Découverte du framwork Apache OFBiz. Dévéloppement d'une API HTTP, basé sur le style architectural REST et inégration dans un contexte de projet client.

Auteur :
Artemiy Rozovyk

Tuteur de stage :
Mathieu Lirzin
L'enseignant référent :
Florent Foucaud

24 mai 2019

Table des matières

Remerciements

Merci tout le monde !

Chapitre 1

Introduction

Le présent document expose le travail effectué lors du stage de fin de licence au sein de la société Néréide. Ce stage se décompose en deux parties : De prime abord, il a pour le but de se familiariser avec la suite d'applications libres pour l'entreprise Apache OFBiz et son utilisation dans le contexte de la société d'accueil.

Tout compte fait, il consiste à intégrer un système permettant la définition des API HTTP du style REST, ainsi que la modification d'une API existante afin de fournir une preuve de concept consistante.

Le travail a été effectué en étroite collaboration avec le principal concerné : la communauté Apache, ce qui a contribué à une meilleure cohérence entre le travail réalisé et les besoins des utilisateurs.

Dans un premier temps nous allons présenter l'entreprise d'accueil ainsi que faire une description de l'outil principal utilisé. Dans un deuxième temps nous exposerons la démarche qui a permis une compréhension suffisante du Framework OFBiz nécessaire à la partie finale du stage qui sera décrite dans un troisième temps.

Chapitre 2

Contexte du stage

Intro

2.1 Entreprise

2.1.1 Presentation générale

Néréide est une société de services en logiciels libres crée en 2004 SCOP SARL
parler de méthodes agiles. transparence interne totale ; fonctionnement démocratique (SCOP, co-gérance tournante) ; implication des salariés dans tous les domaines de décision ; salaire unique

2.1.2 Domaine

Retail

2.1.3 Activité

Transparence libre entreprise

Dévéloppement spécifique Maintenance et support applicatif Administration système

2.1.4 Projets

Décathlon Dejbox

2.2 L'architecture de OFBiz

2.2.1 Vue d'ensemble sur l'architecture

DSL en XML, servlettes, jdbc ...

2.2.2 Composants

2.2.3 Container

2.2.4 Web applications

2.2.5 Entity engine

2.2.6 Service engine

2.2.7 Screen engine

2.2.8 Fonctionnel métier

2.3 Sujet de stage

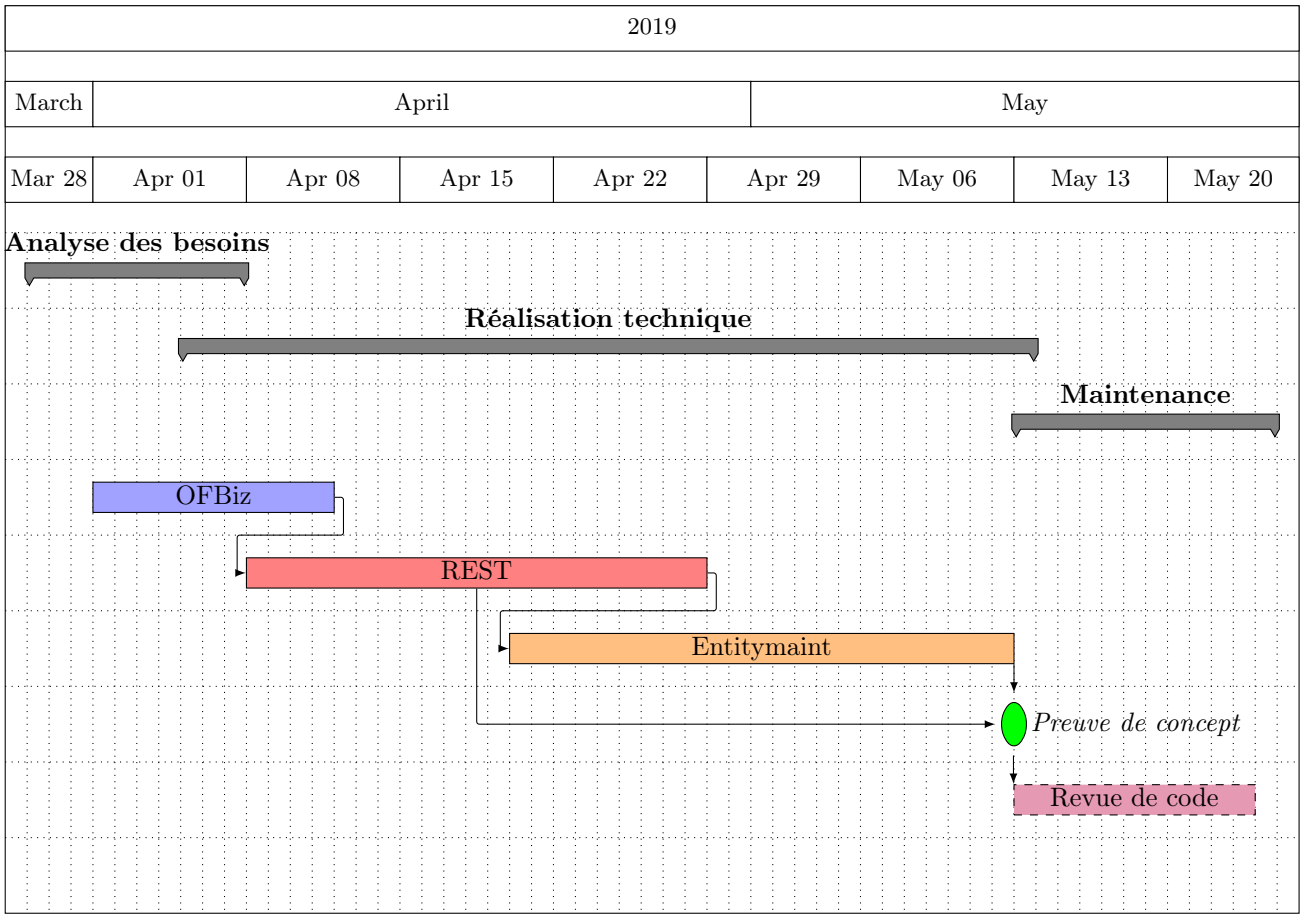
2.3.1 API REST au sein d'OFBiz

Chapitre 3

Travail réalisé

3.1 Aperçu général

Voici la chronologie du travail réalisé en entreprise.



3.2 Environnement

3.2.1 Installation de l'environnement

Avant tout, mon intégration a commencé par l'installation du poste de travail suivi par une discussion sur le choix de distribution Linux [1] , la configuration des outils utilisés par l'entreprise ainsi que par la mise en place des accès aux ressources internes [2]

3.2.2 Conventions

3.2.3 Formation développeur générale

3.2.4 Jira

3.2.5 Approfondissement de Git

3.2.6 Découverte de communauté libre Apache

1. Une question qui a, apparemment, beaucoup d'importance.
2. Contenaient, à mon avis, des information sensibles, mais cela s'explique par le principe de transparence 2.1.3

3.3 Prise en main d'OFBiz

3.3.1 Premier plugin

3.3.2 Projets existants et leur structure

3.3.2.1 Décathlon

RFID et tout ça

3.3.2.2 Dejbox

Pierre et Antoine ont tout géré

3.3.3 Problématique vis-à-vis du développement

What is "fonctionnel"

3.4 Analyse de l'existant

3.4.1 ControlServlet

3.4.2 Mécanisme de résolution des URI

3.4.3 Filtres

Delegateur et Dispatcher

3.5 Analyse des besoins et attentes de la maîtrise d'ouvrage

3.5.1 Structure générale des application web

Les enjeux, les problématiques les solutions, *COURS MAURIZIO*

3.5.2 API en cours

RPC

3.5.3 Controleur

<request-map>...

3.5.4 Besoins d'évolution

Avenir *Discussion communautaire*

3.5.5 Representational state transfer

3.5.5.1 Histoire

Roy Fielding

3.5.5.2 Principe

*Détailles du cours de Maurizio : idempotence, navigabilité par hyperlink, notion de ressource etc.

3.5.5.3 Avantages

3.5.5.4 Examples d'API du style REST

API REST de Twitter, SoundCloud, Wiktionnaire,
les différences entre la définition de Roy Fielding et l'implémentation de ces dernières

3.5.6 Implementations existantes

3.5.6.1 Camel

3.5.6.2 JAX-RS

Tentative d'intégration —
ServletJaxRS fonctionnelle
Particularités techniques (annotations)
Conflit politique car n'est pas dans le même esprit de l'existant.

3.6 Réalisations techniques

3.6.1 Librairie CXF

Problèmatique avec les dépendances supplementaires : Tika contient déjà le CXF

3.6.2 Choix vers URITemplate

description de classe

3.6.3 *OverrideView()* et le conflit avec les URI segmentées

3.6.4 Choix d'intégration en parallèle avec le système existant

3.6.5 Nouveau contrôleur

3.6.5.1 Compromis pour les conflits d'URI

3.6.6 Modification de la partie "Administration : gestion des entités" (entitymaint)

3.6.6.1 Choix de la partie illustrative

3.6.6.2 PUT vs POST

3.6.6.3 Clés composées

3.6.6.4 Formulaires génériques

Create update dans un même formulaire.

3.6.7 Stateless

3.6.7.1 Les réalisation par la communauté

Jaques Le Roux Token en gardant la session.

3.6.8 RESTClient pour la communauté

3.6.8.1 Généralisation de code

3.6.8.2 Correction d'incohérences

Chapitre 4

Conclusion

4.1 Lien avec les connaissences obtenu lors de la formation universitaire

METTRE DANS LA CONCLUSION

4.1.1 MVC

4.1.2 Servlet

4.1.3 FreeMarker -JSP

4.1.4 Notion d'entié - Symfony

4.1.5 Routage