Année 2017/2018

RAPPORT DE STAGE DE FIN D'ÉTUDES

Présenté par **WANG Yuwei**

Sujet du stage :

Gestionnaire de Données d'Études :

mise en place du moteur d'indexation et de recherche

Directeur de stage : **Cédric AGUERRE**

Conseiller de stage : **Sophie CHABRIDON**

DU 01/07/2017 AU 31/12/2017

EDF LAB PARIS-SACLAY

7 Boulevard Gaspard Monge, 91120 Palaiseau

Table des matières

Liste des figures et des tableaux

Glossaire

GDE	Gestionnaire de Données d'Études
JAVA EE	Java Enterprise Edition
JSP	JavaServer Pages
EJB	Enterprise JavaBeans
JPA	Java Persistence API
JTA	Java Transaction API
REST	REpresentational State Transfer
AJAX	Asynchronous JavaScript And XML
JSON	JavaScript Object Notation
SDK	Software Development Kit
XML	Dynamic Execution Object Model
HTTP	HyperText Transfer Protocol
POJO	Plain Old Java Object
DAO	Data Access Object
SGBD	Système de Gestion de Base de Données
JSF	JavaServer Faces
SQL	Structured Query Language
CSS	Cascading Style Sheets
API	Application Programming Interface

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué à la réalisation de mon stage de fin d'études et qui m'ont aidée lors de la rédaction de ce rapport.

Tout d'abord, mes remerciements s'adressent à mon chef de groupe Madame Anne PICAULT, qui m'a accordée sa confiance et m'a permis d'effectuer mon stage au sein de son équipe à EDF Lab.

Je tiens tout particulièrement à remercier mes maîtres de stage Monsieur Cedric AGUERRE et Monsieur Kavoos BOJNOURDI pour m'avoir consacré du temps dès mon arrivée, pour m'avoir guidée tout au long du projet, et pour m'avoir toujours vivement encouragée. Grâce aussi à leurs conseils avisés, leur partage de connaissances et leur soutien lors de cette expérience, j'ai pu mener à bien mon stage.

Je tiens à exprimer toute ma reconnaissance aussi à Madame Sophie CHABRIDON, ma tutrice académique, qui m'a suivie pendant la durée de mon stage et a été un soutien pendant tout le stage.

J'exprime également ma gratitude à tous les membres de l'équipe d'I2A, pour leur accueil, leur disponibilité, leur bonne humeur et leur sympathie qui ont favorisé mon intégration dans l'entreprise.

Enfin, j'adresse mes remerciements à toutes les personnes qui m'ont conseillée et relue lors de la rédaction de ce rapport de stage et tout particulièrement, mon ami Silun ZHANG.

Résumé

EDF R&D réalise des études pour ses clients d'exploitation. Il assure un rôle de support technique et d'innovation scientifique au sein du groupe EDF. EDF R&D dispose aujourd'hui d'une base de données gigantesque des études réalisées et des données de calculs scientifiques. Afin de faciliter l'accès rapide et la reprise de ces données d'études, un projet intitulé « Gestionnaire de données d'études(GDE) » a été lancé au sein du groupe I2A.

Les objectifs principaux du système GDE sont l'archivage d'études et l'aide à la création d'études par la recherche de cas similaires. Ce système se base sur une architecture trois tiers : les clients en C++, Python et Web pour la couche de présentation, un serveur applicatif en JAVA pour le traitement et une base des données relationnelle pour le stockage des données persistantes.

Ce stage se compose de deux parties. La première contribue à l'intégration d'un moteur d'indexation et de recherche dans le GDE. Le travail a consisté à rechercher un module d'extraction des données, analyser les APIs du moteur de recherche Lucene et réaliser l'intégration du moteur de recherche dans le GDE.

La deuxième partie concerne la conception et le développement du module d'administration en client Web. Ce module est accessible depuis les pages Web qui sont réalisées en HTML, CSS, Javascript, grâce au Framework Bootstrap, la bibliothèque jQuery et la technologie Ajax. Son rôle est de faciliter, pour les administrateurs ou utilisateurs autorisés, la gestion des paramètres de la configuration et des données du système.

Abstract

EDF R&D carries out studies for its operating customers. It ensures a role of technical support and scientific innovation within the EDF Group. EDF R&D now has a huge database of studies and scientific data. To facilitate rapid access and retrieval of these study data, a project entitled "Gestionnaire de données d'études (GDE)" has been launched within the I2A group.

The main objectives of the GDE system are the archiving of studies and the help in creating studies by searching similar cases. This system is based on a three-tier architecture: clients based on C++, Python and Web for the presentation layer, an application server based on Java for data processing and a relational database for storing persistent data.

This internship is made up of two parts. The first contributes to the integration of an indexing and search engine into GDE system. The work involved searching for a data extraction module, analyzing the Lucene search engine APIs, and integrating the search engine into GDE.

The second part concerns the design and development of the administration module through a web client. This module is accessible by web pages that are realized by HTML, CSS, Javascript, Bootstrap framework, jQuery Library and Ajax technology. Its role is to facilitate, for administrators or authorized users, the management of system configuration and data settings.

I. Introduction

Dans le cadre de ma formation d'ingénieur à Télécom Sudparis, mon stage de fin d'étude a eu lieu à EDF Lab Paris-Saclay, entre les mois de juillet 2017 et décembre 2017. Il s'est déroulé au sein du département Performance et prévention des Risques Industriels du parc par la simulation et les Études (PERICLES) au sein duquel se trouve le groupe Architecture des Systèmes d'Information et Calcul Scientifique (I2A).

Le groupe EDF, qui est le premier producteur et fournisseur d'électricité dans le monde, s'est fait connaître avec succès pour les énergies nucléaires et les produits innovants. Les ingénieurs d'EDF Lab implantent des codes de calcul scientifiques dans la plate-forme SALOMÉ qui permet de construire des modèles physiques et de résoudre une large gamme de problèmes de simulation numérique.

Par conséquent, une « étude » se définit comme le travail à réaliser pour résoudre les problèmes par simulation numérique et analyser les résultats obtenus. En général, les données des études peuvent être considérées comme un ensemble de fichiers, par exemple les cahiers des charges, les entrées des calculs, les schémas de calcul, etc.

Dans ce contexte, EDF a souhaité développer un système de gestion des données d'étude. L'objectif est la capitalisation des données produites ainsi que la recherche de données pour les utiliser à l'occasion d'activités similaires. Ce stage s'inscrit dans les parties de l'intégration du moteur de recherche dans le système et la conception du module d'administration.

Ce rapport est composé de quatre parties. La première partie présente brièvement l'entreprise EDF. La seconde partie est une introduction du contexte du stage. Elle contient une explication du projet GDE, les méthodologies appliquées dans le travail et les technologies utilisées pendant ce stage. La troisième partie expose de façon détaillée les différentes missions effectuées. En conclusion, une analyse de mon expérience professionnelle et personnelle est présentée.

II. Présentation de l'entreprise

2.1 Groupe Électricité De France(EDF)

2.1.1 EDF en bref

Premier électricien mondial, le groupe EDF est le leader mondial des énergies bas carbone. Solidement implanté en Europe, notamment en France, au Royaume-Uni, en Italie, en Belgique ainsi que sur le continent américain, le Groupe rassemble tous les métiers présents sur la chaîne de valeur de l'électricité – de la production à la distribution en passant par le transport de l'énergie et les activités de négoce – pour équilibrer en permanence l'offre et la demande.

Le groupe EDF est un énergéticien intégré, présent sur l'ensemble des métiers de l'électricité : la production nucléaire, renouvelable et fossile, le transport, la distribution, la commercialisation, les services d'efficacité et de maîtrise de l'énergie, ainsi que le négoce d'énergie.

Sa production d'électricité, marquée par la montée en puissance des énergies renouvelables, s'appuie sur un mix énergétique diversifié et complémentaire autour du nucléaire. EDF propose des offres commerciales et des conseils pour accompagner ses clients particuliers dans la maîtrise de leur consommation, contribue à la performance énergétique et économique des entreprises et aide les collectivités locales à adopter des solutions durables.

Dans le contexte de transition énergétique, EDF a défini une stratégie baptisée CAP 2030 qui porte l'ambition du Groupe : Être l'électricien performant et responsable, champion de la croissance bas carbone. Les trois objectifs de CAP 2030 sont d'accroître la proximité avec ses clients, de doubler la production d'énergies renouvelables d'ici à 2030, et de tripler la part du business réalisé à l'international d'ici à 2030.

En même temps, EDF s'est fixé six Objectifs de Responsabilité d'Entreprise, en résonance aux 17 objectifs de développement durable de l'ONU qui sont le changement climatique, le développement humain, la précarité énergétique, l'efficacité énergétique, le dialogue et la concertation, ainsi que la biodiversité. Des engagements majeurs et prioritaires, dont le Groupe présentera chaque année les résultats.

2.1.2 EDF en chiffres 2016

Voici quelques chiffres clés en ce qui concerne l'activité d'EDF :

- 71,2 milliards d'euros de chiffre d'affaires ;
- 16,4 milliards d'euros d'EBITDA (Le bénéfice avant intérêts, impôts, dépréciation et amortissement) ;
- 4,1 milliards d'euros de résultat net part du Groupe ;
- 14,4 milliards d'euros d'investissements opérationnels bruts ;
- 154845 collaborateurs dans le monde ;
- 37,1 millions de clients dans le monde ;

- 584,7 TWh de production d'électricité du Groupe ;
- 88 % de production sans CO2 pour le Groupe.

Figure 1 : EDF EN CHIFFRES EN 2016

2.2 EDF R&D

La branche Recherche et Développement du groupe compte 3 sites en France et 7 sites à l'international, représentant 2100 collaborateurs et 180 doctorants, divisés en 15 départements. La R&D a également 14 laboratoires communs avec des partenaires industriels et académiques tels que le CEA, ou encore l'école Polytechnique. Les missions de la R&D d'EDF sont orientées autour de 3 priorités clés, qui sont : consolider et développer des mix de production compétitifs et décarbonés, développer de nouveaux services énergétiques pour les clients et enfin préparer les systèmes électriques de demain.

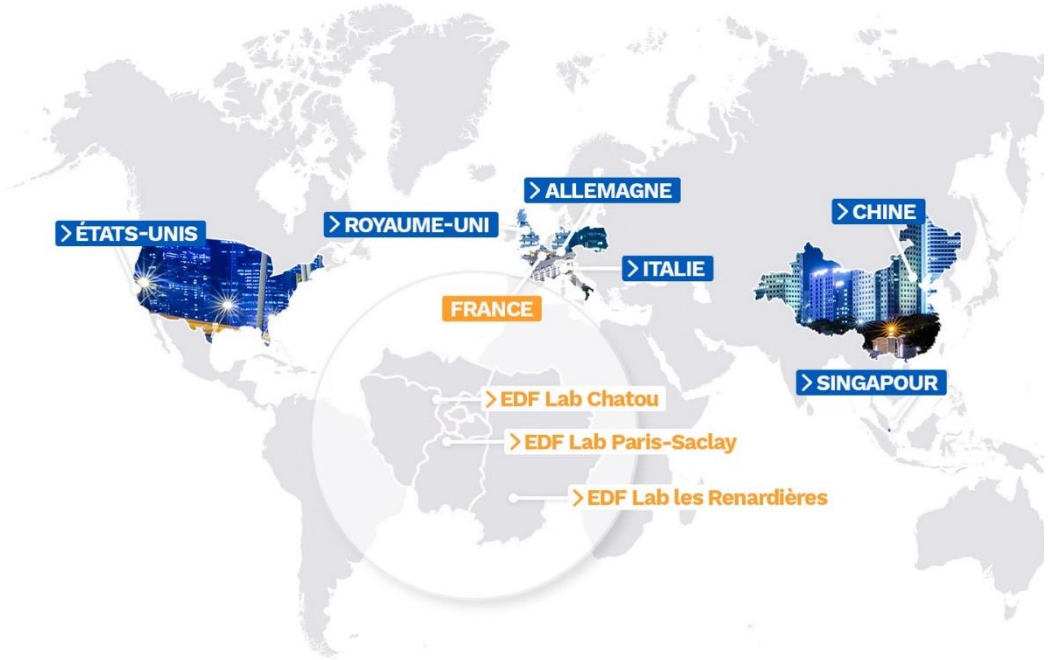


Figure 2 : EDF DANS LE MONDE

Le groupe vient de se doter d'un nouveau centre de recherche au cœur du campus de Paris-Saclay. Implanté sur 12 hectares, EDF Lab Paris-Saclay est la figure de proue de la R&D du groupe. Il s'agit du plus grand centre industriel de recherche et de formation en Europe. Il accueille 1200 chercheurs et un campus de formation.

Ces enjeux font appel à des compétences scientifiques de premier plan : analyses en mécanique avancée, développement des systèmes d'information pour les réseaux électriques, fonctionnement et étude des systèmes énergétiques, technologies et modélisation des infrastructures du système électrique, innovation commerciale, analyse des marchés et de leur environnement, management des risques industriels, simulation neutronique, technologies de l'information, calcul scientifique, mathématiques appliquées.

Figure 3 : VUE DE EDF LAB PARIS-SACLAY

2.3 Département PERICLES

Le présent stage a été réalisé au sein du département Performance et prévention des Risques Industriels du parc par la simulation et les Études (PERICLES) à l'entité R&D EDF de Paris-Saclay.

Le département PERICLES est pôle de compétences dans les domaines de la physique des réacteurs nucléaires et du cycle du combustible, du développement et de la conception des codes de calcul et des systèmes d'information scientifiques et techniques.

Il a pour missions principales de contribuer à une exploitation performante et à la sûreté des réacteurs nucléaires actuels (REP) et futurs en fonctionnement normal ou accidentel et de préparer des solutions polyvalentes dans les domaines du calcul scientifique et de la numérisation des processus métiers au service des métiers du groupe EDF.

Les activités principales sur ces 3 thématiques sont les suivantes : simulation neutronique, simulation numérique et technologies de l'information pour les métiers du producteur, du distributeur, commercialisateur.

Les compétences sont organisées autour de trois grands axes :

- la **Simulation du Combustible Nucléaire** regroupe les savoirs concernant la simulation et l'optimisation des cœurs de centrales REP du parc EDF et les études amont et aval sur le cycle du combustible et sur les effets du transport neutronique ;
- la **Numérisation des Processus Métier** regroupe les thèmes de l'architecture fonctionnelle et technique des systèmes d'information et des architectures techniques de communication, mais aussi de la mise en œuvre des solutions de réalité virtuelle au service des métiers d'EDF et notamment de la maintenance des tranches ;
- la **Simulation Numérique et le Calcul Scientifique** regroupe les compétences nécessaires à la mise en œuvre et à l'exploitation des simulations numériques pour les études de recherche ou d'ingénierie de l'entreprise. Ils couvrent la définition des infrastructures de calcul, l'analyse numérique pour les solveurs, l'architecture logicielle pour les codes de calcul et la visualisation scientifique.

2.4 Groupe I2A

Le département PERICLES est porté par huit groupes. J'ai intégré le groupe Architecture des Systèmes d'Information et Calcul Scientifique (I2A).

Le groupe I2A regroupe des compétences d'architecture de systèmes d'information et d'architecture logicielle pour l'informatique scientifique appliquée au calcul haute performance. Ces deux compétences sont soutenues par une compétence transverse sur les méthodes et les langages de programmation, les outils de développement et les intergiciels utilisées pour le calcul scientifique comme pour les systèmes d'informations et les systèmes complexes.

La mission principale du groupe est donc de fournir une expertise dans ces deux domaines aux projets, départements d'EDF R&D voire aux directions/filiales d'EDF. Cela conduit à une participation active dans les phases amont de ces activités pour analyser, concevoir et choisir les bonnes solutions d'architecture, mais aussi par le développement et la diffusion de technologies comme la plate-forme SALOMÉ comme solution d'intégration du calcul scientifique et notamment du calcul parallèle et la plate-forme de simulation des villes durables CURTIS.

En particulier, dans le groupe, la compétence « Architecture des SI » apporte une expertise et des technologies sur les choix et les orientations d'architecture pour ces grands systèmes. Cela se traduit par :

- des activités de conseil sur l'analyse des besoins et des contraintes, la modélisation des processus métier, des données et des traitements, la conception d'ensemble et l'organisation architecturale ;
- des activités de conseil, de test et de choix sur les technologies SI en jeu et de développement de solutions innovantes d'architecture pour des besoins métier ;
- par ailleurs, une activité de recherche sur les thèmes de l'ingénierie des modèles, de la simulation des systèmes d'information et des systèmes complexes est menée dans le cadre de projets en partenariat.

III. Contexte du stage

3.1 Projet Gestionnaire de Données d'Études (GDE)

Le contexte principal d'utilisation de ce projet « Gestionnaire de données d'études » (on utilise l'acronyme GDE dans la suite du document) est la plate-forme SALOMÉ et ses applications métier.

3.1.1 SALOMÉ

Au cours de la dernière décennie, les progrès matériels et logiciels ont apporté des changements significatifs dans les capacités des plateformes de simulation notamment dans le domaine des applications nucléaires. La puissance des nouvelles machines a permis l'émergence de simulations plus réalistes, plus rapides et plus robustes. Depuis 2001, afin de faciliter et d'améliorer ce processus, le CEA et EDF ont développé une plateforme logicielle nommée SALOMÉ qui fournit des outils pour construire des applications de simulation intégrées.

La plate-forme SALOMÉ est un environnement logiciel ouvert dans lequel les ingénieurs peuvent implanter des codes de calcul scientifiques pour construire des modèles physiques et mettre en œuvre une large gamme de problèmes de simulation numérique. La plate-forme SALOMÉ est livrée nativement avec les outils généralistes de la simulation, en particulier un système de conception de géométries, des outils de maillages, des fonctions de traitement et de visualisation des données, ainsi que des outils de construction de schémas de calcul pour l'exécution sur les ressources HPC (High performance computing) de l'entreprise. SALOMÉ est aussi une équipe d'ingénieurs-chercheurs en physique numérique et génie logiciel, capable de mettre en place des solutions de simulation sur mesure pour les projets d'ingénierie d'EDF.

Dans le contexte CEA-EDF, les systèmes à l'étude sont typiquement les équipements industriels des moyens de production, avec pour enjeux la conception de la nouvelle génération de réacteurs, la gestion du combustible, la fiabilité et la sûreté des installations, l'analyse du vieillissement des équipements et l'optimisation de leur cycle de vie.

Dans ce contexte, une « étude » se définit comme le travail à réaliser pour comprendre une situation industrielle à partir des phénomènes physiques qui sont à l'œuvre. L'étude est initiée sur la base d'une question industrielle à résoudre, implique en général l'utilisation de la simulation numérique, et s'achève par une analyse des résultats obtenus. L'activité de simulation numérique proprement dite peut être vue comme l'exécution de calculs produisant ou manipulant des informations appelées données d'étude et dont le support informatique est en général un ensemble de fichiers. Parmi les données d'étude, on compte notamment les informations de contexte et les éléments de prescription (cahier des charges), les paramètres et données d'entrée des calculs, la définition du schéma de calcul (identification des composants de calcul et de leur dépendance d'exécution), les données de résultat produites, les éléments d'analyse (notes techniques et propositions).

3.1.2 GDE

En coordination avec le CEA et dans le cadre du co-développement EDF/CEA de la plate-forme SALOMÉ, EDF souhaite développer un système de gestion des données d'études permettant de compléter les fonctionnalités actuellement offertes par SALOMÉ pour les applications métier. Donc, le projet du système GDE a été initié en 2014 avec un nouveau positionnement dans le contexte des données d'étude de SALOMÉ, mais avec une exigence de généricité et d'indépendance technique vis-à-vis de SALOMÉ.

Le GDE se définit comme un socle technique pour construire facilement des environnements de gestion d'études, dans le respect du référentiel de l'entreprise (assemblage de produits au référentiel DSP). L'objectif principal du système est la capitalisation des données produites ou utilisées à l'occasion des activités de simulation, c'est-à-dire l'archivage d'études de référence puis l'aide à la création d'études par la recherche de cas similaires.

La préoccupation essentielle est d'abord de pouvoir caractériser les données par des éléments de contexte (métadonnées) qui permettent de définir précisément leur rôle dans la simulation, et qui sont exploitables par d'autres systèmes clients. Dans ce cadre, la caractérisation des données est faite au moyen d'un jeu de propriétés, certaines choisies dans un thésaurus du domaine métier et d'autres libres.

La recherche d'étude se fait alors sur la base de ces propriétés (métadonnées). Un soin tout particulier doit donc être porté sur la définition de ces propriétés pour chaque domaine de simulation ou situation de calcul qu'on voudra être capable de gérer dans le système GDE. Pour accompagner cette discipline, le GDE doit fournir un mécanisme d'indexation rigoureux (basé sur des propriétés issues d'un thésaurus du domaine métier) et ouvert (extensible à de nouveaux domaines métier et capable d'intégrer de nouvelles contraintes de recherche d'études).

3.1.3 Objectifs du stage

Ce stage s'inscrit aussi bien dans une perspective de développement professionnel que personnel.

Les objectifs principaux du stage sont les suivants :

- intégrer l'outil Apache Lucene dans l'environnement du GDE ;
- définir et mettre en place le système d'indexation automatique ;
- mettre à disposition des fonctionnalités permettant de s'interfacer avec ce système.

Du point de vue du développement personnel, il s'agit pour moi d'apprendre à mieux me connaître, de renforcer mon savoir-être et mon professionnalisme.

- Réfléchir à mon orientation professionnelle ;
- Apprendre à m'intégrer à une équipe de projet;
- Améliorer mon autonomie ;
- Me perfectionner en français ;
- Améliorer ma confiance en moi.

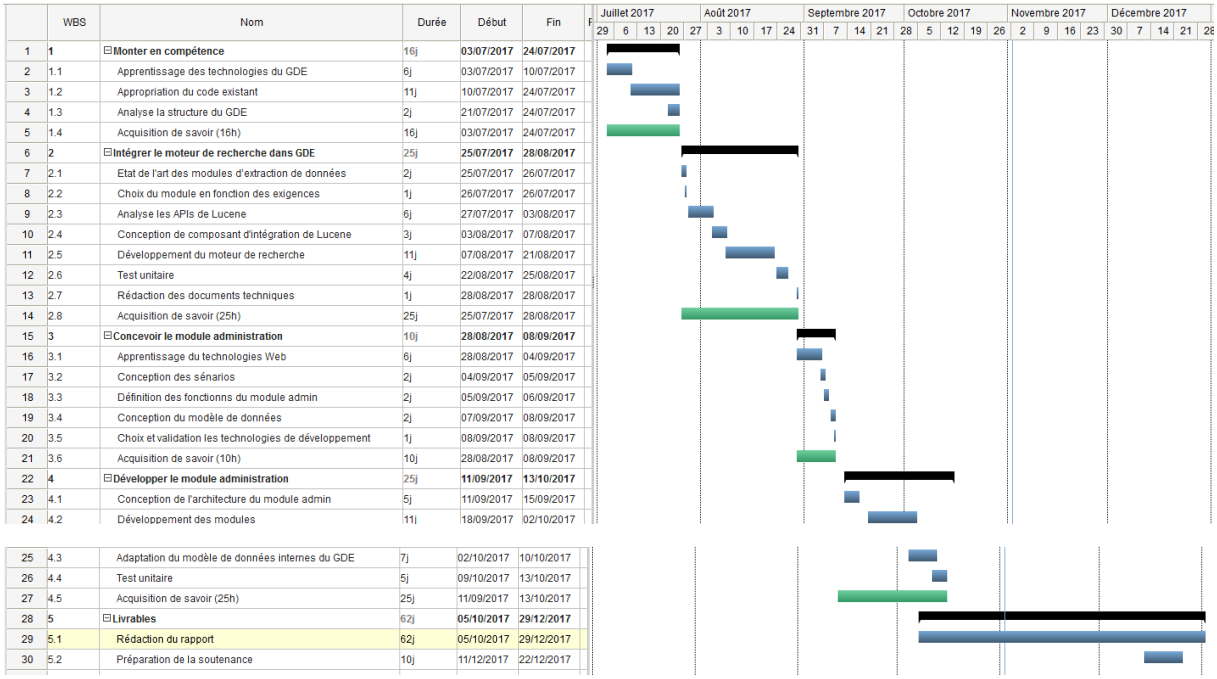
3.2 Méthodologies de travail

3.2.1 Conduite de projet

Le stage de fin d'études a duré 6 mois avec plusieurs tâches différentes. Pour réussir à atteindre les objectifs, il est nécessaire et important de répertorier toutes les tâches à accomplir pour mener le projet à bien. Les figures 4 et 5 présentent le planning final qui a été réalisé. Par rapport au planning prévu, tous les tâches a été accomplis en avance, même un nouveau défi du module d'administration a été effectué.

	WBS	Nom	Durée	Début	Fin
1	**1**	**⊟Monter en compétence**	16j	**03/07/2017**	**24/07/2017**
2	1.1	Apprentissage des technologies du GDE	6j	03/07/2017	10/07/2017
3	1.2	Appropriation du code existant	11j	10/07/2017	24/07/2017
4	1.3	Analyse la structure du GDE	2j	21/07/2017	24/07/2017
5	1.4	Acquisition de savoir (16h)	16j	03/07/2017	24/07/2017
6	**2**	**⊟Intégrer le moteur de recherche dans GDE**	25j	**25/07/2017**	**28/08/2017**
7	2.1	Etat de l'art des modules d'extraction de données	2j	25/07/2017	26/07/2017
8	2.2	Choix du module en fonction des exigences	1j	26/07/2017	26/07/2017
9	2.3	Analyse les APIs de Lucene	6j	27/07/2017	03/08/2017
10	2.4	Conception de composant d'intégration de Lucene	3j	03/08/2017	07/08/2017
11	2.5	Développement du moteur de recherche	11j	07/08/2017	21/08/2017
12	2.6	Test unitaire	4j	22/08/2017	25/08/2017
13	2.7	Rédaction des documents techniques	1j	28/08/2017	28/08/2017
14	2.8	Acquisition de savoir (25h)	25j	25/07/2017	28/08/2017
15	**3**	**⊟Concevoir le module administration**	10j	**28/08/2017**	**08/09/2017**
16	3.1	Apprentissage du technologies Web	6j	28/08/2017	04/09/2017
17	3.2	Conception des sénarios	2j	04/09/2017	05/09/2017
18	3.3	Définition des fonctionns du module admin	2j	05/09/2017	06/09/2017
19	3.4	Conception du modèle de données	2j	07/09/2017	08/09/2017
20	3.5	Choix et validation les technologies de développement	1j	08/09/2017	08/09/2017
21	3.6	Acquisition de savoir (10h)	10j	28/08/2017	08/09/2017
22	**4**	**⊟Développer le module administration**	25j	**11/09/2017**	**13/10/2017**
23	4.1	Conception de l'architecture du module admin	5j	11/09/2017	15/09/2017
24	4.2	Développement des modules	11j	18/09/2017	02/10/2017
25	4.3	Adaptation du modèle de données internes du GDE	7j	02/10/2017	10/10/2017
26	4.4	Test unitaire	5j	09/10/2017	13/10/2017
27	4.5	Acquisition de savoir (25h)	25j	11/09/2017	13/10/2017
28	**5**	**⊟Livrables**	62j	**05/10/2017**	**29/12/2017**
29	5.1	Rédaction du rapport	62j	05/10/2017	29/12/2017
30	5.2	Préparation de la soutenance	10j	11/12/2017	22/12/2017

Figure 4 : PLANNING DU STAGE



	WBS	Nom	Durée	Début	Fin
1	1	**Monter en compétence**	16j	**03/07/2017**	**24/07/2017**
2	1.1	Apprentissage des technologies du GDE	6j	03/07/2017	10/07/2017
3	1.2	Appropriation du code existant	11j	10/07/2017	24/07/2017
4	1.3	Analyse la structure du GDE	2j	21/07/2017	24/07/2017
5	1.4	Acquisition de savoir (16h)	16j	03/07/2017	24/07/2017
6	2	**Intégrer le moteur de recherche dans GDE**	25j	**25/07/2017**	**28/08/2017**
7	2.1	Etat de l'art des modules d'extraction de données	2j	25/07/2017	26/07/2017
8	2.2	Choix du module en fonction des exigences	1j	26/07/2017	26/07/2017
9	2.3	Analyse les APIs de Lucene	6j	27/07/2017	03/08/2017
10	2.4	Conception de composant d'intégration de Lucene	3j	03/08/2017	07/08/2017
11	2.5	Développement du moteur de recherche	11j	07/08/2017	21/08/2017
12	2.6	Test unitaire	4j	22/08/2017	25/08/2017
13	2.7	Rédaction des documents techniques	1j	28/08/2017	28/08/2017
14	2.8	Acquisition de savoir (25h)	25j	25/07/2017	28/08/2017
15	3	**Concevoir le module administration**	10j	**28/08/2017**	**08/09/2017**
16	3.1	Apprentissage du technologies Web	6j	28/08/2017	04/09/2017
17	3.2	Conception des sénarios	2j	04/09/2017	05/09/2017
18	3.3	Définition des fonctionns du module admin	2j	05/09/2017	06/09/2017
19	3.4	Conception du modèle de données	2j	07/09/2017	08/09/2017
20	3.5	Choix et validation les technologies de développement	1j	08/09/2017	08/09/2017
21	3.6	Acquisition de savoir (10h)	10j	28/08/2017	08/09/2017
22	4	**Développer le module administration**	25j	**11/09/2017**	**13/10/2017**
23	4.1	Conception de l'architecture du module admin	5j	11/09/2017	15/09/2017
24	4.2	Développement des modules	11j	18/09/2017	02/10/2017
25	4.3	Adaptation du modèle de données internes du GDE	7j	02/10/2017	10/10/2017
26	4.4	Test unitaire	5j	09/10/2017	13/10/2017
27	4.5	Acquisition de savoir (25h)	25j	11/09/2017	13/10/2017
28	5	**Livrables**	62j	**05/10/2017**	**29/12/2017**
29	5.1	Rédaction du rapport	62j	05/10/2017	29/12/2017
30	5.2	Préparation de la soutenance	10j	11/12/2017	22/12/2017

Figure 5 : DIAGRAMME DE GANTT

3.2.2 Méthode de conception

Dans le déroulement du projet, la programmation représente seulement une petite partie des développements du système. La partie la plus importante réside dans la conception et l'architecture du système. Il faut quitter l'écran de l'ordinateur et prendre un cahier avec un stylo pour analyser et réfléchir à tous les aspects. Le diagramme (Figure 6) présente les étapes qui m'ont permis de comprendre le système existant et de concevoir l'intégration du moteur de recherche ainsi que le module d'administration. Le processus est divisé en deux grandes étapes. L'un est la conception de haut niveau, l'autre est la conception purement technique.

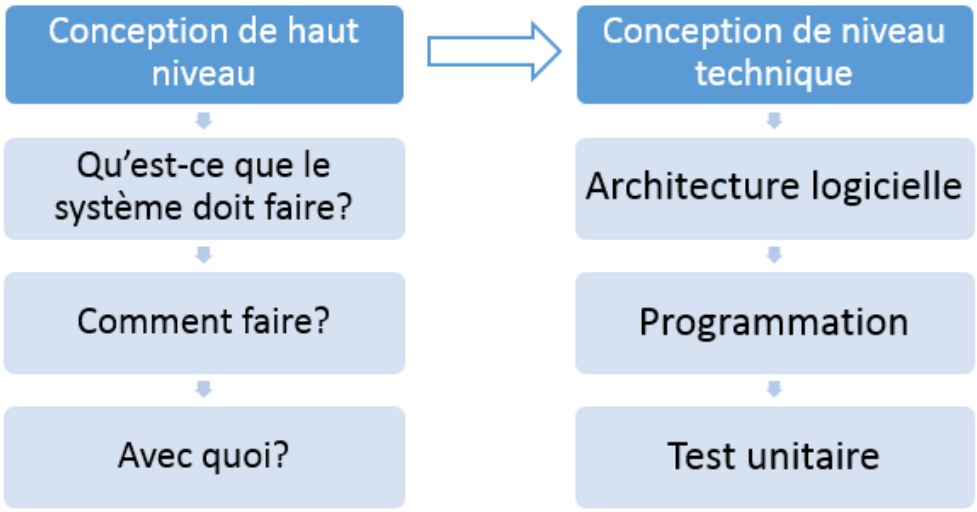


Figure 6 : METHODE DE CONCEPTION

Durant la première étape, la démarche consiste à répertorier les exigences fonctionnelles. Par exemple, quelles sont les entités ? Comment manipuler les données ? Qu'est-ce qu'un scénario ? Par ailleurs, poser des questions pendant la conception est une bonne habitude. Ensuite « comment faire ? » implique de préciser le modèle de données et de déterminer les relations entre les entités. Il n'est pas toujours efficace de définir un seul modèle de données pour le système entier. Les différents modules doivent être couplés de manière aussi faible que possible, cela se fait via la définition d'interface et de couches d'abstraction. A la fin de cette étape, « Avec quoi » consiste à identifier quelles sont les technologies les plus adaptées. Dans la seconde partie sont définis l'architecture logicielle, la programmation et les tests unitaires.

3.3 Technologies utilisées

Le système GDE est basé sur la plate-forme Java EE qui est un framework de développement d'applications d'entreprise. Le système GDE utilise un sous ensemble de Java EE qui va être présenté plus en détail par la suite. En particulier, le moteur de recherche qui est déployé dans le système utilise Apache Lucene, en plus d'Apache Tika pour le module d'extraction des données.

Une application Java EE s'exécute sur un serveur d'applications. Ici, le système GDE utilise le serveur d'application « Oracle GlassFish Enterprise Server ». Pour instancier efficacement le client web, de bonnes connaissances des technologies utilisées pour le frontend comme HTML, CSS, Javascript, le Framework Bootstrap et la bibliothèque jQuery sont également nécessaires. Le système utilise les technologies REST et AJAX pour la communication client/serveur.

Les outils NetBeans IDE et Git sont utilisés pour faciliter le développement et la gestion de configuration dans le projet.

3.3.1 JAVA EE

Java EE est l'acronyme de Java Entreprise Edition. Cette édition est dédiée à la réalisation d'applications d'entreprises. Java EE est basé sur J2SE (Java 2 Standard Edition) qui contient les API de base de Java.

Java EE est une plate-forme fortement orientée serveur pour le développement et l'exécution d'applications distribuées. Elle est composée de deux parties essentielles :

- un ensemble de spécifications pour une infrastructure dans laquelle s'exécutent les composants écrits en Java : un tel environnement se nomme serveur d'applications.
- un ensemble d'APIs qui peuvent être obtenues et utilisées séparément. Pour être utilisées, certaines nécessitent une implémentation de la part d'un fournisseur tiers.

L'utilisation de Java EE pour développer et exécuter une application offre plusieurs avantages :

- une architecture d'applications basée sur les composants qui permet un découpage de l'application et donc une séparation des rôles lors du développement ;

- la possibilité de s'interfacer avec le système d'information existant grâce à de nombreuses APIs : JDBC, JNDI, JMS, JCA ... ;
- la possibilité de choisir les outils de développement et le ou les serveurs d'applications utilisés qu'ils soient commerciaux ou libres.

3.3.2 Oracle GlassFish Server

Oracle GlassFish Server fournit un serveur pour le développement et le déploiement des applications Java EE et des technologies Web basées sur la technologie Java. GlassFish Server fournit les éléments suivants :

- un noyau léger et extensible basé sur les standards « OSGi Alliance » ;
- un conteneur Web ;
- une console d'administration facile à utiliser pour la configuration et la gestion ;
- la prise en charge du clustering de haute disponibilité et de l'équilibrage de charge.

3.3.3 EJB, JPA et JTA

EJB est l'acronyme d'Enterprise JavaBeans. C'est une architecture de composants logiciels côté serveur pour la plateforme de développement Java EE. Ces composants s'exécutent dans le conteneur EJB, un environnement d'exécution au sein du serveur d'applications. EJB permet de faciliter un développement rapide et de simplifier le développement d'applications distribuées, transactionnelles, sécurisées et portables. Il est raisonnable d'envisager d'utiliser EJB quand l'application doit répondre à l'une des exigences suivantes :

- l'application doit être évolutive ;
- les transactions doivent garantir l'intégrité des données ;
- l'application aura de nombreux clients concurrents.

JPA est l'acronyme de Java Persistence API. Cette API fournit une fonction de mappage objet / relationnel pour la gestion des données relationnelles. La persistance avec JPA se décompose en trois thèmes:

- les « Entity Beans » ;
- le langage de requête JPQL;
- les métadonnées de mapping objet / relationnel.

Il existe actuellement trois implémentations principales de JPA : EclipseLink est l'implémentation de référence ; Hibernate est un projet porté par JBoss, appartenant à RedHat ; OpenJPA est un projet de la fondation Apache.

JTA est l'abréviation de Java Transaction API. Cette API permet de démarquer les transactions d'une manière indépendante de l'implémentation du gestionnaire de transactions. Une transaction JTA est contrôlée par le gestionnaire de transactions Java EE. L'un des principaux avantages de JTA est sa capacité à gérer les transactions de plusieurs bases de données simultanément.

3.3.4 Servlet & JSP

Une Servlet est une classe Java qui permet de créer dynamiquement des données au sein d'un serveur HTTP. Ces données sont le plus généralement présentées au format HTML, mais elles peuvent également l'être au format XML ou tout autre format destiné aux navigateurs web. Un ou une servlet s'exécute dynamiquement sur le serveur web et permet l'extension des fonctions de ce dernier, par exemple : l'accès à des bases de données, transactions d'e-commerce, etc.

Le Java Server Pages ou JSP est une technologie basée sur Java qui permet aux développeurs de créer dynamiquement du code HTML, XML ou tout autre type de page web. Cette technologie permet au code Java et à certaines actions prédéfinies d'être ajoutés dans un contenu statique. Les JSP sont compilées par un compilateur JSP pour devenir des servlets.

3.3.5 REST

REST veut dire de REpresentational State Transfer qui est un style d'architecture pour les systèmes hypermédia distribués utiliisant le protocole HTTP. Il s'agit d'un ensemble de conventions et de bonnes pratiques à respecter et non d'une technologie à part entière. Les états d'application sont divisés en ressources qui sont adressés uniquement par une syntaxe universelle comme un identifiant URI. Toutes les ressources partagent une interface uniforme comme celle du protocole HTTP pour transférer l'état entre un client et une ressource.

3.3.6 Ajax

La technologie Ajax (acronyme d'Asynchronous Javascript and XML) permet de construire des applications Web et des sites web dynamiques interactifs sur le poste client en se servant de différentes technologies ajoutées aux navigateurs web.

En utilisant Ajax, le dialogue entre le navigateur et le serveur se déroule la plupart du temps de la manière suivante : un programme écrit en langage de programmation JavaScript, incorporé dans une page web, est exécuté par le navigateur. Celui-ci envoie en arrière-plan des demandes au serveur Web, puis modifie le contenu de la page actuellement affichée par le navigateur Web en fonction du résultat reçu du serveur, évitant ainsi la transmission et l'affichage d'une nouvelle page complète.

En Ajax, comme le nom l'indique, les demandes sont effectuées de manière asynchrone : le navigateur Web continue d'exécuter le programme JavaScript alors que la demande est partie, il n'attend pas la réponse envoyée par le serveur Web et l'utilisateur peut continuer à effectuer des manipulations pendant ce temps.

3.3.7 jQuery

jQuery est une bibliothèque JavaScript libre et multi-plateformes créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web. La bibliothèque

contient notamment les fonctionnalités suivantes : parcours et modification du DOM ; événements ; effets visuels et animations ; manipulations des feuilles de style en cascade ; Ajax ; plugins ; utilitaires (version du navigateur web…).

3.3.8 Apache Lucene

Lucene est une bibliothèque open source haute performance et complète écrite en Java qui permet d'indexer et de chercher du texte. Ce n'est pas une application complète, mais plutôt une API qui peut facilement être utilisée pour ajouter des fonctionnalités de recherche aux applications. Lucene propose des fonctionnalités puissantes à travers une API simple: indexation évolutive et haute performance ; algorithmes de recherche performants, précis et efficaces ; solution multi-plateforme.

3.3.9 Apache Tika

Apache Tika est un toolkit développé par la fondation Apache qui permet de détecter, d'extraire des métadonnées, et de structurer le contenu textuel de nombreux types de documents (.doc, .xls, .ppt, .pdf, .zip,...). Ce projet qui dépend de « Apache Software Foundation », était auparavant un sous-projet d'Apache Lucene. Pour la plupart des formats les plus courants et les plus populaires, Tika propose ensuite des fonctions d'extraction de contenu, d'extraction de métadonnées et d'identification de la langue. Tika est écrit en Java, il est aussi largement utilisé dans d'autres langages de développement.

IV. Contenu du stage

L'objet de cette partie est de présenter les tâches concrètes réalisées pendant ce stage. Dans un premier temps, il est nécessaire de découvrir tous les aspects afin de mieux comprendre le projet. Une fois les technologies prises en main, le travail s'est décomposé en deux grandes parties : la mise en place du moteur d'indexation et de recherche dans le système, ainsi que la conception et le développement du module d'administration côté Web. Tout au long du stage, une heure par jour a été consacrée à l'acquisition de savoirs, à l'amélioration de mon français.

4.1 Monter en compétence

Au début de mon stage, le système GDE était toujours en cours de développement. De nombreuses fonctionnalités étaient déjà implémentées par les membres de l'équipe du projet. Dans ce contexte, ma première tâche a été d'acquérir les compétences nécessaires sur les technologies utilisées dans le système GDE et qui ont été présentées dans le chapitre précédent. Par la suite, l'appropriation des sources existantes et l'analyse de la structure interne du GDE ont représenté une partie importante de mon activité.

4.1.1 Exigences du GDE

Avant de participer au développement du système, il m'a fallu comprendre d'abord toutes les exigences, techniques et fonctionnelles. Car les exigences ont des influences durant toutes les phrases de développement.

4.1.1.1 Exigences non Fonctionnelles

En général, les exigences techniques sont des exigences qui concernent la performance, la robustesse, la maintenabilité d'un système.

Certaines exigences techniques du système GDE concernent surtout la résolution de manière définitive d'un ensemble de préoccupations techniques récurrentes :

- authentification ;
- persistance et stockage ;
- intégrité des données ;
- contraintes réseaux ;
- volumétrie des flux ;
- indexation, recherche et extraction ;
- forte exigence d'adaptabilité de l'interface aux besoins ;
- garantie de cohérence avec la gouvernance des données (les autres projets).

4.1.1.2 Exigences fonctionnelles

Les exigences fonctionnelles sont des exigences définissant des fonctions du système à développer, c'est-à-dire ce que le système doit faire.

Voici, les exigences fonctionnelles principales :

- création d'une étude ;
- ajout d'une donnée (fichier) dans l'étude ;
- caractérisation d'une étude par des propriétés ;
- caractérisation d'une donnée par des propriétés (rôle de la donnée au sein de l'étude) ;
- indexation automatique des données par l'analyse de leur contenu ;
- recherche de données et d'études au moyen des propriétés et/ou d'un moteur de recherche.

4.1.2 Modèle de données du GDE

Après avoir déterminé les exigences du système, il faut déterminer comment le système peut répondre à ces exigences. Le diagramme (Figure 7) donne une vision du modèle conceptuel noyau :

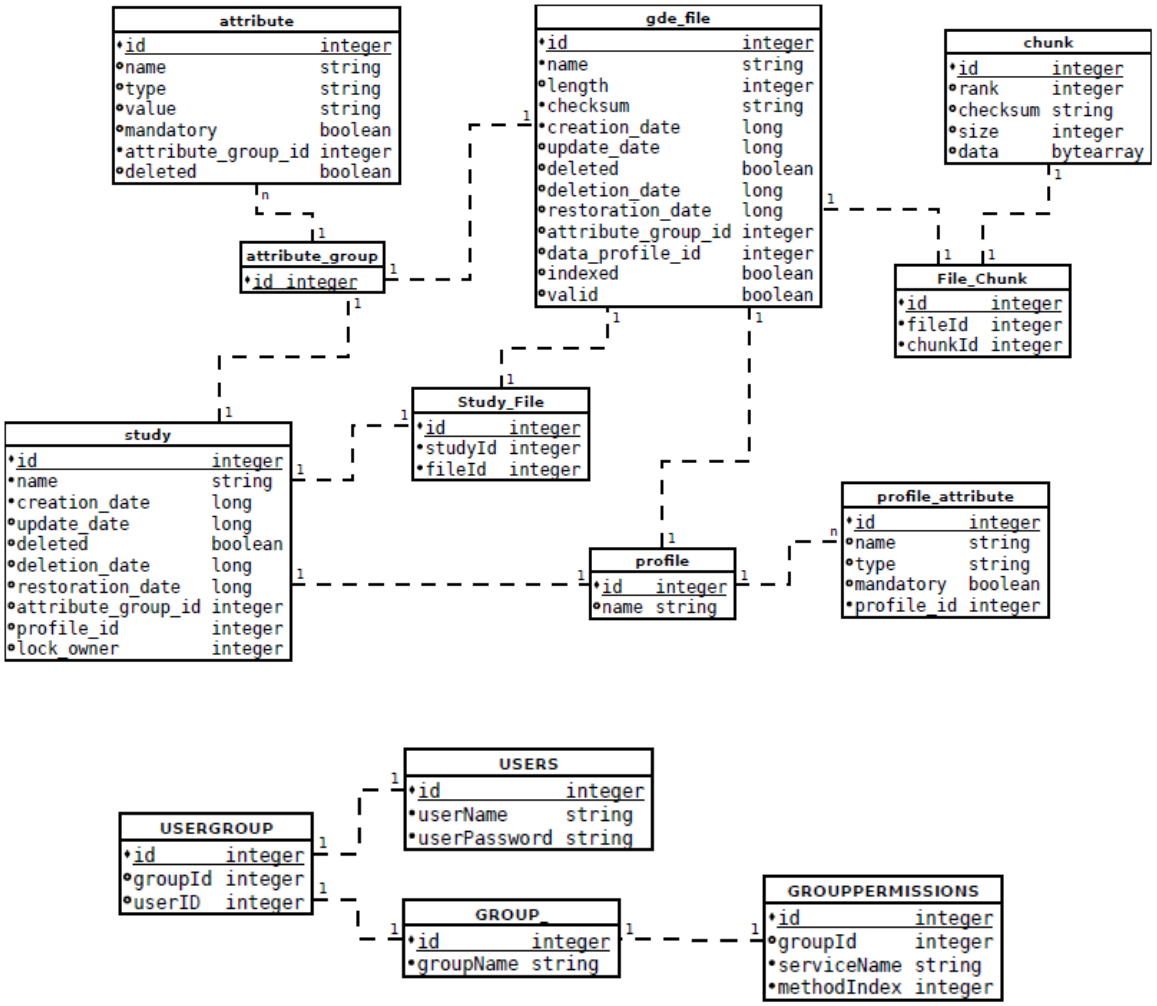


Figure 7 : MODÈLE DE DONNÉES DU GDE, VERSION 1

Ce diagramme introduit toutes les entités définies et les relations entre elles :

Nom des entités	Description des entités
USERS	Les utilisateurs du GDE avec des logins et mots de passe.
GROUP	Les utilisateurs appartiennent à un ou plusieurs groupes pour la gestion des rôles.
GROUPPERMISSIONS	Les droits d'un groupe
Study	L'ensemble des données et des propriétés décrivant le contenu et le contexte de l'étude.
Attribute	Une propriété dans *Attribute_group*.
Attribute_group	L'ensemble de propriétés ajoutées dans une *Study* ou un *File*.
Profile	Un modèle de l'ensemble des propriétés définies pour caractériser un ensemble d'études partageant une cohérence métier.

Profile_attribute	Une propriété dans *Profile*.
Gde_file	Les fichiers attachés à une *Study*, c'est un type de contenu d'une étude.
Chunk	Les fichiers sont divisés en petits morceaux (*Chunk*), ce qui permet de faciliter le stockage des grands fichiers dans la base de données.
Study_File	La relation entre *Study* et *gde_file*.
File_Chunk	La relation entre *gde_file* et *Chunk*.
USERGROUP	La relation (tableau jointure) entre *USERS* et *GROUP*.

TABLEAU 1 : LISTE DES ÉNTITES DU GDE

4.1.3 Architecture du GDE

Le GDE est basé sur l'architecture trois tiers qui est divisée en trois couches logicielles dont le rôle est clairement défini :

- la présentation des données, premier niveau, correspondant à la partie visible et interactive pour les utilisateurs ;
- le traitement métier des données, deuxième niveau, correspondant à la partie fonctionnelle d'implémentation de la logique métier, la mise en œuvre de l'ensemble des règles de gestion et de la logique applicative ;
- l'accès aux données persistantes, troisième niveau, correspondant aux données qui sont destinées à être conservées dans la base de données.

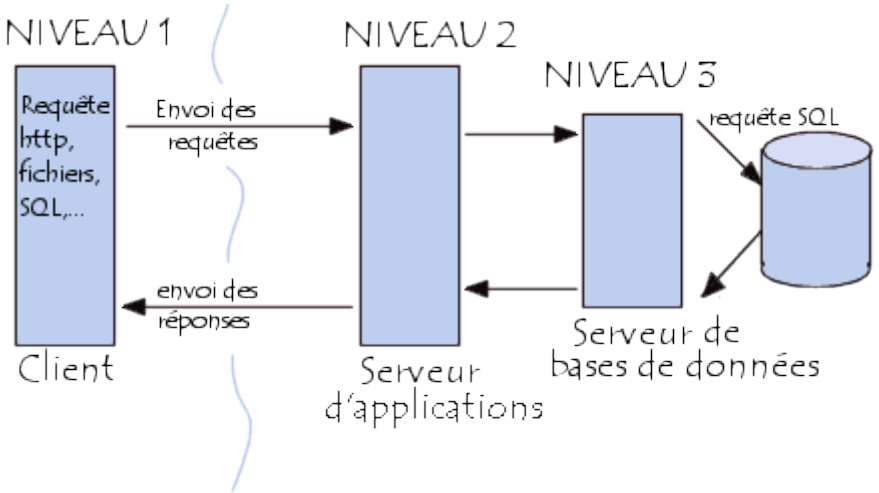


Figure 8 [1] : MODÈLE D'ARCHITECTURE TROIS TIERS

Dans cette architecture, les couches communiquent entre elles en utilisant un ensemble d'interfaces. Chaque couche ne communique qu'avec ses voisins immédiats. La couche de présentation envoie les requêtes de l'utilisateur à la couche de traitement qui va ensuite

demander les données du système à la couche d'accès aux données persistantes. Après le traitement des informations récupérées, elle retourne des réponses aux clients.

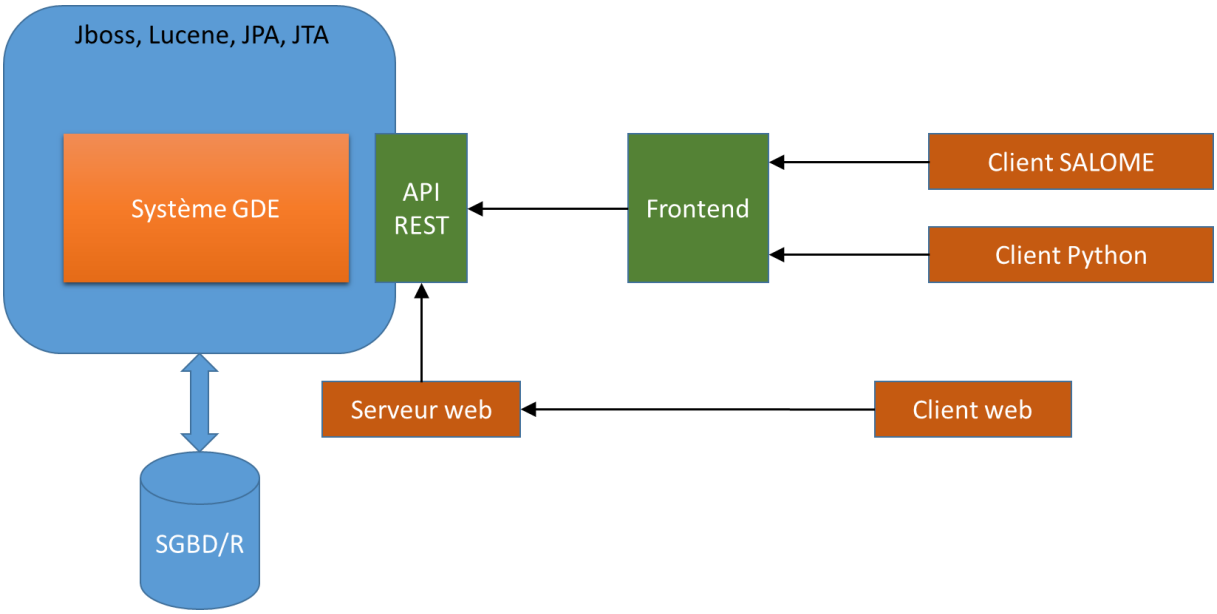


Figure 9 : ARCHITECTURE D'ENSEMBLE DU GDE

Le schéma (Figure 9) représente un point de vue de l'architecture d'ensemble du GDE. Dans ce schéma, le GDE se compose de plusieurs composants :

- Pour la couche de présentation, une API Python / C++ est fournie afin de permettre d'interfacer les applications métier ;
- Pour la couche de traitement métier, il s'agit du cœur du système. Ce dernier est déployé dans un serveur d'application Glassfish qui se charge de la communication bidirectionnelle multitâche entre les clients et le système, l'authentification des utilisateurs et la gestion complète des bases de données ;
- Le GDE utilise une base de données relationnelle et transactionnelle pour garantir l'intégrité des données.

Les clients peuvent récupérer les données dont ils ont besoin et en transmettre au serveur via l'API REST.

Pourquoi utiliser une architecture trois tiers ?

Le plus grand avantage de l'architecture trois tiers c'est son faible couplage. C'est-à-dire que chaque composant joue un rôle spécifique. Le remplacement d'un des modules n'impacte pas les autres tant que les contrats définis par interfaces sont respectés. Ceci permet de maintenir le système aisément et à coûts raisonnables.

4.1.4 Analyse des codes sources existants

Le système GDE est écrit en langage Java. La Figure 10 présente l'architecture logicielle du serveur GDE. Les *Entity Beans* utilisant JPA, sont des objets Java de type POJO (*Plain Old Java Object*) mappés vers une table de la base de données. Un POJO est une classe Java qui

n'implémente aucune interface ni n'hérite d'aucune classe parent. Les *Entity Beans* permettent d'encapsuler les données d'une occurrence d'une ou plusieurs tables. La couche DAO (*Data Access Object*) se charge de l'accès aux données et de leur manipulation indépendamment du SGBD (système de gestion de base de données) choisi. La couche métier en EJB fait un lien entre la couche DAO et les couches supérieures. Enfin, les servlets reçoivent les requêtes HTTP venant des clients et les traitent en appelant les méthodes de la couche de métier.

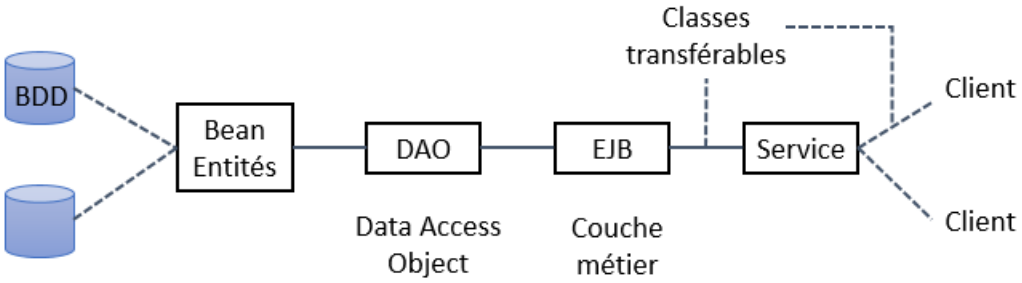


Figure 10 : L'ARCHITECTURE LOGICIELLE DU SERVEUR GDE

Le schéma (Figure 11) montre l'architecture mise en place pour le cas de la gestion des fichiers. Cette architecture est également utilisée pour les autres données (*Study, User, Permission, Profiles, Attributes*). Le code *GDE.sql* initialise la base des données au juste après un déploiement. La classe *GDEFile* est mappée vers la table *Gde_file* stockée dans la base des données. L'interface *FileDAO* et son implémentation *FileDAOImpl* qui contiennent a minima les méthodes CRUD (*Create, Read, Update, Delete*) permettent d'éviter la communication directe entre la couche métier *FileEJB* et le système de stockage. L'objet transférable *FileTO* est une classe POJO sérialisable avec des méthodes « getter » et « setter » qui peuvent être transférées à travers le réseau.

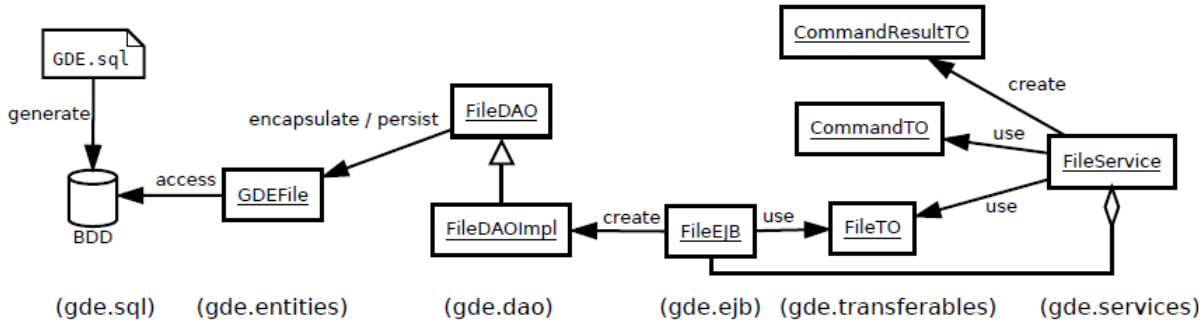


Figure 11 : ILLUSTRATION DU FONCTIONNEMENT DU SERVEUR SUR LE CAS DU CONCEPT DE FICHIER

Les objets transférables *CommandTO* et *CommandResultTO* sont conçus comme des enveloppes permettant de faire de manière générique des requêtes et par la suite de transporter les réponses quelle que soit leur forme et ainsi de faciliter la communication entre les services et les clients via la technologie REST (Figue 12).

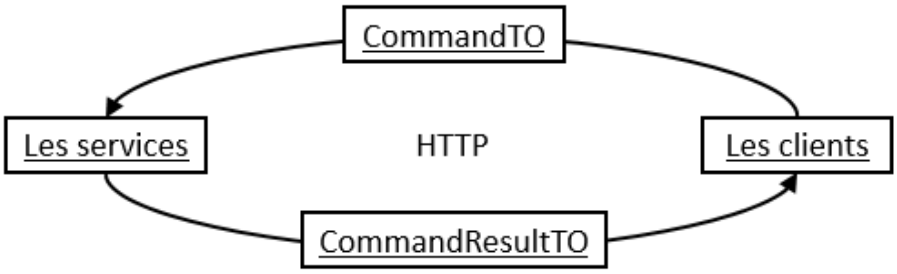


Figure 12 : COMMUNICATION ENTRE LES CLIENTS ET LE SERVEUR

4.2 Intégration du moteur de recherche dans le système GDE

Une fois accomplies les tâches d'analyse et de compréhension du système, on peut passer à l'étape suivante : la mise en place du moteur de recherche dans le serveur GDE. Pour commencer, ma tâche a consisté à faire l'état de l'art des modules d'extraction de données. Suite à quoi, on a choisi un composant qui s'appelle Apache Tika, ce dernier répond aux exigences technico-fonctionnelles définies. Ceci avec l'analyse des APIs de Lucene permet de commencer sereinement la conception de l'intégration de Lucene dans le GDE pour en faire son moteur de recherche. Puis, on passe au développement et à la mise en place des tests unitaires.

4.2.1 Module d'extraction de données

L'indexation et la recherche dans le GDE sont plus précisément ceux des « études » et des fichiers attachés sur la base de leurs caractéristiques sous forme de propriétés et de leurs contenus. Le but du module d'extraction de données est d'extraire des métadonnées, le contenu textuel des différents formats de données, par exemple, PDF, Microsoft Office, images, etc. et ensuite de les transférer au moteur d'indexation et de recherche.

4.2.1.1 État de l'art des modules d'extraction de données

Dans le cadre du GDE, on a besoin d'extraire au moins les données aux formats PDF, WORD, EXCEL, POWERPOINT. Le support d'autres formats représente un bonus. Après avoir étudié les outils existants d'extraction de données, les plus couramment utilisés sont :

- **Apache POI** : une bibliothèque open source développée et distribuée par « *Apache Software Foundation* » pour concevoir ou modifier des fichiers Microsoft ;
- **Apache PDFBox** : une bibliothèque Java open-source qui prend en charge la création et la conversion de documents PDF. En utilisant cette bibliothèque, il est possible de développer des programmes Java qui créent, convertissent et manipulent des documents PDF ;
- **Tagsoup Library** : une bibliothèque pour analyser HTML / XML. Elle prend en charge la spécification HTML 5 et peut être utilisée pour analyser du XML bien formé ou du HTML non structuré et malformé à partir du Web ;
- **Metadata-extractor** : une bibliothèque Java simple pour lire des métadonnées à partir de fichiers image ;

- **Common compress library** : une bibliothèque qui implémente la plupart des formats de compression ;
- **Metadata Extraction Tool** : un outil développé par la Bibliothèque nationale de Nouvelle-Zélande pour extraire les métadonnées à partir de différents formats de fichiers tels que les documents PDF, les fichiers image, les fichiers audio, les documents Microsoft Office et bien d'autres.
- **Apache Tika** : Voir 3.3.9

Le tableau suivant présente la comparaison que j'ai réalisée entre tous les outils cités plus haut: (X représente la possession d'une fonctionnalité, - représente l'absence)

	PDF	Microsoft Documents	Images	HTML	Formats scientifiques	Formats de compression
Apache POI	-	**X**	-	-	-	-
Apache PDFBox	**X**	-	-	-	-	-
Tagsoup Library	-	-	-	**X**	-	-
Metadata-extractor	-	-	**X**	-	-	-
Common compress library	-	-	-	-	-	**X**
Metadata Extraction Tool	**X**	**X**	**X**	**X**	-	-
Apache Tika	**X**	**X**	**X**	**X**	**X**	**X**

TABLEAU 2 : COMPARAISON D'OUTILS D'EXTRACTION DE DONNÉES

4.2.1.2 Choix du module en fonction des exigences

D'après l'état de l'art effectué, Apache Tika a finalement été choisi pour le GDE. Selon le tableau 2, il est évident qu'Apache Tika supporte le plus de formats, et surtout ceux dont nous avons besoin. Ensuite, le projet Apache Tika est un sous-projet d'Apache Lucene qui est la base du moteur de recherche. Enfin, il est facile de l'intégrer dans le système grâce aux interfaces génériques et au mécanisme de détection intelligent qui masque la complexité de la détection de format. Enfin, la faible utilisation de la mémoire et le traitement rapide sont aussi les points forts de l'outil. Par conséquent, le choix d'Apache Tika est le plus logique.

4.2.2 Analyse des APIs de Lucene et de Tika

4.2.2.1 APIs de Apache Lucene

Apache Lucene est une bibliothèque de moteurs de recherche de texte performante et complète. L'API Lucene est divisée en plusieurs paquets.

org.apache.lucene.analysis : Le paquet d'analyse fournit le mécanisme pour convertir les données en *Tokens* qui peuvent être indexés par Lucene. Il y a deux classes principales dans le paquet dont dérivent les processus d'analyse :

- *Analyzer* : Un « Analyzer » est responsable de fournir une séquence « *TokenStream* » des éléments (*Token* en anglais) qui peuvent être consommés par les processus d'indexation et de recherche ;
- *Tokenizer* : Un « *Tokenizer* » est un « *TokenStream* ». Il est responsable de la séparation du texte en mots ou petits éléments d'indexation (*Token*). Dans de nombreux cas, un analyseur utilisera un « *Tokenizer* » comme la première étape du processus d'analyse.

On peut bien sûr personnaliser l'analyseur, mais, Apache Lucene fournit déjà de nombreux type d'Analyzer permettant de traiter la plupart des cas. Le plus commun d'entre eux est le « *StandardAnalyzer* » pour l'anglais et le « *FrenchAnalyzer* » pour le français.

org.apache.lucene.document : Le paquet de document fournit la représentation logique du contenu à indexer et à rechercher. Le paquet fournit également des utilitaires pour travailler avec deux classes Documents et IndexableFields :

- *Document* : Un « *Document* » est l'unité d'indexation et de recherche, il est composé d'un ensemble de champs (*Field* en anglais). Chaque champ possède un nom et une valeur textuelle ;
- *IndexableField* : Un « *IndexableField* » est un champ à indexer et qui est une représentation logique du contenu qui doit être indexé ou stocké. Il a un certain nombre de propriétés qui indiquent à Lucene comment traiter le contenu.

org.apache.lucene.index : Ce package fournit les outils pour maintenir et procéder à l'indexation. Il a deux classes principales :

- *IndexWriter :* Crée et ajoute des « *Document* » à l'indexation ;
- *IndexReader :* Il accède aux données d'indexation.

IndexWriter et *IndexReader* sont complètement thread-safe, ce qui signifie que plusieurs threads peuvent appeler n'importe laquelle de leurs méthodes, en même temps.

org.apache.lucene.search : le package de *Search* fournit des structures de données pour représenter les requêtes (par exemple « *TermQuery* » pour les mots individuels, « *PhraseQuery* » pour les phrases et « *BooleanQuery* » pour les combinaisons booléennes de requêtes) et « *IndexSearcher* » qui transforme les requêtes en « *TopDocs* ». Un certain nombre de « *QueryParsers* » sont fournis pour produire des structures de requête à partir de chaînes de caractères ou de XML via *QueryParser.parse (Query)*.

- *IndexSearcher* : Les applications n'ont généralement besoin que d'appeler la méthode héritée *IndexSearcher.search* ;
- *TopDocs* : Représente les résultats renvoyés par la méthode *IndexSearcher.search*.

org.apache.lucene.store : le package Store définit une classe abstraite pour stocker des données persistantes : « *Directory* », qui est une collection de fichiers écrits par un « *IndexOutput* » et lus par un « *IndexInput* ». Plusieurs implémentations sont fournies, notamment « **FSDirectory** », qui utilise un répertoire de système de fichiers pour stocker les

fichiers, et « *RAMDirectory* » qui implémente les fichiers en tant que structures de données résidant en mémoire.

Pour utiliser les APIs de Lucene, une application doit suivre les processus dans la figure 13.

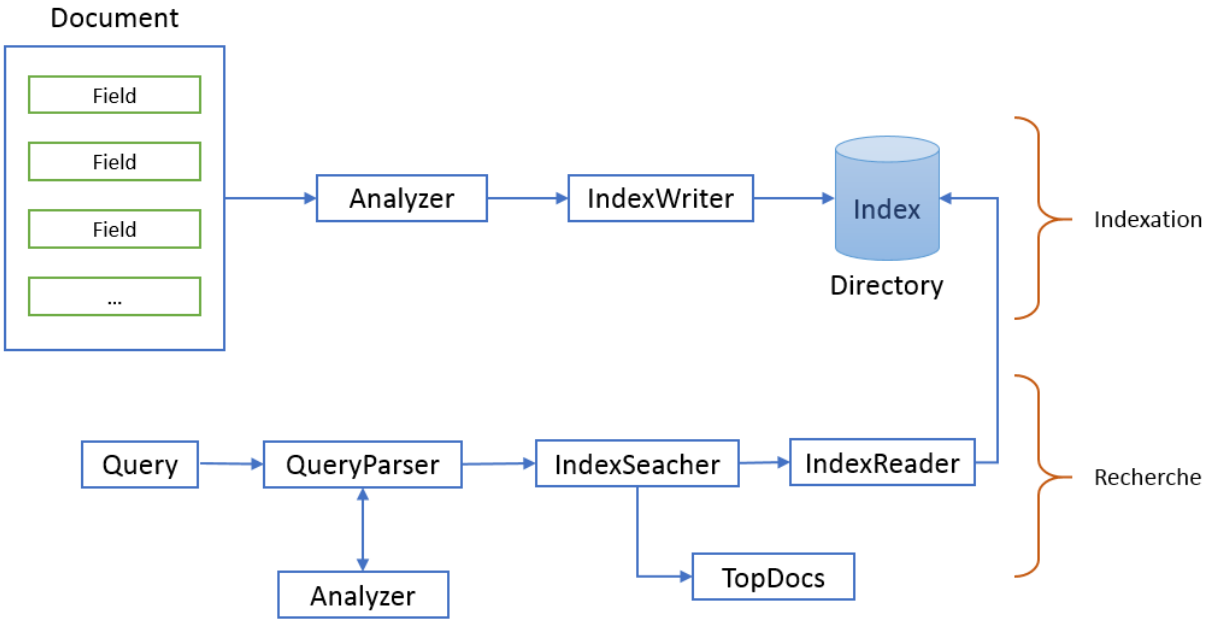


Figure 13 : PROCESSUS D'INDEXATION ET DE RECHERCHE DE LUCENE

Pendant le processus d'indexation, les *Documents* contenant des *Field* sont ajoutés à *IndexWriter* qui fait une analyse à l'aide de l'Analyzer. Puis l'index est créé / ouvert / édité selon les besoins et stocké / mis à jour dans un Directory. *IndexWriter* est utilisé pour mettre à jour ou créer un index mais pas pour le lire.

Pendant le processus de recherche, il faut d'abord créer un *Directory* contenant les index et puis transmettre à *IndexSearcher* qui va ouvrir le *Directory* avec *IndexReader*. Ensuite, un *Query* est créé pour effectuer une recherche avec un *IndexSearcher* qui va transmettre le *Query* au chercheur et renvoie un *TopDocs* contenant les résultats des détails de Document.

4.2.2.2 APIs de Apache Tika

Puisque Lucene accepte uniquement l'entrée de texte brut, le moteur de recherche a besoin d'un module d'extraction de données pour obtenir les textes à partir de fichiers de différents formats. Le GDE utilise Apache Tika qui est décrit au chapitre 4.2.1.

Dans l'architecture de Tika, il y a quatre modules importants.

Tika Façade class - org.apache.tika : Utiliser la classe de façade Tika est la manière la plus simple et la plus directe d'intégrer Tika, et elle suit le pattern de façade. Le module représente une abstraction pour toutes les implémentations internes et fournit des méthodes simples pour accéder aux fonctionnalités de Tika.

Parser Interface - org.apache.tika.parser.Parser : Une interface clé pour l'analyse de documents dans Tika qui va extraire le texte et les métadonnées d'un document. Tout ceci est

réalisé avec une seule méthode « *parse* ». La méthode prend en entrée le flux de document à analyser et les métadonnées associées.

Content Detection Mechanism - org.apache.tika.detect.Detector : une interface pour la détection de type de contenu.

Language Detection Mechanism - org.apache.tika.language : Tika est capable d'identifier la langue d'un texte, ce qui est utile lors de l'extraction de texte à partir de formats de documents qui n'incluent pas les informations de langue dans leurs métadonnées.

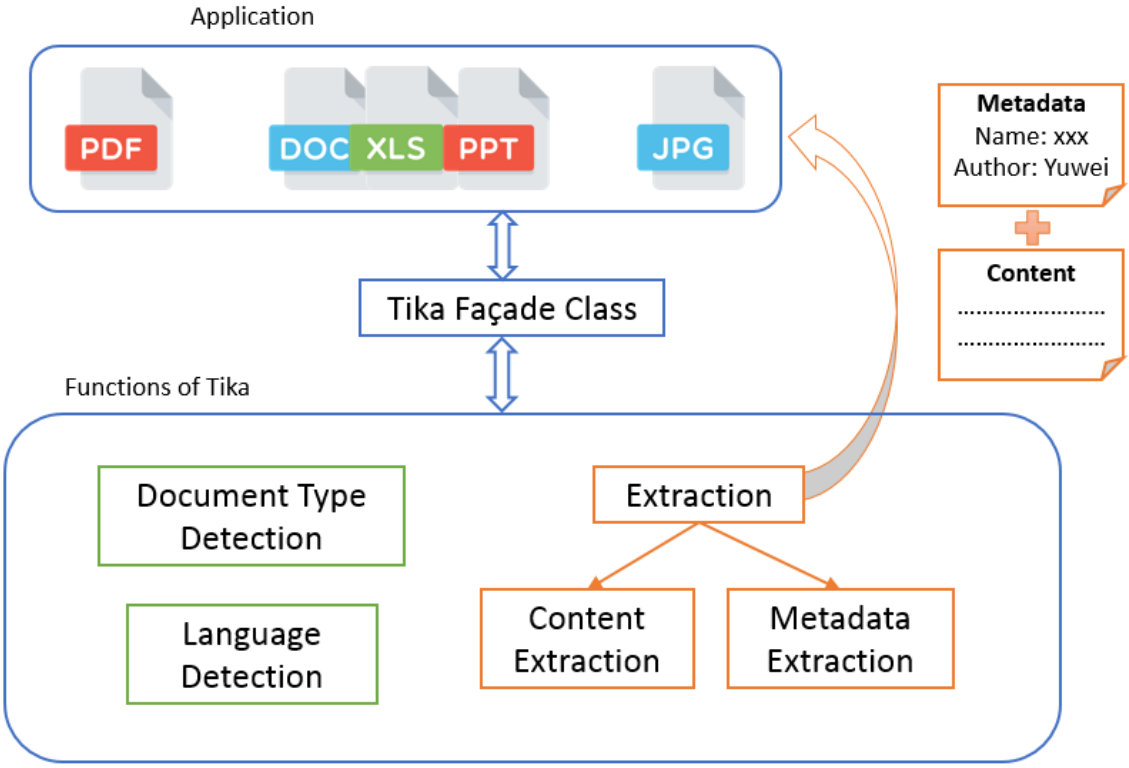


Figure 14 : STRUCTURE DE TIKA

Tika est largement utilisé lors du développement de moteurs de recherche pour indexer le contenu textuel de documents numériques. Le contenu extrait par le module d'extraction de Tika est transmis à l'indexeur du moteur de recherche.

4.2.3 Conception de composant d'intégration de Lucene et Tika

Afin d'assurer l'indépendance entre les composants, il faut concevoir les interfaces qui permettent de découpler de manière efficace les composants afin d'assurer l'évolutivité du système sur le long terme.

Dans le schéma 15, en utilisant le « patron adaptateur », *LuceneDocumentIndexer* et *LuceneDocumentSearcher* implémentent les interfaces *GDEDocumentIndexer* et *GDEDocumentSearcher.* Ces implémentations s'interfacent avec Apache Lucene pour intégrer le module d'indexation et de recherche dans le GDE. Dans le futur, ces interfaces peuvent être implémentées pour utiliser d'autres technologies de moteur de recherche.

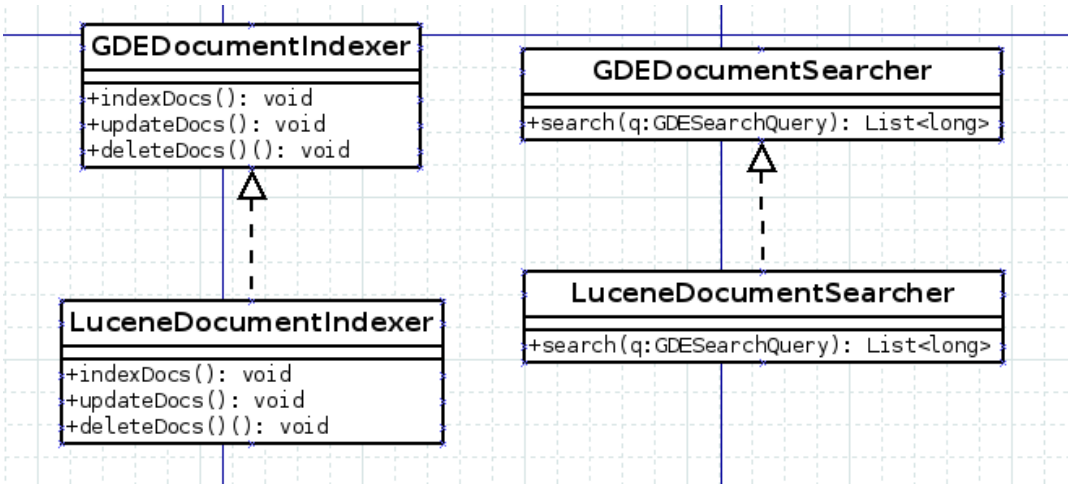


Figure 15 : INTERFACES DE MODULES D'INDEXATION ET DE RECHERCHE

Le schéma 16 montre la conception de l'intégration du module d'extraction de données pour le GDE. Le *DataExtractor* est l'interface de transformation des données en *GDEDocument* qui peut être utilisé dans le moteur de recherche. Une des exigences du GDE est qu'il doit être possible d'effectuer une recherche à partir des métadonnées d'une étude et des fichiers associés à l'étude. Afin de répondre de façon efficace à cette exigence, une interface commune pour ces deux fonctionnalités a été conçue. Ainsi, le *FileDataExtractor* et le *StudyDataExtractor* implémentent l'interface *DataExtractor*. En particulier, la classe enfant *TikaFileDataExtractor* hérite de *FileDataExtractor* pour extraire le contenu des fichiers. De même, il est facile de remplacer Apache Tika par d'autres outils d'extraction de données en héritant simplement de *FileDataExtractor*.

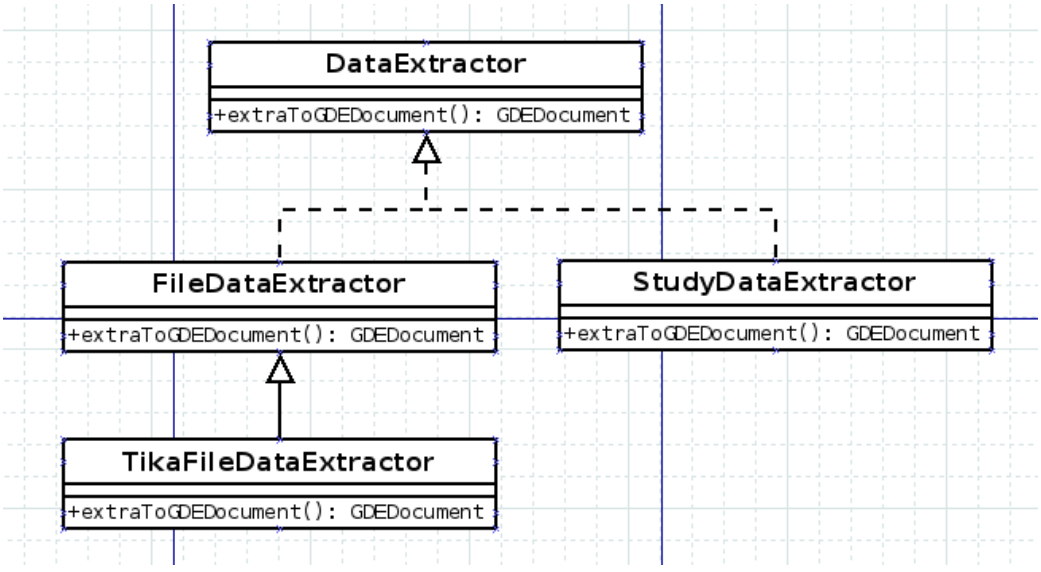


Figure 16 : INTERFACES DE MODULES D'EXTRACTION DE DONNEES

Grâce à ces interfaces, Apache Lucene et Apache Tika sont considérées comme des « boîtes noires » indépendantes, ce qui permet de cacher les complexités et autres parties non intéressantes dans ce contexte. Il est ainsi possible de remplacer les composants à faible coût. Par ailleurs le code source reste simple et facile à maintenir.

4.2.4 Développement du moteur de recherche

L'architecture logicielle de la partie moteur de recherche suit la même philosophie que le reste du GDE. Le code métier du moteur de recherche se trouve dans un EJB dédié. Comme on peut le voir sur la figure 17, un ensemble de classes a été développé pour permettre non seulement l'intégration de Lucene mais également son abstraction. Ceci résulte de la volonté de ne pas être dépendant d'une technologie de moteur de recherche particulière.

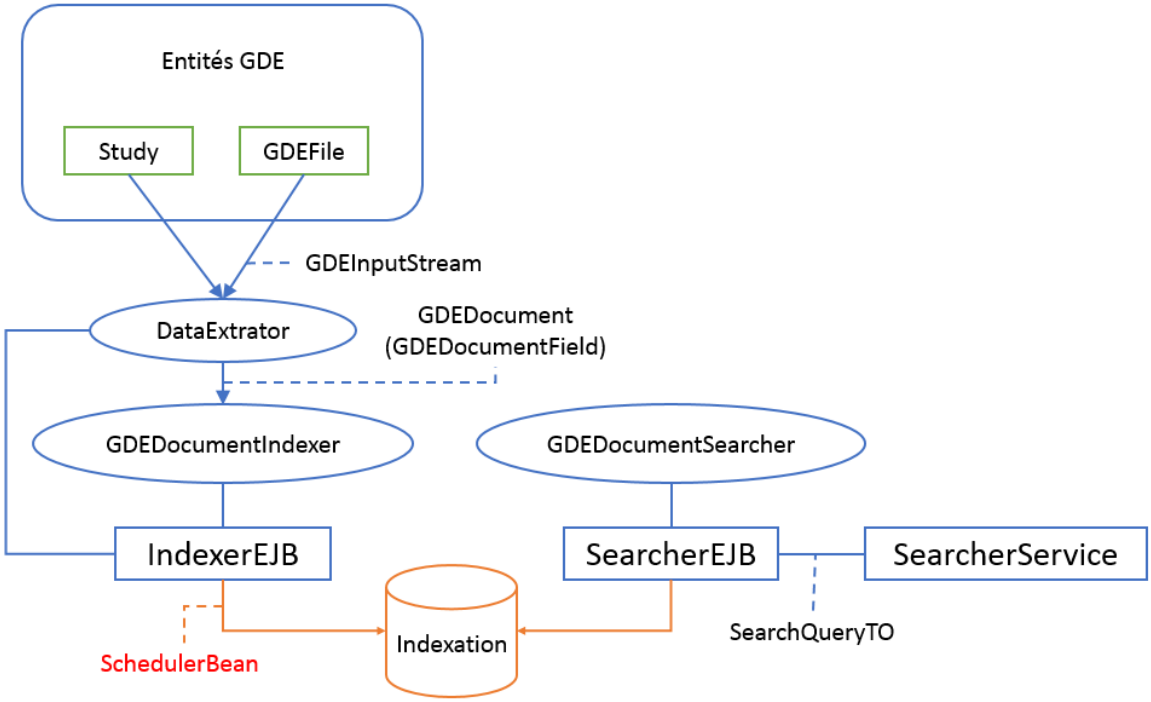


Figure 17 : STRUCTURE DU MOTEUR DE RECHERCHE DANS LE GDE

4.2.4.1 Syntaxe de recherche souple

Avec le moteur de recherche du GDE, il est possible de chercher de différentes manières.

La manière la plus simple est de chercher par mot clé dans les études elles-mêmes ou à l'intérieur des fichiers. Par exemple, si nous cherchons les fichiers caractérisés par le mot « architecture », le moteur de recherche va lister tous les fichiers qui contiennent le mot clé « architecture ».

Il est également possible de chercher par une phrase en utilisant des guillemets. Par exemple, « 'architecture du système d'information' » permet de lister les fichiers contenant précisément cette phrase.

De plus, il est possible de chercher grâce à des équations de recherche en utilisant des opérateurs tel que AND, OR, NOT, « + », « - », etc. Par exemple, on peut trouver tous les fichiers qui concernent l' « architecture du système d'information » mais qui ne contiennent pas « Thomas » par l'équation de « 'architecture du système d'information' - 'Thomas' ».

Les attributs des études et des fichiers peuvent être utilisés dans les équations de recherche. C'est-à-dire si nous voulons chercher les fichiers dont l'auteur est « Yuwei » qui parlent d' « architecture du système d'information » et dans lesquelles nous ne trouvons pas « Thomas », l'équation de recherche correspondante s'écrit ainsi : « auteur : 'Yuwei' 'architecture du système d'information' - 'Thomas' ».

En fin, le moteur de recherche du GDE supporte les recherches génériques de termes simples en utilisant le symbole « ? » pour un seul caractère et « * » pour plusieurs caractères. Par exemple, « informati* » pour « informatif », « informatique » ou « information ».

Tout ceci est réalisé basé sur un analyseur de requête (*QueryParser*) de Lucene.

4.2.4.2 Standardisation des études et des fichiers

Les études dans le GDE sont définies par des propriétés : le nom d'étude, une date de création, de mise à jour, de suppression, de restauration, et des propriétés personnalisées. Ce sont les métadonnées des études. Les fichiers dans le système sont définis par les mêmes types de métadonnées en plus du contenu du fichier.

Pour uniformiser les entrées des interfaces et faciliter la recherche des données, les études et les fichiers sont considérés comme étant la même chose : des métadonnées et un contenu. La figure 18 présente le concept évoqué ici.

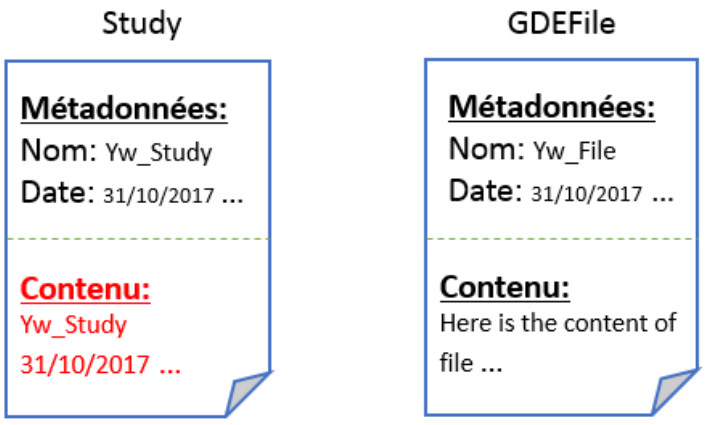


Figure 18 : STANDARDISATION DE STUDY ET GDEFILE

Donc, tant pour les études que pour les fichiers, la recherche est soit dans les métadonnées quand c'est spécifié dans la requête, soit par défaut dans le contenu.

4.2.4.3 Implémentation d'InputStream pour les fichiers du GDE

Dans le GDE, le contenu des fichiers n'est pas directement sauvegardé dans le système de fichier mais dans la base de données. Cependant les bases de données ne permettent pas le stockage de fichiers très volumineux. Donc les fichiers sont divisés en petits morceaux (*Chunk*) pour faciliter le stockage. Lorsque nous voulons lire le contenu du fichier, il faut parcourir tous les morceaux associés au fichier dans l'ordre. Par ailleurs, l'interface d'extraction Tika a besoin d'un flux d'octets en entrée. Donc, une classe *GDEFileInputStream* a été développée. Cette

dernière hérite de l'interface standard *InputStream* permettant ainsi de représenter les données stockées en base comme des flux binaires tels qu'attendus par Apache Tika.

4.2.4.4 Indexation automatique

L'indexation n'étant pas de la responsabilité de l'utilisateur, il est nécessaire de l'automatiser et ce sans perturber le fonctionnement du serveur.

La création, la mise à jour et la suppression déclenchent immédiatement le processus d'indexation car le volume des données à indexer est petit. En revanche, l'indexation du contenu des fichiers se fait de manière asynchrone afin de ne pas se retrouver dans la situation où plusieurs dizaines de fichiers seraient analysés et indexés en même temps ce qui aurait un impact négatif sur les performances du serveur. Nous avons fait le choix d'indexer les fichiers de façon périodique. Le service EJB Timer du serveur d'application permet de planifier l'exécution d'une fonction. Une classe *SchedulerEJB* a été développée en utilisant l'annotation *@Schedule* pour déclarer un minuteur. Ceci permet d'effectuer une indexation automatiquement avec une période définie.

Un verrou permet d'éviter l'exécution d'un nouveau processus d'indexation tant que la précédente n'est pas terminée.

4.2.5 Rédaction des documents techniques

La transmission des connaissances est une part importante du travail effectué par les ingénieurs. Une fois le développement du moteur de recherche terminé, j'ai rédigé la documentation correspondante en y rajoutant les APIs de haut niveau pour le développement de clients. Ces documentations sont compilées à l'aide de l'outil Sphinx.

Sphinx est un outil qui facilite la création d'une documentation intelligente et agréable à consulter. Il a été créé à l'origine pour la documentation de Python mais est aujourd'hui très utilisé pour les autres langages. Sphinx utilise reStructuredText comme langage à balise qu'il convertit, entre autres, en HTML, PDF, EPUB, ou man pages.

La syntaxe est simple et intuitive, un exemple est montré dans la figure 19 :

```
.. Copyright 2015-2017 EDF

.. py:classmethod:: GDESession.search(search_query)

   Chercher par un search query dans le système GDE.

   :param SearchQuery search_query: la requête de recherche.
   :return: une liste des identifiants des résultats trouvés.
   :rtype: liste de Long
```

Figure 19 : EXEMPLE DE LA DOCUMENTATION

Sphinx permet de spécifier le format de sortie qu'on souhaite obtenir, par exemple, le format HTML dans la figure 20.

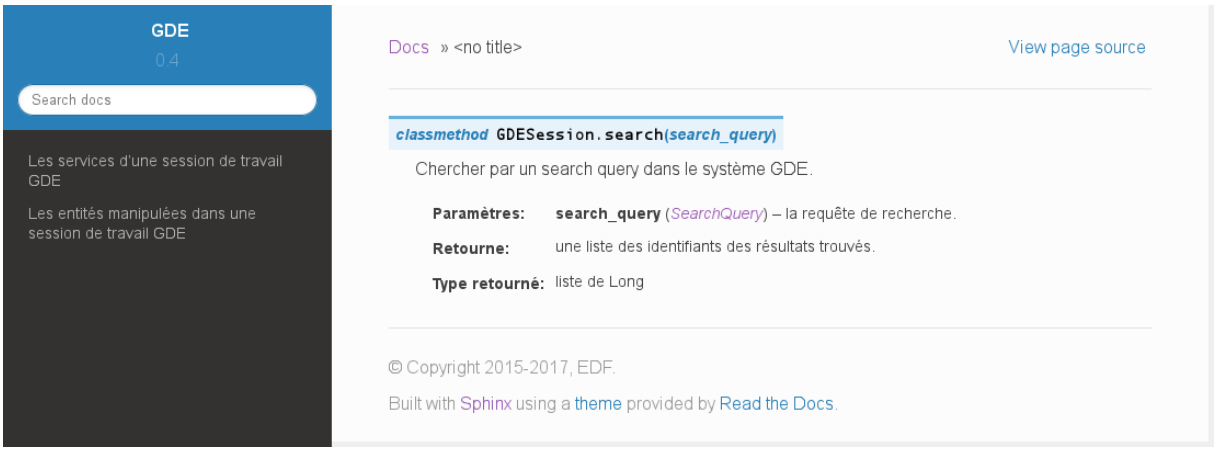


Figure 20 : EXEMPLE DE FICHIER HTML GÉNÉRÉ

Ce projet est développé par une équipe de plusieurs de personnes, donc une documentation de qualité est importante. D'un côté, il aide les autres membres dans l'équipe à comprendre et utiliser plus facilement les APIs offertes par le moteur de recherche pour le développement du côté client. Et de l'autre, il est comme une sorte de tutoriel pour les personnes qui me succéderont dans mon travail, leur permettant de reprendre rapidement et efficacement mon travail.

4.3 Un nouveau défi

À la fin du mois d'août, ce qui constituait le sujet de mon stage, à savoir l'intégration d'un moteur de recherche et d'indexation dans le GDE était effectivement terminé. Après discussions avec mon tuteur de stage et l'architecte du projet, nous avons décidé que je pouvais relever ce qui pour moi était un nouveau défi.

Une des futures phases pour le GDE consistait à concevoir et ensuite faire développer un module d'administration Web. N'ayant encore jamais fait de développement Web j'ai trouvé qu'il s'agissait d'une opportunité pour moi d'apprendre de nouvelles technologies et d'explorer les méthodes de conception et d'architecture autour de ces technologies.

4.4 Conception et développement du module d'administration

Comme expliqué dans le chapitre précédent, il y a quelques paramètres à configurer par les administrateurs du système. Donc, il est nécessaire de développer un module d'administration pour gérer le système du GDE. Par exemple, ajouter des utilisateurs, gérer des profils de l'étude et du fichier, modifier les valeurs de paramètres, etc. Ce module doit être en client léger, c'est-à-dire en client Web.

En partant de zéro, et en appliquant la méthode introduite dans le chapitre 3.3.2 à la conception et au développement du module d'administration, le déroulement de cette partie se compose de deux grandes phases. La première phase consiste à définir précisément le rôle d'administrateur et toutes les opérations possibles qu'il peut effectuer, puis les scénarios et les relations entre chaque module fonctionnel, et enfin le choix des technologies web les plus

adaptées en fonction du contexte. Dans la seconde phase plus technique, comme toujours, avant de programmer et tester, la conception de l'architecture logicielle est indispensable.

4.4.1 Conception du module d'administration

L'objet de la partie qui suit est de présenter la méthode adoptée et les résultats obtenus pour effectuer la conception de haut niveau du module d'administration.

4.4.1.1 Conception fonctionnelle

La première question qui doit être proposée est « qu'est-ce qu'un module d'administration et qu'est-ce qu'il peut faire? »

Un module d'administration, en d'autres termes, est un module de configuration, un module de gestion de données. Dans le GDE, il y a des entités à manipuler : *User*, *Group*, *Study*, *GDEFile*, *Profile* et *Configuration du système*. Pour chaque entité, toutes les opérations possibles doivent être précisées et organisées dans une espace de gestion spécifique. Le tableau 3 présente les opérations que les administrateurs peuvent effectuer sur ces entités.

Entité	Opérations
User	○ Création avec un login unique et un mot de passe ○ Suppression logique dans le système, mettre un « *flag* » : « *deleted* » ○ Modification de tous les champs même du mot de passe mais pas l'identifiant ○ Consultation, mais ne pas afficher le mot de passe qui doit être chiffré
Group	○ Création avec un nom ○ Suppression physique ○ Modification ○ Consultation ○ Ajoute un ou des *User* dans le groupe ○ Suppression d'un ou des *User* du groupe ○ Consultation de User du groupe ○ Gestion de droits (*Permissions*) : ajout et suppression
Study	○ Pas de droit à créer, ce sont les utilisateurs qui créent les études ○ Suppression logique dans le système, mettre un « *flag* » : « *deleted* » ○ Modification ○ Consultation de l'étude et des fichiers attachés ○ Recherche
GDEFile	○ Pas de droit à créer, ce sont les utilisateurs qui ajoutent les fichiers ○ Suppression physique par les administrateurs mais logique par les utilisateurs ○ Pas de droit à modifier ○ Consultation des métadonnées et le contenu ○ Recherche
Profile	○ Création avec un nom ○ Suppression physique ○ Modification ○ Consultation

	○ Ajout des propriétés dans le profil ○ Suppression les propriétés du profil
Configuration	○ Modification ○ Consultation

TABLEAU 3 : LES OPÉRATIONS POSSIBLES DES ENTITES

L'étape suivante est de faire des scénarios animés et de simuler toutes les opérations possibles. Quand un administrateur se connecte au système GDE, il doit d'abord s'identifier par la saisie de son login et de son mot de passe dans une page avec un formulaire. Le système GDE authentifie cet administrateur. Une fois que la vérification est faite, l'accès à la page du module d'administration est accordé, sinon, le système va afficher un message d'erreur, par exemple « *Login or Password invalid !* » ou « *Authorization Error !* ».

Une page d'administration est un espace fonctionnel, c'est-à-dire un espace de gestion des entités et d'administration de l'administrateur lui-même. La première page est une page d'accueil qui affiche un menu principal fixé en haut de la fenêtre, un message d'accueil et des données statistiques du GDE, par exemple, le nombre d'études dans le GDE, le nombre d'utilisateurs en ligne, etc. La figure 21 est une maquette d'une page d'accueil.

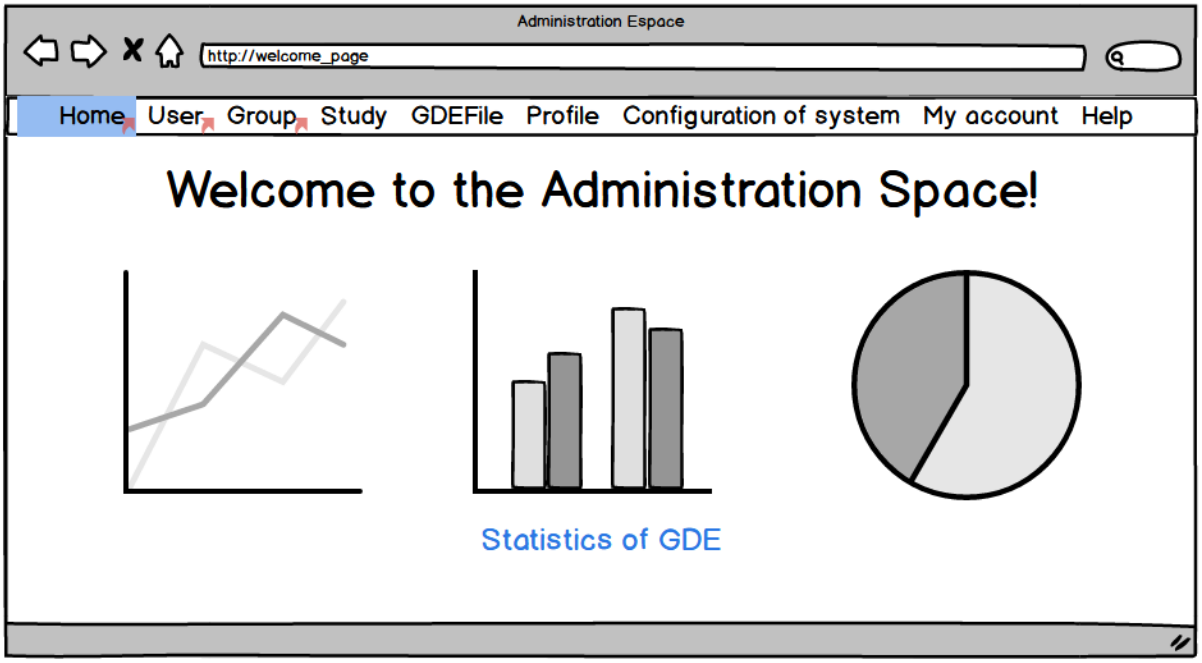


Figure 21 : MAQUETTE D'UNE PAGE D'ACCUEIL

Cliquer sur « User » dans le menu principal, donne accès à l'espace de gestion des utilisateurs (la maquette est en Figure 22). Dans cet espace, il y a une liste de tous les utilisateurs existants avec leur « Id » et leur nom. Pour chaque utilisateur, les informations détaillées peuvent être affichées en cliquant dessus. Une liste de groupes est là pour effectuer un filtrage afin de sélectionner les utilisateurs plus facilement. Par exemple, si nous cliquons sur le « groupe 1 », le système va filtrer et lister tous les utilisateurs du « groupe 1 ». Un ensemble de boutons permet les actions standard sur les utilisateurs, à savoir la création, la modification, la suppression, etc.

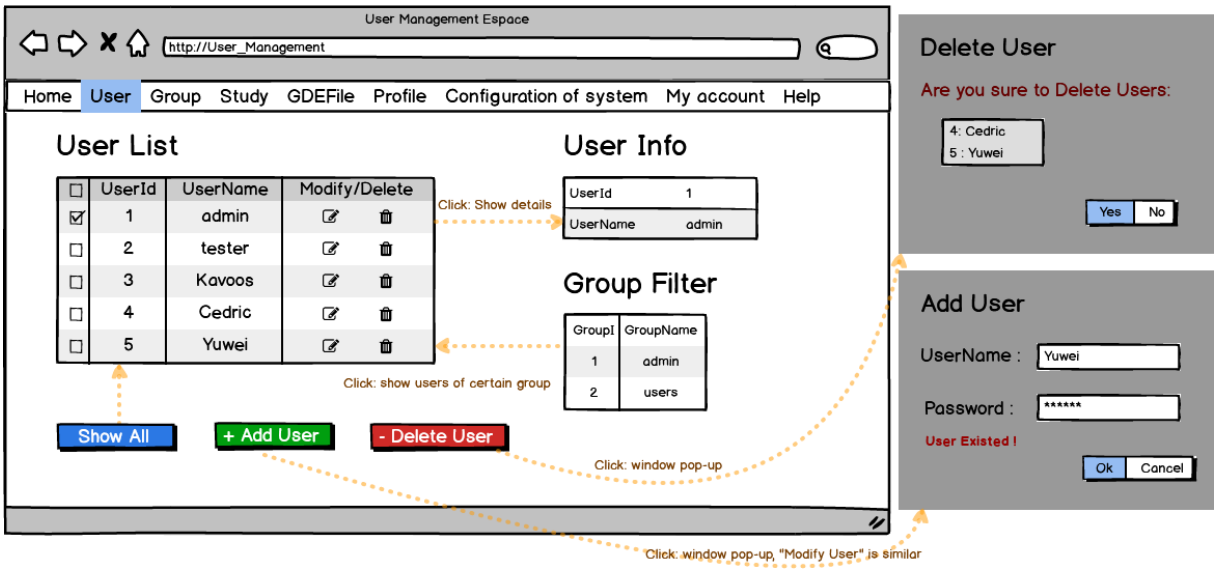


Figure 22 : MAQUETTE D'UN ESPACE USER

L'espace de gestion des groupes se comporte comme celui de la gestion des utilisateurs, on y retrouve les fonctions standards de gestion de groupes d'utilisateurs.

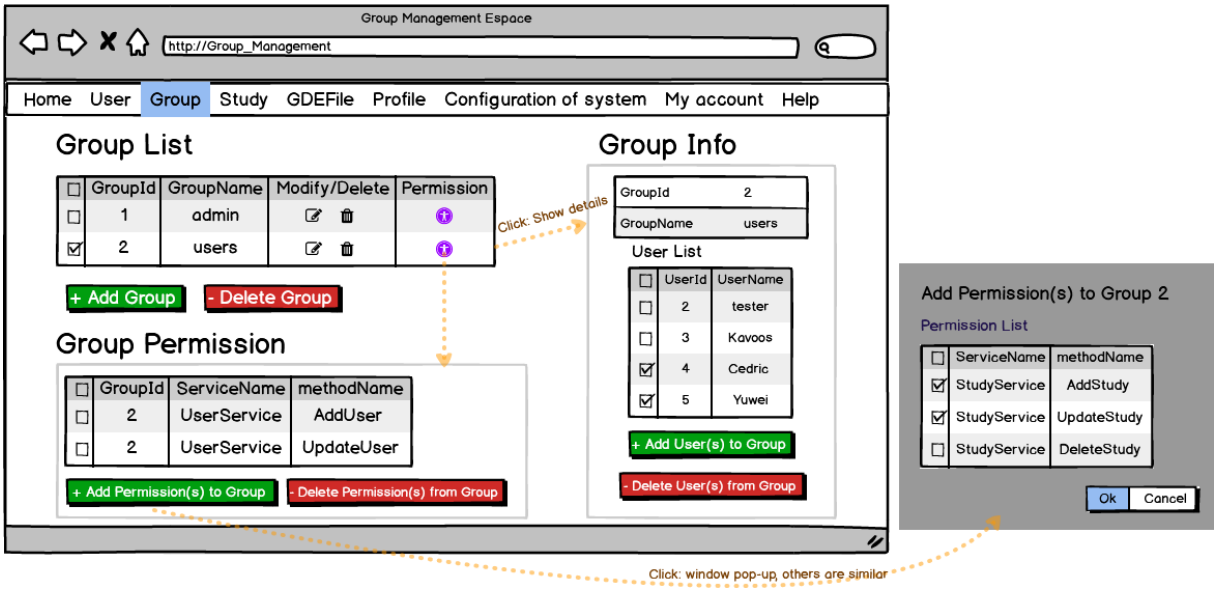


Figure 23 : MAQUETTE D'UN ESPACE GROUP

Le scénario des autres espaces, l'espace de gestion de « *Study* », de « *GDEFile* », de « *Profile* » fonctionne de manière similaire.

4.4.1.2 Conception du modèle de données

Le GDE dispose d'un modèle physique de données qui est projeté sur une base de données relationnelle. Ce modèle de données est adapté aux besoins du GDE. Mais il n'est pas tout à fait adapté aux besoins d'un module d'administration. Pour les besoins du module d'administration, une transformation vers un modèle de données logique plus adapté est

effectuée. Ceci permet de découpler les besoins du module d'administration de ceux du GDE et donc de ne pas alourdir inutilement le modèle physique du GDE.

Le schéma (Figure 24) présente le modèle de données pour le module d'administration. Toutes les propriétés nécessaires proviennent du modèle physique (Figure 7 et Figure 28) et sont réorganisées afin de répondre aux besoins du module d'administration.

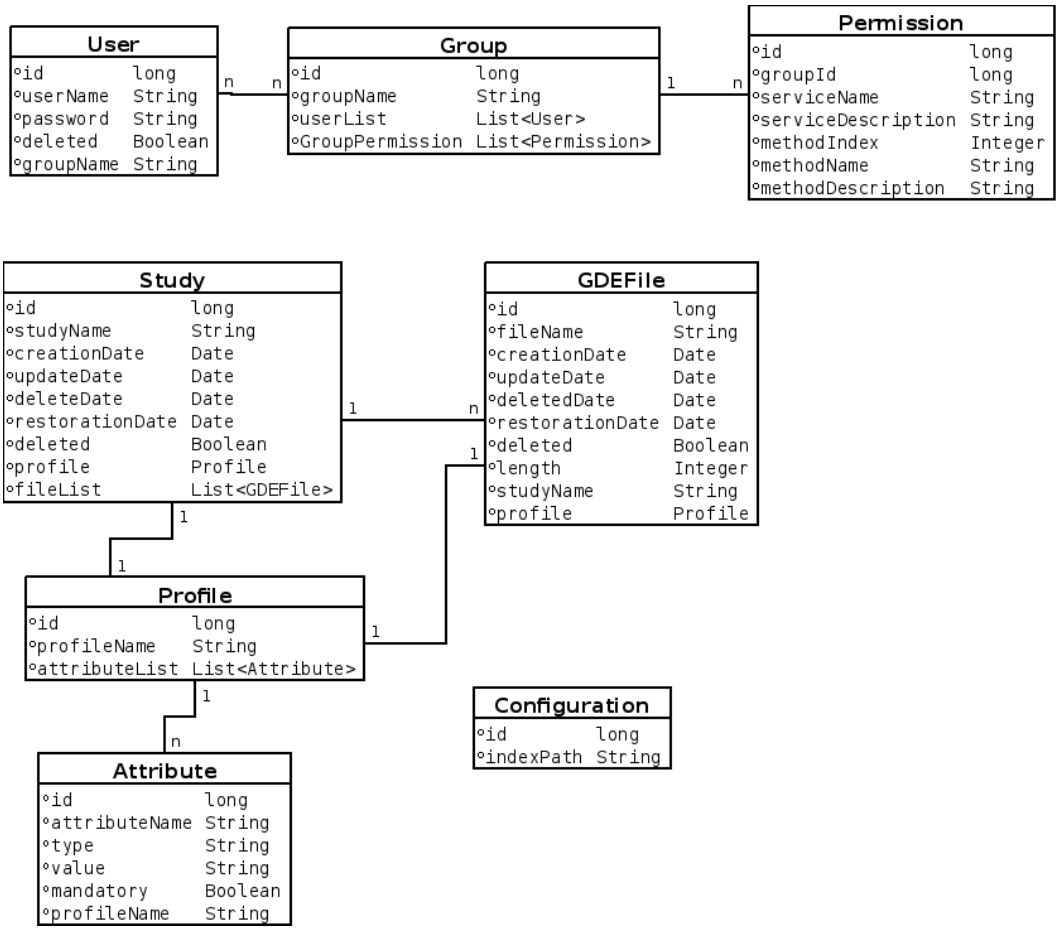


Figure 24 : MODÈLE DE DONNÉES DU MODULE D'ADMINISTRATION WEB

4.4.1.3 Choix et validation les technologies

La troisième étape consiste à choisir les technologies pour réaliser le module. Évidemment, s'agissant d'un client léger, plusieurs technologies de base s'imposent telles que : le langage HTML, les CSS (*Cascading Style Sheets*) et le langage Javascript.

Nous avons fait le choix d'un comportement dynamique et de l'utilisation de communications asynchrones avec le serveur. Pour cela, nous avons adopté la technologie AJAX en utilisant le *Framework* jQuery.

Avant de choisir le *Framework* jQuery, j'ai expérimenté d'autres *Frameworks* tel que JSP, JSF, et l'utilisation des servlets.

4.4.2 Développement du module d'administration

Cette partie présente la conception technique du module. Le module administration est une petite application Web qui utilise plusieurs technologies Web simultanément. Pour que la solution soit facile à maintenir, le rôle de chaque technologie doit être compris, les interfaces doivent être clairement identifiées, les règles de communications doivent être respectées. Pour atteindre ces objectifs, avant de commencer la programmation, il est essentiel de prendre le temps de concevoir l'architecture logicielle.

4.4.2.1 Conception de l'architecture logicielle

Le module d'administration se compose de plusieurs pages web. Chaque page (Figure 25) construite en HTML avec JSP est un ensemble de composants. Il y a trois types de composants : les composants statiques qui ne changent jamais comme le titre de la page, les composants dynamiques qui obtiennent dynamiquement les données du serveur, et les composants d'actions tels que les boutons. Dans un composant dynamique, il peut y avoir un ou plusieurs sous-composants, qui peuvent être n'importe quel type des composants décrit précédemment. Quand on opère sur un composant d'action, par exemple cliquer un bouton, un nouveau composant comme une boîte de dialogue modale apparaît, qui peut posséder des sous-composants également.

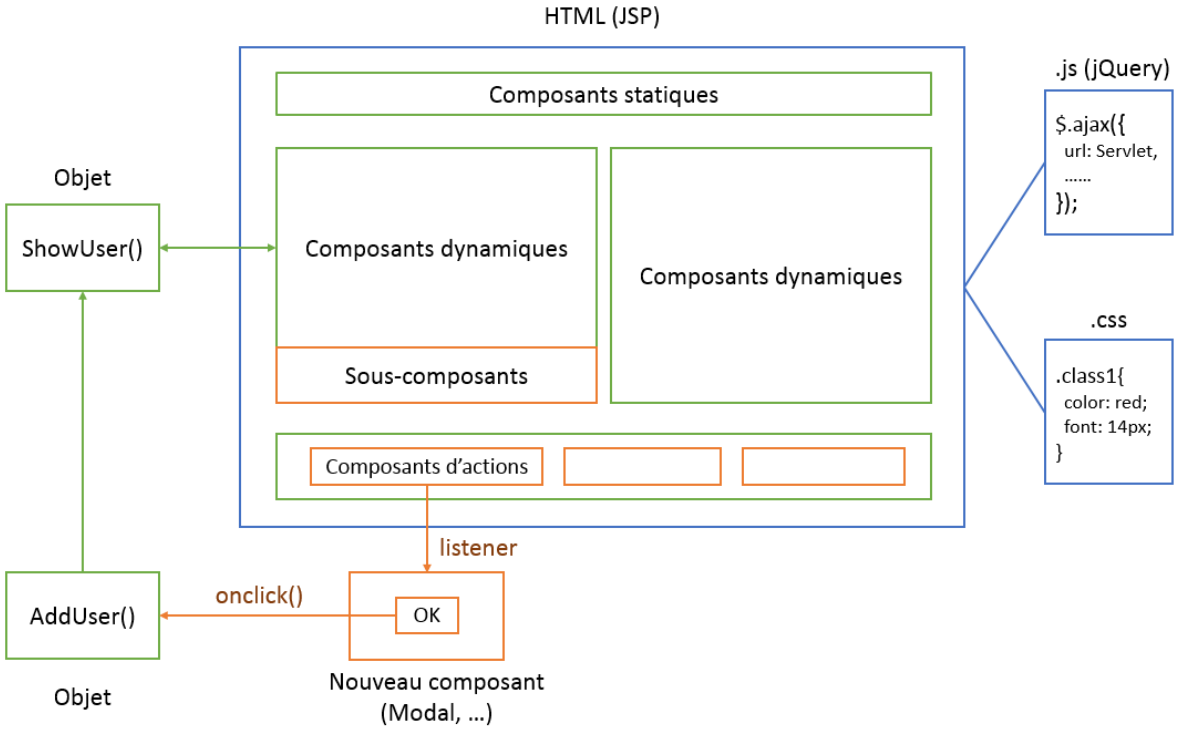


Figure 25 : ARCHITECTURE LOGICELLE DU MODULE D'ADMINISTRATION

Dans les composants dynamiques, les données viennent du serveur. La page envoie une requête à une servlet qui est le contrôleur de la vue, et reçoit en guise de réponse les données demandées ou des messages d'erreur. La communication est basée sur la technologie Ajax, et le format de données transmis est du JSON (*JavaScript Object Notation*).

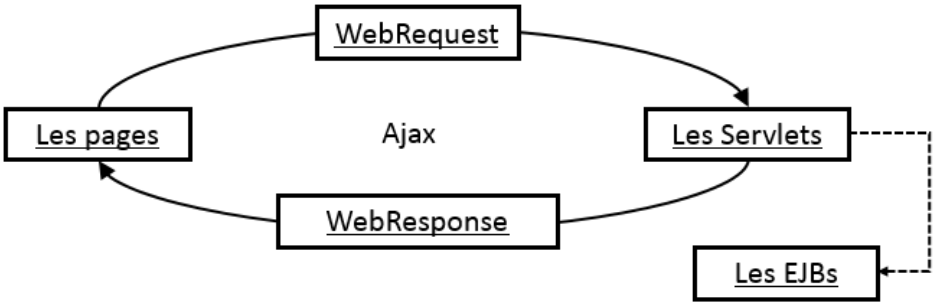


Figure 26 : COMMUNICATION ENTRE LES PAGES ET LES SERVLETS

Par exemple, quand la page a besoin de la liste des utilisateurs, les requêtes Ajax sont effectuées comme décrit dans les figures 26 et 27 pour envoyer un « *WebRequest* » avec la demande et recevoir un « *WebReponse* ».

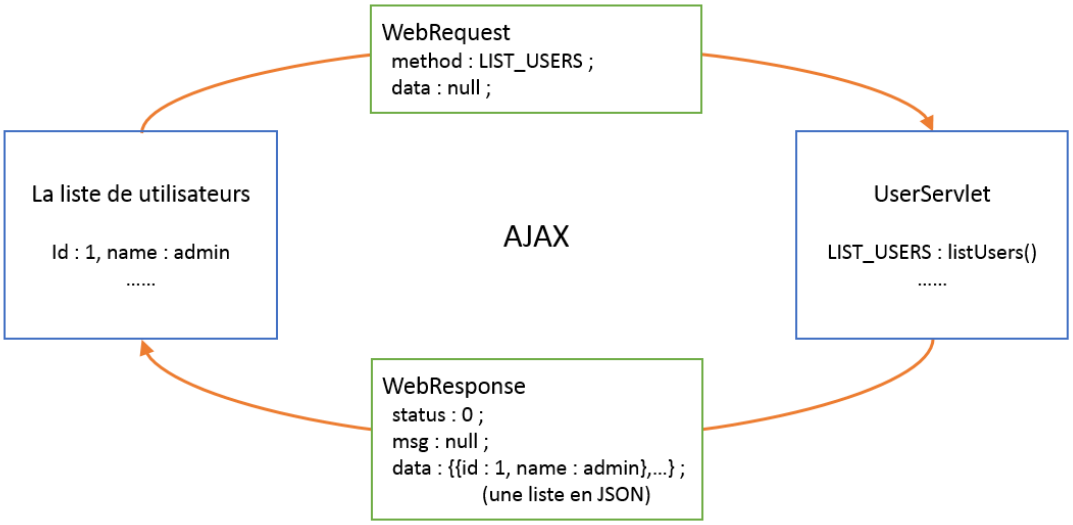


Figure 27 : EXEMPLE D'AFFICHAGE DE LA LISTE DES UTILISATEURS

La couche présentation est pilotée par du JSP. Les interactions avec l'utilisateur sont dynamiquement gérées par l'utilisation de jQuery. La couche contrôleur est formée par un ensemble de servlets qui communiquent avec la couche métier qui est implémentée dans des EJB.

4.4.2.2 Adaptation du modèle de données internes

Le développement de la gestion des permissions dans le module d'administration a mis au jour un défaut dans le modèle de données interne du GDE. En effet, les permissions ne possédaient pas une description humainement compréhensible. Concrètement, dans le module d'administration, j'étais dans l'obligation d'afficher des identifiants de permission sans pouvoir dire qu'il agissait par exemple de la « permission de suppression de fichier ».

Après concertation avec l'architecte du GDE, j'ai décidé avec l'accord de mon tuteur de stage de procéder aux changements nécessaires dans le modèle interne. La figure 28 illustre le

nouveau modèle de données de la gestion des permissions. J'ai pu faire ces changements sans modifier les APIs existantes.

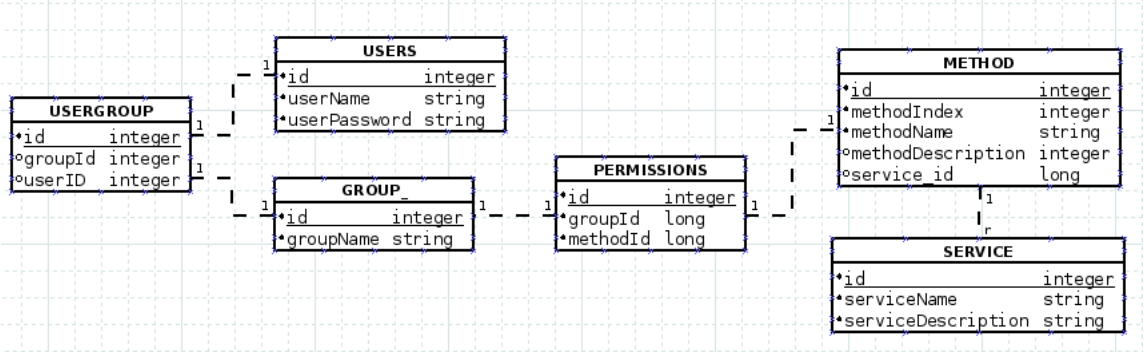


Figure 28 : ADAPTATION DU MODÈLE DE DONNÉES DU GDE, VERSION 2

V. Bilan

Mon stage de fin d'étude au sein d'EDF Lab m'a donné une expérience très enrichissante dans le monde professionnel. Le travail effectué lors de mon stage a permis de faire avancer le projet GDE. En même temps, il m'a beaucoup apporté, tant sur le plan professionnel que sur le plan personnel.

5.1 Apports pour l'entreprise

Grâce à mon travail sur l'intégration du moteur de recherche dans le GDE, le système fonctionne correctement, et les APIs sont disponibles pour que les clients puissent utiliser ces fonctions. En outre, la conception du module administration a permis de démarrer un nouveau chantier important qu'il reste à finaliser.

5.2 Bilan professionnel

D'un point de vue technique, ce stage a été un excellent complément à ma formation d'ingénieur. Il m'a permis d'approfondir les connaissances et les méthodes de travail en informatique surtout sur tous les aspects du système d'information tout au long de mes études dans un vrai contexte d'entreprise avec une pression jamais ressentie auparavant. Les nouvelles compétences acquises sur des technologies comme J2EE, REST, Ajax, Javascript, etc. m'ont permis d'améliorer mes compétences en développement et en conception logicielle. Faire l'état de l'art des outils d'extraction de données m'a permis de comprendre comment effectuer une recherche de l'existant sur un sujet, comment collecter les informations utiles sur Internet et comment faire le meilleur choix pour le projet actuel. Avec l'aide de mes tuteurs, j'ai conçu et intégré de nouveaux modules dans le système GDE. Ceci m'a appris à penser comme une informaticienne et une architecte logicielle.

Concernant mon projet professionnel, ce stage m'a permis de confirmer mes orientations qui sont de travailler dans le domaine de l'architecture du système d'information, car j'ai beaucoup aimé les missions qui m'ont été confiées. Mon intérêt pour le développement et la conception de système d'information s'est trouvé renforcé.

5.3 Bilan personnel

D'un point de vue personnel, en ayant travaillé dans une équipe sympathique et dynamique, ce stage m'a donné une occasion de découvrir la vie au sein d'une grande entreprise. Il m'a permis de développer mes capacités d'apprendre rapidement et effectivement, d'être autonome en m'appuyant à la fois sur les tutoriaux Internet et l'aide ponctuelle des collègues et des tuteurs, d'avoir des responsabilités au travail. De plus, ce stage m'a permis de me familiariser avec le mode de fonctionnement d'une équipe projet de taille humaine, ainsi qu'avec l'organisation et la hiérarchie d'une équipe dédiée à la réalisation de systèmes d'informations.

D'ailleurs, les pauses-café et les pauses déjeuner m'ont énormément apporté car c'était une occasion pour moi de discuter de manière informelle avec mes collègues, et ainsi d'obtenir plus d'information sur les différences culturelles pour mieux m'intégrer en France.

VI. Conclusion

Pour conclure, mon stage de fin d'étude m'est apparu comme une expérience très satisfaisante et enrichissante. Ce stage a été effectué au sein d'EDF Lab dans le groupe I2A du département PERICLES, une équipe accueillante et chaleureuse. Avoir travaillé dans un milieu industriel, m'a procuré l'opportunité de me plonger dans un milieu d'excellents ingénieurs et de rencontrer des professionnels qui ont envie de partager leurs expériences.

Le bilan de ce stage est totalement positif au point de vue technique et personnel. L'objectif initial du stage (chapitre 3.1.3) par rapport à l'intégration du moteur de recherche dans le GDE a été accompli avec succès en à peu près deux mois. Les tâches supplémentaires concernant la conception et le développement du module d'administration se sont bien déroulées. Cela m'a permis d'acquérir et mettre en pratique de nouvelles connaissances sur des technologies telles que J2EE, EJB, Web etc. et la méthodologie de la conception d'un système. Avant la fin de chaque journée, une heure était consacré à l'acquisition de savoirs complémentaires, il pouvait s'agir soit de formation technique, soit de cours de français, soit l'auto-évaluation, soit l'observation et la participation à la vie d'un ingénieur.

Ce stage n'est pas seulement un stage pour moi, je le considère également comme un approfondissement de ma compréhension de la culture française et une vision globale qui me permet de perfectionner encore plus mon français.

La suite du projet GDE s'attachera principalement à mettre en œuvre des projets pilotes dans certains départements d'EDF R&D, ainsi que de réaliser les interfaçages avec les outils métiers en place.

Liste de références

[1] http://www.commentcamarche.net/contents/221-reseaux-architecture-client-serveur-a-3-niveaux : Réseaux - Architecture client/serveur à 3 niveaux

[2] https://lucene.apache.org/core/6_6_0/index.html : site officiel d'Apache Lucene

[3] https://tika.apache.org/ : site officiel d'Apache Tika

[4] https://jquery.com/ : site officiel de jQuery

[5] https://www.w3schools.com/ : site pour apprendre la technologie Web

[6] https://www.tutorialspoint.com/ : site pour apprendre la technologie informatique

[7] https://www.jmdoudoux.fr/java/dej/ : site pour apprendre la technologie Java