Université de Caen Basse-Normandie

Rapport de stage

1 avril 2015 - 30 septembre 2015

Open orchestra

Auteurs :
Amaury Lavieille

Enseignants :
François Rioult

5 août 2015

UNICAEN
université de Caen
Basse-Normandie

Remerciements

Table des matières

Introduction

Dans le cadre de ma deuxième année du master DNR2I (Master document numérique en réseau, ingénierie de l'Internet), j'ai effectué un stage chez Business & Decision au sein de l'agence Interakting à Caen.

Pour une durée de six mois, du 1er Avril au 30 Septembre 2015, j'ai étais intégré au sein de l'équipe de développement du projet Open Orchestra.

Dans un premier temps, je vais vous présenter succinctement l'entreprise. Ensuite, je vous décrirais le projet Open Orchestra, l'origine de celui-ci, ces particularités, etc . Enfin, je vous parlerais de l'organisation et des méthodes mises en place pour la réalisation du projet.

Première partie

L'entreprise

Chapitre 1

Présentation de l'entreprise

1.1 Business & Decision

Business & Decision est un groupe international spécialisé dans trois grand domaines, la Business Intelligence [1] (BI), la gestion de la relation client et enfin le e-business.

Le groupe a était créé en 1992 par Patrick Bensabat, aujourd'hui il est présent dans 15 pays et emplois plus de 2500 personnes.

Événement majeurs et évolution du groupe depuis 1992 :

1992 : Création de Business & Decision par Patrick Bensabat autour de projets de Business Intelligence

1993-1996 : Mise en place des premiers Datawarehouses [2] et applications de prévisions financières

1999-2000 : Création de la division CRM (Gestion de la relation client)

2002-2003 : Acquisition en Grande-Bretagne et au Benelux. Création d'agence régionales en France

2004 : Acquisition en Suisse et aux Pays-bas et implantation en Tunisie

2005 : Implantation aux Etats-Unis et acquisition de Metaphora [3]

2007-2008 : Création de la marque Interkating

2011 : Déploiement d'offres Cloud et Mobilité. Inauguration du Datacenter éco-responsable d'Eolas

2013 : Lancement de Datalyse, programme de recherche en Big Data. Implantation au Pérou

2014 : Création de Herewecan, agence de communication digitale, Lancement des pôles d'expertise Big Data, Transformation et Hub Mobile)

1. La Business Intelligence ou l'informatique décisionnelle représente les différentes solutions informatiques qui permettent l'exploitation des données du entreprise dans le but de faciliter la prise de décision

2. Les Datawarehouses ou entrepôt de données désigne une base de données utilisée pour stocker et structurer des données de production d'une entreprise afin de fournir des informations stratégiques pour la prise de décision

3. Métaphora est une société de service spécialisée dans la conduite du changement. Elle intervient dans toutes les étapes d'un projet pour faciliter l'appropriation du futur système d'informations par les utilisateurs finaux.

1.2 Interakting

Interakting est l'agence web du groupe B&D composé de 340 employés répartis principalement entre Caen et Paris, les équipes de l'agence interviennent sur des projets web de grande envergure (Canal +, Bnp Paribas, Inra, Moët Hennessy, PSA Peugeot Citroën, ...).

Les différents projets de l'agence sont développés avec différent outils : eZ publish (système de gestion de contenu créé par l'entreprise eZ Systems As), PHP Factory (plateforme réalisé par conjointement par Interakting et Zend Technologies) et de plus en plus avec Symfony2 (framework php écrit par SensionLabs) avec notamment Open Orchestra.

Deuxième partie

Open Orchestra

Chapitre 2

Le projet

2.1 Présentation

Open Orchestra est un CMS (Content Management System) open source en développement maintenant un peu plus d'un an. TODO : Encore en dévellopement v1 septembre

Il est basé sur le framework PHP MVC Symfony 2 et MongoDB. Open Orchestra offre bien-sûr les fonctionnalités attendus par tous CMS : gestion de contenu, médiathèque, contribution de page, gestion de version, utilisateurs, « worklow », rôles, multi-site, multi- langue, multi-device , etc.

La grande particularités d'Open Orchestra est son faible couplage du code mais aussi au niveau des solutions techniques, c'est-à-dire que l'on peut facilement remplacer ou étendre un de ses composants. Je reviendrais plus en détail sur ce point dans une section ultérieur.

2.2 Architecture

Une des particularités d'Open Orchestra est que le « Back-office » et le « Front office » peuvent être dissociés dans deux applications Symfony différente.

Ce découpage a deux nombreux avantages, tous d'abord un seul « Back-office » peux gérer plusieurs sites (« Front-office »), de plus cela permet de mettre « Back-office » et le « Front-office » sur des serveurs différents, la seule condition est qu'ils doivent utiliser la même base de données.

Pour finir, Open Orchestra utilise une API RESTFull [1] pour accéder, gérer ces différentes entités (pages, utilisateurs, contenus, etc).

Le schéma présente l'architecture d'Open Orchestra avec un « Back-office » qui gère plusieurs « Fronts ».

1. REST (Representational State Transfer) est un style d'architecture qui définie différentes réglés (ressource identifié unitairement, utilisation des verbes HTTP POST, DELETE, PUT, GET) pour accéder et manipuler des ressources

Chapitre 3

Fonctionnalités

Dans ce chapitre je vais vous présenter les fonctionnalités d'Open Orchestra. Bien sur, je ne vais pas détailler tous les fonctionnalités juste les points clés pour une meilleur compréhension du projet.

3.1 Blocs

Dans Open Orchestra tous les éléments visible en front sont représenté par des blocs. Enfaite un bloc est simplement une entité avec un certains nombre d'attribut qui varient selon le type de block, chaque type de bloc sont indépendant, c'est à dire que eux seul connaissent leurs attributs, leurs façon se d'affiche en front ou en back-office ou encore la façon dont on peut les contribuer en back-office.

Pour permettre cette indépendance, on utilise le design pattern stratégie. Le pattern stratégie permet de rendre une famille d'algorithmes interchangeables et ainsi la possibilité d'exécuter un traitement spécifique selon le contexte.

Comme le présente le schéma pour l'affichage en front d'un bloc, chaque bloc possède une classe qui indique comment doit s'afficher le bloc pour cela elle a deux méthode, la première qui indique le bloc qu'elle supporte (`support`) et la méthode `show` qui fournis le rendu, d'un autre coté il y a une classe que l'on appelle `manager` qui est la seule a connaitre toutes les stratégies des différents bloc.

Ainsi, lorsque que l'on désire afficher un bloc, il suffit de demander au manager d'afficher ce bloc, ce dernier va chercher parmi les stratégies qu'il connait laquelle supporte (méthode `support` le bloc et s'il en trouve une alors il exécute la méthode show de la stratégie.

Par défaut, le CMS propose de nombreux type blocs comme par exemple un bloc pour liste un type de contenu, un bloc pour afficher un texte quelconque formaté, pour afficher un carrousel ou un média, un menu, une carte, un bloc de contact, etc.

Bien-sûr, il possible pour un intégrateur [1], de créer son propre type de bloc, il lui suffit de développer les différentes stratégies (affichage en front et back-office, formulaire en back-office, etc) nécessaire a un bloc.

1. Dans le cadre de ce rapport un intégrateur indique une personne ou une équipe qui utilise Open Orchestra et l'étende pour des besoins particulier

3.2 Nodes

Un des points central d'un CMS est les pages. Sur Open Orchestra les pages sont identifiés comme des noeuds (« nodes »). Les nodes sont simplement des conteneurs qui contient des zones. Les zones permettent d'organiser la pages, elles contiennent des sous-zones ou des blocs ce qui permet un découpage fin de la page pour simplifier sont organisation.

Il existe trois types de nodes :

— Les nodes qui représentent les pages visibles en « front ».

— Les « node tranverse » qui contiennent les blocs transverses, c'est-à-dire les blocs qui sont communs à plusieurs pages comme le bloc « header » ou encore le bloc « footer ».

— Le dernier type de nodes sont ceux qui permettent de contribuer les pages d'erreurs (404, 503) d'un site.

Le schéma montre un exemple typique de page avec trois zones, le header qui contient un bloc menu, le footer un bloc footer et une zone principale découpé elle-même en deux zones avec un bloc de contact pour la zone de droite et un bloc de texte à gauche

3.3 Content type

Un autre aspect important d'un CMS est les contenus avec Open Orchestra il est facile de créer des types de contenus (news, client, etc, ...).

En effet, Open Orchestra propose une approche graphique grâce a un formulaire pour créer ou éditer des types de contenus comme l'illustre la figure. Un type de contenus est composé de différent champs avec différent option, par exemple pour un champ de type texte il peut y avoir une option pour limiter le nombre de caractère ou encore une valeur par défaut.

Par défaut, Open Orchestra offre quelques types de champs (date, texte, monnaie, média, email, entier, zone de texte riche, etc). Comme pour tous les composants d'Open Orchestra il est possible pour les intégrateurs d'ajouter d 'autre type de champ avec ces options.

Chapitre 4

Caractéristiques

4.1 Performance

Open Orchestra a était développé pour supporter de grosse charge de trafic. Pour cela le CMS exploite au niveau du front l'ESI (Edge Side Includes) couplé a un reverse proxy [1] tel que Varnish.

L'ESI est un langage de balisage HTML qui permet de diviser une page en différent éléments dont les rendus est fait dans différentes requêtes par le serveur web. Ce qui permet d'avoir un cache HTTP sur les différents éléments et ainsi rafraichir seulement les éléments obsolètes de la page et non toute la page.

Comme nous l'avons vue dans la section sur Open Orchestra les pages sont déjà découpé en différent blocs, l'utilisation de l'ESI permet nettement l'amélioration des performances.

Si nous reprenons l'exemple de notre page (Numéro), le schéma présente le processus effectué par le serveur dans le cas où les blocs `header` et `footer` sont déjà en cache lorsqu'un utilisateur demande la page.

http ://naholyr.fr/2011/03/varnish-edge-side-includes-cache-partiel/

4.2 Modularités

Comme je vous l'explicais en introduction un des particularités d'Open Orchestra est sa modularité. Pour cela, les différents composants du CMS sont répartis dans différents bundles [2]

- **open-orchestra-base-bundle** : Classes communes au back-office et front-office
- **open-orchestra-base-api-bundle** : Api d'Open Orchestra
- **open-orchestra-base-api-mongo-model-bundle** : Implémentation des models nécessaire à l'api pour `MongoDB`
- **open-orchestra-cms-bundle** : Logique du back-office
- **open-orchestra-front-bundle** : Logique du front-office
- **open-orchestra-display-bundle** : Logique d'affichage des blocs en Front-office

1. Un reverse proxy est un serveur qui traite les requêtes en amont du serveur web. L'intéret d'un rervese proxy est multiple gestion de cache, chiffrement, répartition de chage

2. Sous Symfony2 un bundle est un ensemble de fichiers structurés(contrôleur, entité commande, listener, formulaire) qui permettent d'implémenter une ou des fonctionnalités qui dans l'idéale peut être réutilisé dans différent projets

— **open-orchestra-media-bundle** : Médiathéque d'Open Orchestra
— **open-orchestra-media-admin-bundle** : Administration de la médiathéque d'Open Orhcestra
— **open-orchestra-model-interface** : Interface des models utilisé par les autres bundles
— **open-orchestra-model-bundle** : Implémentation des models pour `MongoDB`
— **open-orchestra-user-bundle** : Gestion des utilisateurs
— **open-orchestra-workflow-function-bundle** : Système de fonction worklow pour la gestion en back-office

Ainsi ce découpage fin a de nombreux avantages. Une application front-office n'a aucun intérét à charger toute la logique du back-office ainsi grâce a ce découpage chaque application (front-office ou back-office) charge uniquement les composant qui lui sont nécéssaire.

De plus, cela permet de désactiver facilement une fonctionnalités. Par exemple, si un intégrateur n'a pas bessoin de gérer des médias il peut alors désactiver la médiathèque en n'utilisant par les bundles (`open-orchestra-media-bundle` et `open-orchestra-media-admin-bundle`).

Un autre avantage est la facilité de remplacement d'un composants. Par défaut Open Orchestra utilise `MongoDB` comme système de gestion de base de données mais si un intégrateur veut utiliser une autre système comme `Mysql` alors il lui suffit de alors d'écrive son propre `open-orchestra-model-bundle` qui utilise biensûr les différentes interfaces de `open-orchestra-model-in` mais adapté a `Mysql`.

Troisième partie

Organisation du développement