Commerce 2.x for the business specific

Feedback on a 2.5k person days project
Fabien Clément

• Drupal contributor for more than 10 years.
• Core contributor Drupal 8.
• Core contributor Drupal Commerce 1.x and 2.x.
• Modules contributor.
• Lead Developer for 3 years at Commerce Guys.

• Lead Developer on the project.
Summary
Summary

• Context
• Customer requirements
• Case study of business specific
• Looking on the weaknesses of Drupal Commerce 1.x
• Problems & Solutions
• Improvements
• Outcomes
Context
We deliver your ambition.

Since 2001, Niji, focus it’s business in helping businesses make a success of the digital transformation. We help our customers – e-retailers and public services – to make the all-important switch to digital technologies in support of their strategies, multi-channel distribution and internal operations.

750 employees
+ 20% growth revenue
200+ international and local customers

100 CONSULTANTS
Voice of customer
Business disruption
Digital Strategy
Technological opportunities

100 DESIGNERS
Customer & User eXperience
Brand & content design
Creative technology
Business performance

500 TECHNICAL EXPERT
Mobile, e-commerce & web
Factory, Labelized Test center
Agile & Scrum management
Smart technologies

50 EXPERTS DRUPAL
Architects, Experts,
Front dev, Project leaders
More than 30 Drupal 8 projects

Our Drupal 8 Customers
Supported by L'ÉQUIPE TECH

• Drupal & Drupal Commerce experts
  • Back-end & Front-end
• +10 years of experience in web & commerce
• Lead teams
• Drupal Commerce support and training
• Help companies for pre-sales
The project team

~2500 person days.

Spread over the duration of the project:

- 1 project manager
- 2 architects
- 1 lead developer
- 4 front-end developers
- 20 back-end developers Drupal & Symfony
- 5 testers
The customer

• A **major player** in car park in France and Belgium.
• Leader in train station parking.
• 40 years of experience in car park.
• Provide on-street and off-street parking management solutions.
• **400 car parks** in **185 cities**.
• 163 M€
Customer requirements
Corporate site

• Introduce the company
• Introduce offers
• Frequently asked questions
• Display media images/videos
• Multilingual site: EN, FR, RU
Online parking spots booking

• Parking spot booking
  • By time or at flat rate.

• Find the best booking price
  • By time.
  • Flat rate.
  • Flat rate + exceeded time.
  • All of them with promotions.

• Take care of booking fees and automatic or manual discounts.

• Subscribe to a car park subscription.

Customer requirements
Online parking spots booking

• From a dashboard, a customer can see and manage:
  • Multiple drivers.
  • Multiple vehicles.
  • See his bookings:
    • Current, past and future
    • Booked from the platform, on site or from a partner.
Data recovery from previous site

• Data recovery for:
  • Users
  • Previous orders
  • Ongoing orders
Business specific needs

• Rates and availability calculated according to criteria:
  • Options
  • Opening hours
  • Date/time range
  • Parking duration time
  • Discounts
  • Yield rules

• Services available for partners.
• Synchronization and data export.
Synchronization with ERP and CRM

- Data exports (orders).
- Data synchronization (import/export) synchronous and asynchronous:
  - Orders
  - Users
  - Drivers
  - Vehicles
  - Prices
  - Opening hours
  - Car parks
  - ...

Customer requirements
Webservices

• Rate calculation provided for partner.
• Availability of a parking spot for a duration in a datetime range.
• Search of parking nearby coords.
• Order creation by partners.
• CRUD:
  • Orders, Users, Drivers, Vehicles.
Case study of business specific
Reminder of requirements

- Multilingual website.
- High editorial requirements.
- Customizable back-office.
- Online booking.
- 2 checkouts (booking + subscription).
- Customized checkout.
- Specific product concept.
- No fixed price per product:
  - Specific rate calculation.
  - Infinite possibilities.
- Specific availability management.
- Webservices and import/export.
Multilingual website

• Drupal is multilingual.
• The commerce part can be segmented according to a language.
High editorial requirements

• Drupal is a CMS: Content Management System.
• Content is what Drupal does the best.
• Use the Paragraph module:
  • Flexibility of the display.
  • Allows the customer to choose how to display its content from a catalog of items.
  • Consistent display throughout the website.
• Use the Media module:
  • Shared media library.
Customized Back-office

• Customized back-office dashboards to:
  • List orders (booking orders, subscription orders, payments state) with data, filters and specific sorts thanks to *views*.
  • List of created entities with data, filters and specific sorts thanks to *views*.
  • Configuration of features in specific screens thanks to the *form api* and the configuration management.
  • Several data exports.

Case study of business specific
Online booking

• Using Drupal Commerce 2.x which already provides:
  • Concept of products
  • Concept of orders
  • Checkout
  • Promotions/Coupons/Discounts
  • Events for price calculation and availability
  • Events at each step of an order
  • Payment helper
2 checkouts

• Booking and subscription are both order bundles.

• Each has its own checkout process:
  • Entering several drivers for the subscription.
  • Different order summaries.
  • Different payment solutions (credit card and/or bank transfer).
  • Different emails and contents depending on the bundle of order.
Customized checkout

• No cart: 1 booking by checkout with direct access.
• Simplify inputs during the process:
  • the driver(s) and their vehicle and reuse existing data.
• Summary and highlights are specific to the current purchase.
• Specific payment (payline / slimpay).
• Display personalized information (access code, booking information).
• Send personalized emails.

Case study of business specific
Specific product concept

• A product is named here a pocket: a parking area in a given place.
• The different booking possibilities lead to an infinite number of possible products if we wanted to represent them all in terms of parking spots.
• The booking will then be qualified according to different parameters:
  • The desired booking period.
  • Some time stayed.
  • Wished options.
No fixed price per product

• A parking spot does not have a fixed price, but different prices depending on many criterias.

• The calculation of the price of a booking is based on:
  • The desired booking period.
  • The rental duration.
  • The wished options.
  • The ongoing promotions (automatic or via coupon).
  • The yield rules.

• The data specific to our booking (period, option, etc) are directly stored in our order.
No fixed price per product

• The **calculation** system is managed directly **in Drupal** (not outsourced).

• All the required data for the calculation of these rates are:
  • **Daily imported** from an **ERP** from CSV files.
  • Stored in entities related to the data models sent.

• **Commerce 2.x** provides a **service** that allows us to integrate our **own price calculation rules**.
The price is dynamic

**Solution**

- Using the Commerce 2.x service `commerce_price.price_resolver`.
- Storage of all data relevant to the calculation in entities.
- Use of this data in calculation rules.
Specific availability management

• The availability of a parking spot depends:
  • The desired booking period.
  • Wished options.
  • Opening hours of the car park.
  • Possible capacity.
  • The number of parking spots already booked over the period according to the options.

• Each car park has its own capacity per period.
• The number of parking spots already booked per option are stored in an entity.
Webservices and import/export

• **Bulk import** via Migrate:
  • Orders
  • Promotions
  • Business data (prices, car parks, opening hours, etc.)

• Expose and consume **web services** in Soap/XML and JSON
  • CRUD orders, users, drivers, vehicles
  • Availabilities
  • Rate calculation

• Export of data file. About ten different **exports in several formats**: CSV and proprietary formats.
Looking on the weaknesses of Drupal Commerce 1.x
Which would have been more laborious (or impossible)

- On-the-fly price calculation.
- Different checkouts.
- Web services.
- Cache management.
- Import/export structure of entities via the features module.
- Using Rules module instead of events.

Looking on the weaknesses of Drupal Commerce 1.x
Problems and solutions
Find the best rate / rate calculation on the display

Solution

• **Dummy order generation** to use calculation commerce features from an order:
  • Adjustments
  • Promotions
  • Price Resolver
Calls to Drupal and external services must not be unavailable

_Solution_

- Add an **abstraction and retention layer** via Symfony + RabbitMQ.
- **Drupal** only provides and manages **JSON**.
- **Symfony** application deals with the **conversion** between Drupal and external if needed.
Calls to Drupal and external services must not be unavailable
Securing critical features

Solution
• Implementation of automatic tests via PHPUnit.
• Manual functional tests by a team of testers.
Quality development in line with good practices

**Solution**

- Systematic **review** of any **code**. Use of Gitlab and merge requests. Development branch blocked and mandatory validation by the Lead dev.

- Use of **Code Sniffer**:
  - PHPCS
  - Eslint and SassLint

- **Code audit** via SonarQube.
The two payment gateways used do not exist on commerce 2.x

Solution

• Creation of two modules.

• The commerce 2.x plugin and interface system
  • shortens the time required to create payment modules
  • helps to maintain consistency between different modules.
Expose web services

Solution

• Drupal 8.x offers by **default a REST module** to expose a CRUD of the data in JSON format.

• Commerce 2.x integrates its entities perfectly into this functionality.

• **Easy creation** of new **REST resources** as needed.
Start before commerce 2.0 release

Solution

• Follow the issues.
• Talk with Commerce Guys to know what current state commerce 2.x is in.
• Help on required issues for the project.
Only lead dev knew about Drupal Commerce

**Solution**

• Training people.

• Drupal Commerce is Drupal + Symfony. Easy to move on, knowing both.

• Reviews help a lot to improve skills.
Maintain consistency between environments

**Solution**

- Use LXC to have containers by application.
- First half of project, always install to get other people stuff back.
- Second half of project, switch to update.
- Everything has to be scripted: No changed has to be made manually.

**Install and update:**

- Bash scripts to run commands and scripts.
- Drush commands to install, update.
- Import configuration with config_split and config filter.
I have an issue / I need this missing feature

Solution

• Participate to issues on drupal.org.
• Create patchs (or PR for commerce 2.x)
  • On core
  • On contrib modules
Improvements

A personal thought about what we could improve.
Price calculation

• Optimization of the price calculation engine to improve its performance (cost and time).

Secure development

• Increase automatic functional tests to avoid regressions on critical features.
• Use of BeHat or equivalent to include the customer in the tests writing.
Back-office inputs

• Management of paragraphs via a "library" more explicit than the interface provided by default.
• Improvement of the interface of nested paragraphs.
Front-Office

• More harmonious use of javascript when searching and calculating prices for a better user experience.
• Reduce page weight.
• The integration of forms in the front-office is still the black spot for front-end developers.
Outcomes
Outcomes

• A business e-commerce project outside the traditional online sales sites.
• Big points of business complexity.
• Commerce 2.x more flexible and better designed than Commerce 1.x.
• Commerce 2.x adapted to business commerce.
• An online site with an ever-increasing number of sales.
Questions ?