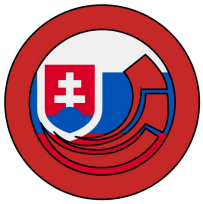
Commerce Connect Components

Peter Prochazka

Commerce Connect Components Documentation

Compiled from official Sitecore Commerce Connect Components Documentation.

About author:



Peter Prochazka (@chorpo / tothecore.sk)

Version 1.0 / 5th July 2018

More Sitecore guidelines and Sitecore related topics can be found on my blog tothecore.sk.

You can find them also directly in my github repositories.

Table of Contents

Commerce Connect Components

Understand Sitecore Commerce Connect and its components.

Introducing Commerce Connect

Sitecore Commerce Connect is the Sitecore commerce API for storefront developers and is an integration layer between a front-end webshop solution and a back-end external commerce system. The back-end system can be any e-commerce system for which Commerce Connect connectors have been created by Sitecore or a third-party vendor.

The Commerce Connect service layer has two purposes:

- Integrating one or more webshops with one or more external commerce systems.
- Applying the unique customer engagement features of Sitecore in e-commerce solutions, regardless of the external commerce system being used.

Commerce Connect enables e-commerce solutions to use customer engagement features such as tracking customer behavior, acting on it with personalization, and following up with engagement automation and reporting. This is facilitated by having common e-commerce related engagement functionalities integrated into the Commerce Connect framework.

Scenarios supported by Commerce Connect

Commerce Connect is designed for end-user business scenarios, including:

- Business-to-consumer (B2C) sales of tangible goods, digital goods, or online service delivery.
- Business-to-business (B2B) scenarios that include:
    - Advanced product pricing.
    - Customers having multiple users acting on their behalf.
    - Sales agents acting on behalf of multiple customers.
    - Multiple shopping carts per customer or user.

Target audience

People who work directly with Commerce Connect include:

- Sitecore partners, developers, and customers who use the Sitecore Experience Platform to build their e-commerce solutions.
- Sitecore technology partners or e-commerce system vendors who want to develop connectors to integrate their commerce systems with Sitecore.

The Commerce Connect framework design

The design of the Commerce Connect framework is based on the following principles:

- Simplicity – the features included in different e-commerce systems vary, therefore, to avoid complicating the architecture of Commerce Connect, it is designed to only support the most common scenarios, and the default domain models provided with Commerce Connect integration layers are kept to a minimum.
- Extensibility – because the domain models are kept simple, developers who integrate Sitecore with external commerce systems must customize the domain models and the request and response parameters that are exchanged with the external commerce system.
- Independence – each integration service layer is able to operate independently – there are no interdependencies. This allows different external commerce systems to handle different parts of the integration. However, some domain model objects are used across service layers.
- Abstraction – each integration service layer is abstract and therefore operates on neutral generic parameters that are not tied to any Sitecore-specific concepts or any external commerce systems.
- Pipelines – Each integration service layer uses Sitecore pipelines to host the business logic and allow for customizations and extensions.
- Fallback functionality – Commerce Connect acts as a fallback mechanism in cases where the most common e-commerce scenarios for which Commerce Connect has been designed are not supported by the external e-commerce system. In these cases, Commerce Connect acts as an intermediate storage that supports the missing scenarios and acts as a bridge between Sitecore and the external commerce system. The following are examples of scenarios that are not supported by every

external commerce system and are therefore supported by Commerce Connect:

- Individual but linked entities like customers and users and the many-to-many relationship between them. Some e-commerce systems do not support both customers and users or do not support the many-to-many relationships.
- Support for multiple shopping carts per user or customer.
- Detailed shopping cart information, for example, nested sublines.

CMS-only mode

You can run Commerce Connect without xDB enabled. In Experience Management mode, a number of features are not available; all the integration with the external system works unaltered, but the customer engagement has the following limitation:

- Tracking works in the session but is discarded after the session ends.

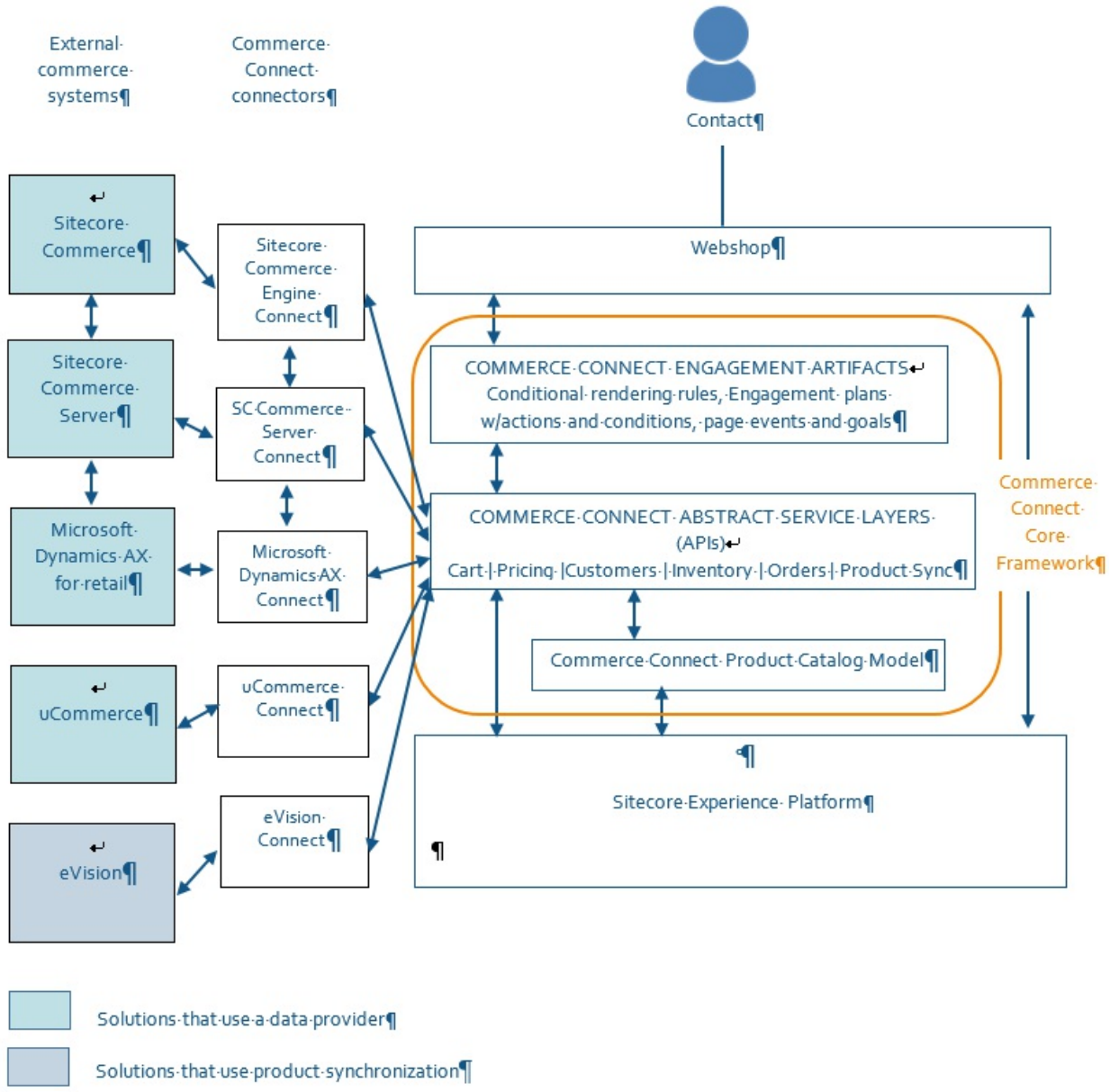
Commerce Connect framework components

A Commerce Connect solution includes the following components:

- The Commerce Connect core framework – this contains the abstract service layers (APIs), page events, goals and outcomes, conditional rendering rules, marketing automation campaigns, actions and conditions, and reports.
- Commerce Connect connectors – these hook into Commerce Connect pipelines, customize domain models, and input and output parameters, and integrate with external commerce systems.
- Commerce reports – these display data gathered by Commerce Connect.

The Commerce Connect core framework

The following diagram illustrates the Commerce Connect framework components, how they are connected, and how they integrate with the Sitecore Experience platform, the front-end webshop, and the external commerce systems:



The previous diagram illustrates the relationships between the components as follows:

- The visitor accesses the webshop.
- The webshop uses the integration service layers that in turn use the connectors to integrate with and access the external commerce server. Because the integration to the external commerce system goes through the Commerce Connect service layers, the customer engagement functionality is triggered indirectly.
- The Sitecore Experience Platform runs in the background to ensure that

both content management functionalities and engagement functionalities are executed.

- The product data model is a separate layer and may or may not be used. If you use product synchronization, the data model is based on items in Sitecore.

Note

Product synchronization is not available in version 9.0 of Commerce Connect. It will be reintroduced in an update to version 9.0.

- The connectors ensure that e-commerce functionality is available from external commerce systems.

Commerce Connect connectors

Commerce Connect cannot run as a standalone system because every external commerce system requires a unique connector. Therefore, the connectors are not a part of the core framework – connectors are provided as individual installation packages.

Developers can use Commerce Connect connectors to integrate Sitecore with one or more commerce systems. Connecters for Sitecore Commerce Server and Microsoft Dynamics AX for Retail are provided by Sitecore. Connectors for other commerce systems are provided by Sitecore technology partners. For connectors developed by Sitecore, see the Sitecore Downloads page.

Note

If data and functionality are managed by different external commerce systems, developers can use one or more connectors in a single solution. For example, you can have product data residing in one system while prices are handled in another system.

Commerce reports

Note

Commerce reports are not available in version 9.0 of Commerce Connect. They are reintroduced in version 9.0 update 2.

Both the Sitecore Experience Profile and Sitecore Experience Analytics applications contain a Commerce tab with reports that display some of the

data gathered by Commerce Connect. Sitecore Experience Profile contains reports showing information about individual contacts, whereas Experience Analytics contains reports showing aggregated data across all contacts.

Commerce Control Panel

The commerce engine and storefront configuration settings

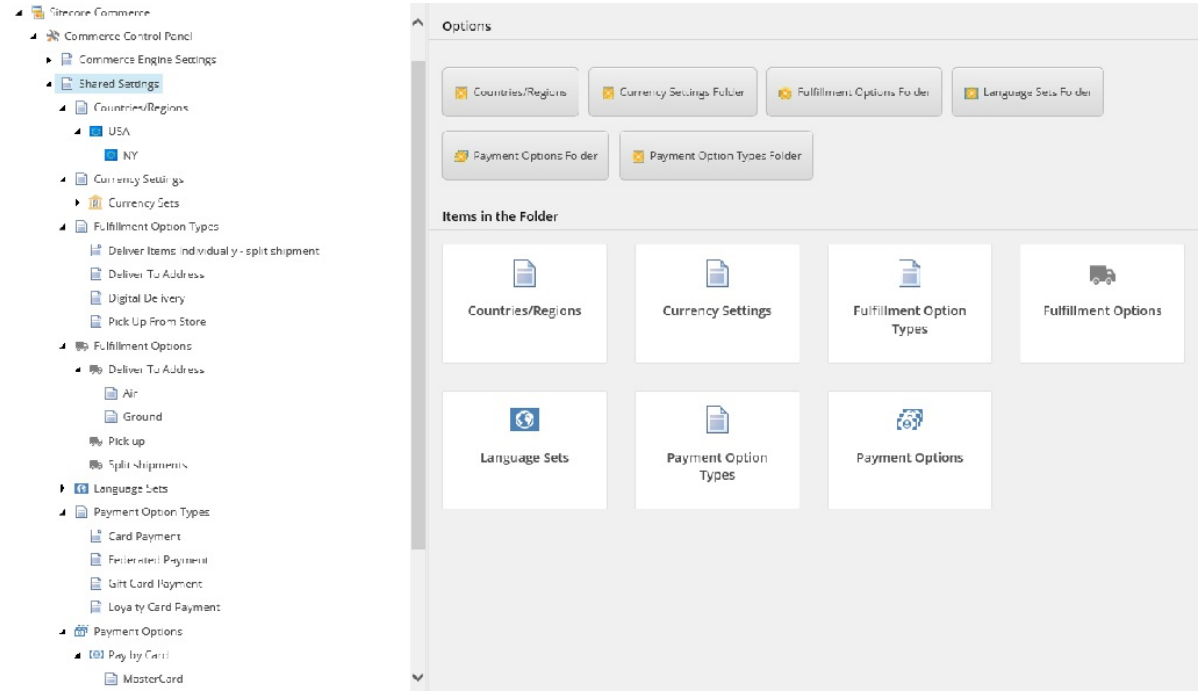
The Commerce Control Panel is installed when you install Commerce Connect. The Commerce Control Panel enables you to configure commerce engines, your storefronts, and business tools (such as Customer and Order Manager, and Pricing and Promotion Manager) by editing content in Sitecore.

You can edit the Commerce Control Panel content in the Content Editor, in: *sitecore/Sitecore Commerce/Commerce Control Panel*.

The control panel content is organized in three folders:

- Shared Settings
- Commerce Engine Settings
- Storefront Settings

Shared Settings



In the *Shared Settings* folder, you can define the options for a number of different configuration settings for your commerce engines and storefronts.

When you configure the content in the *Storefront Settings* folder, you select from among the options that you have defined in the *Shared Settings* folder.

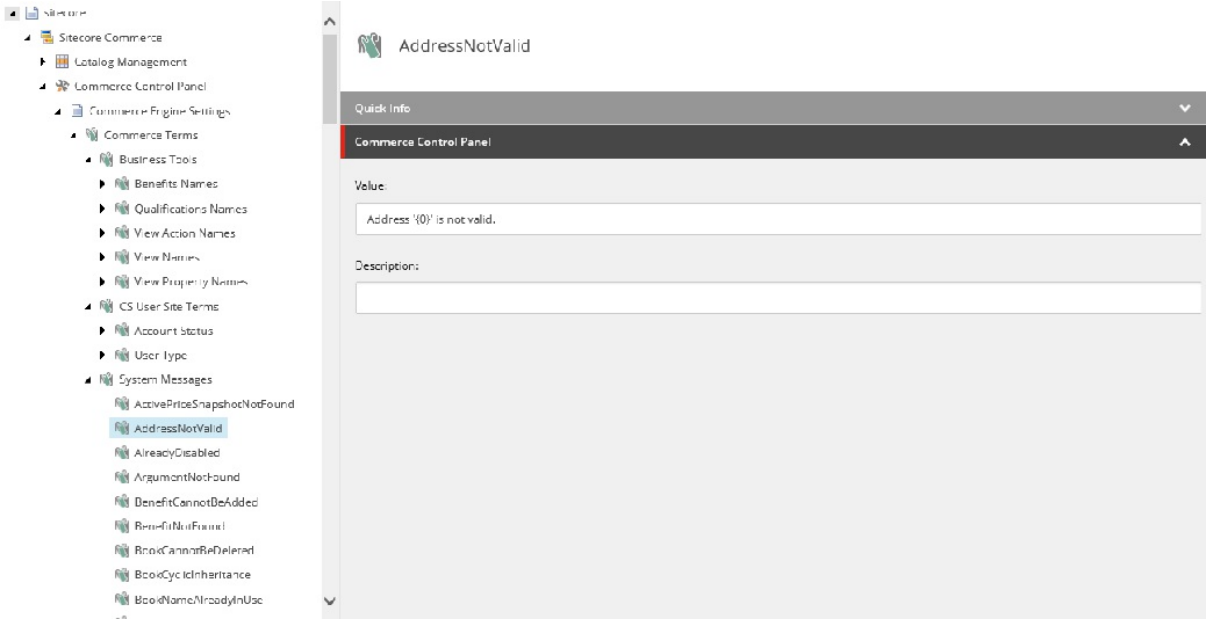
You can define the options for the following:

Setting	Subsetting	Description
Countries/Regions		Create a *Country-Region* item for every country or region that you want to use in your commerce engines or storefronts. In the Code field, enter the ISO 3166-1 country code.
	Subdivisions	Use *Subdivision* items for subdivisions of countries, such as states, provinces, or regions. You must create a *Subdivision* item for every subdivision that you want to use in your commerce engines or storefronts. In the Code field, enter the ISO 3166-2 country subdivision code.
Currency Sets		Define sets of currencies in the *Shared Settings* folder and assign them to storefronts in the *Storefront Settings* folder. For example, you might have a North American storefront that accepts payments in USD and CAD, and a European website that accepts EUR, GBP, and USD. You would then set up a currency set for each storefront. You must include the currency in the Default Currency field in the list of Selected currencies.
Fulfillment Option Types		Use to link fulfillment options to related code in Commerce Connect. The Type ID field refers to a `ShippingOptionType` value that is

		defined in the code. The following types are installed when Commerce Connect is installed, and should not be changed: • *Deliver Items Individually - split shipment* • *Deliver To Address* • *Digital Delivery* • *Pick Up From Store* You can add more fulfillment option types, but then you must add the definition of the `ShippingOptionType` values in the underlying code.
Fulfillment Options		Create one fulfillment option for each fulfillment option type that you want to use on your storefront.
	Fulfillment Methods	Create fulfillment methods for each fulfillment option that requires suboptions. For example: the *Deliver To Address* fulfillment option type could correspond to the *Ship To Address* fulfillment option, which could have two methods, *Standard Shipping* and *Express Shipping*.
Language Sets		Define sets of languages in the *Shared Settings* folder and assign them to storefronts in the *Storefront Settings* folder. For example, you might have a North American storefront that can be displayed in English, French, or Spanish, and a European website that can be displayed in any European language. You would then set up a language set for each storefront.

		You must include the language in the Default Language field in the list of Selected languages. The Fallback Language field is not currently used.
Payment Option Types		Use to link payment options to related code in Commerce Connect. The Type ID field refers to a `PaymentOptionType` value that is defined in the code. The following types are installed when Commerce Connect is installed, and should not be changed: • *Card Payment* • *Federated Payment* • *Gift Card Payment* • *Loyalty Card Payment* You can add more payment option types, but then you must add the definition of the `PaymentOptionType` values in the underlying code.
Payment Options		Create a payment option for each payment option type that you want to use on your storefront.
	Payment Methods	Create payment methods for each payment option that requires suboptions. For example: the *Card Payment* payment option type could correspond to the *Pay by Card* payment option, which could have a method for each type of card, for example, *MasterCard*, *Visa*, and so on.

Commerce Engine Settings

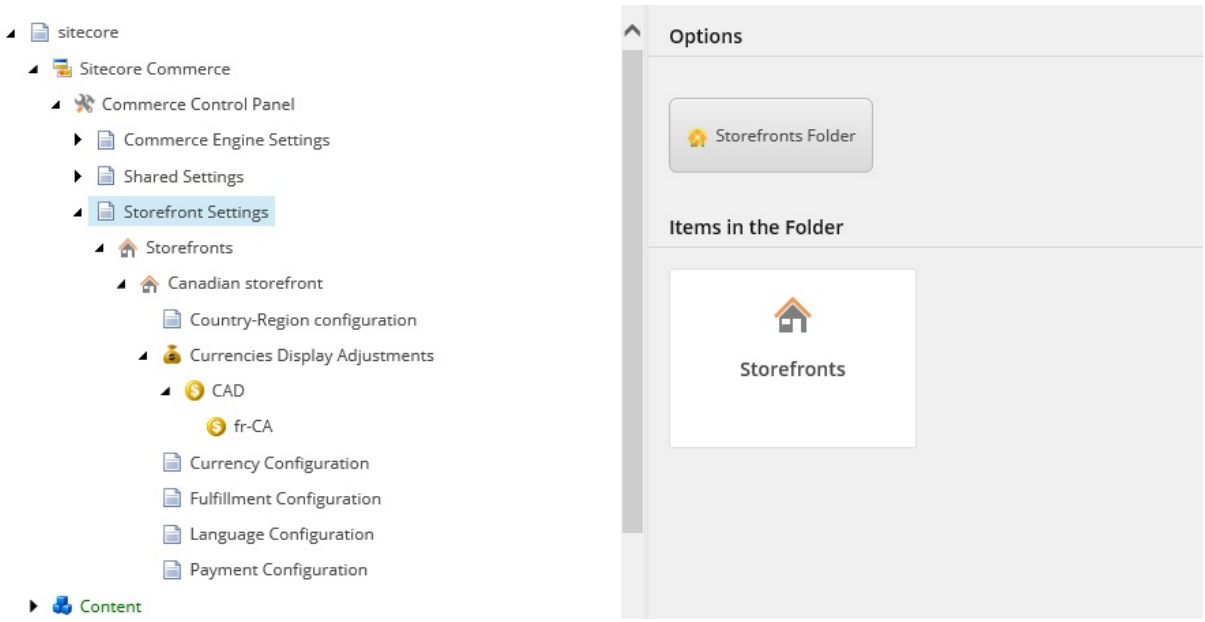


The *Commerce Engine Settings* folder contains *Commerce Term* items for the commerce engine or related business tools. Each Commerce term represents a user interface string or system message that can be translated or localized. A number of terms are created in the *Commerce Terms* folder when the commerce engine is installed; other terms are added when the business tools are installed. These terms are similar to the *Dictionary entry* items in the Sitecore dictionary (in the Content Editor: *sitecore/System/Dictionary*) but are only used for Sitecore Commerce, and are therefore stored within the *Commerce Control Panel* folder.

A *Commerce Term* item has the following fields:

- Value field – contains the localizable text string. The value can be a fixed string or, for example, a system message based on .NET string format that is interpreted by the engine. It is mandatory to fill in this field.
- Description field – contains, for example, a description or explanation of the term, or a tool tip. It can also be empty; using this field is optional.

Storefront Settings



You use the *Storefront Settings* folder to configure your storefront or storefronts with the options that you have defined in the *Shared Settings* folder.

For each storefront, you create a *Storefront* item, and under each *Storefront* item you can insert the configuration items that are needed for that storefront.

Setting	Subsetting	Description
Country/Region configuration		Assign one or more countries or regions to a storefront from the list that you defined in the *Shared Settings* folder. Note When you create a *Country/Region configuration* item, you must rename it without the /, for example, *Country-Region configuration*.
Currencies Display Adjustments		Each currency has a default currency symbol and display format (.NET standard). To use an alternative currency symbol or display format for one or more currencies, insert the *Currencies Display Adjustments*

		folder, and insert the relevant *Currency* items.
	Currency Currency Display	When you create a *Currency* item, the name for the new item must be the currency code. In the *Currency* item, you can fill in the fields to change how the currency symbol or amount is displayed. In the Currency Symbol field, enter a currency code or special character, for example USD or $. The Currency Symbol Position field enables you to determine the position of the currency symbol in relation to the amount. To determine whether the symbol appears before or after the amount and whether or not a space is inserted, enter *0*, *1*, *2,* or *3* according to the following key: *0* = $n, *1* = n$, *2* = $ n, *3* = n $ In the Currency Number Format field, you can specify a .NET culture code (for example, en-US, fr-CA, or en-CA) to determine how amounts are formatted, for example, which character is used to indicate the decimal place. Use a *Currency Display* item if a currency requires more than one display option.
Currency Configuration		Select which currency set is available on the storefront. You can select from the sets that you created in the *Shared Settings* folder.
Fulfillment Configuration		Specify which fulfillment options will be available on the storefront. Some

		fulfillment options might not be compatible with all items (for example, physical items cannot be delivered digitally), so the content of the shopping cart also influences which options are visible on the storefront.
Language Configuration		Select which language set is available on the storefront. You can select from the sets that you created in the *Shared Settings* folder.
Payment Configuration		Specify which payment options will be available on the storefront. Some payment options might not apply to all items (for example, some items might not be eligible for payment with loyalty points), so the content of the shopping cart also influences which options are visible on the storefront.

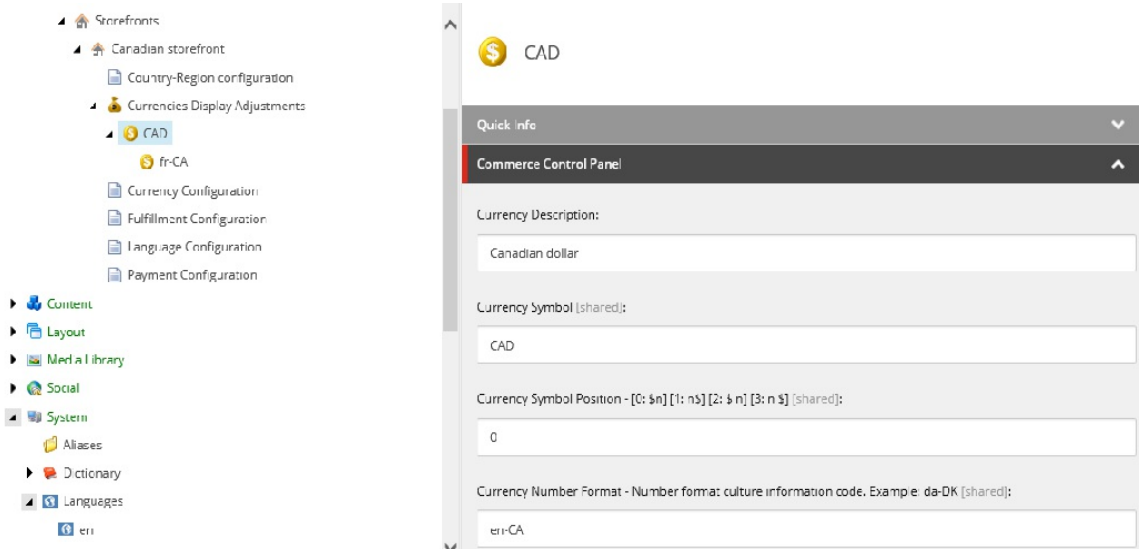
Specify multiple display rules for one currency

In the Commerce Control Panel, the *Currencies Display Adjustments* setting enables you to configure alternative currency display rules if you do not want to use the .NET default currency display rules.

In some cases, you may need to configure multiple display rules for the same currency. For example, the Canadian dollar requires a different number format depending on the language context. The symbol used for the decimal marker differs between English and French (for example 10,000.00 in English is written as 10.000,00 in French). You name each currency display rule using a culture code to determine the context in which the rule applies.

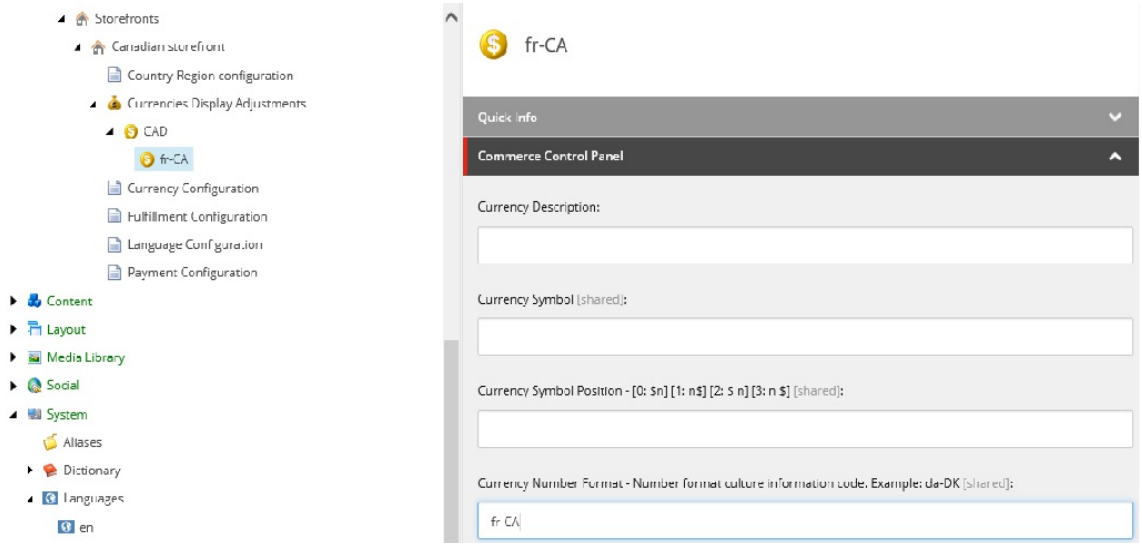
To set up multiple currency display options for one currency:

1. In the Content Editor, navigate to *sitecore/Sitecore Commerce/Storefront Settings/Storefronts*.
2. Select the relevant storefront and insert a *Currencies Display Adjustments* item.
3. Select the *Currencies Display Adjustments* item and insert a *Currency* item. Name the item with the relevant three-letter currency code (for example, CAD) and fill in the fields.



4. Select the *Currency* item that you just created and insert a *Currency Display* item. Name the item with the relevant culture code (for example, fr-CA) and fill in any fields that differ from the

parent item.



In this example, when the French language and Canadian currency are selected on the storefront, currency amounts will be displayed using the French number formatting. When the English language is selected on the storefront, currency amounts will be displayed using the general display settings for Canadian dollars, which are set to use English number formatting.

Abstract service layers

Introducing Commerce Connect abstract service layers

The Commerce Connect core framework contains a number of generic, extensible abstract service layers that work with different external commerce systems.

The Commerce Connect service layers:

- Carry essential e-commerce information between the webshop solution and the back-end e-commerce system(s).
- Encapsulate and provide an abstraction of the business logic of the external commerce system(s).
- Track and follow-up visitor behavior. They provide new goals, page events, marketing automation campaigns, and reports.

When you build an e-commerce solution using Commerce Connect, you should use the APIs in the Commerce Connect service layers instead of directly accessing the APIs in the external commerce system. The reasons for this are:

- The customer engagement is built into the service layers. If the service layer APIs are not used, the built-in commerce scenarios for personalization, reporting, and marketing automation are not triggered or deployed.
- Some external commerce systems do not support all the scenarios that Commerce Connect supports. Therefore, you should use the Commerce Connect APIs to benefit from the fallback functionality built into Commerce Connect.
- If you make customizations and extensions to the APIs to accommodate unique features in one external commerce system, the features will not necessarily be supported in another commerce system. Using the standard Commerce Connect APIs lets you replace the external commerce system with a minimum of effort.
- In case multiple external systems are integrated using the Connect pipelines, you only have to call the Connect APIs as a single point of entry, rather than calling each external system individually.

Note

Only call the external commerce system API directly if the functionality that you want does not exist or is out of the scope of Commerce Connect.

Events

The service layers handle the events that occur in e-commerce scenarios. Events are represented by a service method and corresponding Sitecore pipelines and they can be divided into front-end and back-end events.

The following are examples of front-end events that are triggered by an action on the website:

- Create a customer account
- Update a wish list
- Update a shopping cart
- Place an order

The following are examples of back-end events triggered by an action in the commerce system:

- Update product information
- Update an order status
- Update a product price

Commerce Connect service layer pipelines

Commerce Connect includes pipeline processors that interact with Sitecore. Commerce Connect connectors modify or overwrite the Commerce Connect pipelines.

Pipeline example

The following is an example of a pipeline that adds products to a shopping cart and has been modified for a specific external commerce system:

```
<Sitecore.Commerce.Cart.AddProductToShoppingCart>
  <processor type="CommerceSystem.Cart.ValidateProduct"
/>
  <processor type="CommerceSystem.Cart.CheckInventory" />
  <processor type="CommerceSystem.Cart.PutHoldOnProduct"
/>
```

```
  <processor
type="Sitecore.Commerce.Cart.RecordPageEvent" />
</Sitecore.Commerce.Cart.AddProductToShoppingCart>
```

In this pipeline:

- The processor in bold type is specific to Sitecore and provided by Commerce Connect as part of the default pipeline definition. Developers can modify or remove the processor, but the pipeline must not be removed or renamed.
- The other processors are specific to the external commerce system that is integrated with Sitecore. They are part of what constitutes the connector to the external commerce system and they are merged into the Connect pipeline from a separate configuration file that is part of the connector.

Customizing the service layers for an external commerce system

Each service layer consists of an API, one or more pipelines for each API call, and an extensible domain model:

- Each service layer has a default API that you can customize. Each service layer method takes a `Request` object and a `Result` object, either or both of which can be customized to satisfy the needs of the external commerce system. You can create an extended `Request` object by inheriting the default `Request` class and replacing it by configuration. Similarly, you can create an extended `Result` object by inheriting the default `Result` class and replacing it by configuration.
- Each service layer method executes a pipeline that can call one or more pipelines. Each pipeline can be customized by injecting, removing, or replacing processors within the pipeline.
- All domain model objects and nested domain model objects can be inherited and extended with custom properties. You can create an extended domain model object or extended nested domain model object by inheriting the default class and replacing it by configuration.

All service methods keep the existing defined signature, even when used with customized domain model objects.

Implementation details

Each service layer contains its own configuration file that defines the entities that make up the domain model, pipelines, factories, repositories, and the

service layer API implementation:

- Sitecore Factory is used to instantiate the domain model objects, so you can customize, configure and instantiate the individual entities.
- The Repository design pattern is used in processors that read and write data to Sitecore. This means that you can easily replace or customize the repository if needed.
- The default pipeline configuration in Commerce Connect contains placeholder processors in the places where processors for external commerce systems are expected. Some service layers contain comments instead. The placeholder processors are empty and do not execute any code.
- An external commerce system connector consists of processors that interact with the commerce system and a configuration file that patches the Commerce Connect configuration file, replacing the placeholder processors in the pipelines.

The Commerce Connect abstract service layers

The Commerce Connect core framework contains a number of abstract service layers that carry essential e-commerce information between the webshop solution and the back-end commerce system.

This topic describes the following service layers:

- Shopping Cart
- Inventory
- Pricing
- Customers and Users
- Orders
- Gift Cards
- Loyalty Programs
- Payments
- Shipping
- Wish Lists
- Globalization
- Catalog
- Product Synchronization

For more information, see the Commerce Connect Developer's Guide.

Shopping Cart

The Shopping Cart service layer is used to manage shopping carts from creation and retrieval to updating, locking, and removing. When the API is called, Commerce Connect collects and maintains information about the shopping cart and its content in xDB. The Shopping Cart layer also contains engagement features such as personalization rules and the Storefront Abandoned Cart marketing automation plan.

Inventory

Prices and inventory information change frequently. Inventory data in particular must be delivered in real time and prices might depend on the

customer. Dynamic information of this type is not part of core product catalog information and does not belong in Sitecore content – it belongs in a separate repository. For this reason, prices and inventory information are handled by service layers that are separate from the product data model.

The Inventory service layer provides a number of methods that enable the retrieval of stock locations and stock statuses on products. A separate product index is populated with the product-specific location and stock status.

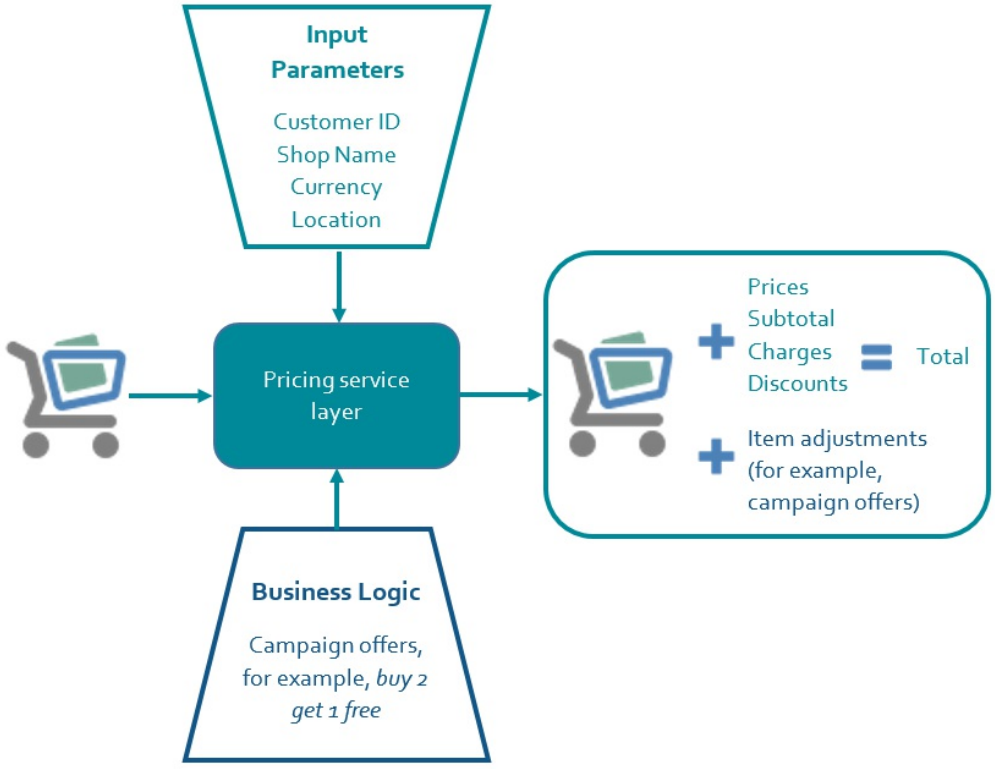
Pricing

Because prices change frequently and can be customer, date and location specific, they are not part of the core product catalog data and can be handled by an external system that is different from the system that owns the product data. For this reason, Commerce Connect contains a separate Pricing service layer.

The Pricing service layer retrieves prices of products according to the following methods:

- The `GetProductPrices` method retrieves various prices of different types for a single product, for example, list price, sale price, volume price, campaign price and customer price.
- The `GetProductBulkPrices` method retrieves prices for a list of products to limit the number of calls to the external system and improve performance. Only one price type is used at a time with bulk prices.
- The `GetCartTotal` method calculates the total and subtotal for a shopping cart.

The following diagram illustrates the `GetCartTotal` method.



The other methods are similar, but each method accepts a different set of input parameters. Other input parameters are, for example, Product ID, Price Type ID, and Quantity.

Customers and Users

A user account is used for authentication (login information) whereas a customer account contains information about who will receive and pay for orders. In B2C scenarios, the customer and user accounts generally represent the same person, whereas in B2B scenarios, the customer account is likely to represent an organization or company and the user is one person acting on the customer's behalf. Therefore, there may not be a one-to-one relationship between customers and users.

The Customers and Users service layer provides a number of methods that enable Create, Read, Update, and Delete (CRUD) actions on customers and users of the site. By default, all customers and users are stored in Sitecore but you can change this by overriding the appropriate pipelines.

Orders

The Orders service layer provides a number of methods for submitting and

retrieving orders.

Gift Cards

The Gift Cards service layer lets you retrieve a gift card for a specific user in a specific store. It also lets a visitor pay for an order using a gift card as payment.

Loyalty Programs

The Loyalty Programs service layer enables the retrieval of loyalty program and card information and the enrollment of a user in a specific loyalty program. It also lets a visitor pay for an order using a loyalty card as payment.

Payments

The Payments service layer includes methods for the retrieval of payment options and methods. The service layer implements support for federated payments. Federated payments are payments handled by a third-party payment service. The vendor does not have access to the customer payment information, such as a credit card number. It is also possible to retrieve costs associated with a given payment option (for example, automatic versus manual versus monthly invoice) and method (for example, choice of credit card).

Shipping

The Shipping service layer provides methods that enable the retrieval of shipping options and methods for a specific store or a particular user cart. It is also possible to retrieve costs associated with a given shipping option (for example, by e-mail message, pick up in store, or deliver to address) and method (for example, choice of delivery service).

Wish Lists

The Wish Lists service layer provides methods for creating and retrieving wish lists as well as adding and removing items from a list.

Globalization

The Globalization service layer contains functionality related to internationalization, such as support for cultures, currencies, languages, and so on.

Catalog

The Catalog service layer contains functionality related to product catalogs, such as raising page events and retrieving and managing products and categories. Page events can be raised for catalog browsing actions, such as visiting a product details page or category page, or performing a sort or keyword search.

Product Synchronization

Note

Product synchronization is not available in version 9.0 of Commerce Connect. It will be reintroduced in an update to version 9.0.

There are different ways to access product data in a solution that uses Sitecore and Commerce Connect. If you choose to store product data in Sitecore, the HYPERLINK "06BCBF15-6EF2-490E-B450-A687B863DE34" Product Synchronization service layer manages synchronization of core product data between the external commerce system and Sitecore. If you use another method of storing and accessing product data, you do not need to use the Product Synchronization service layer.

It is possible to do a full synchronization of products or to synchronize per product. Typically, the products are owned by the external system and made available in Sitecore as a copy, but the synchronization can be two-way, so changes made in Sitecore will be pushed to the external commerce system too.

There is a single repository containing products for all shops hosted in the solution, and data is stored in Sitecore using buckets.

The Shopping Cart service layer

The Commerce Connect core framework contains a number of abstract service layers. The Shopping Cart service layer is one of the most important service layers.

This topic describes:

- Operations in the Shopping Cart service layer API
- User-case scenarios
- Cart-based rendering rule conditions
- The abandoned cart marketing automation campaign

Operations in the Shopping Cart service layer API

The following table describes the operations contained in the Shopping Cart service layer.

Type of operation	Operations

Create, Read, Update, Delete (CRUD), and merge operations on a cart	`CreateOrResumeCart*`
	`DeleteCart*` – not necessary to call if the cart turns into an order
	`UpdateCart*`
	`SaveCart` – typically not necessary to call explicitly as it is done indirectly by calling `UpdateCart` or updating cart lines
	`LoadCart`
	`GetCarts` – supports multiple carts per customer
	`MergeCart`

CRUD operations on cart lines	`AddCartLines*`
	`UpdateCartLines*`
	`RemoveCartLines*`

CRUD operations on associated parties, for example, address and contact information	`AddParties`
	`UpdateParties`
	`RemoveParties`

Add and remove operations	`AddShippingInfo`

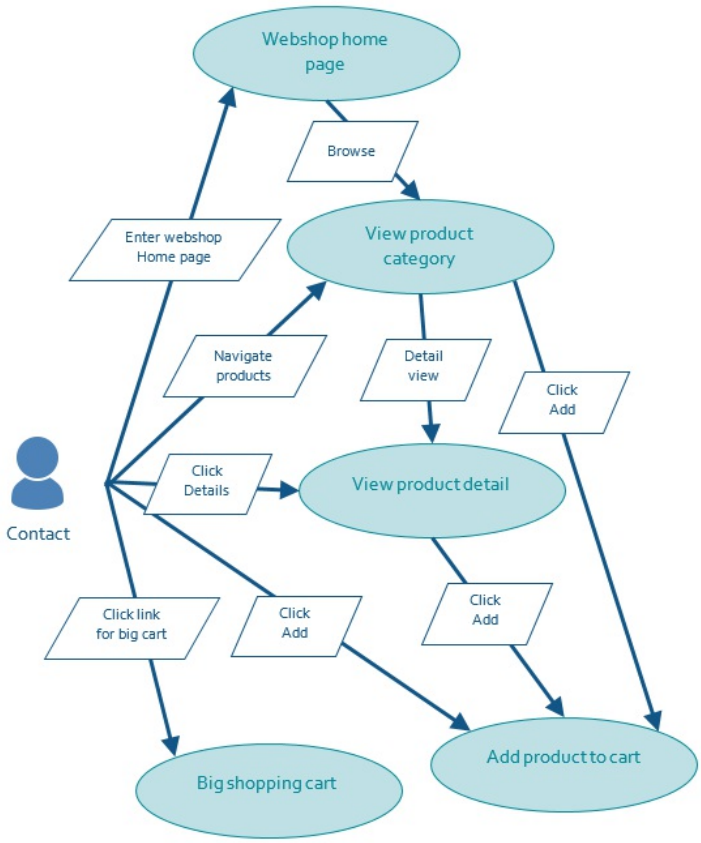
on associated shipping information	`RemoveShippingInfo`
Add and remove operations on associated payment information	`AddPaymentInfo` `RemovePaymentInfo`
Locking and unlocking cart	`LockCart*` `UnlockCart*`

Note

All operations marked with an asterisk (*) trigger a unique page event that allows the contact interaction to be carefully tracked, making it possible to report and act on these operations. Most of the events include information about the cart ID and the added products. The page events are installed when you install Commerce Connect.

User-case scenarios

The following illustrates basic shopping cart user-case scenarios:



Typical storefront usage of the shopping cart API is as follows:

- When the contact first arrives at the webshop, the `CreateOrResumeCart` method is called to retrieve the cart and to show the cart content and total in a mini-cart.
- If the contact has previously registered a profile and left a cart from previous visits, then the `MergeCart` method is called when the contact logs in.
- As the contact starts to add products to the shopping cart, the `AddCartLines` method is called.
- During the checkout process:
  - Shipping and billing addresses are added using the `AddParties` method.
  - Shipping information specifying method and referring to lines and parties is added using the `AddShippingInfo` method.
  - Payment information specifying method and referring to lines is added using the `AddPaymentInfo` method.
- While processing the payment information and reserving money on credit cards, the cart is locked using the `LockCart` method.

For more detailed information, see the Commerce Connect Developer's Guide.

Cart-based rendering rule conditions

Commerce Connect includes three rendering rule conditions that can be used to personalize the contact experience based on the products in the cart, the cart total, and the quantity of products. The cart data is read live from the external commerce system, not from the marketing platform. The conditions can be combined in a rule in a number of different ways. For example:

- Contains product: If [shopping cart contains product A] then …
- Cart total: If [shopping cart total is > Y] then …
- Item quantity: If [shopping cart total quantity of items > N] then …
- A combination of the above: If [shopping cart contains more than N quantity of product A and cart total > Y] then …

The abandoned cart marketing automation campaign

Commerce Connect 9.0.2 contains a default Storefront Abandoned Cart marketing automation campaign that lets you track shopping carts and send emails to contacts when they abandon their cart.