Integrating Oracle CX Commerce and Oracle CX for Communications

F32080-01
August 2020

Integrating Oracle CX Commerce and Oracle CX for Communications,

F32080-01

Contents

1

Integrate Oracle CX Commerce and Oracle CX for Communications

This document is intended to provide instructions on how to install, configure, and use the integration of Oracle CX Commerce (OCC) with Oracle CX for Communications.

Oracle CX for Communications can be thought of as a “verticalized” version of Oracle Engagement Cloud (OEC). It is part of the Sales Cloud portion of OEC.

Keep in mind that this document focuses only on the registered shopper Telco application aspects of this integration and only describes how to configure the integration of Oracle CX Commerce with Oracle CX for Communications.

Commerce is an eCommerce solution designed specifically to run in the Oracle Cloud. It provides you with a range of powerful tools to build a flexible, feature-rich storefront for your customers. The functions that Commerce can perform that are important to this Telco integration are the abilities to store and manage shopper, contact, and profile information.

On the other hand, the functions that the Oracle Engagement Cloud solution (with Oracle CX for Communications as part of the Sales Cloud portion) can perform that are important to this integration include the following:

- Consolidates account and contact data originating from multiple sources
- Standardizes related addresses
- Resolves duplicates in account and customer data
- Ensures that you have a trusted customer profile
- Provide a rich data model to store the relationships within a Comms Household including Customer, Service and Billing accounts

By integrating these two solutions, you extend Oracle CX Commerce to consume account and other customer data maintained by Oracle Engagement Cloud and vice versa. This data includes things such as shopper profiles, shopper addresses, contacts, account information, and more. This integration helps to ensure that the same shared shopper data structure is maintained by integrating the shopper data found on Oracle CX Commerce with the related account data found on Oracle Engagement Cloud.

To better describe the usefulness of this integration, Commerce already has an account-based shopper account data structure that contains both Account and Contact information. A normal registered shopper data information structure on Commerce, however, does not need to have Telco account information - only contact, shopper, and profile information.

In the case of Telco retail situations on Commerce, however, things are a bit different than normal retail situations in that they actually need to use the traditional account-based shopper data structure (i.e., Account, Billing, Contact, and Account Contact role information) rather than the traditional registered shopper retail information structure (i.e., only contact/shopper/profile information).

The purpose of the integration, then, is to provide access to the complete set of data that is needed for a successful Telco transaction by integrating the needed data found on Oracle CX Commerce with the needed data found on Oracle Engagement Cloud.

In summary, this integration lets registered shoppers engaged in business-to-customer commerce use the business-to-customer Comms offering. Business-to-customer horizontal commerce only supports a contact, while business-to-customer Comms vertical commerce requires contact and account structure. The integration with OEC enables OCC for Comms to support the full contact/account structures.

In the end, however, all Telco contact, shopper, and profile information still persists in Commerce and all of the needed Account details such as Billing profile and Contact-Account role information are persisted only in the primary CX customer model on Oracle Engagement Cloud.

This document is written for Oracle CX Commerce and Oracle Engagement Cloud administrators who are configuring the integration of these two systems. Readers of this document should have experience with both Oracle CX Commerce and Oracle Engagement Cloud.

Note: Keep in mind that this document focuses only on the Telco aspects of this integration and describes only how to configure integrations of Oracle CX Commerce with Oracle CX for Communications. The integration configuration described presents just one example of how to set it should be set up. For information about other possible configuration options, refer to the documentation of those specific products.

2

Understand the Integration

This chapter provides you with the information needed to better understand the architecture and components of the Oracle CX Commerce and Oracle CX for Communications integration.

Understand the integration

The goal of Oracle CX Commerce integrating with Oracle CX for Communications (part of the Sales portion of Oracle Engagement Cloud) is to provide a definitive single primary Customer model of shopper data that includes shopper/customer, contact, profile, account, billing, and service information.

As mentioned, the integration uses the Sales Cloud portion of Oracle Engagement Cloud along with its Sales Cloud APIs. Specifically, Commerce is integrating with the primary CX Customer model on Oracle Engagement Cloud. The primary CX Customer model refers to the shopper account data model that underpins the OEC APIs. This is a data model that allows you to manage complex account information about shoppers who belong to a commercial community.

Understand the architecture of the integration

This integration architecture specifically uses the Sales Cloud part of Oracle Engagement Cloud along with its Sales Cloud APIs to create a flow of shopper data between Commerce to Oracle Engagement Cloud via Oracle Integration Cloud (OIC) and back again.

Note: Oracle Engagement Cloud must have the Comms vertical module installed.

Oracle CX Commerce includes an Oracle Integration Cloud adapter (the Commerce adapter) that is used for the integration. The Oracle Engagement Cloud part of the integration uses the Oracle Sales Cloud adapter for communication with Oracle Integration Cloud.

Understand the required integration pieces

This integration consists of the following pieces that must each be configured correctly for the integration to work successfully:

- Oracle CX Commerce – This piece acts as a gateway for shopper and business account information. Commerce communicates to a server-side extension (SSE) which in turn communicates with Oracle Engagement Cloud via connecting to Oracle Integration Cloud. Included as part of this piece, there should also be the introduction of widgets that support the Customer Account model (see the major sections that follow) and therefore support interaction with an external Oracle Integration Cloud system. Instances of these widgets need to be added to the appropriate store layout(s) to support the integration.
- CustomerAccountModel server-side extension (SSE) – This piece implements the logic for sending data from Commerce to Oracle Engagement Cloud via Oracle Integration Cloud as well as receiving updates or changes from Oracle

Engagement Cloud back to Commerce. The server-side extension also lets integration implementers extend and customize the integration.

- Oracle Integration Cloud – Within the integration, you use Oracle Integration Cloud to assign the source application (Commerce) and the target connection (Oracle Engagement Cloud) that the source is being integrated with. It is used to import the necessary integrations to support the Commerce and Oracle Engagement Cloud integration as a whole. Business objects, data, and services are then mapped by activating the necessary integrations so that they can be shared between Commerce and Oracle Engagement Cloud. Oracle Integration Cloud handles both the push and pull of data from Commerce to Oracle Engagement Cloud. Also, by using Oracle Integration Cloud, you have the ability to further customize the integration.
- Oracle Engagement Cloud – This final piece of the integration serves as a manager and repository of business account information. With the integration in place, Oracle Engagement Cloud is able to share this account data with the shopper, contact, and profile data on Commerce. This integration uses OEC Comms APIs to accomplish this.

Note: Oracle CPQ Cloud (Configure, Price, Quote) could also be used to configure new commerce items as a possible enhancement to this integration. CPQ Cloud could act as the primary asset data source and be queried for existing assets and would support asset operations such as suspend, resume, terminate, renew, and modify. Also, alternative/3rd party customer relationship management (CRM) systems could be used.

This document, however, focuses only on integrating Oracle CX Commerce with Oracle CX for Communications (part of the Sales Cloud portion of Oracle Engagement Cloud). Oracle Engagement Cloud is used as the CRM system.

Required versions

A successful integration of Oracle CX Commerce and Oracle CX for Communications requires the following versions (or later) of these products:

- Oracle CX Commerce (OCC) – 19A
- Oracle Integration Cloud (OIC) – 18.4.2
- Oracle Engagement Cloud (OEC) with Comms vertical module installed - 11.13.17.11.0 (REL13.1711) or later

If you do not have one or more of these, please contact an Oracle sales representative for information on how to acquire one: http://www.oracle.com/us/corporate/contact/index.html.

Additional Resources

If you require further information regarding Oracle CX Commerce, you can access the latest product documentation and training videos through the Oracle Help Center page for Oracle CX Commerce.

If you require further information regarding Oracle Integration Cloud, you can access the latest product documentation through the Oracle Help Center page and searching for Oracle Integration Cloud.

If you require further information regarding Oracle Engagement Cloud and Sales Cloud, you can access the latest product documentation through the Oracle Help Center page and searching for Oracle Engagement Cloud and Sales Cloud.

These information sources also contain links to helpful blogs, videos, developer communities, and Support.

3

Understand the Customer Account Model

This chapter provides you with the information needed to better understand the Customer Account Model which is an important part of the Oracle CX Commerce and Oracle CX for Communications integration.

Understand the Customer Account Model

To gain a better understanding of the Oracle CX Commerce and Oracle CX for Communications integration, it helps to know about the asset-based ordering (described below) Customer Account Model used in Telco e-commerce transactions which are the focus of this integration.

This account model is important because services (i.e., cellphone, broadband, cable, and TV services etc.) customarily have a customer account, service account, billing account and profile associated with them. It is critical that this information is passed to downstream fulfillment/provisioning systems when an order is submitted.

To configure the Customer Account model, select the CustomerAccountModel-store SSE and/or the CustomerAccountModel-agent SSE from the Developers tab in the Commerce administration user interface.

Both SSEs enable integration with an external CRM system to retrieve and update the following:

- Contacts
- Accounts (Customer Billing and Service accounts)
- Account Roles (Admin, Buyer and User)
- Billing Profiles

The SSEs serve as the API for the pre-built integration with Oracle Engagement Cloud.

Asset-based ordering lets shoppers create assets based on orders and also lets them create new orders to modify these assets. For example, if a customer orders a specific telephone service, then the Commerce application generates an asset record representing that service. Commerce acts only as the order capture system. Once the order has been successfully placed in Commerce, it is passed to (or is picked up by) a downstream system(s) where the asset is created and maintained.

The heart of the integration described in this document deals with the creation, updating, and maintaining of shopper account information involved in asset-based ordering. The collection of this information is known as the primary Customer Account Model. The primary Customer Account Model, then, is the information shared by both Oracle CX Commerce and Oracle Engagement Cloud as the integral data items of this integration. Commerce does not persist customer account data, however. This is handled only by Oracle Engagement Cloud.

There are a number of different account types associated with a shopper using asset-based ordering within Oracle CX Commerce. This section provides some detail about these account types and their function.

There are three account types available within Commerce relating to billable services:

- Customer account
- Service account
- Billing account

The details for these three accounts are captured when an order is placed and their relationship with the service is maintained after an order has been fulfilled. Also, where a shopper or contact already has an existing customer account data model, the information can be retrieved and used during checkout or the placing of an order.

In many instances, these three accounts may all refer to the same person or organization, but there may also be instances when they differ. It is important to understand the relationship between the different types of account.

In addition to the three account types, there is also a Billing Profile, which includes information such as billing preferences.

All of the information required for the Customer, Service, and Billing accounts, and for the Billing Profile is captured during the order process in Commerce.

Also, taking notice of the account types just discussed, we can discuss the variations made up of these different types of account models involved. These are the following:

- A simple account model – This refers to a customer account with a billing profile and account address along with a role associated with contact.
- A complex account model – This refers to an account model with customer, billing and service accounts along with some addresses and billing profiles associated to each. A contact role relationship is established to each of these accounts.

Let's now look in detail at each type of account that makes up the overall account model and its hierarchical structure. Again, the importance of knowing about all of these accounts and their related information is that this is the data that will be shared in the Oracle CX Commerce and Oracle CX for Communications integration

Note: Address information is something used extensively in Commerce transactions. For all procedures and SSEs that require address information for endpoint inputs, in addition to using Commerce's default address formats, you can also use the REST API to create multi-country custom address formats. This lets you create country-specific address formats to ensure that your address formats align with the requirements of any external service that you might use. This means that addresses appearing in profiles, accounts, registration requests, order addresses and more can be customized. For more complete information on creating custom addresses and understanding how to use custom address formatting, refer to the following:

- Customize Address Formats using the API in *Extending Oracle CX Commerce*
- Work with address types in *Extending Oracle CX Commerce*
- Account Details in *Using Oracle CX Commerce*
- Work with account addresses in *Using Oracle CX Commerce*
- Work with account registration requests in *Using Oracle CX Commerce*

Understand a customer account

This type of account represents the person or organization that owns the service (i.e., the Telco service). It includes basic customer information such as name and address and can receive both services and bills. Customer accounts represent the highest level in the account hierarchy and can perform all customer, service, or billing functions.

Understand a service account

This type of account represents the person or organization that receives the service.

The address associated with the Service account defines the physical location where the service must be delivered. This address is used to verify service and ordering eligibility.

Service accounts are required when the location and/or party receiving the service differ from the Customer account. If a Service account is required, it is always a child of a Customer account. There can be multiple Service accounts associated with a single Customer account.

A Service account cannot be used to perform any of the functions of a Customer or Billing account.

Understand a billing account

This type of account represents the person or organization that pays for the service.

Billing accounts are required when the location and/or party paying for a service differ from the Customer account. If a Billing account is required, it is always a child of a Customer account. There can be multiple Billing accounts associated with a single Customer account.

A billing account cannot be used to perform any of the functions of a Customer or Service account.

A billing profile may be associated with either a Customer account or a Billing account. It captures information such as billing preferences, method of payment, and contact details. There may be more than one billing profile associated with a Customer or Billing account, and the shopper must choose which billing profile to use when placing an order for a service.

Understand contacts, buyers, and contact roles

In addition to the accounts just described, a contact record is also required for Oracle Engagement Cloud (the CRM focused on in this integration). The contact record typically stores identity information such as first name, last name, email address, phone number.

Contacts also have a role based relationship on accounts. The supported roles are “admin”, “buyer” and “user.” Roles are hierarchical (for example, the administrator has buyer and user role). A buyer also has a user role. A contact must have a direct role relationship on the account, so for a complex account model, the account owner would have an “admin” role on every account in the model. For complete information contacts and account-based roles for contacts, refer to Configure Business Accounts.

Add payment details to customer billing profile

In Telco transactions there is critical contact information that must be passed downstream to fulfillment and provisioning systems. Based on the Customer Account Model, this information is the following:

- Customer Account
- Service Account
- Billing Account
- Billing Profile

This section covers the processes involved in the updating of payment details in a Billing Profile.

Understand how billing profiles are handled

A Contact (that is a user, shopper, or customer) may have the following information in Commerce transactions:

- A Customer Account (Account of type “Customer”).
- A link to other Customer Accounts. This would occur where the merchant supports account models such as “Family Account,” “Household,” or “Family and Friends.”
- A link to one or more Service Accounts (Accounts of type “Service”).
- A link to one or more Billing Accounts (Accounts of type “Billing”).
- A link to one or more Billing Profiles.

In this type of transaction model, Commerce has the important ability to create or update a billing profile with payment details. This is important because the payment information for the billing profile needs to be captured and passed on to the primary CRM (Customer Relationship Management) system in a PCI compliant manner. For the Oracle Telco CX Solution, the primary CRM system is Oracle Engagement Cloud (OEC).

Note: In an integration like this, transaction payment details that are stored in the CRM system are used for recurring payments. This info is pulled by the billing system from CRM. To compare this with Commerce, Commerce handles one time/upfront payments by interacting with payment gateways.

At this point in time, Commerce is PCI compliant whereas Oracle Integration Cloud and Oracle Engagement Cloud are not. Commerce supports the storing of credit cards against a shopper profile. The card details are captured, however, in the store as part of the checkout flow and subsequently a tokenized version of the card is obtained from an integrated payment system as part of the payment authorization process. This token, along with a masked version of card number and the expiration date are then stored against the shopper’s profile so that the shopper can easily use the same card for future purchases.

A feature is now supported that offers a generic horizontal benefit to Commerce and contributes to the Telco specific vertical requirement. This feature uses a store API endpoint that allows a shopper to store credit cards as part of their profile without actually purchasing an order. This endpoint can then be used to enable Commerce to pass credit card information to Oracle Engagement Cloud as part of the shopper billing profile when creating or for updating accounts in OEC.

Understand the Update Profile endpoint and Generic Payment webhook

As mentioned, Commerce provides an Admin and Agent Update Profile store API endpoint that lets you add and store customer credit cards as part of a shopper Billing Profile without actually purchasing an order.

The name of the endpoint is `addCreditCard`. The Admin URI for the endpoint is `POST /ccstoreui/v1/current/creditCards/`. The Agent URI for the endpoint is `POST /ccagentui/v1/profiles/{id}/creditCards`.

The endpoint can be used to invoke Add Card requests multiple times to let you add more than one card to a shopper Profile. Each new card is then stored against the profile. The inputs of this endpoint are:

- cardType
- nameOnCard
- cardNumber
- expiryMonth
- expiryYear

Both versions of this endpoint trigger the Generic Payment webhook for a Tokenize operation on the payment system. The payment system is expected to return a tokenized value of the card which is then saved against the billing profile. The endpoint then returns back a stored card ID.

Note: The ability to add credit cards directly to a shopper profile requires configuring the Generic Payment webhook and enabling 3D-Secure support by specifying the `card3ds` processor in the gateway extension's `config.json` file. See Incorporate 3D-Secure support for information about how to do this.

Finally, the Admin Get Profile endpoint can then be used to get the token value of the card using the stored card ID. See Configure Payment Processing and Create a Generic Payment Gateway Integration for more complete information on this subject.

Add and update a Billing Profile to include a card token

There are two processes/flows that Commerce uses to capture the billing profile information that can be passed on to the primary CRM system. These are the following:

- Commerce creates an account(s) for a contact which includes creating one or more billing profiles.
- Commerce updates an existing billing profile.

To assist in these processes, Commerce provides a Customer Account Model Server Side Extension (SSE). The sections that follow provide the details on the various processes and flows that this SSE supports

Understand the OCC to OEC Account Create flow

The basic information on this flow is the following:

- SSE Name - Customer Account Model
- Endpoint Name - Create Accounts
- Flow Name - OCC OEC Comms: Account Create

In this process, the SSE first identifies the payment type. If the payment type is Card, a check is done to see if the token for the card has been passed in (i.e., an existing card stored on the shopper's Commerce profile). If the token has been passed in, then

a check is done to see that the basic card information has also been passed in the form of `maskedCardNumber` and `expiryMonth`. If the token has not been passed in (i.e., a new card is being introduced), then a check is made to look for the following “full card” information being passed in:

- Card Type
- Name on Card
- Card Number
- CVN (card verification number)
- Expiry Month
- Expiry Year

There is also a step in the SSE execution whereby an API call can be made to the storefront Profiles/Update Profile endpoint to retrieve a tokenized version of the card. The billing profile information passed to the OCC OEC Comms: Account Create flow includes:

- Payment Method=Card
- Masked Card Number
- Card Expiry Date
- Tokenized representation of the card

The next section provides details on an update process/flow that the SSE handles.

Understand the OCC to OEC Account Update flow

The basic information on this flow is the following:

- SSE - Customer Account Model
- Endpoint Name - Update Accounts
- Flow Name - OCC OEC Comms: Account Update

In this process, the SSE first identifies the payment type. If the payment type is Card, a check is made to see if the token for the card has been passed in (i.e., there is an existing card stored on the shopper’s Commerce profile). If the token has been passed in, then a check is made that the basic card information has also been passed in the form of `maskedCardNumber` and `expiryMonth`.

If the token has not been passed in (i.e., a new card has been introduced), then a check is made to look for the following “full card” information:

- Card Type
- Name on Card
- Card Number
- CVN (card verification number)
- Expiry Month
- Expiry Year

The billing profile information passed to the OCC OEC Comms: Account Update flow includes:

- Payment Method=Card

- Masked Card Number
- Card Expiry Date
- Tokenized representation of the card

Note: Keep in mind the following additional details regarding the OCC OEC Comms: Account Update flow:

- A check is made to verify that the payment type is credit card/debit card. If it is credit card/debit card, then a check is made to verify whether creditCardNumber is masked or non-masked.
- If creditCardNumber is masked, an additional check is made to verify that the masked creditCardNumber and creditCardId values provided are valid. If both are valid, then only creditCardNickname is allowed to update. All other fields/properties are not allowed to update.
- If creditCardNumber is unmasked, then the card is considered a new card. Tokenization occurs with the provided card details is similar to the Create Account flow. The process ends with the new card and token details stored in CDM.

Understand fields and properties supported by the billing profile

For the credit card or debit card payment type, the following fields/properties are supported:

- paymentMethod: “debitcard”/”creditcard”
- creditCardNumber (mandatory field)
- creditCardExpiryMonth (mandatory field)
- creditCardExpiryYear (mandatory field)
- creditCardContactName (mandatory field)
- creditCardType (mandatory field)
- creditCardSecurityCode
- creditCardNickname
- creditCardIin

For the bank transfer payment type, the following fields/properties are supported:

- paymentMethod: “bankTransfer”
- bankAccountNumber (This is the only mandatory field for bankTransfer payment type.)
- bankRoutingNumber
- bankAccountType
- bankAccountName
- bankSortCode
- bankName
- bankAddress
- bankIban
- bankSwiftCode

Understand the process changes as seen in the store interface

With a standard checkout flow where the shopper does not yet have a customer account model and is purchasing a service, the store interface captures the payment card details. This includes the following:

- The shopper is prompted to identify the payment type. If the payment type is “Credit/DebitCard,” the shopper is prompted to select either an existing card or enter new card details.
- If it is a new card the user interface captures the following required card information:
  - Card Type
  - Name on Card
  - Card Number
  - CVN (card verification number)
  - Expiry Month
  - Expiry Year

For a billing profile update checkout flow where the shopper, who does not have a customer account model and is purchasing a service, the store interface captures the payment card details. These details include the following:

- The shopper is prompted to identify payment type.
- If the payment type is “Credit/DebitCard,” the user interface captures the following the required card information:
  - Card Type
  - Name on Card
  - Card Number
  - CVN (card verification number)
  - Expiry Month
  - Expiry Year

4

Configure the Oracle CX Commerce and Oracle CX for Communications Integration

This chapter provides you with the information needed to correctly configure the Oracle CX Commerce and Oracle CX for Communications integration.

Configure the Oracle CX Commerce and Oracle CX for Communications integration

Now that you know more about the Customer Account Model pieces used in this integration, this section describes how to install and configure all of the pieces that make up the Oracle CX Commerce and Oracle CX for Communications integration so that this data can be shared.

Configure Oracle Engagement Cloud

As a prerequisite when configuring Oracle Engagement Cloud, the Applications Cloud environment being used must have the OEC Comms extension configured and enabled.

To configure the Customer Data Management portion of this integration, do the following:

1. Create the new `defaultShippingAddress` and `defaultBillingAddress` custom field in the Oracle Engagement Cloud Application Composer. The CustomerAccountModel SSE used in this integration requires the use of these custom fields and requires that you create them.
   These fields can be modeled in Oracle Engagement Cloud by creating fields under the Address SDO (Service Data Object) located in the pane on the left side in the Application Composer. In Oracle Cloud Services, you use a new sandbox environment to create these fields and publish them. To do this, perform the following steps:

   Note: For more complete information, refer to the Oracle Application Composer and Oracle Engagement Cloud documentation.

   - While in Oracle Engagement Cloud, click on your user name dropdown and select **Manage Sandboxes**. The Sandbox environment that is in use is displayed in the middle beside the search bar of the main page.
   - Create a new sandbox environment, name it, and save it.
   - From the Sandbox list, set the newly created sandbox as **Active**.
   - Go to the Navigator on the Home page and select **Tools** and then **Application Composer**.
   - Select **Objects** then **Standard Objects** in the Objects pane on the left side of the Application Composer.
   - Select **Address** under **Standard Objects** and then **Fields**.

- Create the new `defaultShippingAddress` and `defaultBillingAddress` fields in the Application Composer.
- Once you have created and configured the new fields, publish the currently active sandbox. To do this, go to **Manage Sandboxes** from the User drop-down list and select the active sandbox from the list.
- Click the **Publish** button. This publishes the changes you just made to mainline/

2. Make sure that the newly created custom address fields `defaultShippingAddress` and `defaultBillingAddress` have been mapped to the correct OIC mapping. For more information, refer to Creating Oracle Sales Cloud Custom Objects and Fields to Support Extensibility in the Oracle Engagement Cloud documentation.
Note: For this release, all other field mappings between the other connections and integrations has already been done as part of the overall installed integration.
3. Create custom Relationships for Role configuration in Oracle Integration Cloud. The CustomerAccountModel SSE currently supports 3 roles: Admin, Buyer and User. These need to be modeled in Oracle Engagement Cloud by creating custom relationships. Refer to the Oracle Engagement Cloud documentation for more details on creating relationships.
To do this, perform the following steps:
   - In the Navigator on the Home page, click **Setup and Maintenance**.
   - In the **Search Tasks** box region, search for Manage Relationship Types.
   - Select this task.
   - Create the Relationship Type on the panel that is provided.
   - The roles that need to be configured in **OCC-OEC_ROLE_LOOKUP** are already created in the OIC package.
4. Create a Source System Reference for Oracle CX Commerce in Oracle Integration Cloud. Source systems are external sources of data that are used to import data into Oracle Integration Cloud. In this integration the external source is Oracle CX Commerce.
Source system references identify the source of the data and specify the references to existing source systems and base tables of Oracle Engagement Cloud. Oracle Integration Cloud uses source system references to create references between Oracle CX Commerce IDs and the Oracle Integration Cloud IDs.

   To do this, perform the following steps:
   - In the Navigator, click **Setup and Maintenance**.
   - In the **Search Tasks** box region, search for Manage Trading Community Source Systems.
   - Select this task and create and create a new source system for Commerce with the code **COMMERCE_CLOUD**. This is the source system code that will be used in the OIC mappings while storing the Commerce profile ID in Oracle Integration Cloud.
5. As a final optional Oracle Integration Cloud configuration task, the address type whether `SHIP_TO` or `BILL_TO` can be set using Address purposes of Oracle Integration Cloud. A list of valid values is defined in the lookup `PARTY_SITE_USE_CODE`. Review and update the codes using the Manage Standard Lookups task in the **Setup and Maintenance** work area.

Configure Oracle Integration Cloud

To configure the Oracle Integration Cloud portion this integration, perform the following steps:

1. Go to the Integrating Oracle Commerce and Oracle CX for Communications information on My Oracle Support.
2. Download the Oracle Commerce and Oracle CX for Communications integration package. The package file `OCC-OECComms_Integration_X.X.par` can be obtained from My Oracle Support. Be sure to download the file to a location that is accessible for importation into Oracle Integration Cloud.
   Besides providing the necessary separate integrations needed for the overall integration, this package also handles all of the data mapping that needs to be done between Commerce and Oracle Engagement Cloud via Oracle Integration Cloud.
3. Import the integration package into Oracle Integration Cloud by doing the following:
   - Log on to Oracle Integration Cloud as an administrator user.
   - Click the **Packages** icon.
   - Click the **Import Package** button.
   - Click **Browse** to open a navigation pane.
   - Browse for and select the packages archive (`.par`) file when prompted.
   - Click **Import**. The package should be added to the Packages list. Clicking on the name of the package in the Package list displays the integrations that are included in the package.
4. Configure the three connections in Oracle Integration Cloud that are needed for each separate integration in the package to work. These connections define information about the instances of each configuration you are integrating. The following three connections are used across all of the integrations provided in the package (described in the step that follows this one):
   - Oracle Engagement Cloud Comms connection - `ORACLE_ENGAGEME_CLOUD_COMMS`.
     For this connection, you need to provide 3 connection URLs: OSC Services Catalog WSDL URL, OSC Events Catalog URL (optional), and Interface Catalog URL (optional). This provides the necessary information to connect to your application or endpoint and process requests.

     You must also provide a user name and a password to provide security credentials to access your application or endpoint.
   - Oracle Commerce connection - `ORACLE_COMMERCE_CLOUD`
     For this connection, you first need to provide the Connection base URL. This provides the information to connect to your application or endpoint and process requests.

     The second part of configuring this connection deals with the security of the connection. The Commerce connection uses the OAuth security policy, so you must enter a Security token for the connection. This token is generated in Oracle Commerce. By generating a security token for the integration, you register the integration within Oracle Commerce and are granted access to the data.

- REST connection - `ORAC_COMM_CLOU_TELC_SSE_ICS_TRIG` (Oracle Commerce SSE OIC Trigger). For this configuration you need only to open the connection in the Oracle Integration Cloud user interface and save it again by selecting the **Save** button and then the **Close** button.
  This is a static connection provided by the integration. If you wish, you also have the optional ability to add a Connection Administrator email address as well as a user name and a password to provide security credentials to access your application or endpoints.

5. Activate all the integrations provided when you have imported the integration package. By default these are not activated. To do this, locate the integration you want to activate in the **Integrations** list and go to the far right end. Switch the sliding button icon to the right to activate the integration.
   An integration includes at the least a trigger (source) connection (for requests sent to Oracle Integration Cloud) and invoke (target) connection (for requests sent from Oracle Integration Cloud to the target). All field mapping between the connections has already been done as part of this overall integration.

   The **Integrations** list names for the integrations that need to be activated are the following:

   - OCC OEC Comms: Account Find - This integration workflow gets the details of accounts (with contacts) from OEC Comms for a profile on Commerce. For clarification, the OCC profile is in fact the Telco contact.
   - OCC OEC Comms: Contact Create - This integration workflow checks to see if a contact exists in OEC before creating the profile on OCC and tries to match it. If it does not exist, it creates a new one in OEC and links it to OCC profile.
   - OCC OEC Comms: Contact Sync - This integration workflow syncs the contact on OEC Comms for a profile created on Commerce. This integration is triggered via the Profile Create Webhook when a new shopper registers on Commerce.
   - OCC OEC Comms: Account Create - This integration workflow creates a customer account on OEC Comms for a profile on Commerce.
   - OCC OEC Comms: Account Update - This integration workflow updates the account on OEC Comms when a billing profile is modified and addresses are added to an account on Commerce.
   - OCC OEC Comms: Contact Find - This integration workflow gets the details of a contact on OEC Comms for a profile on Commerce.

Install and configure the server-side extension in Commerce

For the next step of the integration configuration process, you install and configure the CustomerAccountModel server-side extension (SSE) to allow communication between Commerce and Oracle Integration Cloud as part of the data flow.

Commerce integration functionality is provided through server-side extensions. In addition to providing REST APIs and webhooks for integrating with external systems as well as widgets for extending your storefront, Commerce also includes support for developing server-side extensions written in JavaScript.

Server-side extensions (SSEs) are built using the Express web framework and executed in the Node.js runtime environment. These extensions implement custom REST endpoints, which have the prefix `/ccstorex/custom` (for store version) and `/ccagentx/custom` (for agent version). For more information about developing server-side extensions, see the following post on the Oracle Partner Network.

Note: Address information is something used extensively in Commerce transactions. For all procedures and SSEs that require address information for endpoint inputs, in addition to using Commerce's default address formats, you can also use the REST API to create multi-country custom address formats. This lets you create country-specific address formats to ensure that your address formats align with the requirements of any external service that you might use. This means that addresses appearing in profiles, accounts, registration requests, order addresses and more can be customized. For more complete information on creating custom addresses and understanding how to use custom address formatting, refer to the following:

- Customize Address Formats using the API in *Extending Oracle CX Commerce*
- Work with address types in *Extending Oracle CX Commerce*
- Account Details in *Using Oracle CX Commerce*
- Work with account addresses in *Using Oracle CX Commerce*
- Work with account registration requests in *Using Oracle CX Commerce*

The integration SSE .zip files used in this integration are called

- `CustomerAccountModel-store.zip` – the store version
- `CustomerAccountModel-agent.zip` – the agent version

These SSEs contain the logic for sending data from Commerce to Oracle Engagement Cloud via Oracle Integration Cloud as well as receiving updates or changes from Oracle Engagement Cloud back to Commerce.

The URLs for these look something like the following:

```
https://<admin_host>:<admin_port>/occs-admin/sse/customer-account-model/
CustomerAccountModel-store.zip
```

```
https://<admin_host>:<admin_port>/occs-admin/sse/ customer-account-
model/CustomerAccountModel-agent.zip
```

Before you install the integration server-side extensions, first make sure your custom `Node.js` server is associated with your Commerce environment. You can check this using the `GET /ccadmin/v1/extensionServers` endpoint in the Admin API. Refer to Learn about the APIs for more complete information on how to do the tasks described in this section.

The response lists the extension servers associated with your Commerce environment. If there are no extension servers associated with your environment, use the `POST /ccadmin/v1/extensionServers` endpoint to associate your extension server.

The response lists the extension servers associated with your Commerce environment. If there are no extension servers associated with your environment, use the `POST /ccadmin/v1/extensionServers` endpoint to associate your extension server.

To install the extension, you use the `POST /ccadmin/v1/serverExtensions` endpoint. Specify the Content-Type as `multipart/form-data` and include a reference to the file in the body of the request. For example, the request header might look like this:

```
POST /ccadmin/v1/serverExtensions  HTTP/1.1
Content-Type: multipart/form-data
Authorization: Bearer <access_token>
```

The request body should consist of the `OCC/CustomerAccountModel-store.zip` file, uploaded as multipart form data.

The response should look similar to this:

```
{
    "result": {
        "unzipped": false,
        "failedImages": 0,
        "allImagesFailed": false,
        "failedImagesReasons": {},
        "modifiedImages": 0,
        "newImages": 1,
        "assignedImages": 0
    },
    "success": true,
    "links": [
        {
            "rel": "self",
            "href": "http://myserver.example.com:7002/ccadmin/v1/
serverExtensions"
        }
    ],
    "token": "d63c663af7f15_cd3d"
}
```

Finally, you must make changes to the server-side extension's `config.json` file by providing correct URLs to complete the SSE configuration portion of the integration.

The typical steps used for working with the SSE code and making changes to the `config.json` include the following:

- Obtain the correct SSE zip file.
- Extract the SSE zip file.
- Edit and save the `config.json` file.
- Zip the files using the same zip file name as the original.

The following configuration information must be added to the `config.json` file for both the store and agent model:

```
"hostname": "icshostname.example.com",
"port": "7003",
"timeout": 50000,
"username_env_var": "CRM_USERNAME",
"password_env_var": "CRM_PASSWORD",
```

```
"QUERY_CONTACTS": "/ic/api/integration/v1/flows/rest/
OCC_OEC_GET_PROFILE_SSE/1.0/contacts",
"CREATE_CONTACT": "/ic/api/integration/v1/flows/rest/
OCC_OEC_CONTACT_CREATE_SSE/1.0/contacts",
"QUERY_ACCOUNTS": "/ic/api/integration/v1/flows/rest/
OCC_OEC_GET_ACCNT_DETLS_PROF_SSE/1.0/accounts",
"CREATE_ACCOUNTS": "/ic/api/integration/v1/flows/rest/
OCC_OEC_ACOUNT_CREATE_SSE/1.0/contacts/{currentContactId}/accounts",
"UPDATE_ACCOUNT": "/ic/api/integration/v1/
flows/rest/OCC_OEC_ACCOUNT_UPDATE_SSE/1.0/contacts/{currentContactId}/
accounts/{accountId}"
```

All of the URLs (paths) specified starting from the "QUERY_CONTACTS" to "UPDATE_ACCOUNT" keys in the config.json file just shown are coming from Oracle Integration Cloud and are necessary for a successful integration activation between Commerce and Oracle Engagement Cloud as described in the next section.

Understand the primary Customer Account Model SSE endpoints and flows

The integration SSE just described is used to transfer information about the customer account model for a logged-in shopper or to update the customer account model when required. This is done via endpoints found in the SSE. The following provides additional information on these endpoints:

- CREATE_ACCOUNTS endpoint - This endpoint is triggered if the Query Accounts endpoint does not return any accounts for the shopper. The inputs for this endpoint are:
  - User Token for the logged-in shopper.
  - Account Type
  - Account Name
  - Primary Contact
  - Billing Profile(s)
  - Address(es)
  - Contact ID(s)
  - Contact Role(s)

  The returns for this endpoint are the accounts, roles, addresses, and business profiles now associated with the shopper.

Note: A request can be made to the Create_Accounts endpoint at any time. It is not dependent on the response from Query_Accounts (see below). Typically, however, it makes most sense to call Create_Accounts when a contact does not have any accounts. Also, this endpoint creates a new customer account even if there is an existing account for a contact.

- CREATE_CONTACT endpoint - This endpoint can be invoked at any time (not just when a shopper logs in to Commerce.
  - The input for this endpoint is the User Token for the logged-in shopper.

  The return for this endpoint is an existing contact if a matching contact is found. If not, a new contact is created and returned.

- `QUERY_ACCOUNTS` endpoint - This endpoint is triggered when a shopper logs in to Commerce and when they go to checkout for an order that contains service items.
  - The input for this endpoint is the User Token for the logged-in shopper.

  A call to create contact will return an existing contact if a matching contact is found, otherwise a new contact is created and returned. The returns for this endpoint are the accounts, roles, addresses, and business profiles associated with the shopper.
- `QUERY_CONTACTS` endpoint - This endpoint is triggered when a shopper logs in to Commerce.
  - The input for this endpoint is the User Token for the logged-in shopper. The return for this endpoint is the External Contact ID for the shopper.
- `UPDATE_ACCOUNTS` endpoint - This endpoint is triggered when a shopper saves an account address. The inputs for this endpoint are:
  - User Token for the logged-in shopper.
  - The Account ID of the account to which the billing profile is linked.
  - The new address as provided by the shopper.

  The returns for this endpoint are the accounts, roles, addresses, and business profiles associated with the shopper.

Configure Commerce

To begin the integration configuration process, Commerce first has to be configured to communicate with the CustomerAccountModel SSE which allows communication between Commerce and Oracle Integration Cloud. With this portion configured, then Oracle Integration Cloud can communicate with Oracle Engagement Cloud. This is first done through the introduction of widgets that support the Customer Account model (see the previous section) which therefore support interaction with an external Oracle Integration Cloud system. Instances of these widgets are then added to the appropriate store layout(s) to support the integration.

Based on the information provided in this document, a customer developer or system integrator should be able to design and implement their own custom widgets to suit their particular requirements or workflows needed for this integration.

By using the provided SSEs, REST APIs, and framework view models, a customer developer or system integrator can build extensions (i.e., widgets, global widgets, and application JavaScript modules) to help them meet their integration requirements

The introduction of these widgets to the appropriate store layout(s) is a two-step process:

- Import all of the necessary widgets that you have created that support the integration into Commerce.
- After having imported the widgets, add them to your appropriate store layouts.

The following steps show an example of this process:

1. Log into the administration interface and choose **Settings** from the **Menu** icon.
2. Choose **Extensions** and then **Upload Extension** to install each extension (in this case a widget).
3. Create a new Checkout layout by choosing **Design** from the **Menu** icon.

4. Select **Layout** to create the layout where you will add the new widget instances.
5. Clone the current default Checkout layout (currently Checkout Layout with GiftCard) and
    - Give the clone a name (e.g. Checkout OEC Accounts Integration).
    - Make it the default layout by selecting Make Default Layout.
    - Click Save.
6. Switch to **Grid** view and
    - Remove the existing Checkout Cart Summary widget.
    - Add a new instance of the Checkout Cart Summary OEC Accounts Integration widget.
    - Remove the existing Checkout Order Summary widget.
    - Add a new instance of the Checkout Order Summary OEC Accounts Integration widget.
7. Publish changes by choosing Publishing from the administration menu.
8. Choose **Publish** and then **Publish Now** to publish your widget changes.

Understand additional configuration options

The integration configuration described provides one example of how it can be set up. The following items suggest some additional pieces that can be used outside of the example:

- Oracle CPQ (Configure, Price, Quote) can be used to configure new commerce items as a possible enhancement to this integration. Oracle CPQ could act as the primary asset data source and be queried for existing assets and would support asset operations such as suspend, resume, terminate, renew, and modify. Also, alternative/3rd party customer relationship management (CRM) systems could be used.
- You can create and introduce widgets that support the Customer Account model and therefore support interaction with an external Oracle Integration Cloud system. Instances of these widgets need to be added to the appropriate store layout(s) to support the integration.

5

Use the Integration

This chapter provides you with the information needed to use the features provided by integrating Oracle CX Commerce with Oracle CX for Communications.

Use the integration

This integration has many uses for Telco shoppers.

These include the following:

- Shopper information is retrieved from Oracle Engagement Cloud when the Commerce contact/shopper during a shopper browser session. This information is retained in memory for the duration of the shopper browser session (i.e. it is not persisted in Commerce). Account changes made by shoppers in Commerce are updated in Oracle Engagement Cloud. Those changes are only ever persisted in OEC, not in OCC.
- New contact creations and existing contact updates in Commerce can be updated on Oracle Engagement Cloud via Oracle Integration Cloud at login time.
- Updates to the account details in Commerce are sent to Oracle Engagement Cloud.
- Any updates to the account details in Oracle Engagement Cloud are sent to Commerce at login. For example, if there are changes made to the account outside of Commerce then they are picked up at the next login on the storefront.
- Contact profile information (personal details, address book, etc.) is kept updated on Commerce and contact account information is kept updated in Oracle Engagement Cloud. Note that only contact profile information (and, even then, only a subset of what is in OEC) is persisted in Commerce.
- Emails when a shopper account is created or modified can be triggered from Commerce. You can choose where you want these emails to be triggered from (as they exist on Oracle Engagement Cloud). For information on how to control email notifications on Commerce, see Configure Email Settings.
- Attributes of an account such as account definition and addresses are persisted on Oracle Engagement Cloud. Commerce can call SSE endpoints to maintain data on OEC (but, again, the data is only persisted on OEC). A customer developer/ system integrator could develop a Commerce user interface to maintain OEC account data using the appropriate SSE endpoints.
- Dynamic attributes created on either system can be mapped using Oracle Integration Cloud.
- The default address, email, billing address, shipping address, address book of the account and contact can be managed across both Commerce and Oracle Engagement Cloud. There is no automatic data sync between OCC and OEC, however.
  So, for example, if a shopper adds a new address to their OCC address book and makes it the default address that address will not appear in their OEC customer

account model. A customer developer or system integrator would have to call the correct SSE endpoints to send the data to OEC.

- Contacts in Oracle Engagement Cloud can be updated on Commerce at login time via Oracle Integration Cloud.
- Contact mapping to a specific site can be maintained with a site equivalent entity on Oracle Engagement Cloud (i.e., a territory).
- Emails can be triggered for a new contact registrations (triggered from Commerce).
- Emails can be triggered for a new account registrations (triggered from Commerce). This includes emails based on all the states including accept, reject and in progress.
- All profiles can be mapped under a family, a corporation, or any group like (e.g., friends).