**Rapport d’étude préliminaire**

MultiMIF 2015 - 2016

Symp’le Event Manager (SEM)

Membres du projet

|  |  |  |
| --- | --- | --- |
| Alix Gonnot  Clément Blaise | Jordan Martin Jules Sauvinet | Landry Mélaine Lebatto  Tom Dusseaux |

*Version et date du document : V1 - 11/11/2015*

Sommaire

[1. Introduction 2](#_Toc435039973)

[1.1. Contexte 2](#_Toc435039974)

[1.2. Objectif 2](#_Toc435039975)

[1.3. Contraintes 2](#_Toc435039976)

[2. Cas d’utilisations 3](#_Toc435039977)

[2.1. Création d’un compte 4](#_Toc435039978)

[2.2. Création d’une conférence 5](#_Toc435039979)

[2.3. Enrichissement de la description d’une publication 5](#_Toc435039980)

[2.4. Import des données 6](#_Toc435039981)

[2.5. Export du dataset 7](#_Toc435039982)

[2.6. Edition de données 7](#_Toc435039983)

[3. Méthode de travail 8](#_Toc435039984)

[3.1. Méthode agile SCRUM 8](#_Toc435039985)

[3.2. Organisation des sprints 8](#_Toc435039986)

[3.3. Rôles 9](#_Toc435039989)

[4. Outils et technologies 10](#_Toc435039990)

[4.1. Service web REST 10](#_Toc435039991)

[4.2. Application web 10](#_Toc435039992)

[4.3. Base de données 10](#_Toc435039993)

[5. Architecture du projet 11](#_Toc435039994)

[5.1. Schéma global 11](#_Toc435039995)

[5.2. Diagramme de classe 12](#_Toc435039996)

[6. Risques Identifiés 13](#_Toc435039997)

[6.1. Choix de la modélisation des données 13](#_Toc435039998)

[6.2. Utilisation de Spring 13](#_Toc435039999)

[6.3. Utilisation d’un SGBD orienté graphe 13](#_Toc435040000)

[6.4. Développement de fonctionnalités de faible importance 14](#_Toc435040001)

1. Introduction
   1. Contexte

Afin de nous permettre de mettre en œuvre les connaissances acquises au cours de ce semestre dans différents domaines, un projet requérant l’utilisation de toutes ces notions nous a été confié.

Ce projet met en jeu 4 unités d’enseignements :

* **MIF13 - Développement d’Applications Web :** Nous utiliserons différentes technologies web afin de développer une application client/serveur.
* **MIF18 - Base de Données Appliquées au Web :** Le serveur devra interagir avec une base de données.
* **MIF16 - Conduite de Projet :** Afin de mener le projet à bien, il nous faudra faire preuve de communication et d’une bonne gestion des différentes tâches dans l’équipe.
* **MIF17 - Génie Logiciel :** L’utilisation des bons outils (Diagramme de classe, Cas d’utilisation, …) nous permettra d’avoir une meilleure vision des tenants et des aboutissants du projet.

Nous sommes 6 étudiants à travailler sur ce projet, et nous disposons de 6 semaines pour le réaliser.

* 1. Objectif

Le but du projet est de créer un *backend* pour Sympozer, une application web permettant de gérer des évènements de type Conférence.

Notre application devra pouvoir gérer l’inscription et la connexion des utilisateurs, la modification des données d’une conférence, l’exportation ainsi que l’importation de certains fichiers décrivant les données relatives à une conférence.

* 1. Contraintes

La réalisation du projet sera soumise à différents types de contraintes.

Premièrement, des contraintes temporelles puisque plusieurs rendus sont prévus à différentes étapes du projet :

* Document d’étude préliminaire à rendre le 11/11/2015
* Démonstration du travail effectué le 4/11/2015
* Soutenance de projet le 7/12/2015

Deuxièmement, des contraintes techniques, afin de satisfaire au mieux les besoins du client. Nous utiliserons donc les outils et technologies de développement suivants :

Coté client (HTML, JavaScript)

Coté serveur (Java, Servlet, JSP)

Modèle: deux bases de données différentes: une relationnelle et l’autre non relationnelle (utilisation de RDF via l’API Jena ou MongoDB). Concernant les requêtes associées, elles dépendront des BDD utilisées (SPARQL/RDF, SQL/relationnelles).

Les formats entrant en ligne de compte dans la communication entre le client et le serveur pourront être du XML, du JSON, du CSV ou du HTML selon le choix du client. Le Framework de développement sera Spring MVC, qui permet d’employer les patterns MVC et DAO que nous utiliserons dans notre architecture. Pour travailler de manière efficace, nous nous conformerons à la méthode de travail Itérative Scrum, et nous nous servirons notamment de la forge de l’université comme gestionnaire de version.

1. Cas d’utilisations

L’application va proposer de nombreuses fonctionnalités à ses utilisateurs, mais celles-ci ne seront pas accessibles à tous car certaines d’entre elles nécessitent de posséder des droits particuliers. Nous allons brièvement présenter ces fonctionnalités. Des explications plus détaillées viendront ensuite sous forme de cas d’utilisations.

Les utilisateurs non-connectés pourront créer un compte dans l’application en fournissant simplement un nom, un prénom et une adresse email. Cela leur permettra ensuite de se connecter à l’application pour être en mesure de créer une nouvelle conférence.

Les utilisateurs identifiés comme *Auteur* d’une publication pourront facilement enrichir la description de cette publication en y ajoutant un lien vers la page web de leur choix.

Et enfin, un utilisateur *Chair* aura la possibilité d’importer des fichiers de format variés (XML, CSV, JSON-LD, etc.) pour ajouter des informations concernant la conférence, ou la partie de conférence, dont il est responsable dans la base de données.

Il pourra aussi, s’il est *Chair* d’une conférence, choisir d’exporter l’ensemble des données concernant cette conférence (le *dataset*) au format JSON-LD. C’est ce format qui est utilisé par l’application Sympozer.

L’utilisateur *Chair* pourra également modifier manuellement, via une interface graphique, les informations de la conférence ou de la partie de conférence dont il est responsable.

* 1. **niveau_mer.png**Création d’un compte

**Nom :** Créer un compte

**Portée :** Système (Boîte noire)

**Niveau d’objectif :** Utilisateur

**Acteur principal :** Utilisateur

**Évènement déclencheur :** L’utilisateur clique sur le bouton “création de compte”

**Garantie minimale :** Aucune

**Garantie succès :** Le compte de l’utilisateur est créé et l’utilisateur est connecté à l’application

**Précondition :** L'utilisateur n'est pas connecté

**Scénario nominal :**

1. Le système affiche un formulaire composé de plusieurs champs texte vides nommés : nom, prénom, adresse email, nom de connexion, mot de passe, confirmation mot de passe ainsi qu’un bouton “valider”.
2. L'utilisateur renseigne les différents champs du formulaire et clique sur “valider”.
3. Le système vérifie la cohérence des données saisies.
4. Le système remarque qu’il existe une personne dont l’adresse email correspond au nouveau compte utilisateur.
5. Le système créé le compte utilisateur et attribue au nouveau compte les droits correspondant aux différents rôles de la Personne sur les différents éléments qui la concerne.

**Extensions :**

3a. Les informations saisies sont incohérentes

1. La page d'accueil est rechargée avec un message d'erreur invitant l'utilisateur à corriger les informations
2. *Retour à l’étape 1*

4a. Il n’existe aucune Personne dont l’adresse email corresponde à celle du nouveau compte utilisateur

1. Le système crée le compte utilisateur et n’attribue aucun droit particulier à l’utilisateur.
   1. niveau_mer.pngCréation d’une conférence

**Nom :** Créer une conférence

**Portée :** Système (Boîte noire)

**Niveau d’objectif :** Utilisateur

**Acteur principal :** Utilisateur

**Évènement déclencheur :** L’utilisateur clique sur le bouton “Créer une conférence”

**Garantie minimale :** Aucune

**Garantie succès :** La nouvelle conférence à été créée

**Précondition :** L'utilisateur est connecté

**Scénario Nominal :**

1. Le système affiche un formulaire composé de plusieurs champs textes vides nommés : *Nom*, *Lieu*, *Thème* entre autres ainsi qu’un bouton *Valider*.
2. L’utilisateur renseigne les différents champs et clique sur le bouton *Valider*.
3. Le système créé la conférence.
4. Le système attribue à la Personne associée à l’utilisateur le rôle de *Chair* sur cette conférence.
5. Le système attribue à l’utilisateur les droits de niveau *Chair* pour cette conférence.

**Extensions :**

4a Il n’existe pas de personne correspondant à l’utilisateur

1. Le système créé cette personne
2. *Retour à l’étape 4*
   1. niveau_mer.pngEnrichissement de la description d’une publication

**Nom** : Enrichir la description d’une publication

**Portée** : Système (Boîte noire)

**Niveau** **d’objectif** : Utilisateur

**Acteur principal** : Utilisateur *Auteur*

**Évènement déclencheur** : L’utilisateur clique sur le bouton *Modifier*

**Garantie minimale** : La page est affichée.

**Garantie succès** : La modification à été appliquée et est visible sur la page.

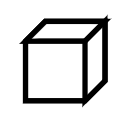
**Précondition** : L'utilisateur est connecté et est identifié en tant qu’*Auteur* de la publication.

**Scénario nominal :**

1. Le système affiche un formulaire composé de deux champs texte vides nommés *URL de la page* et *Nom du lien* ainsi qu’un bouton *Valider*.
2. L'utilisateur renseigne les différents champs du formulaire et clique sur *Valider*.
3. Le système modifie la description de la publication.
4. Le système rafraîchit la page pour que la modification s’affiche.

**Extensions :**

1a Il existe déjà des données de type “informations supplémentaires” pour cette publication.

1. Le système affiche un formulaire composé de deux champs texte, pré remplis avec les informations obtenues, nommés *URL de la page* et *Nom du lien*.
2. *Retour à l’étape 2.*
   1. niveau_mer.pngImport des données

**Nom :** Importer des données

**Portée :** Système (Boîte blanche)

**Niveau d’objectif :** Stratégique

**Acteur principal :** Utilisateur *Chair*

**Acteurs internes :** Module de parsing, Module d’upload et Module d’accès à la base de données

**Évènement déclencheur :** L’utilisateur clique sur le bouton *Importer des données*

**Garantie minimale :** La base de données reste dans un état cohérent

**Garantie succès :** Le fichier est uploadé et les données ont été correctement parsées

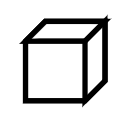
**Précondition :** L'utilisateur est connecté en tant que *Chair*

**Scénario Nominal :**

1. L’utilisateur sélectionne le type et le format du fichier à uploader.
2. L’utilisateur choisit un fichier, d’un type et format reconnus par l’application, sur son disque local.
3. L’utilisateur lance l’upload du fichier.
4. L’application demande au module de parsing de parser le fichier.
5. L’application affiche un récapitulatif des données extraites.
6. L’utilisateur confirme le parsing des données.
7. L’application demande au module d’accès à la base de données d’enregistrer les nouvelles données.

**Extensions :**

2.a. Le format du fichier n’est pas connu (mais le type oui) :

1. L’utilisateur effectue manuellement un mapping pour décrire les données du fichier et valide ce mapping.
2. *Retour Étape 3.*
   1. niveau_mer.pngExport du dataset

**Nom :** Exporter le dataset d’une conférence

**Portée :** Système (Boîte blanche)

**Niveau d’objectif :** Utilisateur

**Acteur principal :** Utilisateur *Chair*

**Acteurs internes :**Module d’export, module d’accès à la base de données et module de stockage de fichiers.

**Évènement déclencheur :** L’utilisateur clique sur le bouton *Générer le dataset*

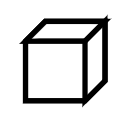
**Garantie minimale :** Aucune

**Garantie succès :** Le dataset est correctement généré et téléchargé par l’utilisateur

**Précondition :** L'utilisateur est connecté en tant que chair de la conférence dont on souhaite générer le dataset

**Scénario nominal :**

1. L’application récupère les données nécessaires auprès du module d’accès à la base de données.
2. Le module d’export génère un document JSON-LD à partir des données récupérées.
3. Le document généré est enregistré par le module de stockage de fichier.
4. L’application lance le téléchargement du dataset ou propose un lien pour le télécharger.

* 1. niveau_mer.pngEdition de données

**Nom :** Edition de données

**Portée :** Système (Boîte blanche)

**Niveau d’objectif :** Utilisateur

**Acteur principal :** Utilisateur *Chair*

**Acteur Interne :** Module d’accès à la base de données

**Événement Déclencheur :** L’utilisateur clique sur le bouton *Modifier*

**Garantie minimale :** La base de données est dans un état cohérent.

**Garantie succès :** L’évènement (ou la partie d’évènement) est modifié selon les souhaits de l’utilisateur.

**Précondition :** L'utilisateur est *Chair* de l’évènement qu’il souhaite modifier

**Scénario nominal :**

1. Le système interroge le module d’accès à la base de données pour récupérer les données de l’évènement (ou de la partie d’évènement).
2. Le module d’accès à la base de données fournit les informations au système.
3. Le système affiche un formulaire contenant différents champs textes pré-remplis avec les informations précédemment récupérées ainsi qu’un bouton *Valider.*
4. L’utilisateur remplit les champs correspondants aux données qu’il souhaite modifier et clique sur *Valider*.
5. Le système demande au module d’accès à la base de données de mettre à jour la base de données avec les nouvelles informations.
6. Le module d’accès à la base de données fait les mises à jour et prend soin de répercuter les modifications aux endroits appropriés.
7. Le système rafraîchit la page pour afficher les modifications.
8. Méthode de travail
   1. Méthode agile SCRUM

SCRUM est la méthode de travail que nous avons jugée la mieux adaptée pour notre projet du fait de la flexibilité qu’elle nous offre. Du 26/10/15 au 11/11/15 nous avons défini notre organisation de travail, nous nous sommes attribués des rôles, et nous avons mis en place un environnement de développement commun (avec l’utilisation d’IntelliJ notamment). Cette période a eu pour but de nous permettre à tous de comprendre le sujet et les attentes du client.

* 1. Organisation des sprints

La durée d’un **sprint** sera d’une semaine (du vendredi au vendredi), l’objectif étant de fournir un livrable qui contient au moins une nouvelle fonctionnalité.

Nous pourrons donc présenter un produit fonctionnel chaque vendredi. Nous avons convenu d’un “Daily meeting” de 15 minutes par jour ouvrable afin de se tenir mutuellement au courant de l’avancée des différentes tâches.

A la fin de chaque sprint, nous effectuerons une réunion plus longue pour statuer sur l’état du projet et pour définir les tâches à accomplir pour le prochain livrable. Nous avons, en outre, prévu d’organiser des sessions de travail intensive en commun et de **pair programming** afin de créer un climat d'effervescence et d’entraide.

Chaque développeur prendra en charge les tests unitaires de sa partie. Quant aux tests fonctionnels (test de robustesse,…), chacun sera amené à en faire, indépendamment de la partie qu’il aura développée.

Les acteurs de deux parties du projet qui doivent coexister partageront la responsabilité de la compatibilité de ces deux parties.

* 1. Rôles

**SCRUM master** : Jordan, qui a une expérience plus approfondie des applications WEB.

**Le product owner**: Jules qui se prendra la responsabilité de la validation de certaines fonctionnalités.

**Le “period manager” :** Tom, qui devra s’assurer que les délais de rendu des livrables et des rapports sont respectés et que personne ne prend du retard dans l’implémentation d’une fonctionnalité.

Afin de travailler de manière efficace, nous nous sommes répartis les tâches de la manière suivante :

* Jordan et Tom la partie web Java EE et l’architecture REST
* Alix et Jules la partie base de données
* Clément la partie parsing
* Landry la partie interface web
* A tous, la partie web client et JavaScript

L’affectation de ces rôles n’est pas figée et changera ou sera renforcée par d’autre membre de l’équipe en fonction de l’évolution du projet.

1. Outils et technologies
   1. Service web REST

Les web services REST sont implémentés grâce au Framework Spring et notamment le projet **Spring MVC**. Ces services permettront d’exposer les différentes ontologies ainsi que l’application web pour la gestion et la consultation des données.

Le contrôle d’accès aux différentes pages et ressources sera assuré par **Spring Security**.

* 1. Application web

En plus des services REST, Spring est utilisé pour générer les différentes pages de l’application. Le rendu de ces pages est géré par des **JSP** de l’API **Java EE 7** complété avec la taglib JSTL.

Côté client, nous avons sélectionné le Framework HTML/CSS **bootstrap** afin de faciliter et accélérer la création des différentes pages. Cela nous permet également d’avoir un visuel agréable et facilement personnalisable (avec des thèmes par exemple).

Enfin, pour rendre nos pages dynamiques et interactives nous utilisons JavaScript et la bibliothèque **jQuery** pour simplifier la manipulation du DOM ainsi que les requêtes AJAX.

* 1. Base de données

Nous avons opté pour l’utilisation du SGBD **MySQL** pour la gestion des comptes et des droits utilisateurs. Cette base relationnelle nous permet de maintenir un schéma de données robuste et simple à manipuler.

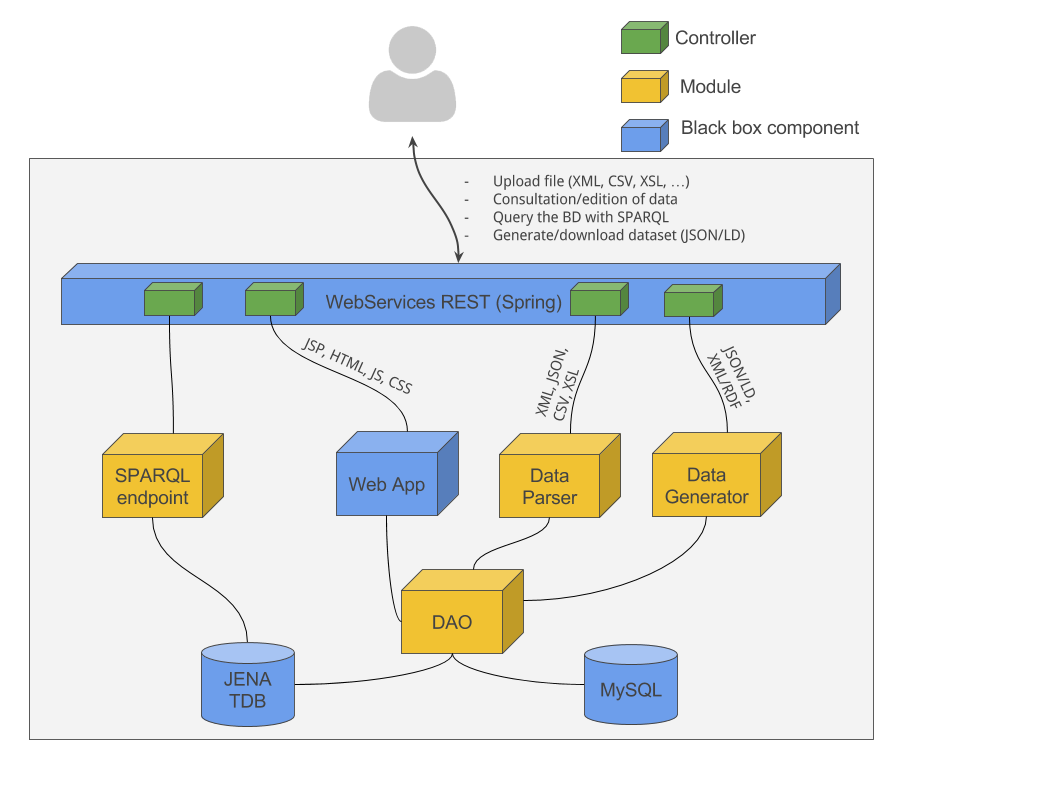
La gestion des tuples du modèle relationnel est assurée par l’ORM **Hibernate** qui nous propose une implémentation de l’API JPA de Java EE.

Pour le stockage et la gestion des différentes ressources de l’application (conférences, personnes, publications, keynotes, …) nous avons choisi d’utiliser le Framework **JENA**.

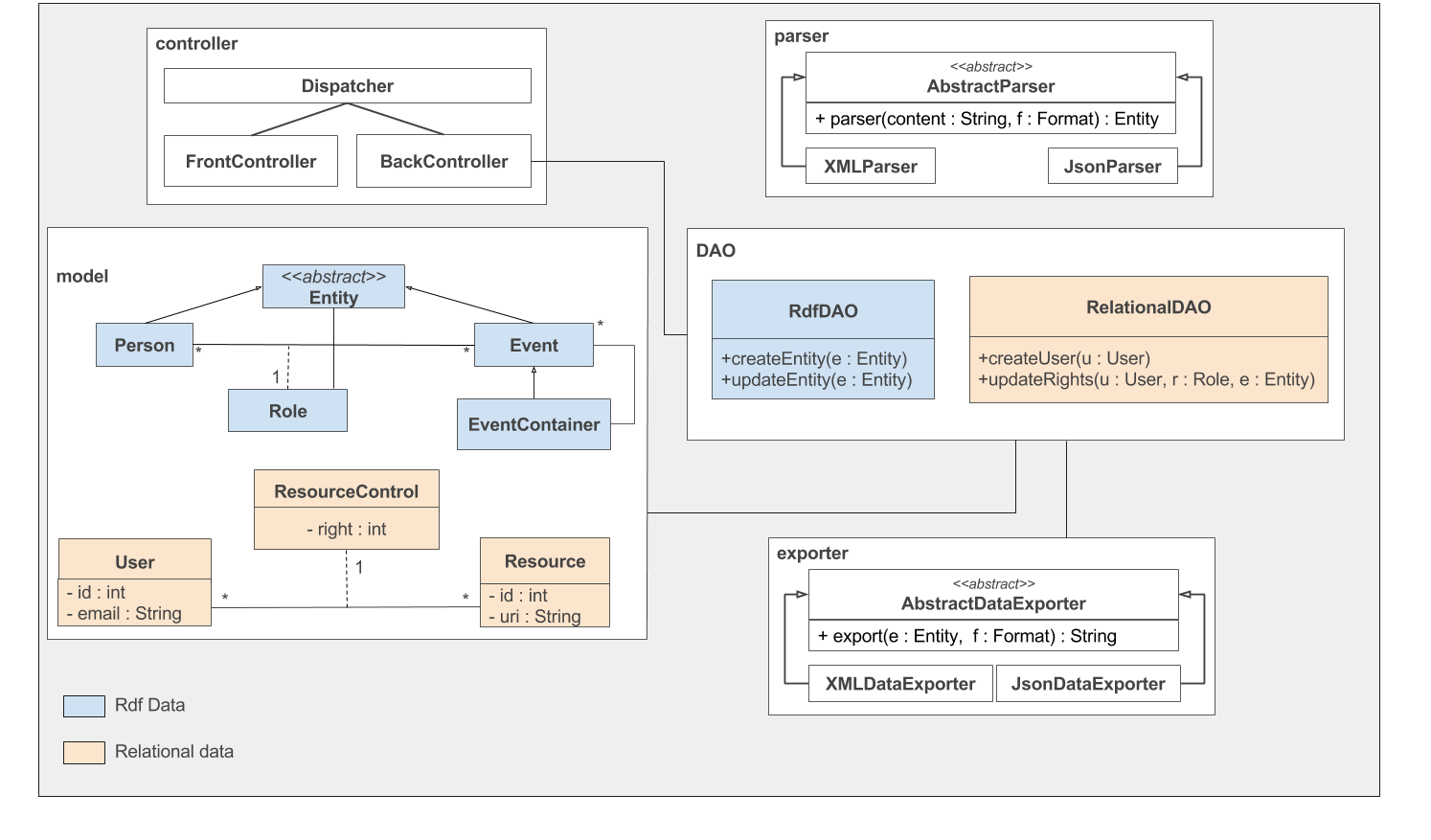
Bien que JENA peut être utilisé avec MySQL nous avons exploité le composant de stockage de triplet **TDB** de JENA pour la persistance des données. La principale raison de ce choix est le gain de performance qu’apporte TDB par rapport à MySQL. De plus, le projet Apache Jena recommande l’utilisation de TDB.

1. Architecture du projet
   1. Schéma global

Un des objectifs que nous nous sommes fixé est d’obtenir une application évolutive et maintenable. Pour cela, nous avons fait en sorte qu’elle soit la plus modulaire possible. Les différents modules de l’application sont donc indépendants les uns des autres et peuvent être modifiés ou remplacés très facilement.

*Schéma global de l’architecture de l’application*

* 1. Diagramme de classe

**

Le diagramme de package ci-dessus décrit de manière succincte la structure de notre application. Le modèle est composé de deux parties, l'une associée au modèle relationnel qui représente les comptes utilisateurs, et l'autre au modèle non relationnel qui représente les entités mises en jeu lors d'une conférence et les relations principales les liant.

Les classes du modèle sont des objets métiers qui vont être créés ou modifiés par le DAO lorsqu'un changement des données de l'application sera enregistré.

Le modèle non relationnel est composé de trois tables, permettant la création et gestion des comptes utilisateur, la création d'une ressource associée à certaines entités du modèle non relationnelle et l'attribution des droits d'un utilisateur sur cette ressource. Ce modèle permet de gérer spécifiquement les droits de chaque utilisateur sur chaque ressource sans redondance au niveau de la base de données.

Le modèle non relationnel de plusieurs entités (qui ne sont pas toutes représentées sur le schéma), qui possède chacun une IRI propre, une liste d'objets avec lesquels elles sont en relations et une liste de propriétés. Ce modèle permet d'établir la correspondance avec le modèle RDF stocké en base.

Le module *parser* sera le module qui sera chargé de traduire les données présentes dans les fichiers uploadés par les utilisateurs en données compréhensibles pour notre application.

Le module *exporter* sera à l’inverse chargé de représenter les données de notre application dans le format choisi par l’utilisateur.

Seuls certains formats de fichiers ont été représentés sur notre schéma par souci de lisibilité.

Le *front controller* fournira à l’utilisateur les différentes pages web et formulaire lui permettant de visualiser et interagir avec les données de l’application.

Enfin, pour orchestrer ces différents modules, le *back controller* permettra le traitement des requêtes de l’utilisateur et l’appel aux différents modules.

1. Risques Identifiés
   1. Choix de la modélisation des données

Une grande partie du succès du projet passe par la gestion des données. Si notre modélisation est mauvaise, tout le projet en subira les conséquences. Afin de minimiser ce risque, l’étude préalable du projet est très importante. De plus, nous n’avons pas hésité à poser nos questions au client.

* 1. Utilisation de Spring

Avant le début du projet, aucun membre de l’équipe n’avait de compétences sur ce Framework. Le risque est d’utiliser un composant du Framework qui n’est pas adapté (ou mal utilisé) dès le début du projet. Spring étant notre base de développement pour ce projet, cela pourrait nous ralentir voir nous bloquer plus tard dans le développement. De plus, une remise en cause des choix faits à la base pourrait induire une quantité importante de modification sur les briques au-dessus de Spring.

Pour minimiser ce risque, les membres de l’équipe les plus expérimentés en Web ont étudié de manière approfondie la documentation officielle.

* 1. Utilisation d’un SGBD orienté graphe

La problématique de modélisation en graphe RDF dans le cadre du web sémantique des conférences est complexe : les chaînages sont nombreux et les liens entre les différents nœuds ne sont pas toujours évidents à établir. De plus, nous n’avons pas beaucoup d’expérience dans l’utilisation de cette architecture de base de données.

Nous risquons donc de perdre du temps lors de la conception de la base de données car nous pouvons avoir du mal à comprendre comment représenter nos données.

Pour minimiser ce risque, il faut commencer par établir un schéma de base de données très simple, qui ne contiendra pas forcément tous les éléments ou qui ne sera pas capable de représenter toutes les relations entre éléments, s’assurer qu’il est fonctionnel et après seulement commencer à le compléter. De cette façon, si nous partons dans une mauvaise direction, il sera facile et assez rapide de revenir en arrière pour corriger nos erreurs.

* 1. Développement de fonctionnalités de faible importance

Au cours du développement, la mise en place de certaines fonctionnalités semblant facile (code source accessible, tutoriel,…) mais pas forcément nécessaire (Connexion Facebook/Twitter,…) pourraient nous faire perdre du temps sur le projet, notamment parce que ces fonctionnalités nous sont inconnues et seront certainement sources d’erreurs. Ces fonctionnalités ne devront être traités que si les fonctionnalités principales sont d’ores et déjà en état de marche et que nous sommes en avance sur notre calendrier.