port}/ouaf{XAI_Server}:{XAI_Port}/ouafNotification</replace>

File: DM_ConDisconStChgNotification.wsdl

Path: ..\SGG-CM-SOA\ConnectDisconnect

line 99: <soap:address location="http://{XAI_Server}:{XAI_Port}/ouaf/xaiserver/DM-ConDisconStChgNotification"/>

File: ProcessCallout.wsdl

Path: ..\SGG-CM-SOA\ConnectDisconnect

line 361: <soap:address location="http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/ProcessCallout/ProcessCallout"/>

OnDemandRead Folder Files

Edit the following files in the “OnDemandRead” folder as noted.

File: composite.xml

Path: C:\SGG-CM-SOA\OnDemandRead

```
line 62: <property name="bpel.preference.CB_MRCallbackEndpoint">http://
{SOA_Server}:{SOA_Port}/soa-infra/services/DM/OnDemandRead/CB_MR</property>
```

```
line 71: <property name="endpointURI">http://{SOA_Server}:{SOA_Port}/soa-infra/
services/DM/ProcessCallout/ProcessCallout</property>
```

```
line 95: <property name="endpointURI">http://{XAI_Server}:{XAI_Port}/
ouaf{XAI_Server}:{XAI_Port}/ouafNotification</property>
```

File: DM_ReadingChanged.wsdl

Path: C:\SGG-CM-SOA\OnDemandRead

```
line 273: <soap:address location="http://{XAI_Server}:{XAI_Port}/ouaf/
xaiserver/DM-ReadingChangedNotification"/>
```

File: DM_ReadingChangedNotification.wsdl

Path: C:\SGG-CM-SOA\OnDemandRead

```
line 273: <soap:address location="http://{XAI_Server}:{XAI_Port}/ouaf/
xaiserver/DM-ReadingChangedNotification"/>
```

File: OnDemandRead_cfgplan.xml

Path: C:\SGG-CM-SOA\OnDemandRead

```
line 73: <replace>http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/
ProcessCallout/ProcessCallout</replace>
```

```
line 88: <replace>http://{XAI_Server}:{XAI_Port}/ouaf{XAI_Server}:{XAI_Port}/
ouafNotification</replace>
```

```
line 103: <replace>http://{AMI_HeadEnd_Server}:{AMI_HeadEnd_Port}/
mockMR_CBSOap</replace>
```

```
line 114: <replace>http://{AMI_HeadEnd_Server}:{AMI_HeadEnd_Port}/
mockMR_CBSOap</replace>
```

File: ProcessCallout.wsdl

Path: C:\SGG-CM-SOA\OnDemandRead

```
line 389: <soap:address location="http://{SOA_Server}:{SOA_Port}/soa-infra/
services/DM/ProcessCallout/ProcessCallout"/>
```

Test Folder Files

Edit the following files in the “test” folder as noted.

File: DM_ConDisconStChgNotification.wsdl

Path: ..\SGG-CM-SOA\test

```
line 99: <soap:address location="http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/
xaiserver/DM-ConDisconStChgNotification"/>
```

File: SGG-CM-SOA-soapui-project.xml

Path: ..\SGG-CM-SOA\test

```
line 660: <con:endpoint>http://{SOA_Server}:{SOA_Port}/soa-infra/services/
${#TestSuite#Partition}/ConnectDisconnect/CB_CD</con:endpoint>
```

```
line 2296: <con:endpoint>http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/xaiserver/
DM-ConDisconStChgNotification</con:endpoint>
```

```
line 2305: <con:endpoint>http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/xaiserver/
DM-ConDisconStChgNotification</con:endpoint>

line 2379: <con:endpoint>http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/xaiserver/
DM-ReadingChangedNotification</con:endpoint>

line 2386: <con:endpoint>http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/xaiserver/
DM-ReadingChangedNotification</con:endpoint>

line 2927: <con:value>{SOA_Server}</con:value>

line 2931: <con:value>{SOA_Port}</con:value>

line 2935: <con:value>{SOA_Partition}</con:value>

line 3008: <con:value>{SOA_Server}</con:value>

line 3012: <con:value>{SOA_Port}</con:value>

line 3016: <con:value>{SOA_Partition}</con:value>

line 3043: <con:value>{SOA_Server}</con:value>

line 3047: <con:value>{SOA_Port}</con:value>

line 3059: <con:value>{XAI_Server}</con:value>

line 3063: <con:value>{XAI_Port}</con:value>

line 3246: <con:value>{SOA_Server}</con:value>

line 3250: <con:value>{SOA_Port}</con:value>

line 3254: <con:value>{SOA_Partition}</con:value>

line 3285: <con:value>http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/
ConnectDisconnect/CB_CD</con:value>

line 3361: <con:endpoint>http://{SOA_Server}:{SOA_Port}/soa-infra/services/DM/
ConnectDisconnect/CB_CD</con:endpoint>

line 3409: <con:value>http://{SOA_Server}:{SOA_Port}/soa-infra/services/
${#TestSuite#Partition}/OnDemandRead/CB_MR</con:value>

line 3506: <con:endpoint>http://{SOA_Server}:{SOA_Port}/soa-infra/services/
${#TestSuite#Partition}/OnDemandRead/CB_MR</con:endpoint>

line 3550: <con:value>{SOA_Server}</con:value>

line 3554: <con:value>{SOA_Port}</con:value>

line 3558: <con:value>{SOA_Partition}</con:value>

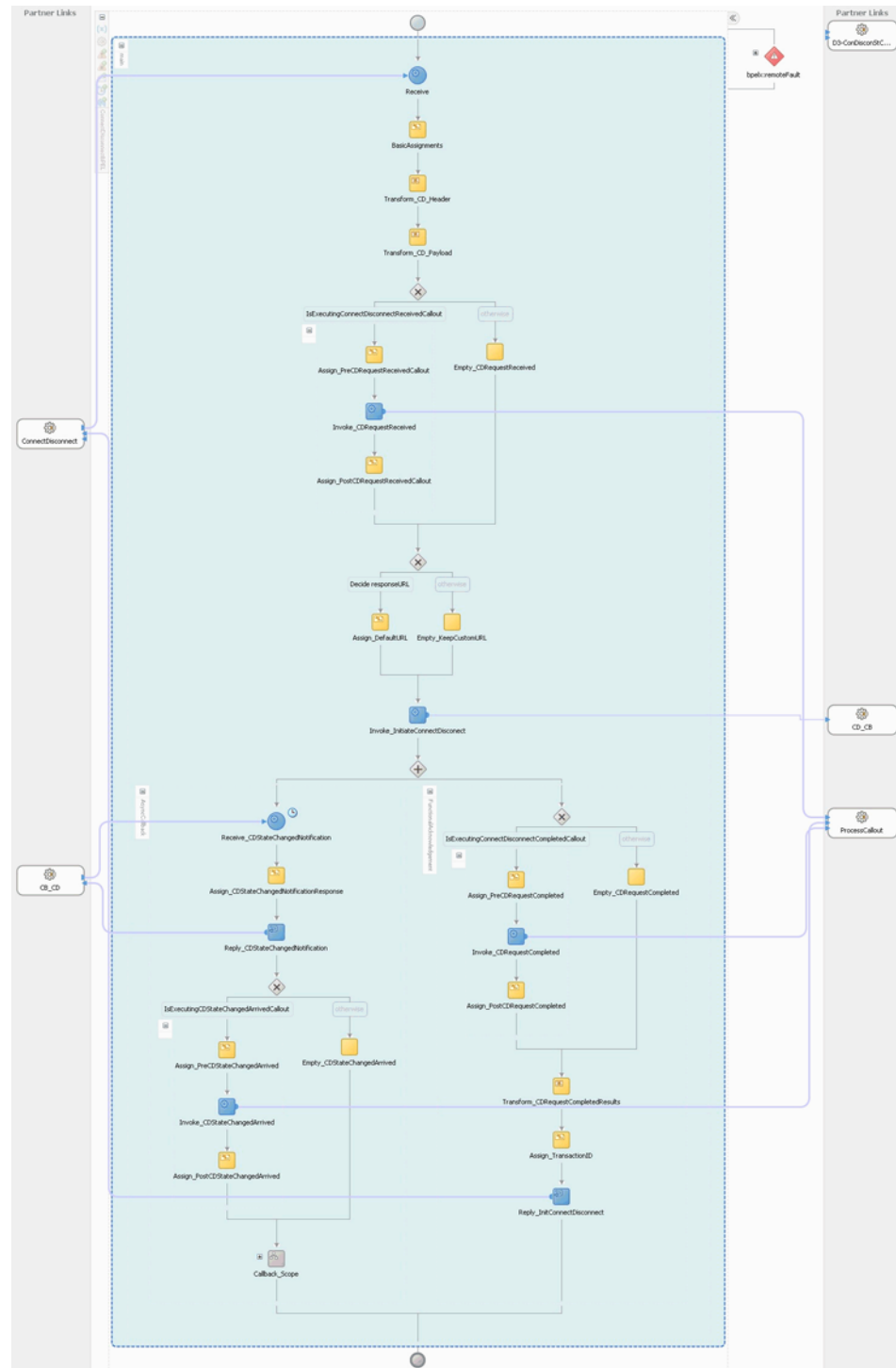
line 3581: <con:endpoint>http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/xaiserver/
DM-ReadingChangedNotification</con:endpoint>

line 3662: <con:endpoint>http://{XAI_Server}:{XAI_Port}/ouaf/XAIApp/xaiserver/
DM-ReadingChangedNotification</con:endpoint>
```

Customizing the BPEL Processes

This section provides an overview of how the sample BPEL processes provided with the Oracle Utilities Smart Grid Gateway generic adapter can be customized to meet specific business requirements, such as using custom XSL transformations and adding/editing the steps involved in the process.

The diagram below illustrates the BPEL process used by the generic adapter for the Remote Connect and Remote Disconnect commands. (Note that these commands are both supported through the same Initiate Connect Disconnect process).



Using Custom XSL Files

This process references XSL files to transform the messages from the Oracle Utilities Smart Grid Gateway standard format into the format used by the “generic” head-end system (based on the Multispeak protocol) when sending messages to the head-end system, and from the head-end system format into the Oracle Utilities Smart Grid Gateway standard format when sending messages back to Oracle Utilities Smart Grid Gateway. These XSL file references can be changed to custom XSL files that transform the messages into and from the format used by your head-end system. Refer to Oracle BPEL and Oracle JDeveloper documentation for more information about referencing XSL files within a BPEL process.

Making Changes to the Process

In addition to referencing custom XSL files for transformation, you can also add additional steps to the process to meet the specific business requirements of your implementation, or make other changes to the existing steps. Refer to Oracle BPEL and Oracle JDeveloper documentation for more information about adding and/or changing steps within BPEL process.

Note: Any changes made to this process for a particular AMI vendor and head-end system needs to be modeled based on the head-end system/AMI vendor’s business processes, including configuring the required request/response transactions required for a specific command (such a remote connect).

Packaging and Deploying the Processes

After you’ve updated the project to specify your server definitions and any required customizations, you can package and deploy your processes.

Use the following procedure to package your project:

1. Confirm that the correct SOA server properties, including username and password for the WebLogic Server, are set in the “build.properties” file in the “SGG-CM-SOA\deploy” folder.
2. Execute the “package.bat” file in the “SGG-CM-SOA\deploy” folder to trigger the build and packaging process. This process builds the jar files that will be deployed to the SOA Suite server.

Note: The details for the files to be processed are in the “package.xml” file. Messages related to the progress of the package build are saved in a log file and errors are saved in an error file.

Once the packaging is successful, you can deploy the jar files onto the SOA server.

Use the following procedure to deploy the project:

1. Execute the “deploy.bat” file in the “SGG-CM-SOA\deploy” folder to trigger the deployment process. This process deploys the jar files onto the SOA Suite server.

Note: Messages related to the progress of the deployment are saved in a log file and errors are saved in an error file.

A Note About Files and Locations

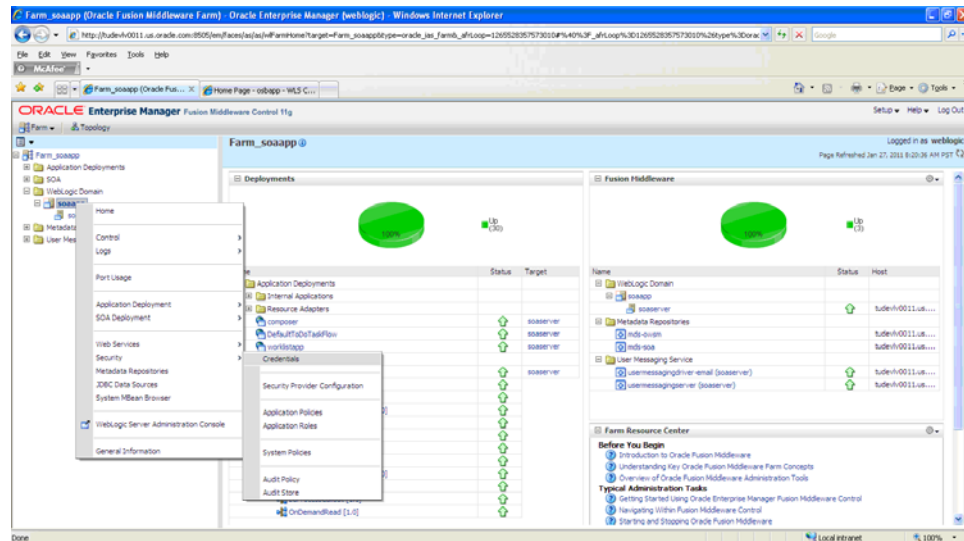
Once deployed, the jar files used by the generic adapter are the same ones used for the Oracle Utilities Smart Grid Gateway Adapter for Landis+Gyr, except that the file names are pre-fixed with “DM” instead of “D3.” The same jar files are created, but they are deployed to the DM partition on the SOA server instead of the LG partition.

Configuring BPEL Security

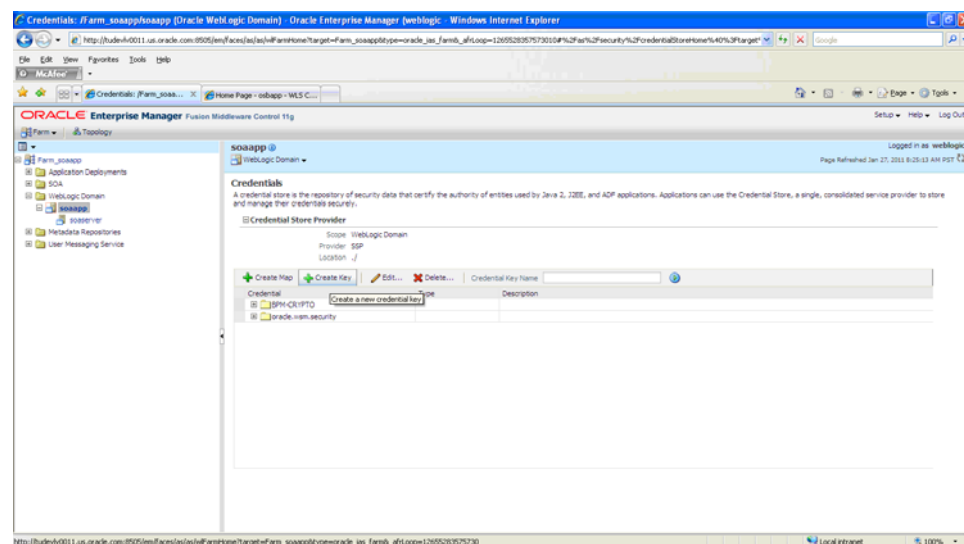
Security to allow the BPEL process to communicate with Oracle Utilities Smart Grid Gateway is configured in the WebLogic Enterprise Manager.

Use the following procedure to enable security:

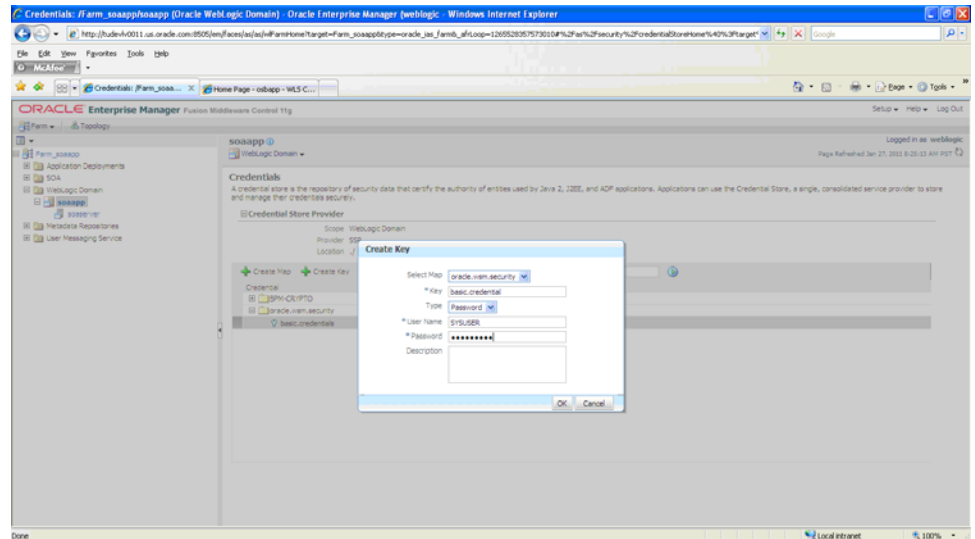
1. Expand the WebLogic domain, right-click on the domain, and choose **Security > Credentials**.



The Credentials screen opens.

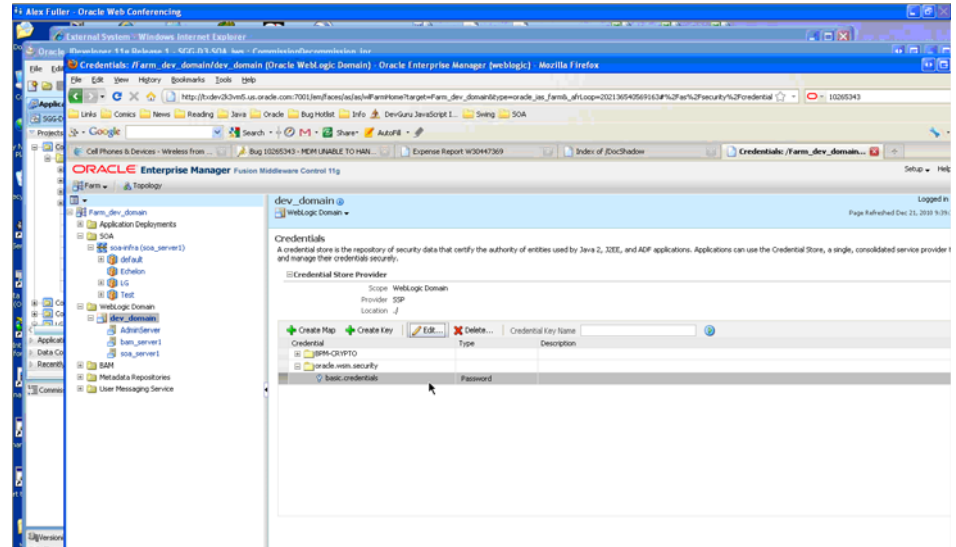


2. Click **Create Key**.
3. On the Create Key dialog:
 - Select a map from the **Select Map** drop-down list.
 - Enter a key (basic.credentials) in the **Key** field.
 - Enter a user name and password in the **User Name** and **Password** fields.



4. Click **OK**.

The key you created will appear on the Credentials screen.



Emulating a Head-End System

The head-end system accessed by the Oracle Utilities Smart Grid Gateway generic adapter can be emulated using tools that can receive and send web service calls. This allows you to test the generic adapter to make sure your server definitions are correct and that your application servers can communicate with each other as needed.

One such tool that can be used for this is soapUI, an open source tool available from SourceForge.net. Note that soapUI is not required to use Oracle Utilities Smart Grid Gateway and it is not provided as part of the generic adapter. However, the generic adapter does contain sample configurations that can be imported into soapUI for testing purposes.

A sample generic adapter emulator is provided in the “SGG_DEMO_EMULATOR.7z” file.

Use the following procedure to set up a generic adapter emulator using soapUI:

1. Download and install soapUI from <http://sourceforge.net/projects/soapui/>.
1. Locate the “SGG_DEMO_EMULATOR.7z” file.
2. Extract the contents of the “SGG_DEMO_EMULATOR.7z” file that contains the soapUI generic adapter emulator.
3. Open the soapUI application.
4. Right-click Projects and select “Import Projects.”
5. Navigate to and select the “SGG-DM-SOA-soapui-project.xml” project file from the “..\SGG_DEMO_EMULATOR\test” folder.

Once you have opened the generic adapter emulator project, you can start the MockService that will emulate the head-end system for Remote Connect and Remote Disconnect commands.

Use the following procedure to start the MockService:

1. Right-click “CD_CBSOap MockService” (connect disconnect) under “Projects\SGG-DM-SOA-DEMO” in the soapUI Navigator.
2. Select “Restart.”

Once the MockService has been started, you can use the soapUI emulator for to test the generic adapter.

Once installed and started, soapUI (or the tool of your choice), will receive web service calls from Oracle Utilities Smart Grid Gateway when Remote Connect or Remote Disconnect commands are initiated, and then send back valid responses. Without a head end simulation tool of this type, all command requests will remain in the “Communication in Progress” state as they will never receive a response.

Message Driven Bean Configuration

This section describes the steps for configuring the Message Driven Bean (MDB) feature of Oracle Utilities Smart Grid Gateway to listen to inbound JMS messages. This feature is used when importing usage reading and device events from head-end systems. This section includes:

- **JMS Configuration**
- **Message Driven Bean Configuration**

JMS Configuration

JMS configuration involves setting up JMS queues which will receive inbound usage readings and device events. The JMS queues need to be created first on the application server where the OSB component is deployed. This server is referred to as remote server in the sections below. In the following section the JMS queue on the remote server is assumed to be created with the name **DestinationQueueWatch-CM**.

Note: The JMS changes described in the following sections are not persistent during patches or upgrades. They will need to be re-created after applying any patches or upgrades to Oracle Utilities Smart Grid Gateway. It is recommended to keep a backup of the \$SPLBASE/splapp/config.xml file.

Create a new JMS module

Log in to the Oracle Utilities Smart Grid Gateway Weblogic console and create a JMS Module with an appropriate name. Specify the following values for this JMS module:

- **Name:** the name of JMS module. For example, JMSModule-CM
- **Target:** the name of the target server where the Oracle Utilities Smart Grid Gateway application is running. This should be specified as myserver.

Create a Foreign JMS server

Create a Foreign JMS server under the JMS module created in the above step. Specify the following values for this foreign JMS server:

- **Name:** Name of the foreign server. For example, JMSFAServer-CM
- **Target:** This should be specified as myserver
- **JNDI Initial Context Factory:** This should be specified as weblogic.jndi.WLInitialContextFactory
- **JNDI Connection URL:** The URL of the server where OSB is deployed. For example: t3://osbserver:7001
- **JNDI Properties Credential:** Password for the OSB server user.
- **JNDI Properties:** The java.naming.security.principal additional property should be specified and set to the OSB server user. For example, java.naming.security.principal=weblogic

Create a Foreign Destination

Create a Foreign destination for each remote queue. Specify the following values for this foreign destination:

- **Name:** Name of foreign destination. For instance, DestinationQueue-CM
- **Local JNDI Name:** Local JNDI name for the foreign JMS queue. For example, ForeignDestinationQueue-CM
- **Remote JNDI Name:** JNDI name of the queue on the remote server. For example, DestinationQueueWatch-CM

Create a Remote Connection Factory

Create a remote connection factory for the foreign JMS server. Specify the following values for this remote connection factory:

- **Name:** Name of remote connection factory. For example, DestinationQueueConnectionFactory-CM
- **Local JNDI Name:** Local JNDI name for the Remote Connection Factory. For example, ForeignDestinationQueueConnectionFactory-CM
- **Remote JNDI Name:** JNDI name of the JMS Connection Factory on the remote server. For example, weblogic.jms.XAConnectionFactory

Message Driven Bean Configuration

Configuration of message driven beans (MDB) involved modifying the **ejb-jar.xml** and **ejb-weblogic-jar.xml** configuration files delivered with Oracle Utilities Smart Grid Gateway. It is recommended that instead of modifying these files directly you create “Customer Modification” (CM) versions of these files to make changes to these configuration files. This ensures that your modifications are not overwritten by future application patches.

The following section describes the changes required in the CM files for configuring the MDBs to read from the foreign JMS queues set up in the steps above (see **JMS Configuration** on page D-37). This requires creating the following files under \$SPLEBASE/templates -

- cm_ejb-jar.xml.wls.jms_1.include
- cm_ejb-jar.xml.wls.jms_2.include
- cm_weblogic-ejb-jar.xml.jms.include.

Note: After making these changes the initialSetup script needs to be run and Oracle Utilities Smart Grid Gateway application needs to be redeployed. However the initialSetup script will overwrite the JMS configuration changes made in the steps above. So it is recommended to keep a backup of the \$SPLEBASE/splapp/config.xml file before running this script.

Changes to cm_ejb-jar.xml.wls.jms_1.include

Below is an example of the cm_ejb-jar.xml.wls.jms_1.include file:

```
<message-driven>
  <description>MDB for DestinationQueue-CM</description>
  <display-name>DestinationQueueWatcher-CM</display-name>
  <ejb-name>DestinationQueueWatch-CM</ejb-name>
  <ejb-class>com.splwg.ejb.mdb.MessageProcessor</ejb-class>
  <messaging-type>javax.jms.MessageListener</messaging-type>
  <transaction-type>Bean</transaction-type>
  <message-destination-type>javax.jms.Queue</message-destination-type>
</message-driven>
```

The values specified in the above file include the following:

- **ejb-name:** This is the name of the MDB.

Changes to cm_ejb-jar.xml.wls.jms_2.include

Below is an example of the cm_ejb-jar.xml.wls.jms_2.include file:

```
<assembly-descriptor>
  <security-role>
    <role-name>cisusers</role-name>
  </security-role>
  <container-transaction>
    <method>
      <ejb-name>DestinationQueueWatch-CM</ejb-name>
      <method-name>onMessage</method-name>
    </method>
  </container-transaction>
</assembly-descriptor>
```

```

    <trans-attribute>NotSupported</trans-attribute>
  </container-transaction>
</assembly-descriptor>

```

The values specified in the above file include the following:

- **ejb-name:** This is the name of the MDB.

Changes to cm_weblogic-ejb-jar.xml.jms.include

Below is an example of the cm_weblogic-ejb-jar.xml.jms.include file:

```

<weblogic-enterprise-bean>
  <ejb-name>DestinationQueueWatch-CM</ejb-name>
  <message-driven-descriptor>
    <pool>
      <max-beans-in-free-pool>5</max-beans-in-free-pool>
      <initial-beans-in-free-pool>1</initial-beans-in-free-pool>
    </pool>
    <destination-jndi-name>ForeignDestinationQueue-CM</destination-jndi-name>
    <connection-factory-jndi-name>ForeignConnectionFactory-CM</connection-
factory-jndi-name>
  </message-driven-descriptor>
</weblogic-enterprise-bean>

```

The values specified in the above file include the following:

- **ejb-name:** This should be the name of the MDB as specified in ejb-jar.xml.
- **destination-jndi-name:** This should be the JNDI name of the foreign destination as provided in JMS module ' Foreign server ' Foreign destination ' Local JNDI name.
- **connection-factory-jndi-name:** This should be the JNDI name of the connection factory as provided in JMS module ' Foreign server ' Remote Connection Factory ' Local JNDI name.

Index

M

Meter Data Management Overview 1-1, 4-1, 10-1, D-1

S

setup sequence 3-7

