Drupal Commerce Nuts and Bolts

Julien DUBOIS, October 2011
@Artusamak

Evolving from Ubercart to Drupal Commerce

What can you *do* with it?

Ubercart offers a wide **feature set**, but operating outside of the core feature set is difficult at best.

“Application” mindset focusing on the modules’ **performance out-of-the-box**.

Sacrificed API functionality and i18n for UI based customizations.

What can you *build* with it?

Commerce modules offer **core e-commerce systems and components**, providing basic functionality with amazing flexibility.

“Framework” mindset focusing on loosely coupled modules and **adaptability**.

Privileges developers and site builders at the core level, administrators and reusable feature development at the profile level.

Leveraging core systems

Commerce depends *heavily* on the core fieldable entity system in Drupal 7:

Demonstration and examples:

1. **Fieldable entities**

   Building product types including “attribute” fields.

2. **Entity field query**

   Query your entities and their field data without writing SQL or knowing schemas.

   Example: commerce_product_reference.module, line 166 ff.

Leveraging contributed systems

Commerce depends *heavily* on Rules, the contributed Entity API, and Views:

Demonstration and examples:

1. **Rules**

   Configuring all sorts of conditional behavior; checkout completion rule example.

2. **Entity API**

   Use the entity metadata wrapper to easily access and manipulate field data and referenced entities on Commerce entities.

   Example: commerce_cart.module, line 770 ff.

3. **Views**

   Administrative Views for all entities on the back end and the Cart on the front.

   Views can now include area handlers and be used to build forms.

Core Commerce systems

Commerce defines its own set of systems to interact with its entities:

Demonstration and examples:

1. **Product vs. Product display**

   Products do not have default displays, but there are multiple ways for you to build custom displays: node + product reference, Views, Panels.

2. **Product pricing**

   Product sell prices are calculated through Rules via pseudo line items.

   Example: commerce_product_pricing.module, line 77 ff.

3. **Price components**

   Price calculation builds an array of price components into a price field’s data array.

   Example: commerce_tax.module, line 238 ff.

Core Commerce systems

Commerce defines its own set of systems to interact with its entities:

Demonstration and examples (cont.):

4. **Cart**

   Shopping carts are orders with special handling for refreshing prices / checkout.

   Example: commerce_cart.module, line 831 ff.

5. **Checkout**

   The form is highly configurable and updates the order upon each submission.
   Modules can define checkout panes for the drag-and-drop form builder.

   Example: commerce_order.checkout_pane.inc

Core Commerce systems

Commerce defines its own set of systems to interact with its entities:

Demonstration and examples (cont.):

6. **Payment**

   Every service defined by a payment gateway is defined as a payment method. Each method can be instantiated any number of times with different API credentials, transaction settings, and conditional availability.

   Example: commerce_payment_example.module
   Example: commerce_paypal_wps.module

7. **Complex conditions**

   Commerce defines complex conditions that you can use in place of chaining various Rules and Rules components together yourself.

   Example: Order address component comparison
   Example: Order contains a particular product

New core Commerce tools

The following tools are *new* since Commerce 7.x-1.0-beta4:

Demonstration and examples:

1. **Default entity controller**

   Commerce now defines a default entity controller that all its entities use.
   Contributed modules can extend the same controller for their entities.

   Example: commerce.controller.inc

2. **Generic entity access**

   Commerce now defines a generic set of entity access permissions and an access callback function that entities can use to perform access checks.
   Entity view access is extensible through hook_query_TAG_alter().

   Example: commerce.module, line 785 ff.
   Example: commerce_payment.module, line 949 ff.

Any questions?

julien@commerceguys.com

Commerce Guys is looking for skilled Drupal developers, themers, project managers