Oracle® Communications Billing and Revenue Management

Provisioning Services

Release 15.2
G35862-01
January 2026

ORACLE

Oracle Communications Billing and Revenue Management Provisioning Services, Release 15.2

G35862-01

Contents

About This Content

This guide describes how to configure multiple levels of service for the same service and to rate each level differently by using provisioning in Oracle Communications Billing and Revenue Management (BRM).

Audience

This guide is intended for developers who customize the provisioning framework in BRM and for pricing analysts and others who configure pricing components in Pricing Design Center.

1

About Provisioning

Learn how to set up services framework for provisioning telco services in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About Service Provisioning

You use service provisioning to make changes on the network based on changes in service status in BRM or changes to the product offerings that customers purchase. For example:

- If a customer purchases a charge offer that includes voice mail, you can send commands to create the voice mailbox on the network when the charge offer is purchased.
- If a customer changes the status of a service, you can send a command to the network to activate or deactivate the service.
- You can activate or deactivate service extensions and supplementary services, such as call forwarding.

You can use one of the following methods to define provisioning tags, depending on what the tag is for:

- For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions. See "Provisioning Telco Services by Using Services Framework Manager".
- For non-telco services, the provisioning tag framework is the recommended method of creating provisioning tags. However, you can also use Services Framework Manager to provision non-telco services. See "Provisioning Non-Telco Services by Using Services Framework".

2

About Services Framework Manager Provisioning

Learn how Oracle Communications Billing and Revenue Management (BRM) Services Framework provisioning works.

Topics in this document:

> **Note**
>
> - For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions. See "Provisioning Telco Services by Using Services Framework Manager".
> - For non-telco services, the provisioning tag framework is the recommended method of creating provisioning tags. However, you can also use Services Framework Manager to provision non-telco services. See "Provisioning Non-Telco Services by Using Services Framework".

About Provisioning Services

Service provisioning enables you to assign and change network resources on external networks when the status of a prepaid service or device changes; for example, when a service is activated or deactivated.

When the status changes, service provisioning does the following:

1. Creates a service order, which provides information about the service and the provisioning action required.
2. Sends the service order to the external network through the Provisioning Data Manager (DM), **dm_prov_telco**.
3. Updates the service order's status.
4. Updates the status of the service and device associated with the service order.

Provisioning occurs whenever customer data on the network must be changed. For example:

- Activating, changing, and inactivating GSM services and supplementary services. (During service activation and inactivation, phone numbers and SIM cards can also be provisioned or unprovisioned.)
- Pre-provisioning SIM cards.

- Changing SIM cards, phone numbers, and other service attributes, such as call forwarding.

BRM can provision both telco services and non-telco services.

- All telco services (**/service/telco/*** objects) are automatically recognized and provisioned.
- Non-telco services are provisioned only if they are listed in the **/config/service_framework/permitted_service_types** object. You specify the non-telco services that Services Framework supports by using the "load_pin_service_framework_permitted_service_types" utility. See "Specifying the Non-Telco Services Supported by Services Framework".

About Service Orders

Services Framework provisioning uses *service orders* to alert external provisioning agents about account and service changes. The service order includes details about what changed, including the following:

- POID of the device, service, or profile object that changed.
- The current status of the service order. See "About Service Order Status".
- The provisioning action to perform: activation, deactivation, suspension, reactivation, change, or ignore.
- Specified fields from the **/service** or **/device** object. See "About Adding Details to the Service Order".

The service order is sent to the external provisioning agent and stored in the BRM database in an **/event/provisioning/service_order/telco/***service_name* object.

About Creating Service Orders for Supplementary Services

Supplementary services are added to or removed from service objects (**/service/telco/***service_name*) when charge offers and bundles are purchased or canceled. Services Framework service management initially sets the service status to NEW. If supplementary services are not included with the service, Services Framework provisioning adds them to the service order.

> **Note**
>
> Some features may be part of two or more charge offers purchased separately. In this case, the one added later takes the status of the existing one and is not included in the service order.

You can localize the status flags for the service and the status for the supplementary service by using the **load_localized_strings** utility, which updates the **/strings** storable class based on the localized configuration file. See "Creating a Localized Version of BRM" in *BRM Developer's Guide*.

About Creating Service Orders for Devices

Device service order creation is controlled by the device status stored in the configuration object (**/config/device_state**). If the configuration object lists the device status, a service order is created. The action associated with this service order is the one specified in the configuration for this device status entry.

The **/event/device/associate** and **/event/device/disassociate** events occur during an update services action. These events retrieve the device information needed to create the service order from the device objects. Only events for device types listed in the **/config/telco/provisioning** object are processed. The fields read from the device objects are specified in the **/config/telco/provisioning** object for this type of device.

When a device is changed, the **/event/device/state** event is generated as part of the device state change. The PCM_OP_TCF_CREATE_SVC_ORDER opcode reads the configuration object (**/config/telco/provisioning**) to determine if the device state requires that a service order be created. If so, this opcode creates it as part of the service order event (**/event/provisioning/service_order/telco/***service_name*).

You can customize the PCM_OP_TCF_PROV_POL_CREATE_SVC_ORDER policy opcode to modify certain values in the service order. This opcode calls the opcodes that update the service order.

About Service Order Status

Like services, service orders have different statuses in their lifetimes. Table 2-1 shows the default values for service order status.

Table 2-1 Service Order Status Default Values

Service Order Status	Description
NEW	This is the initial service order state.
READY	A READY service order is ready to be sent to the provisioning agent. The provisioning FMs have all the necessary data to completely fill in the service order and send it to the network provisioning interface.
PROCESSING	After the service order is received by the network provisioning interface, the state is changed to PROCESSING.
COMPLETED	If the service order is successfully processed and the devices are provisioned, the service order state changes to COMPLETED.
FAILED	If there are errors during device provisioning or in the network, the status of the service order is set to FAILED.

About the Provisioning Process

BRM uses event notification to alert Services Framework provisioning that one of the following occurred:

- A service was created or modified.
- A device was associated with or disassociated from a service.
- A device was replaced with a new device.
- A device's state changed.
- A profile was modified.

> **Note**
>
> To configure event notification to alert Services Framework provisioning that other events occurred, see "Setting Up Event Notification for Provisioning".

Services Framework provisioning then performs the following main functions:

- Generates a service order and adds specified details to it. See "About Adding Details to the Service Order".
- Retrieves the service order state transitions for the specified service or device type. See "About Valid Service Order State Transitions".
- Optionally tests the provisioning process by using the Network Simulator. See "Testing Provisioning Using BRM Network Simulator".
- Publishes the service order and, depending on the provisioning mode, either finishes processing or waits for a response from the network provisioning agent. See "About Provisioning Modes".

About Adding Details to the Service Order

Service orders contain information about the service, device, or profile that changed as well as the provisioning action to perform. You can also have other details about the service, device, or profile added to the service order before it is sent to the external provisioning agent. For example, you can add the quality of service (QoS) values and APN names to GSM service orders.

You specify the service, device, or profile object fields to add to the service order in the *BRM_home*/**sys/data/config/pin_telco_provisioning** configuration file. You then load the file into the *provisioning configuration object* (**/config/telco/provisioning** and **/config/telco/provisioning/fieldlist**) by using the "load_pin_telco_provisioning" utility.

During the provisioning process, Services Framework determines the object fields to include in the service order by reading the provisioning configuration object:

- For telco services, the provisioning configuration object is **/config/telco/provisioning/***ServiceType*, where *ServiceType* is the service passed in the input flist. For example, if the service is **/service/telco/gprs/telephony**, the provisioning configuration object is **/config/telco/provisioning/gprs/telephony**.
- For non-telco services, the **/config/service_framework/permitted_service_types** object lists the provisioning configuration object to use.

To configure the service, device, and profile object fields to add to the service order, see "Specifying the Details to Add to the Service Order".

About Valid Service Order State Transitions

Service orders can be set to a NEW, READY, PROCESSING, COMPLETED, or FAILED state. When a service order is first created, it is set to the NEW state by default. The service order can then transition from the NEW state to a list of permitted states that you specify.

For each service order state, you define the valid states to which it can transition. For example, you specify whether service orders can transition from a NEW state to only a READY state or from a NEW state to either a READY state or a PROCESSING state.

You specify the valid state transitions for each service in the *BRM_home*/**sys/data/pin_telco_service_order_state** configuration file. You then load the file into the **/config/telco/service_order_state/*** database object by using the **load_pin_telco_service_order_state** utility.

> **Note**
>
> You can also configure BRM to call an opcode when a service order transitions from one state to another.

To specify the valid state transitions, see "Specifying the Available States for Each Service Order".

About Provisioning Modes

After service orders are published, the provisioning process cannot continue until the network provisioning agent returns a response. You can configure whether Services Framework provisioning sends the service order directly to the agent and waits for a response or publishes the service order to a queue by setting the provisioning mode:

- **Queued provisioning mode**. Services Framework provisions service orders in two separate transactions. In the first transaction, Services Framework generates a service order and queues it for the external provisioning agent. In the second transaction, the external provisioning agent responds that provisioning failed or was successful and then Services Framework updates the service in the BRM database. This is the default provisioning mode.
- **Confirmed provisioning mode**. Services Framework publishes the service order immediately to the external provisioning agent, waits for a response, and updates the service in the BRM database in one transaction. Specifically, Services Framework:
  - Processes the request and converts the service order information into a provisioning payload object in XML format.
  - Sends the service order to the network provisioning agent through a TCP/IP connection.
  - Waits for a response from the network provisioning agent.

    If the wait exceeds the timeout value, the transaction is rolled back. See "Setting a Timeout Value for Requests Sent in Confirmed Mode".

You specify the provisioning mode on a service-by-service basis by using the **pin_service_framework_permitted_service_types.xml** configuration file. You then load the XML file into the BRM database's **/config/service_framework/permitted_service_types** object by using the "load_pin_service_framework_permitted_service_types" utility. See "Specifying the Non-Telco Services Supported by Services Framework".

You can also customize the PCM_OP_TCF_POL_PROV_HANDLE_SVC_ORDER policy opcode to set an event's provisioning mode based on service order attributes. The policy opcode overrides the provisioning mode set in the **/config/service_framework/permitted_service_types** object.

> **Note**
>
> The **/config/service_framework/permitted_service_types** object and the PCM_OP_TCF_POL_PROV_HANDLE_SVC_ORDER policy opcode are used primarily to configure *non-telco* services. However, they can be used to specify the provisioning mode for telco services.

About Delayed Provisioning

GSM service provisioning is triggered according to the charge offer or bundle purchase date or end date:

- If the purchase or end date is not specified, the date is by default the current date. This means that the service is provisioned or unprovisioned as soon as the charge offer or bundle is purchased or canceled.
- If the purchase or end date is in the future, the service is provisioned at the future date. This is known as *delayed provisioning*.

You should use delayed provisioning whenever possible. For example, instead of performing a charge offer upgrade on the same day it is requested, schedule the upgrade for the following day.

The advantage of using delayed provisioning is that provisioning is triggered for only the net difference between the existing supplementary services and the new supplementary services. For example, an account might own a charge offer that includes these supplementary services:

- Voice mail
- Call forwarding

Using delayed activation, you cancel the existing charge offer and add a charge offer that includes these supplementary services:

- Voice mail
- Call blocking

In this case, BRM unprovisions call forwarding, and provisions call blocking. The customer's voice mail configuration, including any existing messages, is unchanged.

3

Provisioning Telco Services by Using Services Framework Manager

Learn how to set up services framework for provisioning telco services in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Setting Up Event Notification for Provisioning

BRM uses event notification to start the Services Framework provisioning process. You specify which notification events trigger provisioning by editing the event notification configuration file and then loading it into the database with the **load_pin_notify** utility.

To configure event notification for provisioning:

1. Open the *BRM_home***/sys/data/config/pin_notify_telco** file in a text editor.
2. If your system has multiple configuration files for event notification, merge them with the **pin_notify_telco** file.
3. Add the following entry for each service or device type that you want to provision:

   ```
   OpcodeNumber    Flag    Event
   ```

   where:

   - *OpcodeNumber* specifies the hard-coded number for an opcode. To find an opcode's number, see the opcode header files (***.h**) in the *BRM_home***/include/ops** directory.
   - *Flag* is the name of the flag to pass to the opcode when it is called by the event notification feature. **0** means no flag is passed.
   - *Event* is the name of the event that triggers the processing of the opcode. You can use any BRM default or custom event defined in your system.

   The default **pin_notify_telco** file includes the following lines, which indicate that PCM_OP_TCF_PROV_CREATE_SVC_ORDER (opcode number 4016), PCM_OP_TCF_PROV_HANDLE_SVC_ORDER (opcode number 4017), and PCM_OP_TCF_PROV_UPDATE_PROV_OBJECT (opcode number 4019) are called whenever these notification events occur:

   ```
   4016    0       /event/notification/service/pre_create
   4016    0       /event/notification/service/create
   4016    0       /event/notification/service/pre_change
   4016    0       /event/notification/service/post_change
   4016    0       /event/device/associate
   4016    0       /event/device/disassociate
   ```

```
4016    0       /event/device/replace
4016    0       /event/notification/profile/pre_modify
4016    0       /event/notification/profile/modify
4016    0       /event/device/state
4017    0       /event/provisioning/service_order/telco
4017    0       /event/provisioning/service_order/telco/gsm
4017    0       /event/provisioning/service_order/telco/gsm/telephony
4017    0       /event/provisioning/service_order/telco/gsm/data
4017    0       /event/provisioning/service_order/telco/gsm/fax
4017    0       /event/provisioning/service_order/telco/gsm/sms
4017    0       /event/provisioning/service_order/telco/gprs
4019    0       /event/provisioning/service_order/telco
4019    0       /event/provisioning/service_order/telco/gsm
4019    0       /event/provisioning/service_order/telco/gsm/telephony
4019    0       /event/provisioning/service_order/telco/gsm/data
4019    0       /event/provisioning/service_order/telco/gsm/fax
4019    0       /event/provisioning/service_order/telco/gsm/sms
4019    0       /event/provisioning/service_order/telco/gprs
```

4. Save and close the file.
5. Load your final event notification list (*pin_notify_file*) into the BRM database by using the **load_pin_notify** utility:

   ```
   load_pin_notify pin_notify_file
   ```

6. Restart the Connection Manager (CM).

For more information, see "Using Event Notification" in *BRM Developer's Guide*.

Specifying the Details to Add to the Service Order

You specify the services and associated devices (if any) to include in a provisioning service order by editing the **pin_telco_provisioning** file. You load the file into the BRM database's **/config/telco/provisioning** object by using the "load_pin_telco_provisioning" utility.

> **Note**
>
> Including associated devices in service order is not the same as pre-provisioning devices, although you use the **pin_telco_provisioning** file for both operations.

To specify the service, device, and profile object fields to add to service orders:

1. Open the *BRM_home***/sys/data/config/pin_telco_provisioning** file in a text editor.
2. Add the following lines for each *service* that you want to provision:

   ```
   Service provisioning info:
   ServiceType, ProvAction,
   Field1,
   Field2,
   Field3
   ```

   where:

   - *ServiceType* specifies the type of service being provisioned.
   - *ProvAction* specifies the provisioning action sent in the service order. The external provisioning system uses this information to determine the appropriate provisioning

action. Use **A** (activate), **C** (close), **D** (deactivate), **I** (ignore), **P** (provisioning), **R** (reactivate), and **S** (suspend).

- *FieldX* specifies the service object fields to include in the service order.

For example, the following lines specify to add the bearer service name, APN name, and QOS profile name to the service order when a GPRS service is being activated. Note that the fields specified are part of the **/service/telco/gprs** schema.

```
Service provisioning info:
/service/telco/gprs, A,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE,
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME
```

3. (Optional) Add the following lines to specify devices associated with the service. Each device you include requires a separate line, separated by commas.

```
Device provisioning info:
DeviceType, ProvAction, Field1, "String1",
DeviceType, ProvAction, Field2,  "String2"
```

where:

- *DeviceType* specifies the type of device associated with the service.
- *ProvAction* specifies the provisioning action for the device objects in the service order. The external provisioning system uses this information to determine the appropriate provisioning action. Use **A** (activate), **C** (close), **D** (deactivate), **I** (ignore), **P** (provisioning), **R** (reactivate), and **S** (suspend).
- *FieldX* specifies the name of a field that contains a device attribute to include in the service order. Field names are replaced by values when you run **pin_telco_provisioning**.
- *StringX* specifies a string used in the service order to identify the attribute specified by the Field value. You can query for the string in the service order.

For example, the following line activates a phone number. The phone number value is included in the service order identified by the MSISDN string.

```
/device/num, A, PIN_FLD_DEVICE_ID, "MSISDN"
```

4. Save and close the **pin_telco_provisioning** file.
5. Use the following command to load the file into the database:

```
load_pin_telco_provisioning pin_telco_provisioning
```

For the complete command syntax, see "load_pin_telco_provisioning".

6. Restart the CM.

To verify that the data was loaded, you can display the **/config** objects by using the Object Browser or use the **robj** command with the **testnap** utility. See "testnap" and "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Specifying the Available States for Each Service Order

You specify the available service order state transitions for each service by editing the **pin_telco_service_order_state** file. You then load the file into the BRM database's **/config/telco/service_order_state/*** object by using the "load_pin_telco_service_order_state" utility.

To specify the state transitions:

1. Open the *BRM_home*/**sys/data/config/pin_telco_service_order_state** file in a text editor.
2. Add the following lines for each service that you want to provision:

   ```
   ServiceType
   StateID: StateType: OpcodeNum: Flags
               NextState:OpcodeNum:Flags
               NextState:OpcodeNum:Flags
               NextState:OpcodeNum:Flags
   ```

   where:

   - *ServiceType* specifies the service being provisioned.
   - *StateID* specifies the starting service order state: NEW (**1**), READY (**2**), PROCESSING (**3**), COMPLETED (**4**), and FAILED (**5**).
   - *StateType* specifies the state type: raw (**0**), init (**1**), normal (**2**), and end (**3**).
   - *NextState* specifies the state to which the service order can be transitioned to from the starting state: NEW (**1**), READY (**2**), PROCESSING (**3**), COMPLETED (**4**), and FAILED (**5**).
   - *OpcodeNum* specifies the opcode to call when the transition is made. To not call an opcode, use **0**.
   - *Flags* specifies the flag to pass to the opcode.

   For example, the following lines specify that service orders can transition from a NEW state to a READY state, from a READY state to a PROCESSING state, and from a PROCESSING state to a COMPLETED or FAILED state:

   ```
   /event/service_order/telco/gsm
   1: 1: 0: 0
         2: 0:0
   2: 2: 0: 0
         3: 0:0
   3: 3: 0: 0
         4: 0:0
         5: 0:0
   ```

3. Save and close the **pin_telco_service_order_state** file.
4. Use the following command to run the **load_pin_telco_service_order_state** utility:

   ```
   load_pin_telco_service_order_state pin_telco_service_order_state
   ```

   For the complete command syntax, see "load_pin_telco_service_order_state".
5. Restart the CM.

To verify that the data was loaded, you can display the **/config** objects by using the Object Browser or use the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Configuring Service Status Change for Device-to-Service Associations

By default, when you associate a device with a service, BRM activates the service, provisions the associated supplementary features, updates the status of the service and the associated supplementary features, and generates a service order that contains the service status and the status of the associated supplementary features. When you disassociate a device from a service, BRM deactivates the service, unprovisions the associated supplementary features,

updates the status of the service and the associated supplementary features, and updates the service order.

You can configure BRM to not update the status of a service when you associate a device with a service or disassociate a device from a service.

To enable this feature, run the **pin_bus_params** utility to change the **RestrictDeviceToServiceStatePropagation** business parameter. For information about this utility, see "pin_bus_params" in *BRM Developer's Guide*.

To configure service status change for device-to-service associations:

1. Go to *BRM_home***/sys/data/config**.
2. Use the following command to create an editable XML file from the **TCF** instance of the **/config/business_params** object:

   ```
   pin_bus_params -r BusParamsTCF bus_params_TCF.xml
   ```

   This command creates an XML file named **bus_params_TCF.xml.out** in your current directory. If you do not want this file in your current directory, specify the path as part of the file name.
3. In **bus_params_TCF.xml.out**, set **RestrictDeviceToServiceStatePropagation** to **enabled**:

   ```
   <RestrictDeviceToServiceStatePropagation>enabled</
   RestrictDeviceToServiceStatePropagation>
   ```

   > ⚠ **Caution**
   >
   > BRM uses the XML in this file to overwrite the existing instance of the **/config/business_params** object. If you delete or modify any other parameters in the file, these changes affect the associated aspects of the BRM configuration.

4. Save and exit the file.
5. Rename the **bus_params_TCF.xml.out** file to **bus_params_TCF.xml**.
6. Use the following command to load your changes into the **/config/business_params** object:

   ```
   pin_bus_params bus_params_TCF.xml
   ```

   You should run this command from the *BRM_home***/sys/data/config** directory, which includes support files used by the utility. To run it from a different directory, see "pin_bus_params" in *BRM Developer's Guide*.
7. Read the object with the **testnap** utility or the Object Browser to verify that all fields are correct.

   For general instructions on using **testnap** , see "Using the testnap Utility to Test BRM" in *BRM Developer's Guide*. For information on how to use Object Browser, see "Reading Objects" in *BRM Developer's Guide*.
8. Stop and restart the CM.

   For more information, see "Starting and Stopping the BRM System" in *BRM System Administrator's Guide*.

4

Provisioning Non-Telco Services by Using Services Framework

Learn how to set up services framework for provisioning non-telco services in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

> **Note**
>
> - For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions. See "Provisioning Telco Services by Using Services Framework Manager".
> - For non-telco services, the provisioning tag framework is the recommended method for creating provisioning tags. However, you can also use Services Framework Manager to provision non-telco services. See "Provisioning Non-Telco Services by Using Services Framework".

Specifying the Non-Telco Services Supported by Services Framework

You specify the non-telco services that Services Framework supports by editing the **pin_service_framework_permitted_service_types.xml** configuration file. You then load the XML file into the BRM database's **/config/service_framework/permitted_service_types** object using the "load_pin_service_framework_permitted_service_types" utility.

The **pin_service_framework_permitted_service_types.xml** file specifies the following for each supported service:

- The provisioning configuration object, which defines the fields to include in a service order.
- The service order configuration object, which specifies the service order state transitions.
- The provisioning mode: Queued or Confirmed.

> **Note**
>
> You use the XML file primarily to configure your *non-telco* services, but you can use it to specify the provisioning mode for telco services. To do this, list only the telco service and the provisioning mode.

To specify the supported non-telco services, perform these tasks:

1. Open the *BRM_home*/**sys/data/config/pin_service_framework_permitted_service_types.xml** file in an XML editor.
2. For each non-telco service supported by Services Framework, add the following entries:
   a. Specify the supported non-telco service on the **ServiceType** line. For example, replace *ServiceType* with **/service/cable**.

      ```
      <ServiceType>ServiceType</ServiceType>
      ```

   b. Specify the provisioning configuration object to use on the **ConfigTypeProvisioningDetails** line. For example, replace *ProvDetails* with **/config/telco/provisioning/cable**.

      ```
      <ConfigTypeProvisioningDetails>ProvDetails</ConfigTypeProvisioningDetails>
      ```

      For information, see "About Adding Details to the Service Order".

   c. Specify the service order configuration object to use on the **ConfigTypeServiceOrderState** line. For example, replace *SOstate* with **/config/telco/service_order_state/cable**.

      ```
      <ConfigTypeServiceOrderState>SOstate</ConfigTypeServiceOrderState>
      ```

      For information, see "About Valid Service Order State Transitions".

   d. Specify the provisioning mode on the **ProvisioningMode** line. Replace *Value* with **0** for Queued mode and **1** for Confirmed mode.

      ```
      <ProvisioningMode>Value</ProvisioningMode>
      ```

3. Save and close the file.
4. Use the following command to load the list of non-telco services into the database:

   ```
   load_pin_service_framework_permitted_service_types
   ```

   See "load_pin_service_framework_permitted_service_types".

   > **Note**
   >
   > This utility requires a configuration (**pin.conf**) file in the directory from which you run the utility. See "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

5. Stop and restart the CM.

To verify that the data loaded successfully, you can display the **/config/service_framework/permitted_service_types** object by using Object Browser or by using the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Customizing the Provisioning Mode Based on Service Order Attributes

By default, Services Framework assigns provisioning modes to events based on the service. You can customize Services Framework to assign the provisioning mode based on other attributes, such as service order details, by customizing the PCM_OP_TCF_POL_PROV_HANDLE_SVC_ORDER policy opcode. You can also customize the policy opcode to modify service order event details before they are published to the external provisioning agent.

You customize the policy opcode to find events with specific flist fields, assign the appropriate provisioning tag and change service order details, and then return the provisioning mode in the PIN_FLD_MODE output flist field.

Setting a Timeout Value for Requests Sent in Confirmed Mode

When provisioning service orders in Confirmed mode, **dm_prov_telco** waits for a response from the external provisioning agent before activating the service in the BRM database. If the external provisioning agent encounters an error or fails, **dm_prov_telco** might have to wait indefinitely. To prevent this problem, you can configure **dm_prov_telco** to close the connection and roll back the transaction if the wait exceeds a specified amount of time.

To specify a timeout value, perform these tasks:

1. Open the **dm_prov_telco** configuration file (*BRM_home*/**sys/dm_prov_telco/pin.conf**) in a text editor.
2. Set the **prov_timeout** entry to the amount of time, in seconds, that **dm_prov_telco** waits for a response from the external provisioning agent before timing out. For example, to set the timeout value to 30 seconds, enter the following:

   ```
   -dm_provision prov_timeout 30
   ```

   By default, the timeout value is set to **20**. If you specify **0**, **dm_prov_telco** waits an infinite amount of time for the external provisioning agent to respond.
3. Save and close the file.
4. Stop and restart **dm_prov_telco**.

5

Setting Up Provisioning for GPRS Services

Learn how to set up provisioning for GPRS services in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Creating Provisioning Tags for GPRS Services

To implement service extensions, such as bearer services, APNs, QoS values, and extended rating attributes (ERAs) for GPRS services, you define provisioning tags. You then use PDC to include provisioning tags in charge offers. A tag becomes available to an individual account and service when a charge offer containing the tag is purchased. This is also known as charge offer–level provisioning.

To define provisioning tags for GPRS services, you modify the **pin_telco_tags_gprs** file. You then load the file into the BRM database's **/config/telco/gprs** and **/config/account_era** objects by using the **load_pin_telco_tags** utility.

To specify provisioning tags for GPRS services, perform the following:

1. Open the *BRM_home*/**sys/data/config/pin_telco_tags_gprs** file in a text editor.
2. If necessary, edit the **pin_telco_tags_gprs** file. The sample file includes these entries:

```
provisioning_tag    "/config/telco/gprs "    "Data Premium"    "Data Service"  "y"
service_extn       "BEARER_SERVICE"    "BS 70"
service_extn       "APN"               "apn.example.com"
service_extn       "QOS"               "Platinum"
service_era        "FRIENDS_FAMILY"    12 13 "n"
service_era        "HOME_CELL"         16 17 "y"
service_era        "HOME_REGION"       14 15 "y"
service_era        "SERVICELEVEL"      10 11 "n"
service_era        "RATEPLAN"          18 19 "n"
```

> **Note**
>
> List the APN and QoS entries in order so that the PIN_FLD_APN_ARRAY is populated by taking the APN name along with its associated QoS. In addition, the QoS entry is optional for each APN.

3. Save the **pin_telco_tags_gprs** file.
4. Use the following command to run the **load_pin_telco_tags** utility:

   ```
   load_pin_telco_tags pin_telco_tags_gprs
   ```

   For the complete command syntax, see "load_pin_telco_tags".

5. Restart the Connection Manager (CM).

To verify that the account ERAs were loaded, you can display the **/config** objects by using the Object Browser or use the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Specifying the Provisioning Configuration for GPRS Services

The provisioning configuration data includes information about the fields required for creating a service order for GPRS services. You specify the configuration in the GPRS provisioning configuration file (**pin_telco_provisioning_gprs**) and then load the file in the BRM database's **/config/telco/provisioning** object by using the "load_pin_telco_provisioning" utility.

To specify the provisioning configuration information for GPRS services, perform the following:

1. Open the *BRM_home***/sys/data/config/pin_telco_provisioning_gprs** file in a text editor.
2. If necessary, edit the **pin_telco_provisioning_gprs** file. The default file includes these entries:

```
Service provisioning info:
/service/telco/gprs, A,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME

Service provisioning info:
/service/telco/gprs, C,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME

Service provisioning info:
/service/telco/gprs, D,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME

Service provisioning info:
/service/telco/gprs, R,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME

Service provisioning info:
/service/telco/gprs, S,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME

Service provisioning info:
/service/telco/gprs, I,
PIN_FLD_GPRS_INFO.PIN_FLD_BEARER_SERVICE
PIN_FLD_APN_ARRAY[*].PIN_FLD_APN,
PIN_FLD_APN_ARRAY[*].PIN_FLD_QOS_PROFILE_NAME
```

   For more information on how to edit the file, see "load_pin_telco_provisioning".
3. Save the **pin_telco_provisioning_gprs** file.
4. Use the following command to run the **load_config_provisioning_tags** utility:

```
load_pin_telco_provisioning pin_telco_provisioning_gprs
```

For the complete command syntax, see "load_pin_telco_provisioning".

5. Restart the CM.

To verify that the account ERAs were loaded, you can display the **/config** objects by using the Object Browser or use the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Specifying the Available States for Each GPRS Service Order

You specify the available states for GPRS service orders by editing the **pin_telco_service_order_state_gprs** file. You then load the file into the BRM database's **/config/telco/service_order_state/gprs** object by using the **load_pin_telco_service_order_state** utility.

See "About Service Order Status" for more information.

To specify the available states, perform the following:

1. Open the *BRM_home***/sys/data/config/pin_telco_service_order_state_gprs** file in a text editor.
2. If necessary, edit the entries to support your business needs. The default entries are shown below:

```
1: 1: 0: 0
      2: 0:0
      3: 0:0
      4: 0:0
      5: 0:0
# READY -> PROCESSING
2: 2: 0: 0
      1: 0: 0
      2: 0: 0
      3: 0: 0
      4: 0: 0
      5: 0: 0
# PROCESSING -> FAILED or COMPLETED
3: 2: 0: 0
      1: 0: 0
      2: 0: 0
      3: 0: 0
      4: 0: 0
      5: 0: 0
# Completed Provisioning -> Completed Provisioning (terminating state)
4: 3: 0: 0
      1: 0: 0
      2: 0: 0
      3: 0: 0
      4: 0: 0
      5: 0: 0
# Failed Provisioning -> Failed Provisioning (terminating state)
5: 3: 0: 0
      1: 0: 0
      2: 0: 0
      3: 0: 0
      4: 0: 0
      5: 0: 0
```

For more information about editing the input file, see "Specifying the Available States for Each Service Order".

3. Save and close the **pin_telco_service_order_state_gprs** file.
4. Use the following command to run the **load_pin_telco_service_order_state** utility:

   ```
   load_pin_telco_service_order_state pin_telco_service_order_state_gprs
   ```

   For the complete command syntax, see "load_pin_telco_service_order_state".
5. Restart the CM.

To verify that the account ERAs were loaded, you can display the ***/config*** objects by using the Object Browser or use the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

6

Creating Provisioning Tags

Learn how to create provisioning tags in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

> **Note**
>
> - For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions. See "Creating Provisioning Tags".
> - For non-telco services, the provisioning tag framework is the recommended method of creating provisioning tags. However, you can also use Services Framework Manager to provision non-telco services. See "Provisioning Non-Telco Services by Using Services Framework".

About Provisioning Tags

You use service provisioning to make changes on the network based on changes in service status in BRM or changes to the product offerings that customers purchase. For example:

- If a customer purchases a charge offer that includes voice mail, you can send commands to create the voice mailbox on the network when the charge offer is purchased.
- If a customer changes the status of a service, you can send a command to the network to activate or deactivate the service.
- You can activate or deactivate service extensions and supplementary services, such as call forwarding.

In the example in Table 6-1, the email service is associated with three charge offers. Each charge offer has an associated provisioning tag. Each provisioning tag defines a different service configuration based on mailbox size. When a customer signs up for email service and purchases a package, the mailbox size is configured according to the associated provision tag.

Table 6-1 Service Provisioning Example

Service	Charge Offer	Provisioning Tag	Description
/service/email	Basic Email	Basic	Sets the minimum mailbox size to 0 MB and the maximum mailbox size to 10 MB. It also specifies the path to the mail server.
/service/email	Regular Email	Regular	Sets the minimum mailbox size to 0 MB and the maximum mailbox size to 20 MB. It also specifies the path to the mail server.
/service/email	Power Email	Power	Sets the minimum mailbox size to 0 MB and the maximum mailbox size to 30 MB. It also specifies the path to the mail server.

You can use one of the following methods to define provisioning tags, depending on what the tag is for:

- **Provisioning tag framework**: You create provisioning tags by defining them in an XML file and loading them into a configuration object in BRM. You can create provisioning tags for any service or account.

  You may also need to add the tag to the provisioning policy source file and compile it.

  > **Note**
  >
  > – For telco services, use Services Framework Manager to define provisioning tags in most cases. You must use Services Framework Manager if the provisioning tag creates supplementary services or service extensions.
  >
  > – For non-telco services, the provisioning tag framework is the recommended method of creating provisioning tags.

- **Telco tag text file**: For telco services only, you can create service provisioning tags by defining them in a text file and then loading the file into BRM. You can include supplementary services in the tags.
- **Provisioning policy source file**: You can define charge offer provisioning tags by editing and compiling the policy file, which is the source file for all provisioning operations.

  > **Note**
  >
  > The provisioning tag framework is the preferable method for creating provisioning tags.

Deciding Which Provisioning Tag Method to Use

For telco services, you should create provisioning tags by using one of these methods. The order of preference is shown below.

1. Provisioning tag framework.

2. Telco tag text file.
3. Provisioning policy source file.

For non-telco services, you should create provisioning tags by using one of these methods. The order of preference is shown below:

1. Provisioning tag framework.
2. Provisioning policy source file.

Using the Provisioning Tag Framework

You can create service or account provisioning tags using the provisioning tag framework. This framework stores provisioning tags in the **/config/provisioning_tag** object.

> **Note**
>
> To create most provisioning tags for telco services, use the **load_pin_telco_tags** utility.

You define a provisioning tag by specifying the services to which it applies and the opcodes to run when the charge offer or discount containing the tag is purchased or canceled. You might need to create custom opcodes for some provisioning tags.

To create a provisioning tag using this framework, do the following:

- Configure the provisioning tag in the **pin_config_provisioning_tags.xml** file.
- Load the provisioning tag configuration into the **/config/provisioning_tag** object with the **load_config_provisioning_tags** utility.
- Add the tag to the provisioning policy source file and compile the file. You do this if you plan to include the provisioning tag in a charge offer and if the tag uses a service associated with the __DEFAULT__ provisioning tag.

Configuring Provisioning Tags

To create provisioning tags, you configure the provisioning tags configuration file, **pin_config_provisioning_tags.xml**. This file is located in the *BRM_home*/**sys/data/config** directory.

To define a provisioning tag in this configuration file, you specify the following:

- A unique name for the provisioning tag.

  > **Note**
  >
  > The provisioning tag for a charge offer or discount must be the name of the offer profile with which the charge offer or discount is associated.

- For a service tag, the permitted services.
- For an account tag, **/account**.

- The name and number of each opcode to run when the charge offer or discount containing the provisioning tag is purchased or canceled. These opcodes contain the business logic to perform the actual provisioning, such as creating a profile.
- Parameters that specify the fields to be added to each opcode's input flist and the value for each field.

> **Note**
>
> – Do not remove existing provisioning tags from **pin_config_provisioning_tags.xml** when adding new tags unless you want existing tags removed from the **/config/provisioning_tag** objects in the database.
>
> – When you load **pin_config_provisioning_tags.xml**, all existing instances of **/config/provisioning_tag** are removed from the database, so only the provisioning tags defined in the file when you load it are in the database.

Table 6-2 lists the elements in the **pin_config_provisioning_tags.xml** file, the syntax to use for each element, and a description of how to specify each element. The syntax is based on the default version of the file.

Table 6-2 Elements in pin_config_provisioning_tags XML File

Element	Syntax	Description
BusinessConfiguration	`<BusinessConfiguration xmlns="http://www.portal.com/schemas/BusinessConfig" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig business_configuration.xsd">`	The root element common to all BRM configurations.
ProvisioningTag Configuration	`<ProvisioningTagConfiguration>`	Opens the provisioning tag configuration. Contains the ProvisioningTagList element.
ProvisioningTagList	`<ProvisioningTagList>`	Contains all the provisioning tag definitions.
ProvisioningTag	`<ProvisioningTag name= "__DEFAULT__">`	Contains the definition of a provisioning tag and specifies the tag's name.
PermittedTypes	`<PermittedTypes>/service/ip </PermittedTypes>`	Specifies a service or account valid for the provisioning tag. One or more of these tags can be part of a provisioning tag definition. When the API for getting a list of provisioning tags is called, the array of permitted types is looked up; only provisioning tags applicable to the specified service are returned.

Table 6-2 (Cont.) Elements in pin_config_provisioning_tags XML File

Element	Syntax	Description
OpcodeList	`<OpcodeList>`	Contains the definition of one opcode. The opcodes specified for a provisioning tag contain the business logic required for provisioning when purchasing and canceling a charge offer. A provisioning tag can include multiple opcodes.

OpcodeName	`<OpcodeName>`
	`PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING`
	`</OpcodeName>`
		Specifies the opcode's name.

OpcodeNumber	`<OpcodeNumber>417` `</OpcodeNumber>`	Specifies the hard-coded number for an opcode. To find an opcode's number, see the opcode header files in the *BRM_home*/**include/ops** directory.
OpcodeMode	`<OpcodeMode>0</OpcodeMode>`	Indicates when the opcode should be called, as follows: • **0**: on charge offer or discount purchase • **1**: on charge offer or discount cancellation • **2**: on both purchase and cancellation
OpcodeParamsList	`<OpcodeParamsList>`	Defines a parameter for an opcode. You can have multiple parameters. Each parameter is a name-value pair.
OpcodeParamName	`<OpcodeParamName>PIN_FLD_POID` `</OpcodeParamName>`	Specifies a field name to be added to the input flist. If the field is part of a substruct or array, use a period to separate the substruct or array name and the field name. For example: PIN_FLD_EVENT.PIN_FLD_ACCOUNT_OBJ
OpcodeParamValue	`<OpcodeParamValue>$SERVICE$` `</OpcodeParamValue>`	Specifies the value of the field. For certain values not known until run time, you can use a variable, such as $SERVICE$.

Specifying an Opcode in a Provisioning Tag to Create an ERA

When setting up a configuration for the provisioning tag framework, you can specify one or more opcodes to perform actions, such as creating an ERA. You have the option to set an opcode to run when a charge offer or discount is purchased, canceled, or both. Additionally, you can either use an existing opcode or design a custom one.

If the provisioning tag is designed to create an ERA, you can specify that the PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode runs at both purchase and cancellation time. This opcode creates, modifies, or deletes a profile (**/profile** object). ERAs are stored in profiles.

The actions the PCM_OP_SUBSCRIPTION_PROVISION_ERA opcode takes depend on the value specified for the PIN_FLD_ACTION parameter in the provisioning tag:

- If PIN_FLD_ACTION is set to **Purchase**, the opcode checks if a profile already exists. If the profile does not exist, the opcode calls PCM_OP_CUST_CREATE_PROFILE to create

the profile. If the profile does exist, the opcode calls PCM_OP_CUST_MODIFY_PROFILE to add data passed in from the input flist and to increment the reference counter by 1.

- If PIN_FLD_ACTION is set to **Cancel**, the opcode decrements the reference counter by 1. If the counter is 0, the opcode calls PCM_OP_CUST_DELETE_PROFILE to delete the profile.

For example, you can create a friends and family ERA for a service by calling the PCM_OP_SUBSCRIPTION_PROVISION_ERA at purchase time with the parameters in Table 6-3.

Table 6-3 PCM_OP_SUBSCRIPTION_PROVISION_ERA Parameters

OpcodeParamName	OpcodeParamValue
PIN_FLD_POID	0.0.0.0 /profile/serv_extrating -1
PIN_FLD_ACCOUNT_OBJ	$ACCOUNT$
PIN_FLD_FLAGS	0
PIN_FLD_SERVICE_OBJ	$SERVICE$
PIN_FLD_STR_VAL	12, 13
PIN_FLD_NAME	FRIENDS_FAMILY
PIN_FLD_INHERITED_INFO. PIN_FLD_EXTRATING. PIN_FLD_LABEL	MYFRIENDS

These name-value pairs indicate that an ERA named FRIENDS_FAMILY and an ERA label named MYFRIENDS are created. Because both the account and service POIDs are specified, the opcode creates a service profile (**/profile/serv_extrating**) object. If the service POID was not specified, the opcode would create an account profile (**/profile/acct_extrating**) object.

> **Note**
>
> - PIN_FLD_POID is the POID, in string format, of the object the provisioning tag creates. This is converted to a POID when the opcode runs. If you are using multiple database schemas, the string is converted to the correct schema database number.
> - For a service profile, the POID type is **/profile/serv_extrating**, as in the previous example. For an account profile, the POID type is **/profile/acct_extrating**.
> - PIN_FLD_FLAGS specifies that PCM_OP_SUBSCRIPTION_PROVISION_ERA is called at purchase time.

You must include the opcode twice in a provisioning tag, once with PIN_FLD_FLAGS set to 0 and once with PIN_FLD_FLAGS set to 1, so that it runs both at purchase and cancellation time.

- PIN_FLD_STR_VAL specifies that the profile name and profile description are localized and are stored in the **/string** object under string IDs 12 and 13.
- PIN_FLD_ACCOUNT_OBJ and PIN_FLD_SERVICE_OBJ use variables.

Variables for Parameter Values

You can use the following variables in Table 6-4 to specify certain values available only at the time the opcode is run:

Table 6-4 Run-Time Variables for Parameters

Variable	Description
$ACCOUNT$	Account object POID
$SERVICE$	Service object POID
$PRODUCT$	Charge offer POID
$DISCOUNT$	Discount offer POID
$OFFERING$	POID of the purchased charge offer or discount offer
$PROVTAG$	Provisioning tag

Default Provisioning Tag

The default **/config/provisioning_tag** object contains the __DEFAULT__ provisioning tag. This tag is defined in the default **pin_config_provisioning_tags.xml** file. The tag calls the following opcodes:

- PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING, on charge offer purchase.
- PCM_OP_SUBSCRIPTION_POL_CANCEL_PROD_PROVISIONING, on charge offer cancellation.

If you have customized these opcodes, they set and clear fields in **/service** objects when a charge offer is purchased and canceled. If you have not customized these opcodes, you do not need to use them. You can specify other opcodes in the provisioning tags file to perform necessary actions.

The __DEFAULT__ provisioning tag is always called for specified services when a charge offer is purchased or canceled, so you do not need to include these opcodes in any other provisioning tags you define. See the default **pin_config_provisioning_tags.xml** file for the list of services.

Using Custom Opcodes

You can create custom opcodes to perform actions not supported by existing opcodes and specify the custom opcodes in the provisioning tags configuration file. For example, you can write an opcode to add fields to a **/service** object.

Modifying Provisioning Tags

You use provision tags as the names of offer profiles associated with charge offers and discounts. If you modify a provision tag used as an offer profile, be sure to modify the corresponding offer profile name accordingly.

> **Note**
>
> - Do not remove existing provisioning tags from **pin_config_provisioning_tags.xml** when adding new tags unless you want existing tags removed from the **/config/provisioning_tag** objects in the database.
> - When you load **pin_config_provisioning_tags.xml**, all existing instances of **/config/provisioning_tag** are removed from the database, so only the provisioning tags defined in the file when you load it are in the database.

Loading Provisioning Tag Configurations

After you configure provisioning tags in the **pin_config_provisioning_tags.xml** file, load the tags into the database with the **load_config_provisioning_tags** utility.

> **Note**
>
> The utility that loads provisioning tags into the database overwrites existing provisioning tags. When updating provisioning tags, you cannot load new provisioning tags only. You must load the complete set of provisioning tags each time you run the utility.

To load provisioning tag configurations:

1. Go to the directory in which the **pin_config_provisioning_tags.xml** file is located. The default location is *BRM_home***/sys/data/config**.
2. Use the following command to run the **load_config_provisioning_tags** utility:

   ```
   load_config_provisioning_tags pin_config_provisioning_tags.xml
   ```

   > **Note**
   >
   > - When you run the utility, the**pin_config_provisioning_tags.xml**and **business_configuration.xsd** files must be in the same directory. By default, both files are in *BRM_home***/sys/data/config**.
   > - This utility needs a configuration (**pin.conf**) file in the directory from which you run the utility.

   If you do not run the utility from the directory in which **pin_config_provisioning_tags.xml** is located, include the complete path to the file, for example:

   ```
   load_config_provisioning_tags BRM_home/sys/data/config/
   pin_config_provisioning_tags.xml
   ```

   For more information, see **load_config_provisioning_tags**.
3. Stop and restart the Connection Manager (CM).
4. To verify that the provisioning tags were loaded, display the **/config/provisioning_tag** object by using the Object Browser or the **robj** command with the **testnap** utility.

Modifying and Compiling the Provisioning Policy Source File

If a provisioning tag defined in the provisioning tag framework uses the PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING policy opcode, you must add the tag to the **fm_subscription_pol_provisioning.c** file and recompile the file.

You can also use the PCM_OP_PROV_POL_UPDATE_SVC_ORDER opcode to customize service orders. See *BRM Opcode Guide*.

A provisioning tag uses PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING opcode if it is included in a charge offer and if it uses a service associated with the __DEFAULT__ provisioning tag.

Modifying and compiling the provisioning policy source file enables PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING to handle the tags.

To modify the provisioning policy source file for provisioning tags created using the provisioning tag framework:

1. Open the *BRM_home***/source/sys/fm_subscription_pol/fm_subscription_pol_provisioning.c** file.
2. Add the provisioning tag name to the **service_info** table for the appropriate service.

   For example, to add a provisioning tag called *test* to **/service/ip**, change this code:

   ```
   static char *tags_ip[] = {
     "example",
     NULL    /* MUST BE LAST! */
   };
   ```

   to the following:

   ```
   static char *tags_ip[] = {
   "example",
   "test",
     NULL    /* MUST BE LAST! */
   };
   ```

3. Add code in the plp function to handle the new tag. You do this in the PROVISIONING FUNCTIONS section. The functions are grouped by service.

   For example, to add the provisioning tag *test* to **/service/ip**, change this code:

   ```
   static void
   plp_ip(pcm_context_t *ctxp, poid_t *svc_obj_p, int32 buy,
     char *tag, pin_errbuf_t *ebufp)
   {
     if (strcmp(tag, "example") == 0) {
       plp_example(ctxp, svc_obj_p, buy, tag, ebufp);
     }
     else{
       plp_ssg(ctxp, svc_obj_p, buy, tag, ebufp);
     }
   }
   ```

   to the following:

   ```
   static void
   plp_ip(pcm_context_t *ctxp, poid_t *svc_obj_p, int32 buy,
     char *tag, pin_errbuf_t *ebufp)
   ```

```
{
  if (strcmp(tag, "example") == 0) {
    plp_example(ctxp, svc_obj_p, buy, tag, ebufp);
  }
  else if (strcmp(tag, "test") == 0) {
    /*skip*/
  }else{
    plp_ssg(ctxp, svc_obj_p, buy, tag, ebufp);
  }
}
```

4. Compile and save the file.

Sample Provisioning Tag XML File

Following is the default provisioning tag XML file. This file defines the provisioning tag named __DEFAULT__, which includes several permitted services and two opcodes:

```
<?xml version="1.0" encoding="UTF-8"?>

<BusinessConfiguration
    xmlns="http://www.portal.com/schemas/BusinessConfig"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig
business_configuration.xsd">

    <ProvisioningTagConfiguration>
    <ProvisioningTagList>

        <ProvisioningTag name="__DEFAULT__">

            <PermittedTypes>/service/email</PermittedTypes>
            <PermittedTypes>/service/ip/gprs</PermittedTypes>
            <PermittedTypes>/service/content</PermittedTypes>
            <PermittedTypes>/service/vpdn</PermittedTypes>
            <PermittedTypes>/service/ip</PermittedTypes>
            <PermittedTypes>/service/ip/gprs</PermittedTypes>
            <PermittedTypes>/service/telco</PermittedTypes>
            <PermittedTypes>/service/telco/gsm</PermittedTypes>
            <PermittedTypes>/service/telco/gsm/data</PermittedTypes>
            <PermittedTypes>/service/telco/gsm/fax</PermittedTypes>
            <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
            <PermittedTypes>/service/telco/gsm/telephony</PermittedTypes>

            <OpcodeList>
                <OpcodeName>PCM_OP_SUBSCRIPTION_POL_PURCHASE_PROD_PROVISIONING</
OpcodeName>
                <OpcodeNumber>417</OpcodeNumber>
                <OpcodeMode>0</OpcodeMode>

                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                    <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
                </OpcodeParamsList>
                <OpcodeParamsList>
                    <OpcodeParamName>PIN_FLD_PROVISIONING_TAG</OpcodeParamName>
                    <OpcodeParamValue>$PROVTAG$</OpcodeParamValue>
                </OpcodeParamsList>
            </OpcodeList>

            <OpcodeList>
                <OpcodeName>PCM_OP_SUBSCRIPTION_POL_CANCEL_PROD_PROVISIONING</OpcodeName>
```

```
                    <OpcodeNumber>418</OpcodeNumber>
                    <OpcodeMode>1</OpcodeMode>

                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                        <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
                    </OpcodeParamsList>
                    <OpcodeParamsList>
                        <OpcodeParamName>PIN_FLD_PROVISIONING_TAG</OpcodeParamName>
                        <OpcodeParamValue>$PROVTAG$</OpcodeParamValue>
                    </OpcodeParamsList>
                </OpcodeList>

            </ProvisioningTag>

        </ProvisioningTagList>
        </ProvisioningTagConfiguration>

    </BusinessConfiguration>
```

7

Implementing Provisioning Tags for Telco Services

Learn how to set up provisioning tags for prepaid services, supplementary services, and extended rating attributes (ERAs) in your Oracle Communications Billing and Revenue Management (BRM) system.

Topics in this document:

About Provisioning Tags for Telco Services

To implement supplementary services, service extensions such as bearer services, and extended rating attributes (ERAs) for telco services, you define provisioning tags. You then use PDC to include provisioning tags in charge offers. A tag becomes available to an individual account and service when a charge offer containing the tag is purchased. This is also known as charge offer–level provisioning.

Use the **pin_telco_tags**_*service* file to define provisioning tags for telco services. For example, for GSM services, use **pin_telco_tags_gsm** with the **load_pin_telco_tags** utility. Use this method to create provisioning tags with custom ERAs. See "Defining Provisioning Tags for Telco Services by Using the pin_telco_tags File ".

Use these methods for provisioning tags for telco services, except for the following cases:

- For a provisioning tag that creates an ERA or other type of profile that you want to associate with a discount. Services Framework Manager does not process provisioning tags associated with discounts.
- For a provisioning tag that populates default values to a profile when creating it. Services Framework Manager cannot populate default values to the profiles.

In both of these cases, use the provisioning tag framework to create provisioning tags.

Service-level provisioning tags are stored in service-specific **/config/telco** objects, such as **/config/telco/gsm/telephony**.

You can also use the **pin_telco_tags**_*service* file to create account-level ERAs, which do not depend on a specific service. These ERAs are stored in **/config/account_era** objects.

Examples of Provisioning Tags for Prepaid Services

You can create different types of provisioning tags; for example:

- A provisioning tag for a single bearer service, such as a type of voice service.

- A provisioning tag for one or more supplementary services without a bearer service. A charge offer with this type of provisioning tag is typically included in an add-on package because the customer must have the service before adding the supplementary services.
- A provisioning tag for one or more service-level ERAs. This type of provisioning tag can be used only in an add-on package.
- A provisioning tag can include combinations of a bearer service, supplementary services, and service-level ERAs.

For example, you might include two different provisioning tags for a GSM telephony service charge offer:

- The VoicePremium provisioning tag implements the following:
  - A voice bearer service
  - The Call Forwarding supplementary service
  - The Home Cell Assignment service ERA
- The VoiceFamily provisioning tag implements the following:
  - A voice bearer service
  - The Caller ID supplementary service
  - The Friends and Family service ERA

You might also create charge offers such as these:

- A charge offer that implements a voice bearer service.
- An add-on charge offer that implements Call Forwarding and the Home Cell Assignment ERA.
- An add-on charge offer that implements Caller ID and the Friends and Family ERA.

> **Note**
>
> You cannot directly change the status of supplementary services. Instead, you change the status of the charge offers they were purchased with.
>
> For example, to inactivate a Call Forwarding supplementary service, you inactivate its charge offer. However, when you do so, you inactivate all other charge offers, supplementary services, and service ERAs purchased with the charge offer.
>
> Therefore, you should create charge offers that enable you to manage services after the charge offers are purchased.

About GSM Supplementary Services

GSM supplementary services are features such as call forwarding and call blocking. They are not implemented as BRM services. Instead, they are implemented by using charge offer provisioning.

For example, you might have a charge offer that includes the GSM telephony service and a provisioning tag that implements the call forwarding supplementary service.

> **Note**
>
> Supplementary services can be used only with GSM services.

Value-added services, such as voice mail, are similar to supplementary services. The difference is that value-added services are not part of the GSM network standard.

> **Note**
>
> In this documentation, supplementary services and value-added services are referred to collectively as *supplementary services*.

You include supplementary services in charge offers by using provisioning tags. You can create charge offers that add supplementary services to an existing account (for example, a charge offer that adds call forwarding).

You cannot activate supplementary services in Billing Care. After BRM provisions a supplementary service, a customer usually activates the supplementary service using the telephone keypad. For example, a customer can define a number to use for call forwarding.

A customer's supplementary services are defined in the service objects owned by the customer's account, such as **/service/telco/gsm**.

You define systemwide supplementary services in **/service/telco/gsm** objects. For example, supplementary voice telephony services are defined in **/service/telco/gsm/telephony**.

Defining Provisioning Tags for Telco Services by Using the pin_telco_tags File

You define provisioning tags through the **pin_telco_tags** file for a specific telco service. For example, you use **pin_telco_tags_gsm** for GSM provisioning tags.

You can include service-level ERAs, supplementary services, and service extensions in a provisioning tag defined in a **pin_telco_tags** file.

To define provisioning tags using the **pin_telco_tags** file:

- Configure provisioning tags in the **pin_telco_tags**_*service* file.
- Load the **pin_telco_tags**_*service* file into the BRM database with the **load_pin_telco_tags** utility.

You can also define account-level ERAs in the **pin_telco_tags** file.

To configure a provisioning tag in the **pin_telco_tags** file:

1. Open the **pin_telco_tags**_*service* file. For example, use **pin_telco_tags_gsm** for GSM services.

   The default **pin_telco_tags** files are in *BRM_home*/**sys/data/config**. They include examples and instructions.

2. Use this syntax to add a provisioning tag, entering each value in quotation marks:

```
provisioning_tag  "Class" "ProvTag" "PTDescription" "DelayedProvReqd"
service_extn      "Extension Type Name" "Extension Value"
features          "One Or More Feature Name String Values"
service_era       "ServiceERA" "StringIdERA" "StringIdServiceERADesc" "ProvBool"
"ERALabel"
```

A provisioning tag can be any combination of service extensions, features, and service-level ERAs. You do not need to include all three types of data in a tag.

Table 7-1 describes the provisioning tag syntax.

Table 7-1 Provisioning Tag Syntax

Tag Element	Value	Description
provisioning_tag	Class	The object that stores the tag. For example: **"/config/telco/gsm/telephony"**
provisioning_tag	ProvTag	The name of the provisioning tag. For example: **"DataPremium"**
provisioning_tag	PTDescription	The description of the provisioning tag. For example: **"Premium Data Service"**
provisioning_tag	DelayedProvReqd	Whether the tag is unprovisioned when the charge offer containing the tag is canceled. Possible values are: • **"y"** specifies that cancellation triggers unprovisioning. In most cases, use this setting. • **"n"** specifies that cancellation does not trigger unprovisioning. Use this setting to leave a customer's service configuration unchanged. For example, you might want to leave a voice mailbox intact.
service_extn	Extension Type Name	The type of service extension. For example: **"BEARER_SERVICE"**
service_extn	Extension Value	The code for a GSM bearer service or other service extension. For example: **"B46"** Codes are defined in the GSM specification. You must use the exact code that the network requires.
features	One or More Feature Name String Values	The GSM supplementary services provisioned when this charge offer is purchased. The services are entered as codes in one line. For example: **"CLIP" "CW"** These codes are defined in the GSM specification. You must use the exact code that the network requires. For a list of codes, see "Supported Supplementary Services".
service_era	ServiceERA	The service ERA code. For example: **"FRIENDS_FAMILY"**
service_era	StringIdERA StringIdServiceERADesc	The IDs for the ERA name and description. For example: **"12" "13"** You define a localized name and description for these IDs in the **era_descr.***locale*file.

Table 7-1 (Cont.) Provisioning Tag Syntax

Tag Element	Value	Description
service_era	ProvBool	Whether or not provisioning is required. The possible values are: • **"y"** specifies that provisioning is required. • **"n"** specifies that provisioning is not required.
service_era	ERALabel	The name of a list within the ERA. An ERA can have one or more lists. For example: **"MYFRIENDS"** **Note:** You cannot localize the ERA label. You cannot have duplicate label names associated with the same ERA code.

This example shows a provisioning tag for a telephony charge offer that includes a bearer service, call waiting and voice mailbox supplementary services, and friends and family service ERAs:

```
# Standard Telephony Package
provisioning_tag  "/config/telco/gsm/telephony"  "Voice Standard" "Voice Standard
Service package with Caller ID, Call Waiting, Voice mail and Friends and Family" "y"
service_extn      "BEARER_SERVICE" "T11"
features          "CLIP" "CW" "VMBOX"
service_era       "FRIENDS_FAMILY" 12 13 "n" "MYFRIENDS"
service_era       "FRIENDS_FAMILY" 12 13 "n" "MYFAMILY"
```

Loading the pin_telco_tags File

Run the **load_pin_telco_tags** utility to load the contents of the **pin_telco_tags**_*service* file, such as the **pin_telco_tags_gsm** file, into the BRM database. This utility creates or updates **/config/telco/***service* and **/config/account_era** objects.

> **Note**
>
> - By default, the **load_pin_telco_tags** utility appends telco provisioning tags and account-level ERAs to the BRM database. But if you use the **-x** parameter, this utility overwrites existing telco provisioning tags and account-level ERAs. Do not use the **-x** parameter unless you are sure you want to overwrite existing objects.
> - The **load_pin_telco_tags** utility requires a configuration file. See "Creating Configuration Files for BRM Utilities" in *BRM System Administrator's Guide*.

To load the **pin_telco_tags** file:

1. Edit the **pin_telco_tags**_*service* file to add the custom account-level ERAs. The default **pin_telco_tags_gsm** file in *BRM_home***/sys/data/config** includes examples and instructions.
2. Save the **pin_telco_tags**_*service* file.
3. Use the following command to run the **load_pin_telco_tags** utility:

   load_pin_telco_tags pin_telco_tags_*service*

   For the complete command syntax, see "load_pin_telco_tags".

4. Restart the CM.

To verify that the account ERAs were loaded, you can display the **/config** objects by using the Object Browser or use the **robj** command with the **testnap** utility. See "Reading an Object and Writing Its Contents to a File" in *BRM Developer's Guide*.

Supported Supplementary Services

Table 7-2 shows the supplementary services supported by GSM Manager. You select from these codes when configuring GSM provisioning tags in PDC.

These codes are defined in the GSM provisioning DTD file (**GSM.dtd**), which is used by the GSM provisioning components. If you customize the GSM provisioning components to add supplementary services, you must be sure to use the new codes in the provisioning tags.

Table 7-2 GSM Manager Supported Supplementary Services

Supplementary Service	Code
Advice of charge (charging)	AOCC
Advice of charge (information)	AOCI
Barring all incoming calls	BAIC
Barring incoming calls when roaming outside the home PLMN country	BAICR
Barring all outgoing calls	BAOC
Barring all incoming calls when roaming outside the home PLMN country	BICRO
Barring all outgoing international calls	BOIC
Barring all outgoing international calls except those directed to the HOME PLMN country	BOICXH
Completion of calls to busy subscribers	CCBS
Call deflection	CD
Call forwarding on mobile subscriber busy	CFB
Call forwarding on mobile subscriber not reachable	CFNRC
Call forwarding on mobile subscriber no reply	CFNRY
Call forwarding unconditional	CFU
Calling line identification presentation	CLIP
Calling line identification restriction	CLIR
Name identification	CNAP
Connected line identification presentation	COLP
Connected line identification restriction	COLR
Call waiting	CW
Explicit call transfer	ECT
Call holding	HOLD
Multicall	MC
Enhanced multilevel precedence	MLPP
Multiparty	MPTY

Table 7-2 (Cont.) GSM Manager Supported Supplementary Services

Supplementary Service	Code
Support of private numbering package	SPNP
User-to-user signaling	UUS
Voice/fax mail service	VMBOX

8

Configuring Provisioning for Policy-Driven Charging

Learn how to enable in-flight changes to provisioning in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Configuring Provisioning Tags for Policy-Driven Charging

Configure the balances that BRM should track by creating provisioning tags for your services. Each provisioning tag contains the balance elements valid for that tag, as well as information on the policy attributes used in the provisioning policy for that balance element.

Policy attributes are the configured characteristics associated with a provisioning policy. They contain the information you must provide to the specific opcode that processes changes to the charge offer or discount associated with that provisioning tag (for example, an offer profile threshold breach notification is required, as well as the language in which the subscriber needs the notification).

> **Note**
>
> BRM stores policy attributes as subscriber preferences (**/profile/subscriber_preferences**) objects containing the following elements:
>
> - Name (required)
> - Type (optional)
> - Value (optional)

BRM provides a default provisioning tag in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file located in *BRM_home***/sys/data/config** directory.

The first section of the definition, as shown in Example 8-1, outlines the valid balance elements for the default provisioning tag. It is followed by information on the opcodes to call when the charge offer or discount containing the provisioning tag is purchased or canceled, and the parameters that specify the fields and values to be added to the opcode's input flist.

Collecting Data for Provisioning Tags

Collect the following data for any additional services that BRM should track for policy-driven charging.

- The name for your provisioning tag. This must be the name of the offer profile with which you associate this provisioning configuration.
- Permitted types of services.
- Subscriber preferences for the service stored in the database as **/profile/subscriber_preferences** objects.
- Opcodes to use in association with the provisioning tag.

  You must set up the following for each opcode included in a provisioning tag:

  – The name and number of each opcode to run and when the opcode must be run. These opcodes contain the business logic to perform the provisioning, such as creating a profile.

  – Parameters that specify the fields to add to each opcode's input flist and the value for each field.

This information can now be configured in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file for use with your offer profiles.

Configuring Provisioning Tags for Offer Profiles

To configure provisioning tags for use with your offer profiles:

1. Go to the *BRM_home***/sys/data/config** directory.
2. Open the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file in an XML editor or text editor.

   Tip

   Save a copy of the default configuration file before you make any changes to it.

3. Do one of the following to update this file:
   - Edit the default provisioning tag contained in this file.
   - Add your provisioning tag by placing your definition just below the default provisioning tag (*Platinum*). Include the following for each provisioning tag:
     – Provisioning tag name:

       ```
       <ProvisioningTag name="YourProvisioningTagName">
       ```

       where *YourProvisioningTagName* is the name of your offer profile.

     – Each service valid for the provisioning tag as a **PermittedType** element as shown in Example 8-1.

     – Specifications for the opcodes to call for this provisioning tag as shown in Example 8-1.
4. Save the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file.

Loading Provisioning Tags for Policy-Driven Charging

Use the **load_config_provisioning_tags** utility to load the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file into the BRM database.

To do so:

1. Ensure that the provisioning tags required for policy-driven charging are configured in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** XML file.
2. Go to the directory in which the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file is located. The default location is *BRM_home***/sys/data/config**.
3. Commit the provisioning tags and policy attributes XML file to the BRM database.

   ```
   load_config_provisioning_tags -d -v
   pin_offer_profile_provisioning_tags_policy_attributes.xml
   ```

   where:

   - **-d** creates a log file for debugging purposes.
   - **-v** displays information about successful or failed processing as the utility runs.

   For more information about the utility's syntax and parameters, see "load_config_provisioning_tags" in *BRM Developer's Guide*.
4. To verify that the provisioning tags were loaded, display the **/config/provisioning_tag** object by using the Object Browser or the **robj** command with the **testnap** utility.
5. Stop and restart the Connection Manager (CM).

The provisioning tags and policy attributes information associated with policy-driven charging are now stored in **/config/provisioning_tag** objects in the database.

Default Provisioning Tag for Policy-Driven Charging

Example 8-1 shows the default provisioning tag provided by BRM in the **pin_offer_profile_provisioning_tags_policy_attributes.xml** file located in the *BRM_home***/sys/data/config** directory.

Example 8-1 Default Provisioning Tag Configuration

```
<BusinessConfiguration xsi:schemaLocation="http://www.portal.com/schemas/BusinessConfig
business_configuration.xsd">
<ProvisioningTagConfiguration>
   <ProvisioningTagList>
     <ProvisioningTag name="Platinum">
        <PermittedTypes>/service/email</PermittedTypes>
        <PermittedTypes>/service/ip/gprs</PermittedTypes>
        <PermittedTypes>/service/content</PermittedTypes>
        <PermittedTypes>/service/vpdn</PermittedTypes>
        <PermittedTypes>/service/ip</PermittedTypes>
        <PermittedTypes>/service/fax</PermittedTypes>
        <PermittedTypes>/service/ip/gprs</PermittedTypes>
        <PermittedTypes>/service/telco</PermittedTypes>
        <PermittedTypes>/service/telco/gsm</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/data</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/fax</PermittedTypes>
        <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
```

```
        <PermittedTypes>/service/telco/gsm/sms</PermittedTypes>
        <OpcodeList>
            <OpcodeName>PCM_OP_CUST_SET_SUBSCRIBER_PREFERENCES</OpcodeName>
            <OpcodeNumber>3916</OpcodeNumber>
            <OpcodeMode>0</OpcodeMode>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                <OpcodeParamValue>0.0.0.0 /profile/subscriber_preferences -1</
OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_DELETED_FLAG</OpcodeParamName>
                <OpcodeParamValue>0</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
                <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
                <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
                <OpcodeParamValue>SET_SUBSCRIBER_PREFERENCES</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_NAME</
OpcodeParamName>
                <OpcodeParamValue>Language</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_
VALUE</OpcodeParamName>
                <OpcodeParamValue>English</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>

<OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[0].PIN_FLD_SUBSCRIBER_PREFERENCE_
ID</OpcodeParamName>
                <OpcodeParamValue>1</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_
NAME</OpcodeParamName>
                <OpcodeParamValue>Channel</OpcodeParamValue>
            </OpcodeParamsList>

            <OpcodeParamsList>
                 <OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_
VALUE</OpcodeParamName>
                 <OpcodeParamValue>IVR</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>

<OpcodeParamName>PIN_FLD_SUBSCRIBER_PREFERENCES[1].PIN_FLD_SUBSCRIBER_PREFERENCE_
ID</OpcodeParamName>
                <OpcodeParamValue>2</OpcodeParamValue>
            </OpcodeParamsList>
        </OpcodeList>
        <OpcodeList>
            <OpcodeName>PCM_OP_CUST_SET_SUBSCRIBER_PREFERENCES</OpcodeName>
```

```
            <OpcodeNumber>3916</OpcodeNumber>
            <OpcodeMode>1</OpcodeMode>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_POID</OpcodeParamName>
                <OpcodeParamValue>0.0.0.0 /profile/subscriber_preferences -1</
OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_DELETED_FLAG</OpcodeParamName>
                <OpcodeParamValue>1</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_ACCOUNT_OBJ</OpcodeParamName>
                <OpcodeParamValue>$ACCOUNT$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_SERVICE_OBJ</OpcodeParamName>
                <OpcodeParamValue>$SERVICE$</OpcodeParamValue>
            </OpcodeParamsList>
            <OpcodeParamsList>
                <OpcodeParamName>PIN_FLD_NAME</OpcodeParamName>
                <OpcodeParamValue>DEL_SUBSCRIBER_PREFERENCES</OpcodeParamValue>
            </OpcodeParamsList>
        </OpcodeList>
    </ProvisioningTag>
  </ProvisioningTagList>
</ProvisioningTagConfiguration>
</BusinessConfiguration>
```

9

Enabling In-Flight Changes to Service Orders

Learn how to enable in-flight changes to provisioning in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

Enabling In-Flight Changes to Service Orders

By default, Services Framework provisioning generates only one service order for a particular service at a time. For example, when a customer adds a service, such as GPRS telephony, Services Framework provisioning generates a service order to activate the service. Services Framework does not generate any additional service orders for that GPRS telephony service until the external provisioning agent sends a response and Services Framework updates the GPRS telephony service's status in the BRM database. This prevents the external provisioning agent from overwriting service changes or processing service orders out of order.

You can enable Services Framework provisioning to generate multiple service orders for a particular service before it receives a response from the external provisioning agent and updates the service's status. This enables you to make in-flight changes to a service's provisioning request. For example, one service order could activate a GPRS telephony service, and a second service order could correct the device ID associated with the service.

> **Note**
>
> When Services Framework sends multiple service orders for a particular service to the external provisioning agent, it waits until it has received a response for all requests before updating the service's status in the BRM database.

To enable Services Framework provisioning to generate new service orders for a service that already has a provisioning request in process:

1. Open the CM configuration file (*BRM_home***/sys/cm/pin.conf**) in a text editor.
2. Add the following **support_multiple_so** entry to the file:

   ```
   - fm_tcf support_multiple_so 1
   ```

   where:

   - **0** prohibits Services Framework provisioning from generating new service orders for a service that already has a provisioning request in process. This is the default.
   - **1** allows Services Framework provisioning to generate service orders for services that already have a provisioning request in process.
3. Save and close the file.
4. Restart the CM.

10

Testing Provisioning Using BRM Network Simulator

Learn how to use the Oracle Communications Billing and Revenue Management (BRM) Network Simulator to test your telco implementation.

Topics in this document:

About the Network Simulator

You use the Network Simulator to simulate a network provisioning agent. It does so by receiving and processing XML provisioning payload files from the Provisioning DM.

> **Note**
>
> To use this simulator, the Provisioning DM must be running.

The Network Simulator includes a Perl-based simulator and a Java-based simulator.

The Perl-based simulator (**agent_sim.pl**) retrieves service order information from the BRM database and generates XML provisioning payload files:

- If the $PIN_HOME environment variable is set, **agent_sim.pl** saves the XML file in *BRM_home*/**apps/telco/service_orders** with the name **SvcOrder**_*EventObject_id*.**xml**.
- If the $PIN_HOME environment variable is not set, the XML file is created in the directory where the simulator is launched.

The Java-based simulator (**RunSimulator**) does the following:

1. Processes the XML file according to the command-line arguments.
2. Updates the service order status by calling the PCM_OP_PROV_UPDATE_SVC_ORDER opcode.

Testing Provisioning

To test how a network provisioning agent would process service orders:

1. Write service order details to an XML provisioning payload file by running the **agent_sim.pl** script. See "Writing Service Orders to an XML File".
2. Simulate how the network provisioning agent would process service orders by running the **RunSimulator** utility. See "Simulating How a Network Agent Processes Service Orders".

Writing Service Orders to an XML File

> **Note**
>
> Before running Network Simulator, enable provisioning.

To write service orders to an XML file:

1. Open the CM configuration file (*BRM_home*/**sys/cm/pin.conf**).
2. Set the **agent_return** entry to **1**:

   ```
   - fm_tcf agent_return 1
   ```

3. Ensure that the internal simulator is turned off by setting the **simulate_agent** entry to **0**:

   ```
   - fm_tcf simulate_agent 0
   ```

4. Save the file.
5. Restart the CM.
6. Start the Perl-based simulator:

   ```
   Perl agent_sim.pl &
   ```

7. Start the Provisioning DM:

   ```
   start_dm_prov_telco
   ```

   > **Note**
   >
   > Always start the Perl-based simulator before you start the Provisioning DM.

8. Create a provisioning event.

Simulating How a Network Agent Processes Service Orders

You use BRM's Java-based simulator (**RunSimulator**) to process an XML provisioning payload file that was created with **agent_sim.pl**.

To simulate how a network provisioning agent would process your service orders, run the **RunSimulator** utility with the following commands:

- To process an XML provisioning payload file, run this command:

  ```
  Java RunSimulator -x xml_file
  ```

- To process an XML provisioning payload file and create a service order with its status set to **Active**:

  ```
  Java RunSimulator -s 0 -x xml_file
  ```

  If you include the parameter to change the state of a supplied service order, the utility changes the status of the associated service. For example, when a successful service order is returned, the service, charge offer, and supplementary service (feature) status is set to **Active**.

For more information about the utility's syntax and parameters, see "RunSimulator".

11

About XML Provisioning

Learn how XML files are used for provisioning in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About XML Provisioning

To initiate service provisioning on a carrier network, GSM Manager sends service order information to a third-party network provisioning agent in the form of XML-formatted *provisioning payload* files. The provisioning agent returns the result of the provisioning request to GSM Manager.

The basic network provisioning process:

1. Provisioning Data Manager (DM) (**dm_prov_telco**):
   a. Receives a service order.
   b. Creates an XML provisioning payload file that includes fields specified in the provisioning configuration file (**/config/provisioning/telco**).
   c. Sends it to the network provisioning agent.
2. The Provisioning DM (**dm_prov_telco**) waits for an acknowledgment from the network provisioning agent.

   This is a sample POID field from a service creation response:

   ```
   0.0.0.1 /event/provisioning/service_order/telco/gsm/telephony 18832
   ```
3. The network provisioning agent returns the provisioning result (Success or Failure) to an opcode which updates the service order using the POID field from the response.

   For more information on supported fields, see "Service Order XML DTD".

Sample XML Document

The following is a sample XML provisioning payload file generated for service creation:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
 <order>
         <POID>0.0.10.2 / 0 0</POID>
         <EVENT_OBJ>0.0.0.1 /event/provisioning/service_order/telco/gsm/telephony 18832
0</EVENT_OBJ>
        <SVC_ORDER>
                  <STATUS>1</STATUS>
         </SVC_ORDER>
         <SERVICE_ORDER_INFO elem="0">
                  <ACTION>A</ACTION>
```

```
  <POID>0.0.0.1 /device/sim 13947 1</POID>
  <PARAMS elem="0">
          <VALUE>000000020001152</VALUE>
          <ACTION>I</ACTION>
          <NAME>IMSI</NAME>
  </PARAMS>
  <PARAMS elem="1">
          <VALUE>0000000000200011526</VALUE>
          <ACTION>I</ACTION>
          <NAME>SIM</NAME>
  </PARAMS>
          </SERVICE_ORDER_INFO>
          <SERVICE_ORDER_INFO elem="1">
          <ACTION>A</ACTION>
          <POID>0.0.0.1 /device/num 10105 1</POID>
  <PARAMS elem="0">
          <VALUE>00493451212</VALUE>
          <ACTION>I</ACTION>
          <NAME>MSISDN</NAME>
  </PARAMS>
          </SERVICE_ORDER_INFO>
          <SERVICE_ORDER_INFO elem="2">
          <NAME>MOBTEL</NAME>
         <ACTION>A</ACTION>
         <POID>0.0.0.1 /service/telco/gsm/telephony 18400 8</POID>
         <PARAMS elem="0">
         <VALUE>T00</VALUE>
         <ACTION>I</ACTION>
         <NAME>BEARER_SERVICE</NAME>
</PARAMS>
<PARAMS elem="1">
         <ACTION>A</ACTION>
         <NAME>VMBOX</NAME>
</PARAMS>
<PARAMS elem="2">
         <ACTION>A</ACTION>
         <NAME>CLIP</NAME>
</PARAMS>
<PARAMS elem="3">
         <ACTION>A</ACTION>
         <NAME>CFU</NAME>
</PARAMS>
<PARAMS elem="4">
         <ACTION>A</ACTION>
         <NAME>CW</NAME>
</PARAMS>
<PARAMS elem="5">
         <ACTION>A</ACTION>
         <NAME>HOLD</NAME>
</PARAMS>
<PARAMS elem="6">
         <ACTION>A</ACTION>
         <NAME>CD</NAME>
</PARAMS>
         </SERVICE_ORDER_INFO>
         <SERVICE_ORDER_INFO elem="3">
         <ACTION>A</ACTION>
         <POID>0.0.0.1 /device/sim 13947 3</POID>
<PARAMS elem="0">
         <VALUE>000000020001152</VALUE>
         <ACTION>I</ACTION>
         <NAME>IMSI</NAME>
```

```
                </PARAMS>
                <PARAMS elem="1">
                        <VALUE>0000000002000011526</VALUE>
                        <ACTION>I</ACTION>
                        <NAME>SIM</NAME>
                </PARAMS>
        </SERVICE_ORDER_INFO>
</order>
```

Table 11-1 describes the main fields of the payload file.

Table 11-1 Payload File Fields

Field	Description
SVC_ORDER.STATUS	Specifies the status of the service order. Possible values are (preappend): • NEW • READY • PROCESSING • COMPLETED • FAILED
SVC_ORDER.STATUS_MSG	Specifies the status message for the service order.
SERVICE_ORDER_INFO[0].ACTION	Specifies the action to perform. Possible values are: • A (Activate) • D (Deactivate) • S (Suspend) • C (Change) • R (Reactivate) • I (Ignore)
SERVICE_ORDER_INFO[0].NAME	Specifies the name of the object to be provisioned. Possible values are: • MOBTEL • MOBFAX • MOBDATA • MOBSMS
SERVICE_ORDER_INFO[0].PARAMS[*].SUB_NAME	Specifies the name of the Parameter/supplementary services/VAS. Examples include: • CFU • CW • CLIP • VMBOX
SERVICE_ORDER_INFO[0].PARAMS[*].SUB_VALUE	Specifies the value associated with the NAME.
SERVICE_ORDER_INFO[1].*	Includes information about the ESN.
SERVICE_ORDER_INFO[2].*	Includes information about the NUM.

Service Order XML DTD

The following lists the XML DTD:

```
<!ELEMENT template (version?,
                    name,
                    start_line,
                    loading_controls,
                    file_type?,
                    file_format,
                    comment_line_prefix?,
                    global_record_info,
                    global_field_info,
                    records)>
          <!ELEMENT version (#PCDATA)>
          <!ELEMENT name (#PCDATA)>
          <!ELEMENT start_line (#PCDATA)>

          <!ELEMENT loading_controls ANY>
               <!ATTLIST loading_controls group_by CDATA #REQUIRED>
          <!ELEMENT file_type (#PCDATA)>
          <!ELEMENT file_format (#PCDATA)>
          <!ELEMENT comment_line_prefix ANY>

          <!ELEMENT global_record_info (record_delimiter?,
                    discard_prefix?,
                    record_type_locator?,
                    record_length_locator?,
                    ignore_record_types?,
                    record_length?)>
          <!ELEMENT record_delimiter ANY>
          <!ELEMENT discard_prefix ANY>
          <!ELEMENT record_type_locator ANY>
               <!ATTLIST record_type_locator name CDATA #IMPLIED>
               <!ATTLIST record_type_locator position CDATA #IMPLIED>
               <!ATTLIST record_type_locator start CDATA #IMPLIED>
               <!ATTLIST record_type_locator end CDATA #IMPLIED>
          <!ELEMENT record_length_locator ANY>
               <!ATTLIST record_length_locator name CDATA #IMPLIED>
               <!ATTLIST record_length_locator position CDATA #IMPLIED>
               <!ATTLIST record_length_locator start CDATA #IMPLIED>
               <!ATTLIST record_length_locator end CDATA #IMPLIED>
          <!ELEMENT ignore_record_types (ignore_record_type*)>
          <!ELEMENT ignore_record_type (#PCDATA)>
          <!ELEMENT record_length (#PCDATA)>

          <!ELEMENT global_field_info (trim_white_space?,
                    padding_char?,
                    field_delimiter?,
                    justification?,
                    attribute_value_separator?,
                    consecutive_delimiters_is_one?,
                    literal_indicator?,
                    missing_field_indicator?)>
          <!ELEMENT trim_white_space (#PCDATA)>
          <!ELEMENT padding_char (#PCDATA)>
          <!ELEMENT field_delimiter ANY>
          <!ELEMENT justification (#PCDATA)>
          <!ELEMENT attribute_value_separator ANY>
          <!ELEMENT consecutive_delimiters_is_one (#PCDATA)>
          <!ELEMENT literal_indicator ANY>
          <!ELEMENT missing_field_indicator ANY>

          <!ELEMENT records (record*)>
          <!ELEMENT record (field_definitions?,
                            user_mapping_info?,
```

```
                    event_mapping_info?,
                    filters?,
                    checks?)>
     <!ATTLIST record name CDATA #REQUIRED>
     <!ATTLIST record type CDATA #IMPLIED>
     <!ATTLIST record record_length CDATA #IMPLIED>
     <!ATTLIST record event_class CDATA #IMPLIED>
     <!ATTLIST record service_class CDATA #IMPLIED>
     <!ATTLIST record event_opcode_name CDATA #IMPLIED>
     <!ATTLIST record event_opcode_num CDATA #IMPLIED>
     <!ATTLIST record user_info_opcode_name CDATA #IMPLIED>
     <!ATTLIST record user_info_opcode_num CDATA #IMPLIED>
<!ELEMENT field_definitions (field*)>
<!ELEMENT field (field*)>
     <!ATTLIST field name CDATA #IMPLIED>
     <!ATTLIST field start CDATA #IMPLIED>
     <!ATTLIST field end CDATA #IMPLIED>
     <!ATTLIST field position CDATA #IMPLIED>
     <!ATTLIST field data_type CDATA #IMPLIED>
     <!ATTLIST field override_value CDATA #IMPLIED>
     <!ATTLIST field default_value CDATA #IMPLIED>
     <!ATTLIST field date_format_string CDATA #IMPLIED>
     <!ATTLIST field inf_data_type CDATA #IMPLIED>
     <!ATTLIST field inf_data_type_num CDATA #IMPLIED>
     <!ATTLIST field inf_field_name CDATA #IMPLIED>
     <!ATTLIST field inf_field_num CDATA #IMPLIED>
     <!ATTLIST field elem_num CDATA #IMPLIED>
     <!ATTLIST field elem_position CDATA #IMPLIED>

<!ELEMENT user_mapping_info (field*)>
<!ELEMENT event_mapping_info (field*)>
<!ELEMENT filters (filter*)>
<!ELEMENT filter (regexp, infix_regexp?)>
     <!ATTLIST filter name CDATA #REQUIRED>
     <!ATTLIST filter discard CDATA #REQUIRED>
     <!ATTLIST filter log CDATA #REQUIRED>
<!ELEMENT regexp ANY>
<!ELEMENT infix_regexp ANY>
<!ELEMENT checks (check*)>
<!ELEMENT check (#PCDATA)>
     <!ATTLIST check name CDATA #REQUIRED>
     <!ATTLIST check type CDATA #REQUIRED>
     <!ATTLIST check expr CDATA #REQUIRED>
     <!ATTLIST check field_name CDATA #IMPLIED>
     <!ATTLIST check field_pos CDATA #REQUIRED>
```

12

Provisioning Utilities

Learn about the syntax and parameters for the Oracle Communications Billing and Revenue Management (BRM) provisioning utilities.

Topics in this document:

agent_sim.pl

Use the **agent_sim.pl** script to generate an XML output file containing one or more service orders. By default, the script creates an XML file named **SvcOrder**_*eventObject_id*.**xml** in the *BRM_home*/**apps/gelco/service_orders** directory.

For more information, see "Writing Service Orders to an XML File".

Location

BRM_home/**apps/telco**

Syntax

```
agent_sim.pl [-p port] [-r responseCode] [-s status] [-f statusFlag] [-t featureStatus]
             [-m statusMessage] [-d delayResponse] [-i]
```

Parameters

-p *port*
Specifies the port number the script listens on. The default is **20000**.

-r *responseCode*
Specifies the response code the script returns for each request. The response code can be one or two digits. The default is **0** (acknowledge).

-s *status*
Specifies the service status to return in the provisioning response.

-f *statusFlag*
Specifies the status flag to return in the provisioning response.

-t *featureStatus*
Specifies the feature status to return in the provisioning response.

-m *statusMessage*
Specifies the status message to return in the provisioning response.

-d *delayResponse*
Delays the response by the specified number of seconds. Use this option to test the timeout in **dm_provision**. The default is **0**.

-i
Runs the script in user interactive mode, getting the provisioning response from the user and sending it back in the provisioning response.

Results

The **agent_sim.pl** script notifies you if it encounters an error.

load_pin_service_framework_permitted_service_types

Use the **load_pin_service_framework_permitted_service_types** utility to load the list of non-telco services supported by the Services Framework into the **/config/service_framework/permitted_service_types** object in the BRM database. You define the supported non-telco services in the *BRM_home***/sys/data/config/pin_service_framework_permitted_service_types.xml** file.

See "Specifying the Non-Telco Services Supported by Services Framework" for information.

Location

*BRM_home***/bin**

Syntax

```
load_pin_service_framework_permitted_service_types [-f xml_file]
                                                   [-v] [-h]
```

Parameters

-f *xml_file*
The name and location of the XML file that defines the list of non-telco services supported by Services Framework. By default, the utility uses the *BRM_home***/sys/data/config/pin_service_framework_permitted_service_types.xml** file.
If you do not run the utility from the directory in which the file is located, you must include the complete path to the file.

-v
Displays information about successful or failed processing as the utility runs.

-h
Displays the syntax and parameters for this utility.

Results

The utility notifies you when it successfully creates the **/config/service_framework/permitted_service_types** object or if it encountered errors. You can view more detailed error messages by looking in the **default.pinlog** file. This file is either in the directory from which the utility was started or in a directory specified in the utility configuration file.

load_pin_telco_provisioning

Use this utility to load your provisioning configuration into the BRM database.

The provisioning configuration specifies the service, device, and profile object fields that must be added to a provisioning service order. You specify the configuration in a provisioning configuration file (*telco_prov_config_filename*) and use that file as input for the utility. The default file is *BRM_home*/**sys/data/config/pin_telco_provisioning**.

This utility creates the **/config/telco/provisioning** and **/config/telco/provisioning/fieldlist** configuration objects.

For information, see "Specifying the Details to Add to the Service Order".

Location

BRM_home/**bin**

Syntax

```
load_pin_telco_provisioning [-d] [-v] telco_prov_config_filename
```

Parameters

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no displayed errors, but mapping between events and opcode numbers does not appear to be loaded correctly.

-v
Displays information about successful or failed processing as the utility runs.

telco_prov_config_filename
The name and location of the *telco_prov_config_filename* file.
If you copy *telco_prov_config_filename* to the same directory from which you run the **load_pin_telco_provisioning** utility, you do not have to specify either the path or the file name. If you run the command in a different directory from where the file is located, you must include the entire path for the file.

Results

By default, **load_pin_telco_provisioning** notifies you only if it encounters errors. However, if you use the **-v** flag, the utility displays confirmation messages when running.

> **Note**
>
> After you run this utility, you must restart the Connection Manager (CM).

load_pin_telco_service_order_state

Use this utility to load service order state configuration data for a specific telco service, such as GSM, into the BRM database.

This utility creates the **/config/telco/service_order_state** configuration object.

> **Note**
>
> Use the instructions and examples included in *BRM_home*/**sys/data/config/pin_service_order_state** file. For more information on BRM configuration files, see "Connecting BRM Utilities" in *BRM System Administrator's Guide*.

For information, see "Specifying the Available States for Each Service Order".

Location

BRM_home/**bin**

Syntax

```
load_pin_service_order_state [-d] [-v] service_order_state_config_file
```

Parameters

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no displayed errors, but mapping between events and opcode numbers does not appear to be loaded correctly.

-v
Displays information about successful or failed processing as the utility runs.

service_order_state_config_file
The path and name of the service order state file that you are loading for a particular service. If you copy *service_order_state_config_file* to the same directory from which you run the **load_pin_service_order_state** utility, you do not have to specify the path. If you run the command in a different directory from where the file is located, you must include the entire path for the file.

Results

load_pin_telco_service_order_state notifies you only if it encounters errors. However, if you use the **-v** flag, the utility displays confirmation messages when running.

> **Note**
>
> After you run this utility, you must restart the CM.

load_pin_telco_tags

Use the **load_pin_telco_tags** utility to load provisioning tags for telco services into the appropriate **/config/telco/service** object and account-level extended rating attributes (ERAs) into the **/config/account_era** object.

You define telco provisioning tags and account-level ERAs in a **pin_telco_tags**_*service* file. For example, you can use the *BRM_home*/**sys/data/config/pin_telco_tags_gsm** file for GSM services.

For more information, see "Setting Up Provisioning for GPRS Services" and "Defining Provisioning Tags for Telco Services by Using the pin_telco_tags File ".

Location

BRM_home/**bin**

Syntax

```
load_pin_telco_tags [-d] [-v] [-h] [-x] pin_telco_tags_file
```

Parameters

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no displayed errors, but mapping between events and opcode numbers does not appear to be loaded correctly.

-v
Displays information about successful or failed processing as the utility runs.

> **Note**
>
> This parameter is always used in conjunction with other parameters and commands. It is not position dependent. For example, you can enter **-v** at the beginning or end of a command to initiate the verbose parameter. To redirect the output to a log file, use the following syntax with the verbose parameter. Replace *filename*.**log** with the name of the log file:
> **load_pin_telco_tags** *any_other_parameter* **–v >** *filename*.**log**

-h
Displays the syntax and parameters for this utility.

-x
Overwrites all account ERAs and telco provisioning tags in the database, including the provisioning tags created using PDC, with those in the file.

> **Note**
>
> Use the **-x** parameter with care.

pin_telco_tags_file
The name and location of the **pin_telco_tags** file that you are loading. The default **pin_telco_tags** files are located in *BRM_home*/**sys/data/config**.
If you do not run the utility from the directory in which the file is located, you must include the complete path to the file. If you copy the **pin_telco_tags** file to the directory from which you run the **load_pin_telco_tags** utility, you do not have to specify the path. If you do not include a file name, the utility loads the default generic telco file, **pin_telco_tags**.

Results

The **load_pin_telco_tags** utility notifies you only if it encounters errors. However, if you use the **-v** parameter, the utility displays confirmation messages when running.

To verify that the objects were loaded, you can display the **/config/telco**_*service* object (for example, **/config/telco/gsm**) and the **/config/account_era** object by using the Object Browser, or use the **robj** command with the **testnap** utility. See "Reading an Object and Fields" in *BRM Developer's Guide*.

> **Note**
>
> You must restart the CM to make new provisioning tags and ERAs available.

RunSimulator

Use the **RunSimulator** utility to simulate how a network provisioning agent would process your service orders. It takes an XML provisioning payload file as input and returns a sample provisioning network agent response.

For more information, see "Simulating How a Network Agent Processes Service Orders".

Location

*BRM_home***/apps/telco**

Syntax

```
RunSimulator [-t featureStatus] [-s status] [-f statusFlags] [-m messageStatus]
             [-x xmlFile] [-d] [-h|?]
```

Parameters

-t *featureStatus*
Specifies the supplementary status to return.

-s *status*
Specifies the status of the service order. Possible values are success (**0**) and failed (**1**). The default is **0**.

-f *statusFlags*
Specifies the service status to return in the provisioning response.

-m *messageStatus*
Specifies the status flag to return in the provisioning response.

-x *xmlFile*
Specifies the name and location of the XML provisioning payload file.

-d
Creates a log file for debugging purposes. Use this parameter for debugging when the utility appears to have run with no displayed errors, but does not generate an output flist.

-h|?
Displays the syntax and parameters for this utility.

Results

If successful, the utility creates an **/event/provisioning/service_order/*** event object. The utility notifies you if it encounters an error.

A

Provisioning Flags

Learn about the flags used for provisioning in Oracle Communications Billing and Revenue Management (BRM).

Topics in this document:

About GSM Service Provisioning Flags

GSM services use these standard BRM status attributes: active, inactive, and closed. Additionally, GSM services include provisioning flags that indicate the provisioning status.

Service orders often include multiple provisioning requests. For example, a service order might include provisioning requests for the following:

- The bearer service
- One or more supplementary services
- A voice mailbox

In some cases, only some of the provisioning requests in an order can be completed, while others fail. In most cases, if a service can be configured so that the customer can at least perform some activities, the service order status is successful. In that case, the remaining provisioning requests can be completed later. Your business policies or provisioning configuration may differ. For example, specify which services or supplementary services must be provisioned before an account can be created.

You can use the provisioning flags to customize your business policies, utilize event notifications to trigger emails, or record status changes in a log file.

If provisioning or unprovisioning fails, you must perform the provisioning operation manually on the network or complete the processing in the provisioning system. You cannot resend the provisioning service order.

Table A-1 lists the GSM service provisioning flags.

Table A-1 Provisioning Flags for GSM Services

Provisioning Flag	Description
Processing-Provisioning	Provisioning for the GSM service is in progress. This flag is set by BRM when you create or activate a service.
Provisioning-Failed	Provisioning for the GSM service has failed. BRM sets this flag: • When provisioning fails during service activation. • When provisioning fails while inactivating or closing a service. If a service is closed but has a Provisioning-Failed flag, BRM returns an error.

Table A-1 (Cont.) Provisioning Flags for GSM Services

Provisioning Flag	Description
Unprovisioning	The GSM service is being unprovisioned. This flag is set by BRM when you close or inactivate a service.
Suspend	The GSM service is suspended. This flag is set by BRM when you inactivate a service.

About Supplementary Service Provisioning Flags

Customer Center displays the provisioning flags for supplementary services in the **Service** tab.

The default supplementary service provisioning flags are shown in Table A-2.

Table A-2 Default Supplementary Service Provisioning Flags

Provisioning Flag	Description
Provisioning	The supplementary service is being provisioned.
Provisioning Failed	Supplementary service provisioning failed.
Active	Supplementary service provisioning was completed successfully.
Suspending	The supplementary service is being inactivated.
Suspending Failed	Unprovisioning failed while the supplementary service was being inactivated.
Suspended	The supplementary service was successfully inactivated.
Unprovisioning	The supplementary service was canceled and is currently being unprovisioned.
Unprovisioning Failed	The supplementary service was canceled, but unprovisioning failed.
Unprovisioned	The supplementary service was successfully canceled and unprovisioned.

About Service ERA Provisioning Flags

Customer Center displays the provisioning flags of service-based ERAs that have a provisioning impact. (Some ERAs do not require provisioning.) The default provisioning flags are shown in Table A-3.

Table A-3 Service ERA Provisioning Flags

Provisioning Flag	Description
No_op	There is no provisioning impact.
Provisioning	The ERA is in the process of being provisioned.
Provisioning Failed	ERA provisioning failed.
Active	ERA provisioning was completed successfully.
Suspending	The ERA is being inactivated.
Suspending Failed	Unprovisioning failed while the ERA was being deleted.

Table A-3 (Cont.) Service ERA Provisioning Flags

Provisioning Flag	Description
Suspended	The ERA was successfully deleted.
Unprovisioning	The ERA was deleted and is currently being unprovisioned.
Unprovisioning Failed	The ERA unprovisioning failed.
Unprovisioned	The ERA was successfully unprovisioned.

B

Using a Policy Source File to Set Up Provisioning

Learn how to enable provisioning by using a policy opcode in Oracle Communications Billing and Revenue Management (BRM). This method does not use the provisioning tag framework; provisioning tags are configured in the opcode.

Topics in this document:

Using a Policy Source File to Set Up Provisioning

You can define provisioning tags directly in the **fm_subscription_pol_provisioning.c** file without using the provisioning tag framework. This file is the single source file for all provisioning operations.

> **Note**
>
> The provisioning tags framework is the preferable method for creating provisioning tags.

To define provisioning tags directly in source code, follow these steps:

1. Open the *BRM_home*/**source/sys/fm_subscription_pol/fm_subscription_pol_provisioning.c** file.
2. Define your provisioning tags by following the instructions in the file.
   a. In each entry in the **provisioning_tags** table, include the name of the service associated with the tag, the tag name, and calls to service-specific functions.
   b. Ensure that each function in the table does the following:
      - Changes fields in the service object when customers purchase the service
      - Clears the appropriate fields when customers cancel the service
3. Compile and save the file.

If you create a provisioning tag in the policy source file, you must modify and recompile the source file to modify or delete the tag.