Rapport de Projet de Fin d'Etude

Conception et réalisation d'une application mobile (Android)

Session Juillet 2015

Réalisé par :
Mlle EL AHMAR Fatima Zahra.

Encadrants Sofrecom :
M. MESSAOUDI Youssef Badreddin.
M. TIGMA Khalid.

Encadrant High Tech :
M. ACHA Nawfal.

Année universitaire : 2014-2015

Rapport de Projet de Fin d'Etude

Conception et réalisation d'une application mobile (Android)

Session Juillet 2015

Réalisé par :
Mlle EL AHMAR Fatima Zahra.

Encadrants Sofrecom :
M. MESSAOUDI Youssef Badreddin.
M. TIGMA Khalid.

Encadrant High Tech :
M. ACHA Naoufal.

Année universitaire : 2014-2015

Dédicace

A mes chers parents, auxquels je dois tout ce que je suis aujourd'hui,

A mes chères sœurs et mon frère, pour l'amour et leur soutient continuel

A tous les êtres qui me sont chers et à mes chers amis,

Je vous dédie ce travail pour exprimer toute ma gratitude et mon amour.

Fatima Zahra

Remerciements

Je tiens à exprimer mes profonds remerciements à toutes les personnes qui ont contribué, de près ou de loin, à l'élaboration de ce travail.

J'exprime ma gratitude à mes deux maîtres de stage M. Khalid TIGMA et M. Youssef Badreddin MESSAOUDI pour leurs orientations et leurs conseils, ainsi que Mlle Hanae EL MEKKI pour son assistance et ses explications durant les premières phases du projet. De même, je tiens à remercier M. Mohammed Amine TAZI pour ses conseils techniques.

A M. Nawfal ACHA, mon encadrant interne, j'adresse un hommage particulier pour sa disponibilité, ses remarques et ses suggestions.

Pour finir, je tiens à remercier les membres du jury, pour avoir accepté de juger ce travail.

Résumé

Le document que vous avez entre les mains est une synthèse de mon projet de fin d'études, effectué dans le cadre d'un stage de quatre mois au sein de la société Sofrecom Service Maroc.

Le projet consiste à créer un prototype mobile Android pour la gestion –et particulièrement la recherche et la visualisation- des devices. Ma mission était de concevoir et créer cette application, en respectant au mieux les normes et fonctionnement de ces fonctionnalités dans l'application web.

Il fallait dans un premier temps que je me familiarise et que je comprenne comment fonctionne Device-DB avant de pouvoir commencer le travail qui m'a été demandé de faire. Ce rapport détaille une à une chaque étape suivi sur ses trois différents chapitres :

- **Chapitre 1 – Cadre du projet** : Ce chapitre va présenter le contexte général du projet ainsi que l'organisme au sein duquel le projet s'est déroulé. La problématique y sera détaillée et il se terminera par un cahier des charges simplifié du projet.

- **Chapitre 2 – Conduite du projet** : Ce chapitre contient le planning suivit pour la réalisation du projet, ainsi qu'une explication de la méthodologie de développement adoptée. Finalement, il traiter de la conception du projet.

- **Chapitre 3 – Réalisation** : Ce chapitre présente quant à lui des éléments liés à la réalisation du prototype, comme la théorie suivie, l'environnement technologique adopté, les tests des services web et finalement une explication des interfaces.

Abstract

The following document is the summary of my graduation project that took place at “Sofrecom Morocco” for a period of four months.

The main purpose of the project is to conceive and develop a mobile prototype to search and list the various devices existing in the Orange database. My mission was to create this application, taking into consideration the existing functionalities of the web application “Device-DB”.

First of all, I had to understand and familiarize with the existing web application before I started working on the project. This document will go through all the details of each of the development phase the project went through, and it will contain the three following chapters:

- **Chapter 1 – Framework of the Project:** this chapter will give a general presentation to the company, will mention the project’s general context and will explain the main challenges of the project.

- **Chapter 2 – Project Management:** this chapter will explain the schedule that the project is supposed to follow, the agile methodology adopted to make it, and the design of the project.

- **Chapter 3 – Production:** this final chapter will sum up all the elements used to make the prototype, and will explain the web service’s test phase as well as the graphic designing of the views of the application.

Liste des figures

Liste des tableaux

Liste des abréviations

ADT	Android Development Tools
ANRT	Agence Nationale de Règlementation des Télécommunications
CPU	Central Processing Unit
DB	Data Base
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
IHM	Interface Homme/Machine
JavaEE	Java Enterprise Edition
JDN	JournalDuNet
JSON	JavaScript Object Notation
MSE	Médiation et Systèmes Embarqués
OS	Operating System
PHP	PHP: Hypertext Preprocessor
R&D	Recherche et Développement
RAD	Rapid Application Development
RAM	Random Access Memory
REST	Representational State Transfer
SDK	Software Development Kit
SGBDR	Système de Gestion de Base de Données Relationnelle
SGBDRO	Système de Gestion de Base de Données Relationnelle Objet
SIG	Système d'Information Géographique
SOAP	Simple Object Access Protocol
SSM	Sofrecom Service Maroc
TMA	Tierce Maintenance Applicative
UP	Unité de Production
URI	Uniform Resource Identifier
XML	eXtensible Markup Language
XP	eXtreme Programming

Table des matières

Introduction

Dans un monde où la technologie évolue sans cesse, l'obtention immédiate de l'information est devenue un des éléments les plus indispensables à notre vie quotidienne, aussi bien sur le plan personnel que sur le plan professionnel. Durant un court laps de temps, la recherche de l'information se fait de plus en plus par l'intermédiaire de téléphones ou de pads, plus qu'elle ne se fait via des ordinateurs. Actuellement, les renseignements que nous souhaitons avoir se doivent d'être accessibles du bout des doigts à tous les moments et en tous lieux.

Dans cette optique d'évolution constante vers des plateformes mobiles, j'ai été mené durant ce stage à réaliser un prototype d'application Android, qui devrait reprendre certaines fonctionnalités de la version web existante, pour les rendre portables et accessibles constamment.

L'existant.

Device-DB est une des diverses parties de l'application web Zoltar, successeur de l'outil DGCT du groupe Orange, lancé en 2007.

Zoltar est un outil qui référencie et regroupe toutes les informations concernant les devices[1] commercialisés par Orange. Il est utilisé par plus de 500 clients dans 23 pays du monde, et contient les informations les plus précises sur plus de 2 000 devices différents.

Device-DB web sert plus précisément à recenser, consulter et modifier ces devices, ainsi qu'à ajouter des commentaires de bugs qui pourraient être exploités par la suite.

[1] Dans notre contexte, un device est tout équipement télécom (téléphone, fax, radio ou autre) commercialisé par le groupe Orange.

L'avenir.

Dans l'optique de réaliser un premier prototype mobile de Device-DB, seules quelques fonctionnalités parmi les fonctionnalités de bases seraient reproduites dans un premier temps.

Device-DB Mobile est une application mobile destinée à rendre les fonctionnalités de recherche et visualisation des devices *–deux fonctionnalités disponibles dans l'application web d'origine-* plus portables et accessibles via des terminaux mobiles Android.

Si approuvé, le prototype donnerait naissance à une application à part entière, complète, et réalisée pour convenir à plusieurs plateformes mobiles (iOS, Windows Phone etc…), cependant, avant d'en arriver là, le choix s'est porté sur le système Android, étant l'OS mobile le plus populaire du monde.

Chapitre 1 :

Cadre général du projet

Introduction : Ce premier chapitre décrit le cadre général du projet. Dans un premier temps, je vais présenter l'organisme d'accueil qui est Sofrecom Service Maroc, avant de procéder à la présentation du cadre dont lequel s'inscrit le projet.

I - Présentation de l'organisme d'accueil

1 - Le groupe Sofrecom

Raison sociale	: Ingénierie, étude technique
Forme juridique	: Autre SA à conseil d'administration
Groupe	: France Telecom Orange
Date de création	: 01 Janvier 1967
Siège social	: 24, Avenue du petit parc 94300 Vincennes France.
Capital social	: 5 000 000 Euro
Téléphone	: +33 143 98 55 55
Télécopie	: +33 143 98 57 96
Site Web	: www.sofrecom.com

Le groupe Sofrecom est une filiale de France Telecom – Orange. Il est spécialisé dans le conseil des technologies de l'information, et plus particulièrement dans le domaine des télécommunications. Il est composé d'une équipe internationale d'experts et de consultants de haut niveau spécialistes en télécommunications. C'est un leader dans le domaine du Conseil, de l'Ingénierie et des Système d'information qui intervient à l'international depuis plus de 30ans. Ses prestations s'adressent à tout type d'opérateur (fixe, mobile, internet,…) et s'appuient sur des compétences pluridisciplinaires et multiculturelles. Il a fait preuve, au fil des projets et à travers le monde, d'un savoir-faire unique dans divers domaines des télécommunications.

Le chiffre d'affaire de Sofrecom consolidé, ayant dépassé pour la première fois de son histoire, la barre symbolique des 100 M€ avec un résultat net social supérieur à 1.5 M€ et une présence accrue à l'international (+55% du CA).

Le groupe Sofrecom est certifié ISO 9001 version 2008 et CMMI niveau 2, et il est en train de préparer le niveau 3 de la certification CMMI.

- **Sofrecom créateur de solutions opérateurs**

Sofrecom apporte une gamme complète de services et de solutions dédiées aux opérateurs à savoir :

- Conseil & Réseaux.
- Système d'Information.
- Intégration & Développement.

- **Ingénierie logicielle**

Sofrecom apporte maîtrise et compétitivité aux projets d'externalisation et de développements de logiciels et de tierce maintenance applicative.

Le front office de Sofrecom s'appuie sur ses « software factories » développées dans des filiales : Maroc, Argentine et Pologne, avec des ressources et des infrastructures dédiées.

Cette politique permet une adaptation rapide aux nouvelles technologies et maîtrise des coûts.

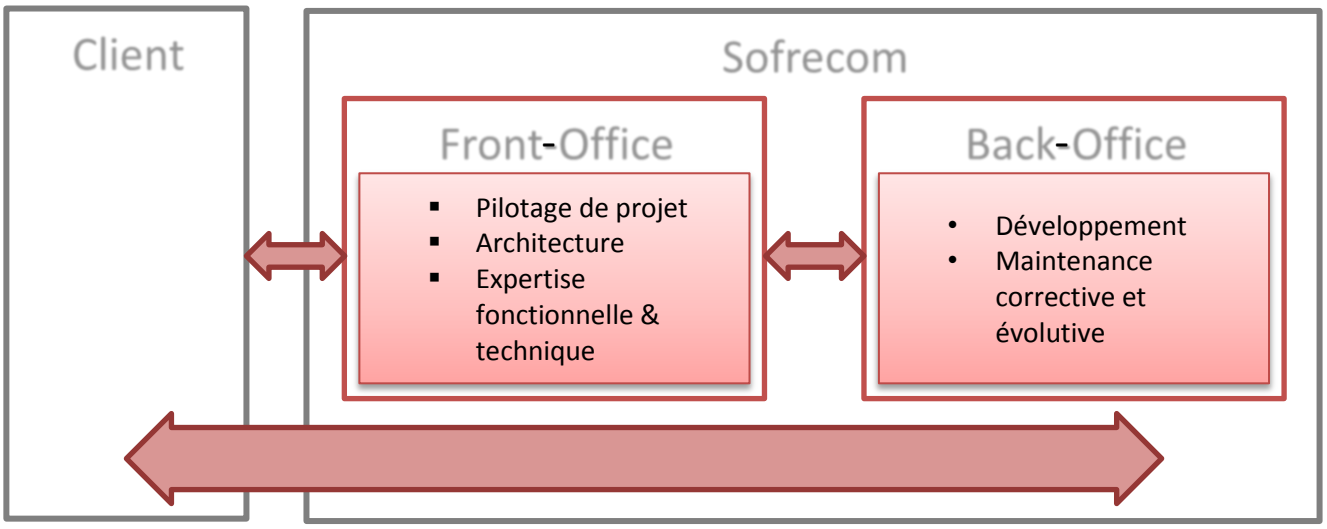

Figure 1: Processus de production Sofrecom

L'ingénierie logicielle répond aux besoins suivants :

- Externalisation de développements logiciels en local, « nearshore » ou « offshore ».
- Intégration et portage de logiciel, de système et d'architecture technique.

➢ Implantation Sofrecom

Sofrecom est représentée par huit filiales. Ces filiales sont implantées en Algérie, Argentine, Indonésie, Maroc, Pologne, Jordanie, Dubaï et Vietnam.

Figure 2 : Implantation de Sofrecom

2 - Sofrecom Service Maroc

Sofrecom développe sa présence au Maroc avec l'aide de sa filiale Sofrecom Services Maroc – SSM créée en 2004, centre de services et d'ingénierie logicielle.

Le haut niveau d'expertise de ses collaborateurs marocains francophones et la proximité avec l'Europe, permettent à ces deux filiales d'offrir flexibilité et réactivité aux demandes de leurs clients : bases de données, systèmes d'information décisionnels ou déploiement de solutions applicatives (TMA).

Principaux clients : ANRT, Maroc Telecom, France Télécom-Orange.

3 - Organisation de Sofrecom Service Maroc

Sofrecom Services Maroc est composée de trois directions, une direction ressources humaines, commerciale, financière et une direction technique organisée selon plusieurs unités de production (UP) dont : UP Web source, UP Réseaux & SIG, UP MSE, UP Portail Self-Services et R&D, UP Gaia & Technologies Oracle (voir organigramme).

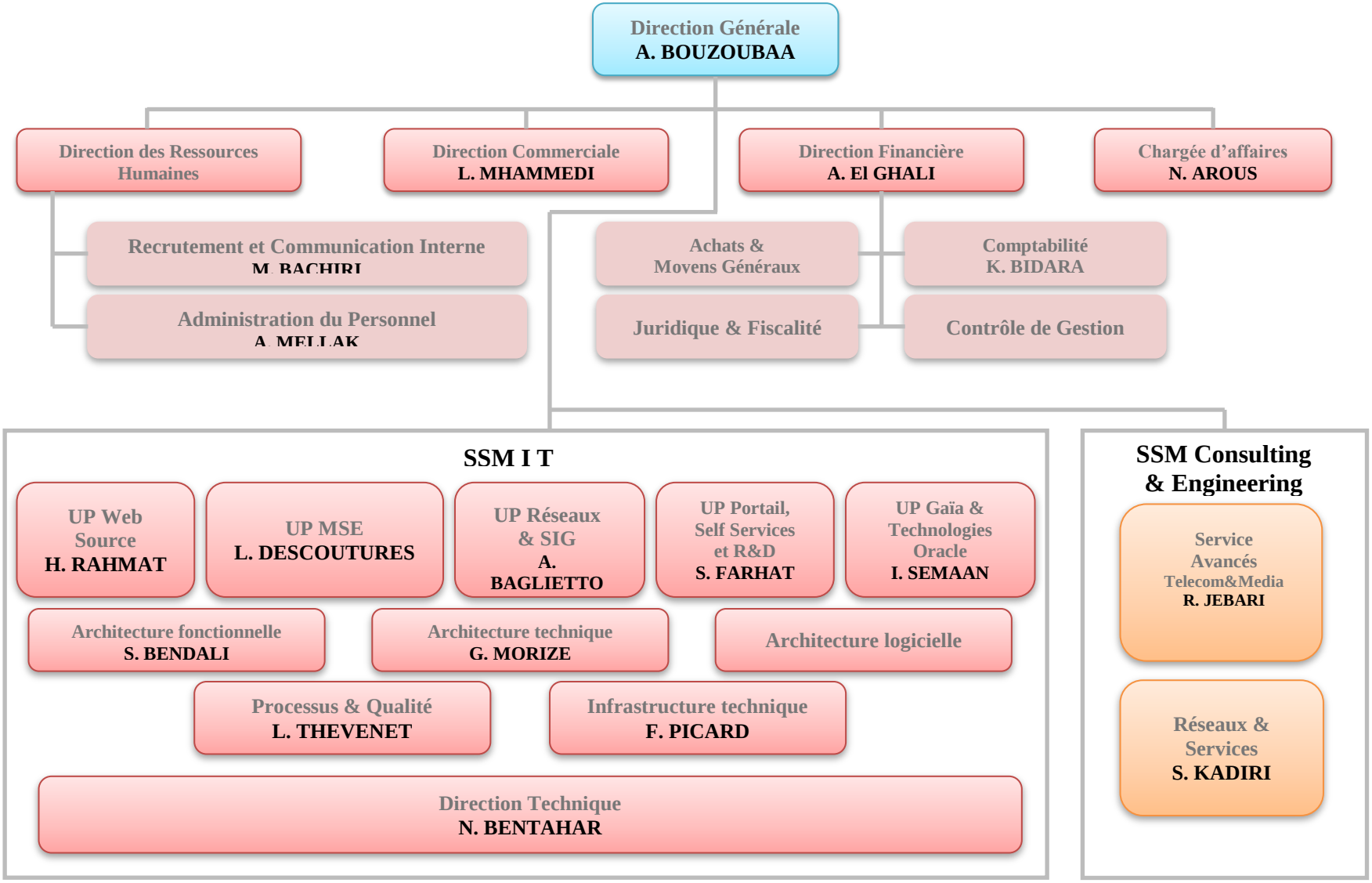

Figure 3 : Organigramme 2013-2014

II – Contexte général du projet

1 – La problématique

En prenant en considération le but et l'utilisation de l'application de base (à quoi sert Device-DB ?) ainsi que les limites qu'une application mobile peut avoir aussi bien sur un plan technique et fonctionnel, que d'un plan de facilité d'utilisation (lequel est plus évident pour un utilisateur d'application mobile : rechercher et consulter ? ou remplir et insérer ?), la première vraie question à laquelle il faudrait trouver une réponse : **quelles sont les premières fonctionnalités à mettre en place pour Device-DB Mobile ?**

Destinée à être utilisée par un nombre précis d'utilisateurs, l'application future devrait au premier abord assurer son rôle qui est de rendre l'accès aux informations concernant les devices facile, rapide, et effectif, et ce à travers les fonctionnalités de recherche et de visualisation, qui seront détaillées par la suite.

A partir de là, il est naturel de se demander **quelle plateforme choisir pour le développement de l'application ?**

Le choix s'est porté sur Android. Ce système d'exploitation est utilisé comme système pour des appareils mobiles de marques variées. De plus, l'application visera la clientèle venant d'Afrique, et selon le site AfriqueItNews, Android détient la partie majeure du marché, en plus d'avoir 85% du marché mobile à l'échelle internationale.

Les appareils mobiles Android sont accessibles et beaucoup moins coûteux que leurs contemporains d'Apple ou Windows, ce qui a motivé le choix de cette plateforme. L'application pourrait éventuellement être adaptée sur d'autres plateformes mobiles dans le futur.

Nous pourrions également nous poser une question sur la nature de l'application à concevoir et créer. **Devrait-elle être native ou hybride ?**

Avant de donner une réponse, je rappellerai brièvement la définition des deux. Une application mobile native est une application développée précisément pour un des systèmes d'exploitation mobile. Son langage de

développement est le langage spécifique du système d'exploitation en question. Une application mobile hybride est une application qui combine des éléments d'une application native et propres à un langage spécifique, en plus des éléments avec HTML5 sous forme de web application mobile.

Pour DeviceDB mobile, le choix s'est porté pour le natif. La refonte de l'application web existante pour la rendre hybride et lui permettre de s'afficher correctement sur les appareils mobiles peut-être une démarche coûteuse, et longue. De plus, une application hybride est limitée et ne permet pas l'utilisation de toutes les fonctions natives de l'appareil.

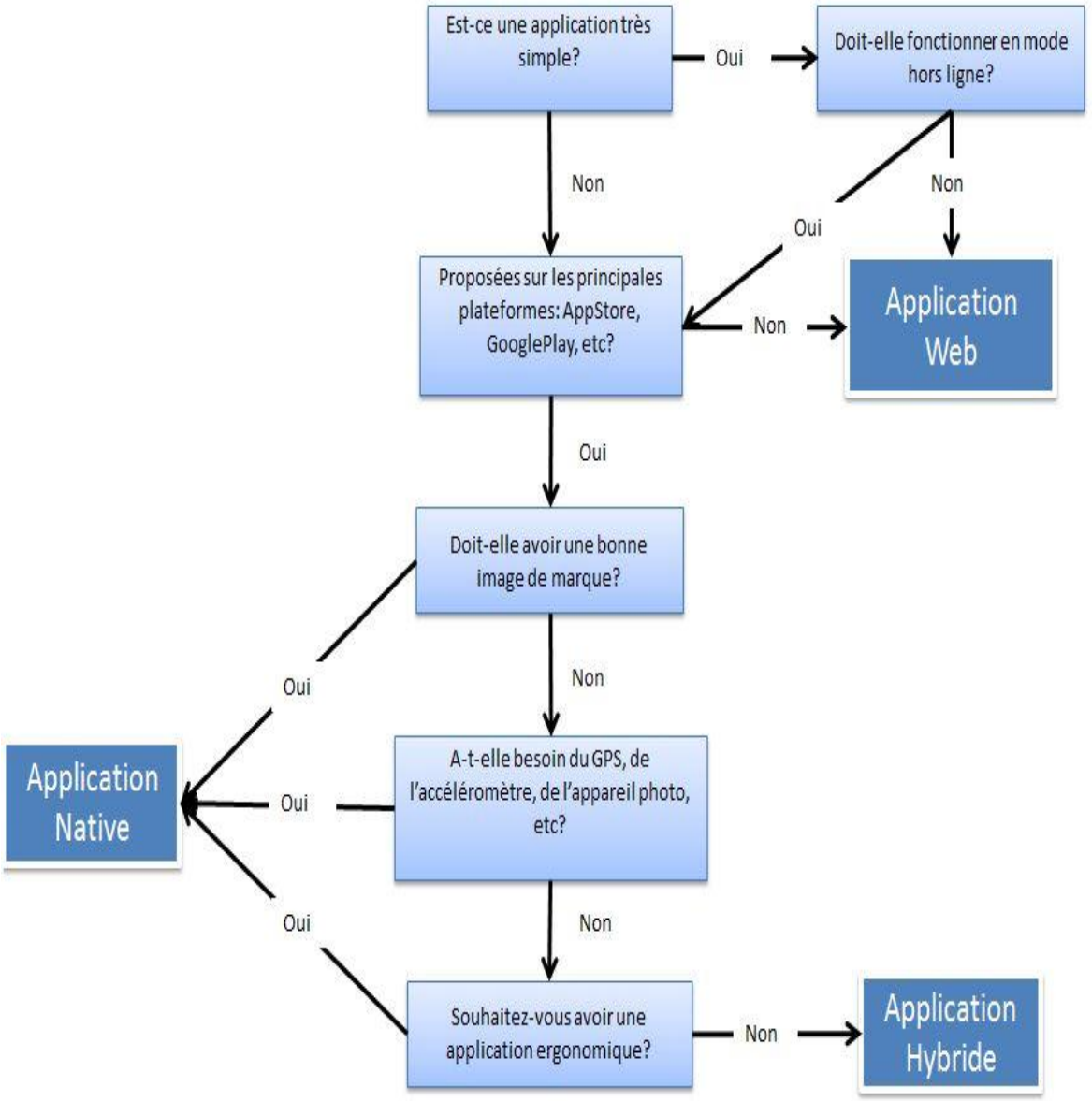

Figure 4 : Application native ou hybride ? (source : AppSolute)

Passons maintenant à une question de nature plus technique. Etant donné la nature de l'application et son langage de programmation (une application mobile, programmée en Android, donc.), la manipulation d'une base de données externe, comme il est le cas ici, peut poser des difficultés : **quelle est la démarche la plus convenable à suivre pour les surmonter ?**

La base de données associée à Android est une base de données interne, SQLite. Pour permettre la communication entre la plateforme Android et une source de données externe, il faut utiliser une troisième couche qui servira d'intermédiaire entre les deux parties.

2 – Cahier des charges simplifié

2.1 – Description du projet

Objectif :

L'objectif principal de l'application mobile serait de faciliter la portabilité, la recherche, et la visualisation des devices disponibles, fonctionnalités qui sont aujourd'hui uniquement disponibles via l'application web interne Device-DB.

Pour des raisons de sécurité mais également d'utilité, le premier prototype de Device-DB mobile portera uniquement sur les fonctionnalités de recherche et de visualisation.

Les besoins fonctionnels :

L'application devrait principalement permettre la recherche et la consultation des devices.

Les besoins non-fonctionnels :

L'application devra être présentable. La qualité de l'ergonomie sera un facteur essentiel pour l'application.

Les fonctionnalités :

Authentification.

Etant donné la confidentialité des éléments à visualiser, l'utilisateur devrait dans un premier temps s'authentifier avant de pouvoir accéder à l'application mobile. Une première vue où il peut insérer ses identifiants apparaît et ce n'est qu'après vérification de ces informations que l'utilisateur peut parvenir aux autres vues. L'authentification représente donc un premier pas nécessaire de sécurité.

Recherche :

Sur cette vue, l'utilisateur peut effectuer une recherche des devices présents sur la base de données. Il a la possibilité de le faire en se basant sur les critères de recherche suivants :

- Recherche par supplier, ou manufacturer : où l'utilisateur effectue sa recherche en se basant sur le fournisseur et/ou constructeur du device. (Exemples : Apple, Samsung, etc…). Cette liste est sous forme d'une liste déroulante.

- Recherche par catalogue : le catalogue représente l'année et le trimestre exact où le device est disponible en vente. (Exemple : si un device est dans le Q4 de l'année 2014, ça voudrait dire qu'il a été rendu disponible durant le 4ème trimestre de 2014.). Le critère catalogue a la forme d'une liste déroulante contenant les combinaisons Q (quarters[2]) et Années.

- Recherche par Name : l'utilisateur peut rechercher un device par son nom complet, ou un mot clé appartenant à son nom, qu'il peut saisir dans un champ texte.

[2] Quarter : quart d'année/trimestre, en anglais.

Visualisation :

Une fois la recherche effectuée et le résultat trouvé, l'utilisateur pourrait sélectionner le device en question et ainsi accéder à ses diverses informations. La vue résultante sera sous formes d'onglets où les diverses caractéristiques de chaque matériel seront listés, selon des catégories (General, Design, Features, etc…)

2.2 – Ressources technologiques

L'application Android sera développée en native avec le SDK[3] Android, et Android Studio[4].

SDK Android :

Le SDK Android représente l'outillage indispensable au développeur Android. Ce kit contient tous les outils nécessaires pour programmer avec Android, exécuter les programmes et les tester.

Android Studio :

Android Studio représente la plateforme officielle, soutenue par Google, pour le développement d'applications Android. Il repose sur IntelliJ[5] et devrait permettre aux développeurs d'être plus rapides et plus productifs. Avant la sortie de la première version stable d'Android Studio, Google recommandait aux développeurs Android d'utiliser l'infrastructure Eclipse, combinée au plugin ADT[6], une infrastructure qui reste une référence connue en matière de développement Java.

[3] SDK : Software Development Kit en anglais, Kit de Développement logiciel en français.
[4] Android Studio : Android Studio est un environnement de développement pour développer des applications Android. Il est le premier de son genre car Google avait jusque-là proposé ses outils de développement pour Android sous la forme d'extensions pour l'environnement Eclipse, notamment l'ADT.
[5]IntelliJ IDEA, IDE (environnement de développement intégré) de JetBrains.
[6] ADT : Android Development Tools. Plugin Eclipse permettant le développement d'application sous Android.

2.3 – Les contraintes

Contraintes de délais :

Le projet devrait respecter le planning établit et qui se déroulera sur 95 jours au total.

Contraintes budgétaires :

Le projet qui représente un premier prototype ne nécessite pas de ressources financières. Il ne nécessite pas de déplacement ou de matériel particulier puisqu'il se concentre sur le développement d'application mobile Android en utilisant l'outil offert par Google qui est Android Studio.

Contraintes associées au projet :

Le développement de cette application apportera également son lot de contraintes, associées à l'application elle-même :

- L'application Android devrait être lisible et compatible avec le maximum des versions de l'OS possibles. Depuis février 2014, Google juge obsolète toute version inférieure à Jelly Bean 4.2.

- L'application mobile devrait respecter au maximum les mesures de sécurité assurées dans la version web.

Conclusion : Après avoir présenté le contexte général du projet, ainsi que l'organisme d'accueil au sein duquel s'est déroulé ce stage dans ce premier chapitre, je procéderai à détailler la conduite du projet dans le chapitre suivant.

Chapitre 2 :

Conduite du projet

Introduction : La première partie de ce chapitre va porter sur la planification et la méthode suivie pour mener à bien le travail, tandis que sa seconde partie se focalisera sur la conception.

I – Méthodologie de développement

1 – Méthode agile RAD

« Si vous achetez une maison sur plans, sans jamais visiter le chantier, vous risquez d'avoir des surprises au moment d'y emménager, n'est-ce pas ? C'est pourtant une approche similaire qui guide la gestion traditionnelle de projets de développement logiciel. », extrait de l'article *Pourquoi choisir la gestion de projet Agile ?* du site LesAffaires.Com. Cette question résume, sans détailler, les raisons qui poussent chaque projet à adopter une méthode agile pour son développement.

Les méthodes de développement traditionnelles se basent sur un enchaînement des activités et des phases, depuis la phase spécification et jusqu'à la mise en place du système, selon un planning précis et qui sert plus à prédire la manière dont les choses devraient se passer. Cette vision est hélas loin de la réalité, car les activités d'un projet informatique ne peuvent, presque jamais, se succéder d'une manière stricte, sans qu'aucun changement ne vienne déranger ce planning préétablit.

Pour la réalisation de ce projet, j'ai adopté la méthode de développement rapide d'applications, notée RAD[7], qui se base sur un cycle de développement dont le délai idéal et de 90 jours minimum, et 120 jours maximum, et composé de cinq phases : initialisation, cadrage, design, construction et finalisation, dont trois principales et qui sont cadrage, design et construction. Son cycle de vie se schématise comme suit :

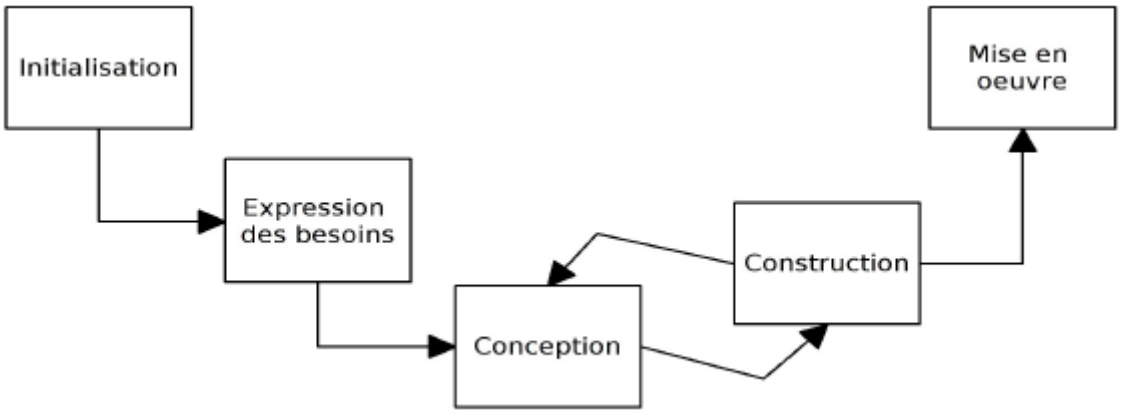

Figure 5 : Cycle de vie RAD

[7] RAD : Rapid Application Development.

Avant de décrire les phases principales de ce projet, il faudrait d'abord faire un bref rappel sur les « ancêtres » des méthodes agiles, et ainsi montrer pourquoi elles ne sont plus au goût du jour. Je vais choisir la méthode cascade, ou waterfall, comme principal exemple. Jusqu'à la fin des années 80, la méthode cascade était la méthode la plus employée par l'industrie du logiciel. La mise en œuvre d'une nouvelle phase dépendait fortement de la complétude de sa précédente, chose qui pouvait engendrer plusieurs difficultés pour le développement informatique.

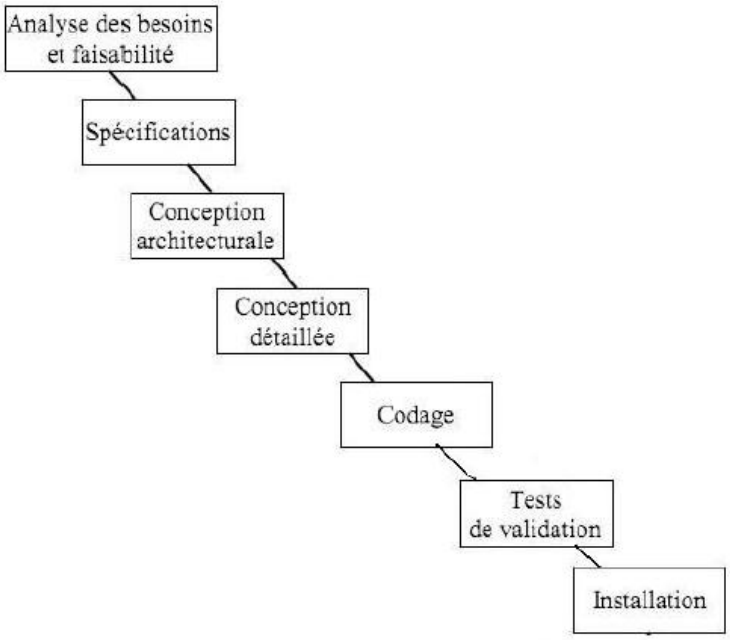

Figure 6 : Cycle de vie en cascade

En informatique, recueillir l'ensemble des spécifications du premier coup n'est pas toujours facile. Décrire exactement le système attendu depuis le départ n'est pas toujours évident. Avec la méthode classique en cascade, le temps de réalisation du système est souvent long et difficilement estimable à l'avance.

Une réponse apportée à ce problème a été proposée avec la méthode RAD. Définie par James Martin dans son live « Rapid Application Development », elle est avant tout pragmatique. Elle regroupe un ensemble de bonnes pratiques et impose un cycle de développement court qui ne dépasse pas les 120 jours.

Contrairement aux méthodes Scrum ou XP qui sont principalement basées sur un travail en binôme ou en équipe de plusieurs personnes, cette méthode peut permettre le travail individuel. Etant donné la taille du projet, j'ai été menée à le réaliser en monôme.

2 – Les phases du projet

Phase 1 – Initialisation.

La première partie de ce projet consistait à définir le contexte de développement, et répondre aux questions concernant la technologie à adopter. Elle représente environ 5 % du projet.

Phase 2 – Cadrage.

La phase de cadrage consistait à définir le paramètre et les fonctions principales que ce premier prototype devait reprendre, en se basant sur une étude de la documentation existante de l'application web initiale et déjà existante. Prenant en considération le besoin principale de l'application mobile qui est, et je le rappelle, de rendre les informations de Device-DB disponibles à être consultés tout moment, les premières fonctionnalités ont été limitées à les fonctions de recherches et de visualisations. Durant cette phase, j'ai pu rédiger un cahier des charges simplifié question de cerner le mieux possible ce qui était demandé de faire. Elle représente environ 10 % du projet.

Phase 3 – Design.

Une fois les vues nécessaires précisées, la phase design pouvait commencer. Elle représente environ 25 % du projet, et j'ai utilisé l'outil Balsamiq et sa version d'essai pour la réalisation des mockups d'interfaces.

Phase 4 – Construction.

Cette phase est constituée de deux sous-phases qui s'alternaient au fur et à mesure : la conception, et le développement. Cette phase représente environ 50% du projet.

Phase 5 – Finalisation.

Cette phase représente environ 10% du projet. C'est l'étape finale et consiste à déployer l'application.

En prenant en considération toutes ces données, le cycle de vie de ce projet pourrait être schématisé de la manière suivante :

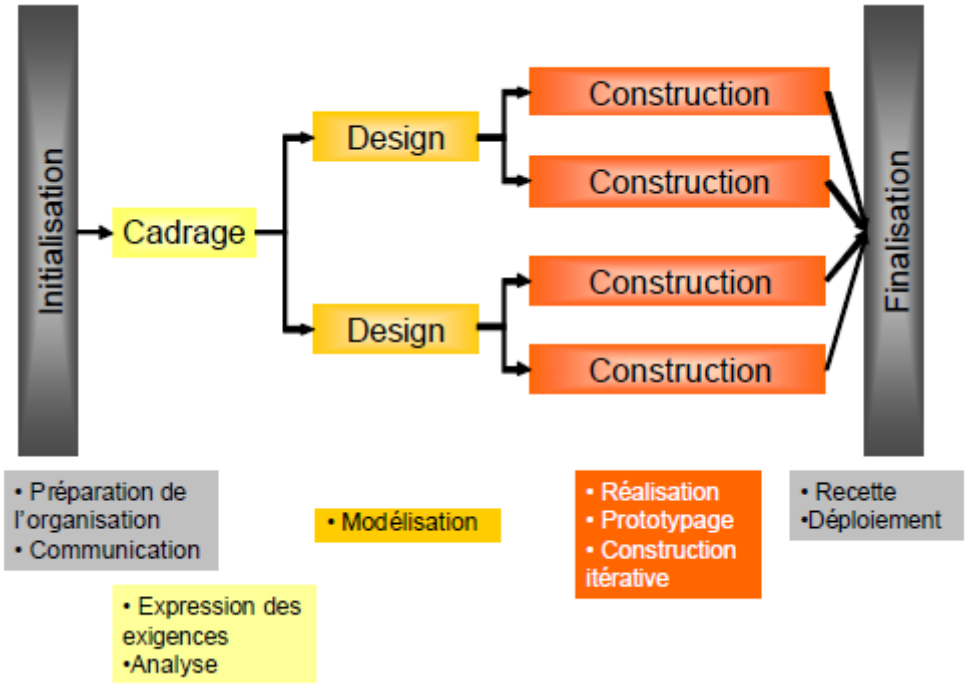

Figure 7 : Cycle de vie RAD détaillé

3 – La planification du projet

La planification du projet est parmi les phases avant-projet, elle consiste à ordonnancer les actions et le chemin de leur déroulement tout au long des phases constituantes du projet.

3.1 – Plan prévisionnel

Le plan prévisionnel que j'ai proposé a été réalisé à l'aide de l'outil Gantt Project, et il est comme suit :

Nom	Durée	Date de début	Date de fin
Phase 1 - Initialisation	5	09/03/15	13/03/15
Étude du contexte générale	5	09/03/15	13/03/15
Phase 2 - Cadrage	16	16/03/15	06/04/15
Documentation initiale	7	16/03/15	24/03/15
Rédaction du CDC	15	16/03/15	03/04/15
Validation du CDC	1	06/04/15	06/04/15
Phase 3 - Design	23	07/04/15	07/05/15
Documentation sur les outils de maquettage	2	07/04/15	08/04/15
Création des mockups des vues/interfaces	20	09/04/15	06/05/15
Validation des mockups	1	07/05/15	07/05/15
Phase 4 - Construction	44	08/05/15	08/07/15
Conception	44	08/05/15	08/07/15
Développement des services REST	14	08/05/15	27/05/15
Test des services REST	2	28/05/15	29/05/15
Développement mobile	25	01/06/15	03/07/15
Test de l'application mobile	3	06/07/15	08/07/15
Phase 5 - Finalisation	7	09/07/15	17/07/15
Déploiement	7	09/07/15	17/07/15

Figure 8 : Les tâches

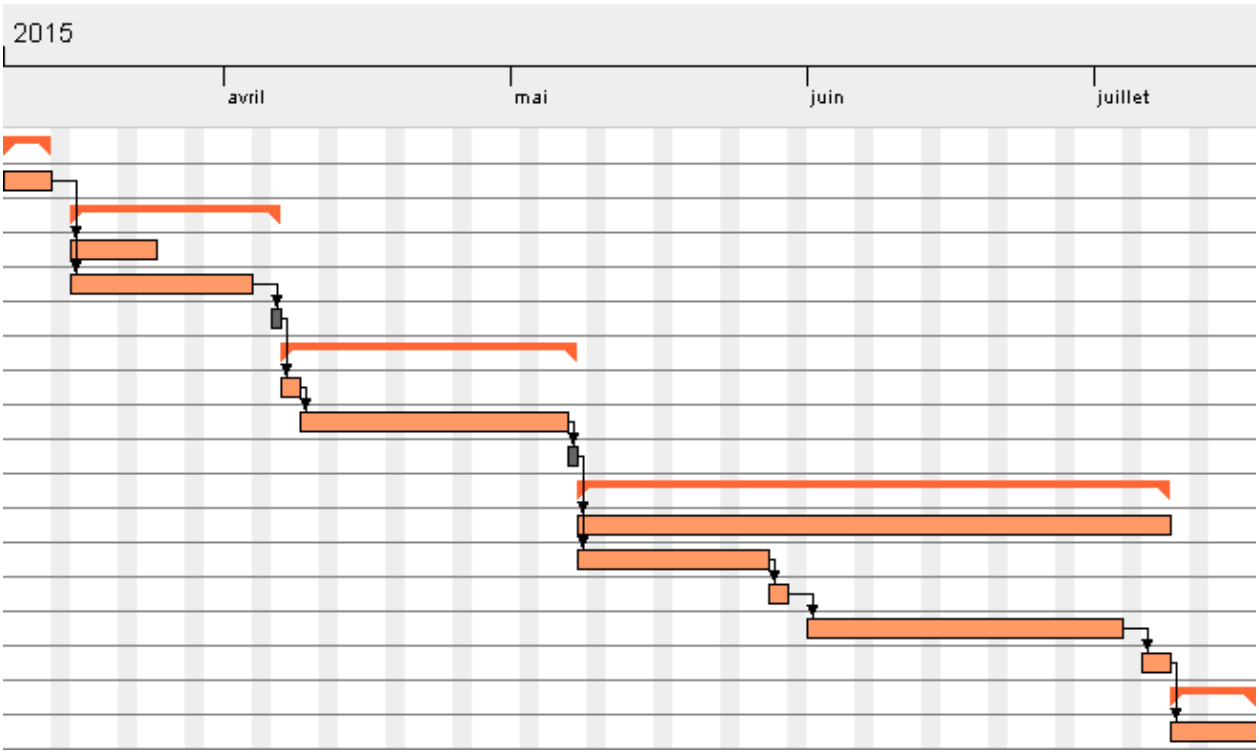

Figure 9 : Diagramme de GANTT

3.2 – Plan réel

Cependant, le plan prévisionnel a subi des modifications tout au long du déroulement du projet, surtout durant son avant dernière phase qui est la phase « Construction ». D'une part, j'ai été confrontée à des blocages techniques, dû au fait que plusieurs des méthodes que j'ai utilisé pour mon code sont devenues obsolètes très récemment. Je me suis donc retrouvée avec un code non-fonctionnel qui devait être retravaillé pour coller aux nouvelles spécifications d'Android. D'une autre, étant donné mon niveau débutant en cette technologie, j'ai dû me documenter encore plus sur les nouvelles méthodes proposées dans la documentation officielle.

Le rapport du plan réel se trouve dans l'annexe.

II – Conception du projet

Ce prototype aura comme source la base de données déjà établie pour l'application web existante, et gérée par ses administrateurs. Vu la taille de cette structure et sa confidentialité, le diagramme de classe représentera que les tables nécessaires pour la réalisation de ce premier prototype. Les diagrammes ont été réalisés avec l'outil PowerAMC.

Certains des diagrammes seront des diagrammes globaux et représenteront, non seulement le prototype, mais également une application future en prenant en considération les perspectives et les ajouts potentiels qui pourront être inclus par la suite, tandis que d'autres seront plus spécifiques et détailleront principalement les fonctionnalités de Device-DB Mobile dans sa première version.

1 – L'acteur

Pour le prototype mobile, l'acteur principal sera le client (utilisateur) qui aura les droits d'accès et de lecture sur les informations contenus dans la base de données.

2 – Diagramme de packages

Le diagramme de packages montre la décomposition du système en deux principale catégorie : le package de la gestion des devices, et c'est la catégorie

sur laquelle va porter la suite de cette conception, ainsi que le package de la gestion des bugs.

La gestion des bugs regroupe plusieurs fonctionnalités futures qui consistent principalement à signaler et rapporter les bugs techniques et défaillants constatés sur les différents devices.

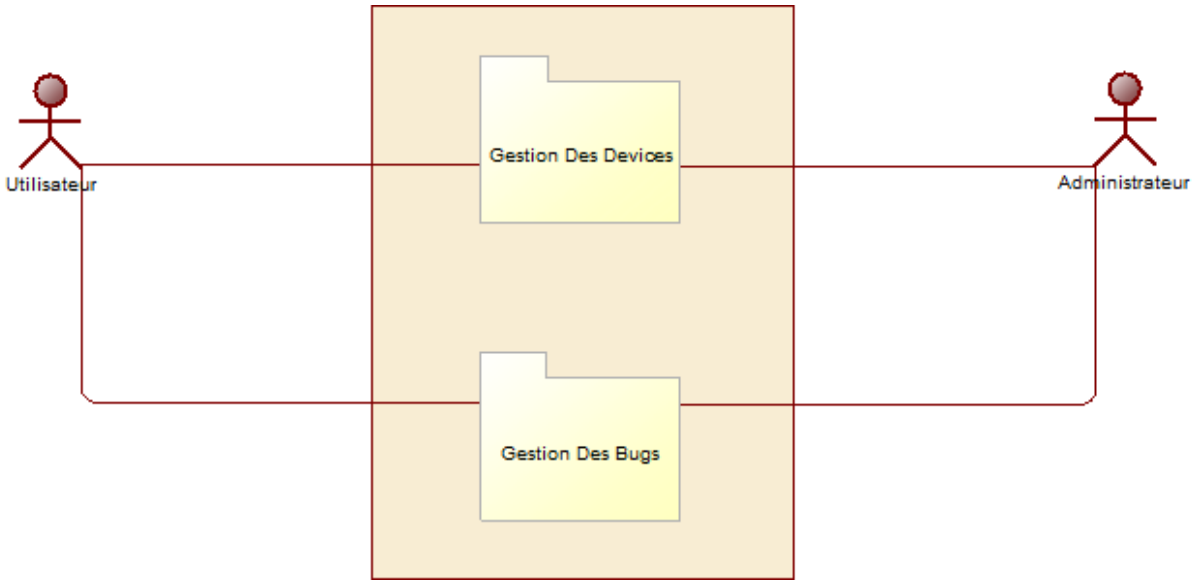

Figure 10 : Diagramme de packages

Je vais maintenant me focaliser sur le package sur lequel se base la partie réalisation qui est le package gestion des devices.

3 – Diagrammes de cas d'utilisation

Le diagramme de cas d'utilisation représente les fonctionnalités nécessaires aux utilisateurs.

La partie réalisation portera sur les cas « Recherche » et « Visualisation ». Concernant le cas « Création », vu les objectifs primaires du prototype, son contenu pourrait être ajouté aux perspectives.

Gestion des devices :

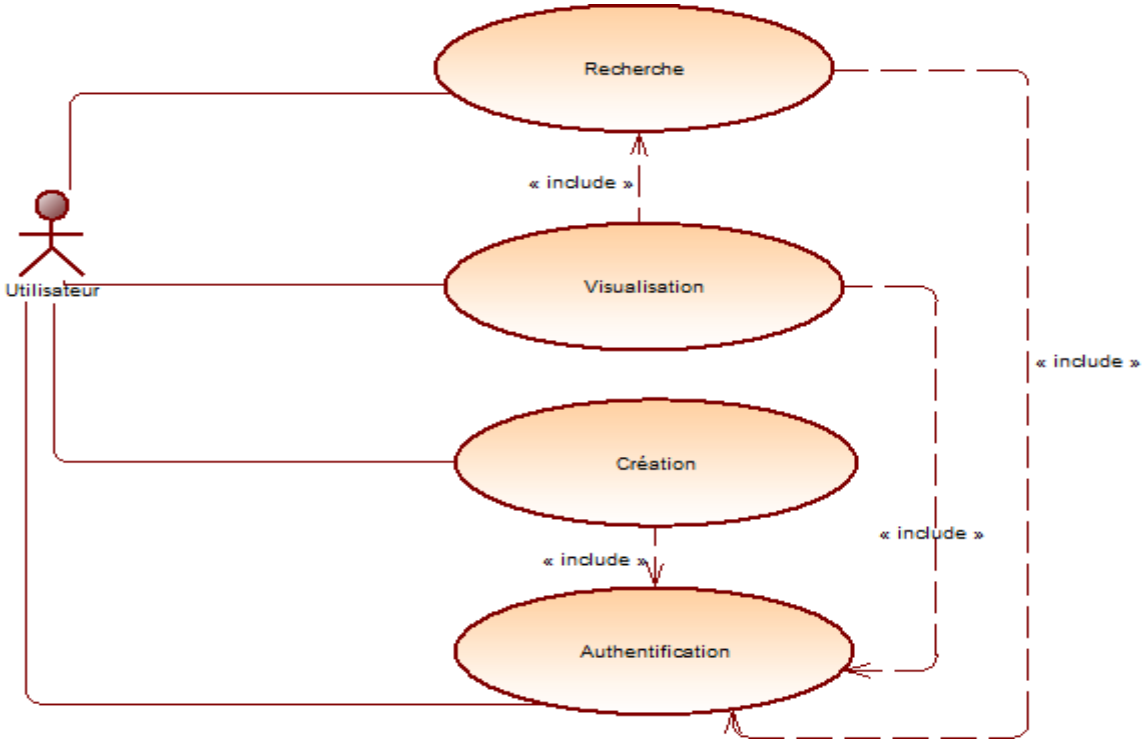

Figure 11 : Diagramme de cas d'utilisation – Gestion des devices

Création des devices :

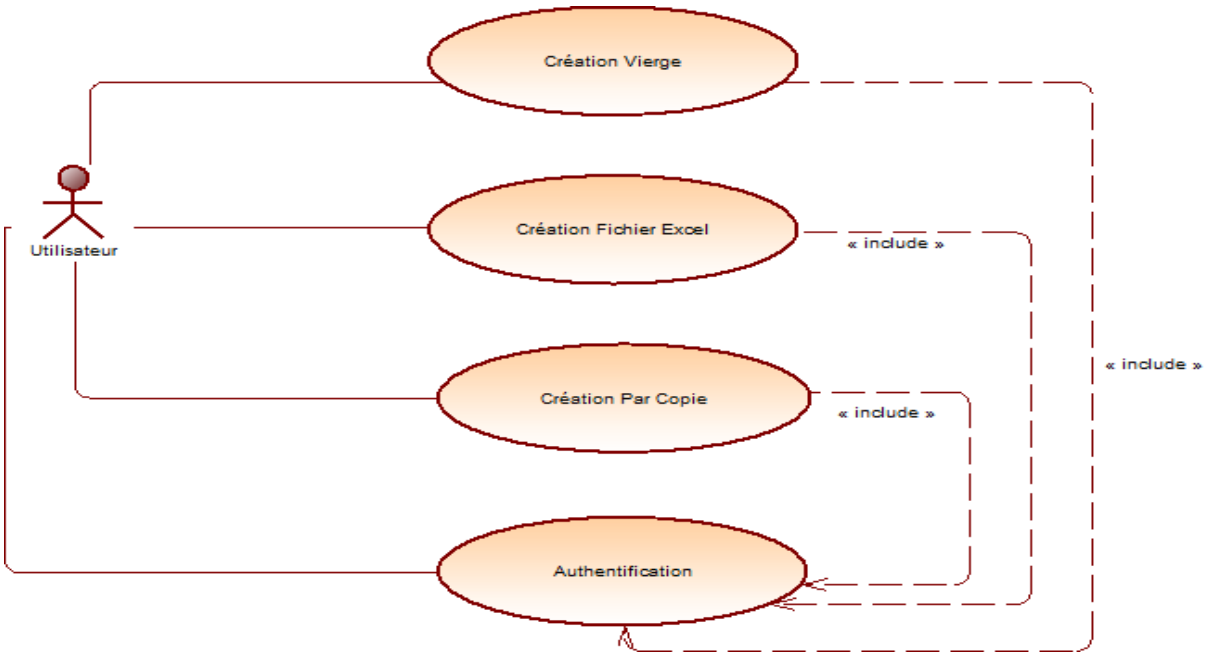

Figure 12 : Diagramme de cas d'utilisation – Création des devices

4 – Diagrammes de séquences

Ce diagramme permet de décrire les différents scénarios d'utilisation des cas d'utilisation « Recherche » et « Visualisation ».

Recherche :

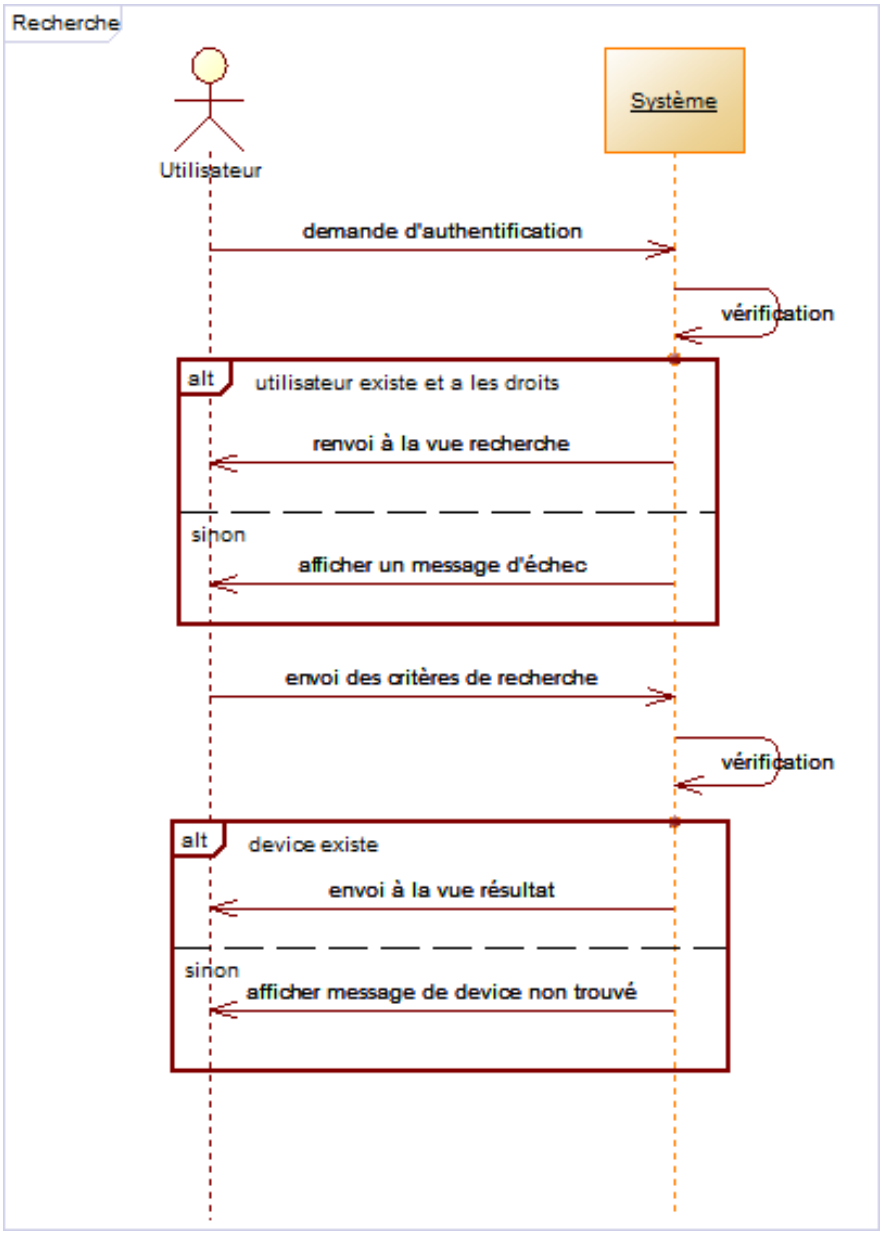

Figure 13 : Diagramme de séquence – Recherche

Visualisation :

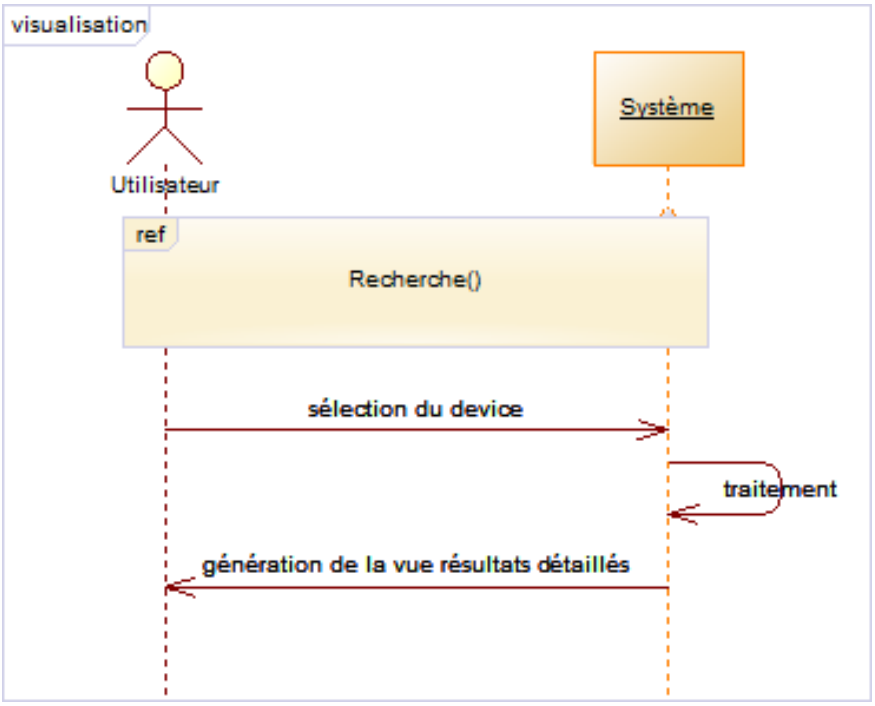

Figure 14 : Diagramme de séquence – Visualisation

5 – Diagrammes de classes

Ce diagramme représente les entités (des informations) manipulées par les utilisateurs.

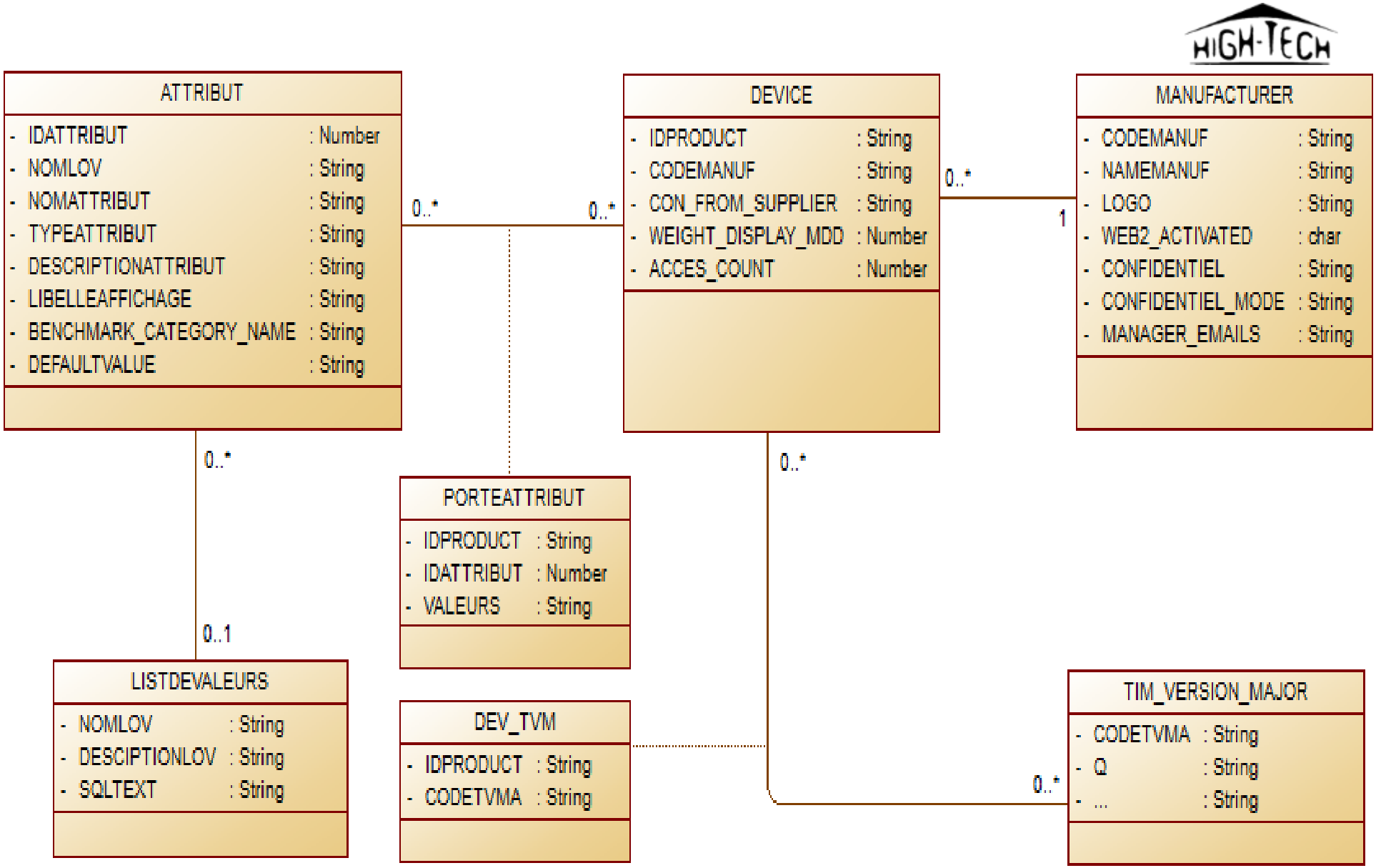

Figure 15 : Diagramme de classes

6 – Schéma logiciel

L'application dans son intégralité (c'est-à-dire le premier prototype ainsi que toutes les fonctionnalités qui peuvent lui être ajoutées) peut être schématisée de la manière suivante :

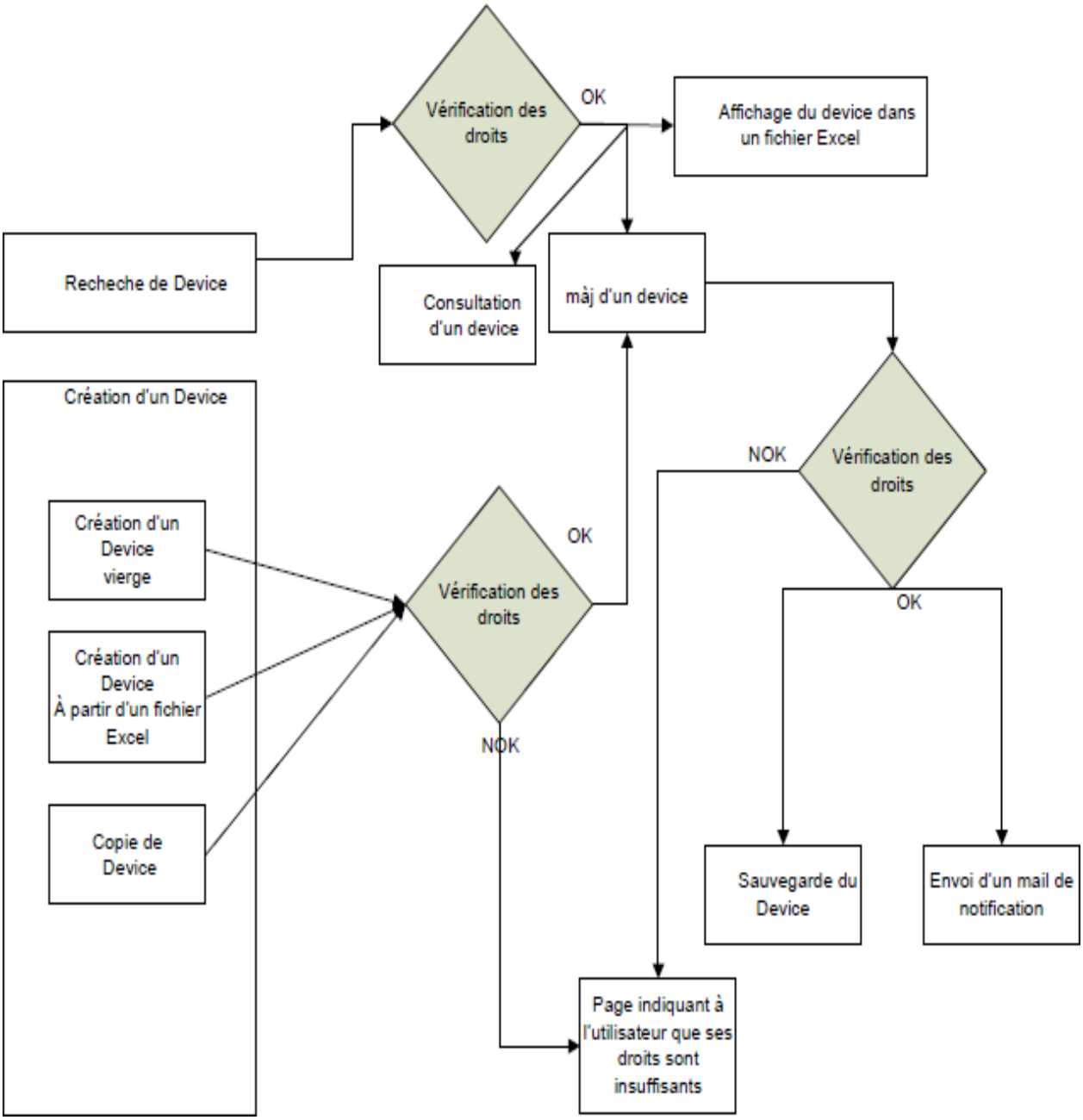

Figure 16 : Schéma logiciel global

Ce rapport porte donc sur cette partie :

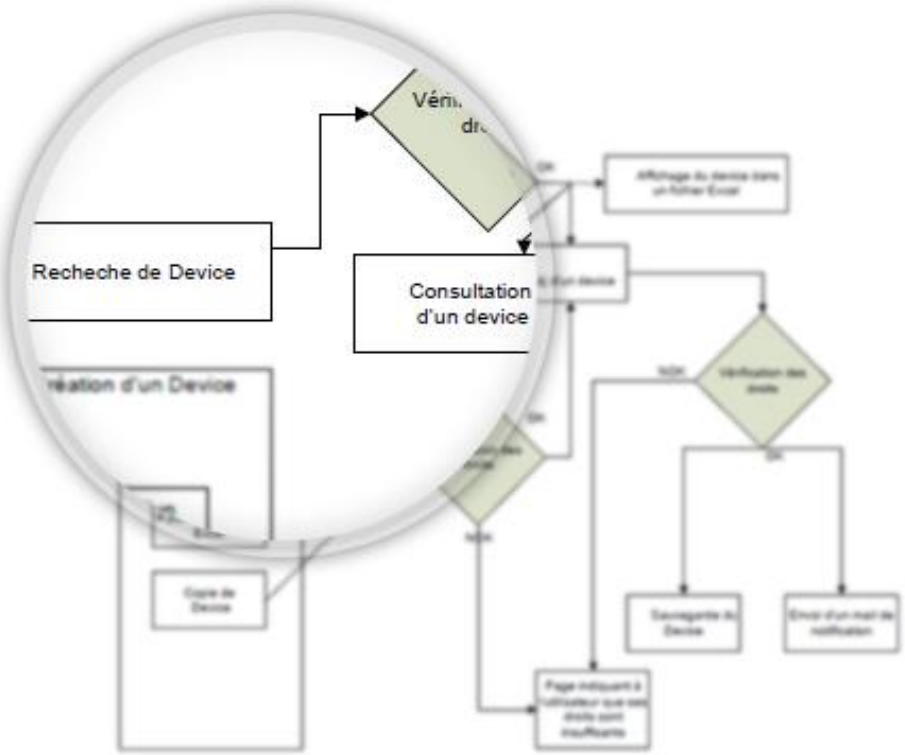

Figure 17 : Schéma logiciel spécifique

Conclusion : Dans ce chapitre, j'ai présenté la planification initiale suivie pour réaliser le projet ainsi que sa conception en présentant ses diagrammes package, cas d'utilisation, séquence. J'ai conclu en présentant un schéma logiciel global d'une future application plus complète et détaillée.

Chapitre 3 :

Réalisation

Introduction : Je vais présenter dans la première partie l'approche théorique que a été adoptée pour la réalisation du projet, ensuite je parlerai brièvement de l'architecture 3-tiers avant de passer aux tests effectués sur les web services, et finalement présenter les différentes IHM de l'application.

I – Approche théorique

Comme l'application web d'origine, Device-DB Mobile devrait communiquer avec une base de données Oracle qui contient les diverses informations qui seraient récupérées et utilisées.

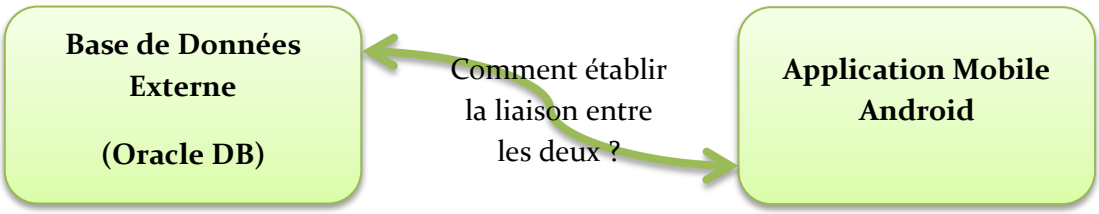

Figure 18 : Comment établir la liaison ?

La base de données associée habituellement à la technologie Android est SQLite. Permettant le stockage interne, cette dernière a toutefois un espace de stockage limité, et étant donné le volume des données que l'application est censée supporter, le mieux serait d'utiliser la base de données Oracle externe.

L'avantage de cette approche est que l'espace de stockage des données, avec une base de données externe, peut être immense, contrairement à une base de données stockée directement sur le Smartphone. Cependant, le temps de réponse et retour des requêtes avec une base de données externe peut être largement supérieur au temps de réponse d'une base de données interne.

Cette différence majeure vient du fait qu'une application Android ne peut pas, et ne devrait pas, communiquer directement avec la base de données externe. Pour faire communiquer les deux entités (application et base de données) il faut avoir recours à une couche intermédiaire, utilisant souvent un web service, qui reçoit les requêtes et sert faire la liaison entre les deux couches pour permettre leur interaction.

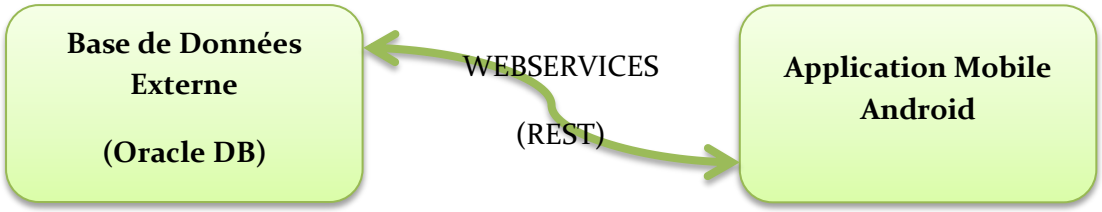

Figure 19 : Lien établit via un web service

Ce service web en question devrait donc organiser, adapter et traiter les échanges de données entre l'application et la base de données externe qui seraient maintenant possibles. Il peut être développé avec des technologies comme le PHP, le .NET, le JavaEE, ou autre. Le choix du langage utilisé pour la création du service web dépend de plusieurs facteurs notamment le choix de la plateforme (Microsoft .NET, Apache Axis, ...)

Les services web, en général :

Globalement, un service web est une méthode de communication entre deux applications ou appareils électroniques, via le Web. Il existe deux types de services web: Simple Object Access Protocol (SOAP) et Representational State Transfer (REST).

SOAP est une spécification d'un ensemble de règles, standards, pour permettre l'échange de messages dans un format XML. REST décrit un ensemble de principes architecturaux par lesquels les données sont transmises sur une interface standard (telle que HTTP).

Pour arriver à choisir le type de service Web qui convient le plus, entre SOAP et REST, il faudrait prendre en considération plusieurs facteurs en se basant sur les fonctions que permet chaque type de service Web. Si les documentations officielles des deux types ont tendance à valoriser et survendre le mérite de chacun des deux, des sources plus neutres comme le JDN[8] ou LeMagIT s'occupent de dresser une liste de fonctions beaucoup plus objective et qui m'a permis de faire un choix : un service web type REST.

[8] JDN : Le JournalDuNet.

Pour citer quelques-unes des raisons ayant motivé ce choix :

- Les services REST proposent une bonne infrastructure de GET via la méthode HTTP GET et cela peut améliorer les performances si les données retournées par le service Web ne sont pas régulièrement modifiées et ne sont pas de nature dynamique.

- REST est particulièrement utile pour les appareils aux profils restreints, comme les mobiles, pour qui la surcharge de paramètres comme les headers ou d'autres éléments SOAP, est compliquée.
- L'implémentation basée sur REST est simple comparée à SOAP.

L'utilisation d'un service web REST est la méthode la plus efficace et la plus utilisée pour établir un lien entre la base de données et l'application mobile, étant donné que le langage Android ne communique pas directement avec une base de données autre qu'une base SQLite.

REST peut être définit comme étant un style d'architecture qui repose sur le protocole HTTP : On accède à une ressource (par son URI unique) pour procéder à diverses opérations (GET lecture / POST écriture / PUT modification / DELETE suppression), opérations supportées nativement par HTTP.

II – Architecture 3-tiers

L'architecture d'une application mobile utilisant une base de données distante pourrait se schématiser comme suivant :

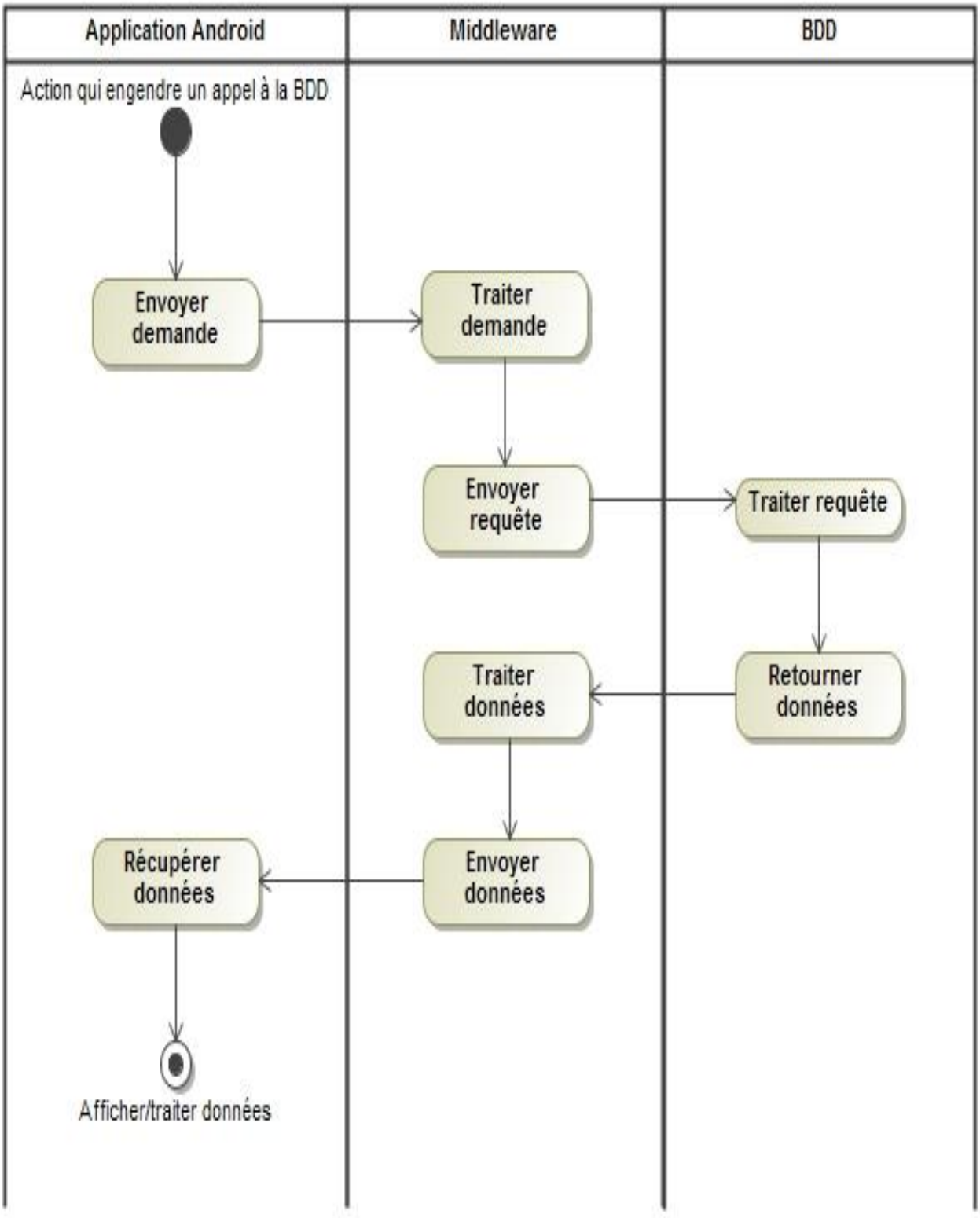

Figure 20 : Architecture 3-tiers, simplifiée (Source : tutorielandroid.)

L'application Android représente la couche client qui envoie des requêtes à une couche intermédiaire (Middleware). Par le bais d'un ou plusieurs web service REST, la communication entre la couche Client et la couche données contenant la base de données peut être établie.

Une architecture 3-tiers est une architecture client-serveur où les couches accès aux données, interface utilisateur et traitement des données sont séparées et maintenue sur des plateformes distantes. C'est une architecture 3 niveaux qui permet à chacune des couches d'être changée ou remplacée de manière indépendante, sans toucher aux autres couches.

Les trois tiers d'une architecture 3-tiers sont :

- « Présentation » : ou la couche client, demandeuse de ressources. Elle est l'interface utilisateur.
- « Application » : ou middleware. C'est la couche « logique », qui contient les fonctionnalités et chargée de fournir la ressource en faisant appel à un autre serveur.
- « Données » : la couche contenant les bases de données et toutes les datas.

Figure 21 : Architecture 3-tiers Mobile

III – Environnement technologique

Cette partie servira à présenter les outils utilisés pour la réalisation de ce projet. Certains d'entre eux ont été imposés par l'entreprise d'accueil tandis que d'autres ont été libre de choix. Les langages utilisés pour la réalisation du service web et de l'application sont Java et Android, respectivement.

Oracle Database :

Oracle Database est un système de gestion de base de données relationnel (SGBDR) qui depuis l'introduction du support du modèle objet dans sa version 8 peut être aussi qualifié de système de gestion de base de données relationnel-objet (SGBDRO). Pour les premiers tests, c'est la version Express 11g qui a été utilisée.

Android Studio :

Depuis le 8 Décembre 2014, Android Studio est devenu l'IDE officiel du développement Android, proposé par Google. Entre 2011 et 2014, Eclipse était utilisé à la place, combiné à un plugin et contenant le SDK d'Android.

Bien que certains utilisent toujours Eclipse et ADT, le choix, pour ce projet, c'est posé sur Android Studio car depuis la sortie de sa première version stable en Avril 2015, Google a entièrement délaissé Eclipse qui ne verra plus de mise à jour ou de réalisation de nouvelles versions à partir de maintenant. De plus, Android Studio est basé sur l'IDE IntelliJ IDEA qui est, selon les professionnels du domaine, un environnement de développement beaucoup plus complet, beaucoup plus performant, qu'Eclipse.

Les différences entre Android Studio et Eclipse sont majeures, et ont toutes motivé ce choix.

	Android Studio	Eclipse
Build	Android Studio utilise le système build Gradle. C'est un système qui se base sur les concepts d'Apache Ant et d'Apache Maven.	Eclipse utilise Apache Ant comme son système build principal.
UID[9]	Le design des interfaces sous Android Studio se fait plus rapidement et propose plus d'options sans avoir à passer en mode XML.	Certains éléments graphiques ne peuvent être insérer que via le mode XML, et le temps de réponse au changement de l'interface est plus lent.

Tableau 1 : Android Studio vs. Eclipse

Concernant la performance et la stabilité des deux, Android Studio bien que plus performant qu'Eclipse, il nécessite une charge considérable quand il s'agit de la RAM et le CPU de la machine utilisée pour le développement. C'est d'ailleurs l'un des premiers soucis qui ont été rencontré durant la phase développement.

Concernant l'approche choisie pour la création du service web, le choix s'est posé sur l'approche REST vu qu'elle est plus convenable que SOAP.

REST - Representation State Transfer :

L'approche REST est plus simple à comprendre et à mettre en œuvre que SOAP[10]. D'après Mike Rozlog[11], « les développeurs qui utilisent cette approche cite la facilité de développement, l'utilisation de l'infrastructure Web existante, et le faible coût de formation comme des avantages clés de ce style. ». Au-delà de ces avantages, le choix s'est porté sur REST car il est communément utilisé pour les applications web. De plus, et contrairement à SOAP, il permet de retourner les

[9] UID : User Interface Design.
[10] SOAP : Simple Object Access Protocol.
[11] Mike Rozlog est un conseiller en services de l'Information avec plus de 10 ans d'expérience dans le domaine.

résultats avec de multiples formats (JASON, XML, etc…) tandis que SOAP ne communique qu'avec du XML !

Son majeur inconvénient reste cependant est qu'il est difficile d'implémenter les protocoles de sécurités avec REST, tandis qu'avec SOAP des protocoles sécurité font partie de ce style par défaut.

	REST	**SOAP**
Avantages	- Relativement simple à implémenter et à maintenir. - L'information peut être enregistrée par le client pour éviter des appels répétés.	- Des protocoles sécurités y sont intégrés de base : WS-Security[12]. - Est décrit de façon simple et lisible sous forme d'un document WSDL.
Inconvénients	- Fonctionne uniquement avec les bases du protocole http. - Y ajouter des protocoles de sécurités peut être plus compliqué, sauf SSL qu'il supporte par défaut.	- Difficile a implémenter et peu populaire pour le développement web ou mobile. - Nécessite une bande passante importante pour faire circuler les données.
Formats de réponse	- Multiples et variés.	- Un format unique : XML.
Quand utiliser	- Pour les applications orientées web ou mobile.	- Pour des applications qui nécessitent une très forte sécurité.
Exemples populaires	- Twitter API, LinkedIn API, etc…	- PayPal SOAP API, etc…

Tableau 2 : REST vs. SOAP

Beaucoup de professionnels du domaine semblent être d'accord sur une chose : l'une des idées principales derrière les Web Services est de pouvoir simplifier au maximum l'accès aux données. C'est là où REST excelle, en comparaison à SOAP.

[12] WS-Security : protocole qui permet d'appliquer de la sécurité aux web service SOAP.

JSON - JavaScript Object Notation :

JSON est un format d'échange de données qui respecte une structure pour véhiculer facilement et légèrement les informations.

	JSON	**XML**
Avantages	- La vitesse de traitement. - La simplicité de mise en œuvre. - Syntaxe simple à parser et à générer par la machine.	- Largement utilisé et reconnu par tous les langages de programmation. - Il est plus facile à lire et convient mieux pour les fichiers destinés aux non programmeurs.

Tableau 3 : JSON vs. XML

JSON, associé à REST, est plus utilisé et préféré pour les applications mobiles que le format XML vu que ces dernières nécessitent rarement de récupérer des données fortement structurées. JSON est rapide, léger et efficace.

Oracle GlassFish Server :

GlassFish est un serveur d'application open-source sponsorisé par Oracle. Sa première version a été lancée par Sun Microsystems en 2005 et il est aujourd'hui à sa version 4.

Le serveur utilisé pour le déploiement des web services REST est Glassfish. L'une des raisons qui ont motivé ce choix c'est que GlassFish propose une console d'administration que d'autres serveurs d'applications ne proposent pas. Il est également parmi les plus rapides quand il s'agit de temps d'exécution, de déploiement et de redéploiement, et supporte les web services REST par défaut, sans avoir à lui ajouter quoique ce soit.

NetBeans :

NetBeans est un environnement de développement open-source. Il a été choisi pour la réalisation des web services REST pour sa facilité d'utilisation, sa riche documentation et le fait qu'il est rapide et ergonomique.

IV – Tests des web services REST

Une fois les web service créés, je devais passer par une période de test de ces derniers, pour les vérifier et les valider, question de m'assurer qu'ils étaient fonctionnels et pourraient être utilisés sans problèmes par la suite.

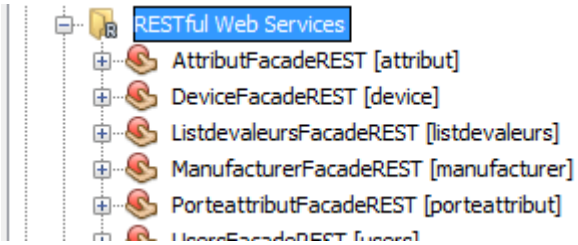

Figure 22 : Les web services REST

Il existe principalement deux catégories de tests, les tests en boîte noire et les tests en boîte blanche. Vu que je suis celle qui a effectué les tests, et étant donné que j'ai connaissance du code, ces tests vont s'inscrire dans la catégorie « boîte blanche ».

Bien que j'ai testé un à un tous les web services, j'ai choisis aléatoirement un seul à présenter dans ce rapport, pour éviter une répétition inutile étant donné que tous les autres ont été testé de la même manière.

Les web service en question est le web service permettant la manipulation des « Manufacturers ».

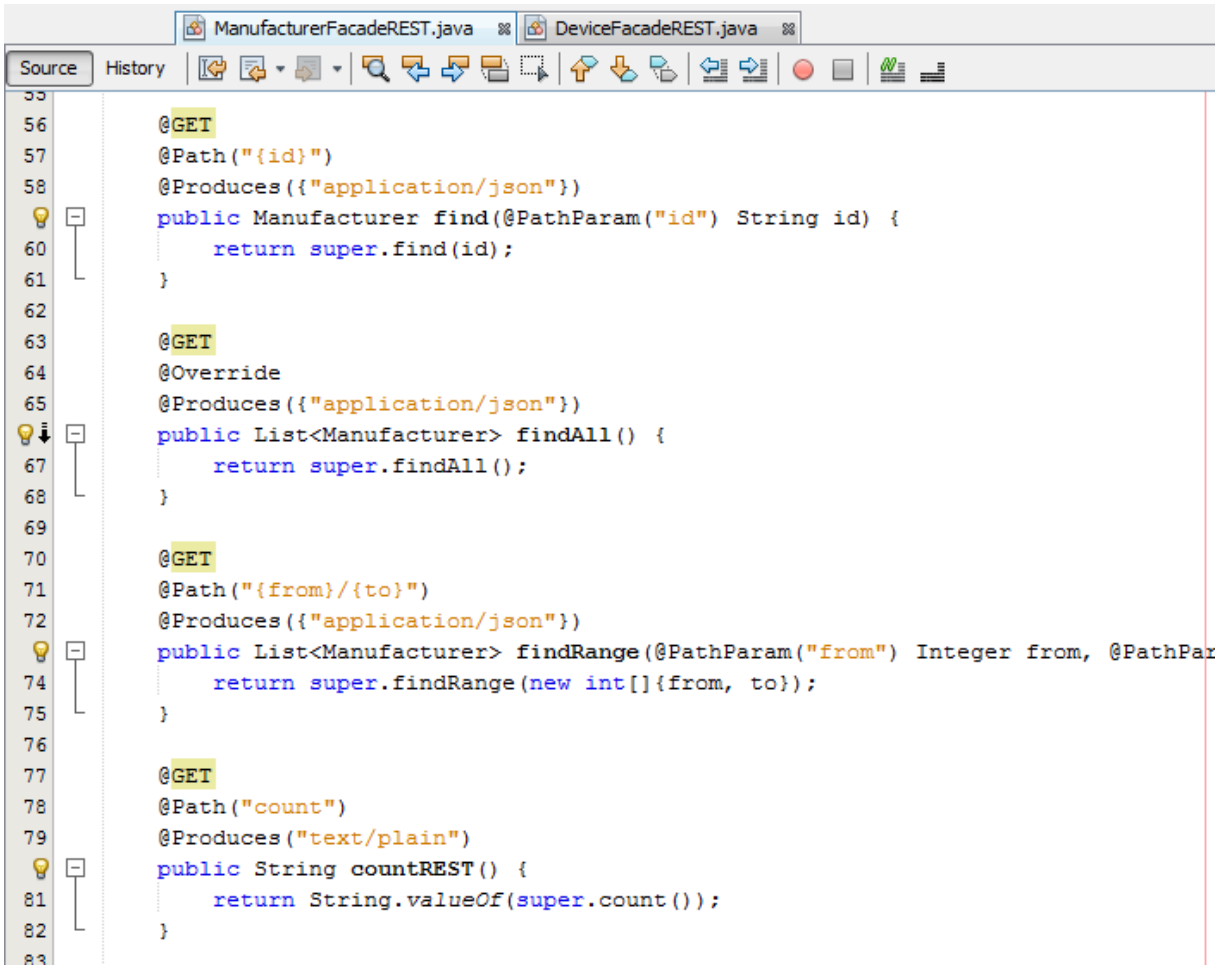

Figure 23 : Web service Manufacturer

Sur cette figure, vous pouvez également constater que le format d'échangé spécifié est JSON.

Le premier type de test effectué était le test unitaire. Les tests unitaires sont les tests de blocs de code, que je testais au fur et à mesure de la création des web services. J'ai également testé le résultat donné via l'URI du service.

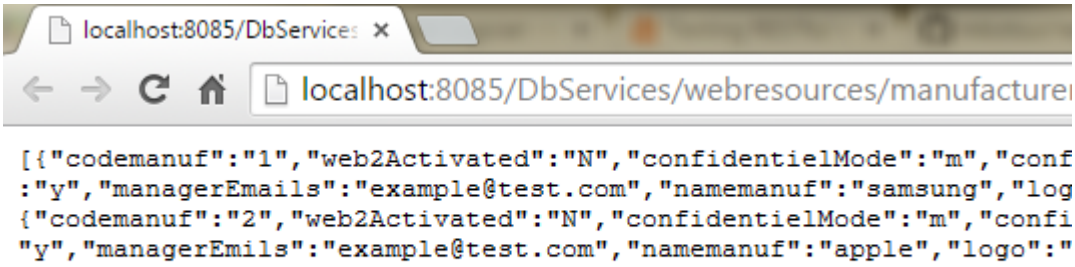

Figure 24 : Web service Manufacturer, résultat JSON

Le deuxième type de test était le test d'intégration qui s'exécute aussi bien en boîte noire qu'en boîte blanche. Ces tests servent à vérifier que les services s'intègrent bien avec d'autres composants, ou systèmes. Pour ce test, j'ai intégré les web services dans une application JavaEE basique.

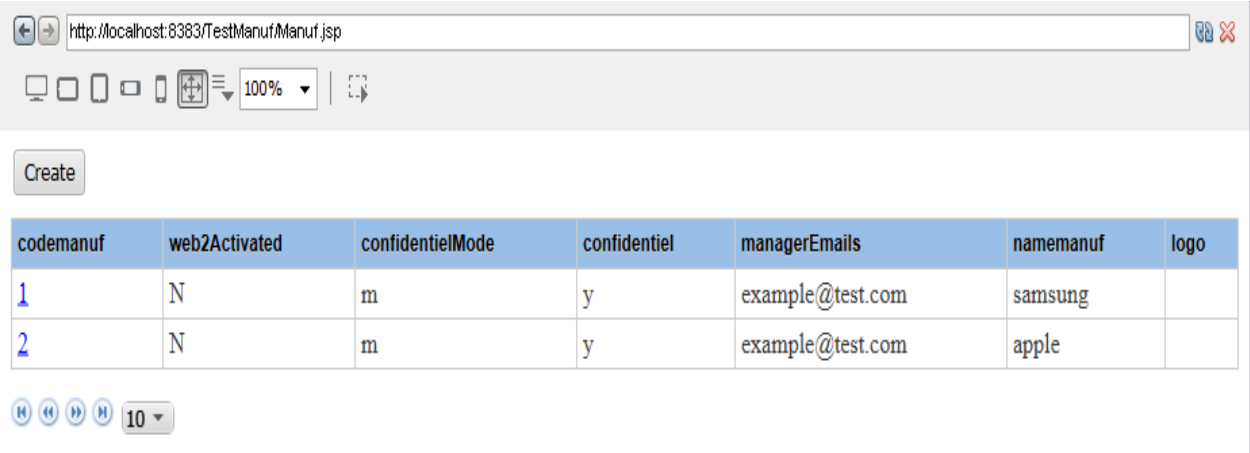

Figure 25 : Web service Manufacturer, test JavaEE

V – Les Interfaces Homme/Machine (IHM)

Dans la dernière partie de ce chapitre, je vais vous présenter les quatre interfaces principales de l'application mobile. Ces illustrations sont les maquettes validées et approuvée des vues finales, et elles sont plus adéquates aux explications car contiennent le plus de détails possible.

Ces maquettes ont été réalisées avec le logiciel Balsamiq.

1 – Authentification (Authetication) :

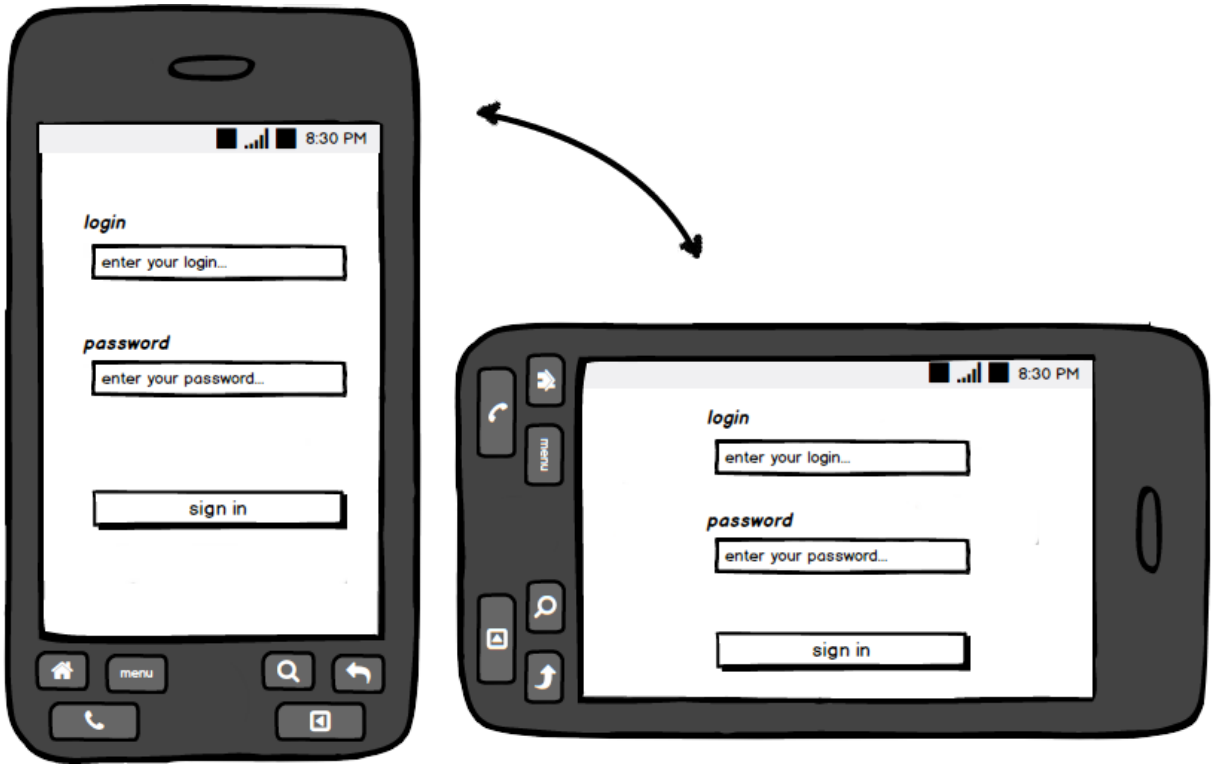

Figure 26 : IHM - Authentication

Constituée de deux champs textes et un bouton, l'interface d'authentification sert à vérifier l'identité de l'utilisateur avant de lui donner le droit d'accès à l'application.

L'utilisateur devrait saisir son nom d'utilisateur dans le champ « login », et son mot de passe dans le champ « password », avant de procéder à la connexion en appuyant sur le bouton « sign in ».

2 – Recherche (Search) :

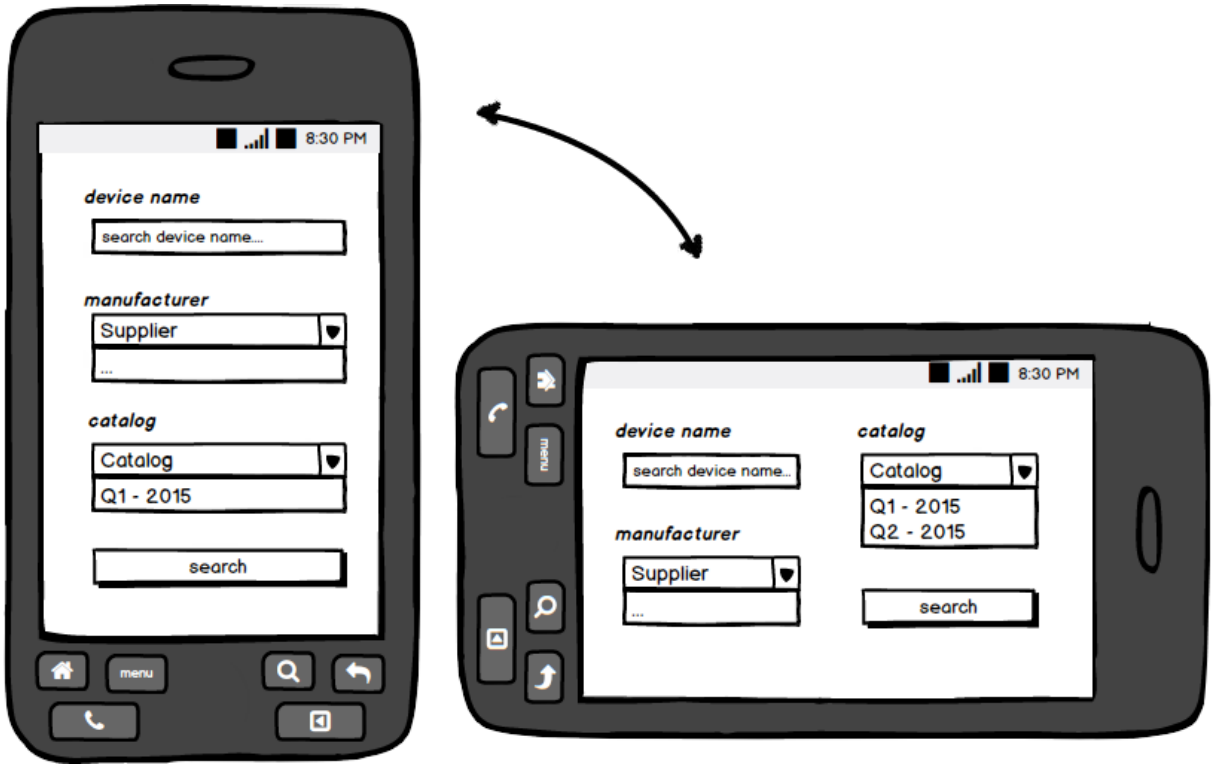

Figure 26 : IHM - Search

Si l'authentification est réussie, l'utilisateur sera dirigé vers la vue de recherche.

Cette vue permet à l'utilisateur d'effectuer sa recherche des devices. Il peut se baser sur l'un des trois critères : nom du device, son fournisseur, ou le catalogue, tout comme il peut se baser en précisant les trois critères.

L'utilisateur devrait entrer le nom entier, ou une partie du nom, du device dans le champ « device name », pour effectuer la recherche par nom. Il devrait également sélectionné un choix des listes déroulantes « manufacturer », et/ou « catalog », s'il souhaite faire une recherche en se basant sur ces critères.

3 – Résultats de la recherche (Search results) :

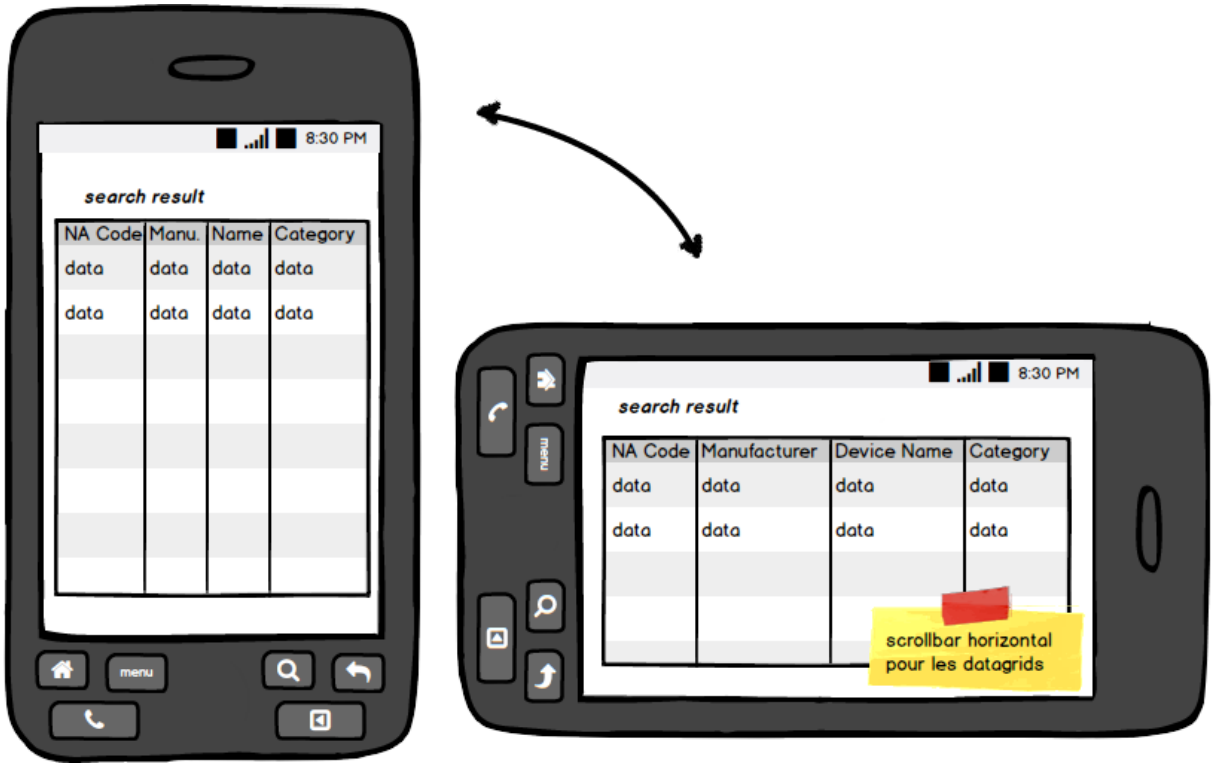

Figure 27 : IHM – Search Results

Une fois la recherche effectuée, si des résultats correspondants aux critères existent, l'utilisateur sera dirigé vers cette vue.

Les résultats seront affiché sur une liste, idéalement un datagrid, mais, les éléments graphiques basique d'Android n'en offrent pas, il faudrait éventuellement avoir recours aux solutions proposées par Xamarin ou AndroidJetPack[13].

A partir de cette liste, l'utilisateur peut sélectionner la ligne correspondante au device de son choix, pour pouvoir accéder à une quatrième vue qui contiendra plus de détails.

[13] Xamarin et AndroidJetPack proposent des solutions payantes pour compléter les éléments graphiques d'Android avec des éléments plus performants et plus personnalisé. Les prix de l'élément DataGrid par exemple peut varier entre 400$ pour une utilisation dans un seul projet, et 1000$ pour une utilisation dans un nombre illimité de projets.

4 – Résultats détaillés (Details) :

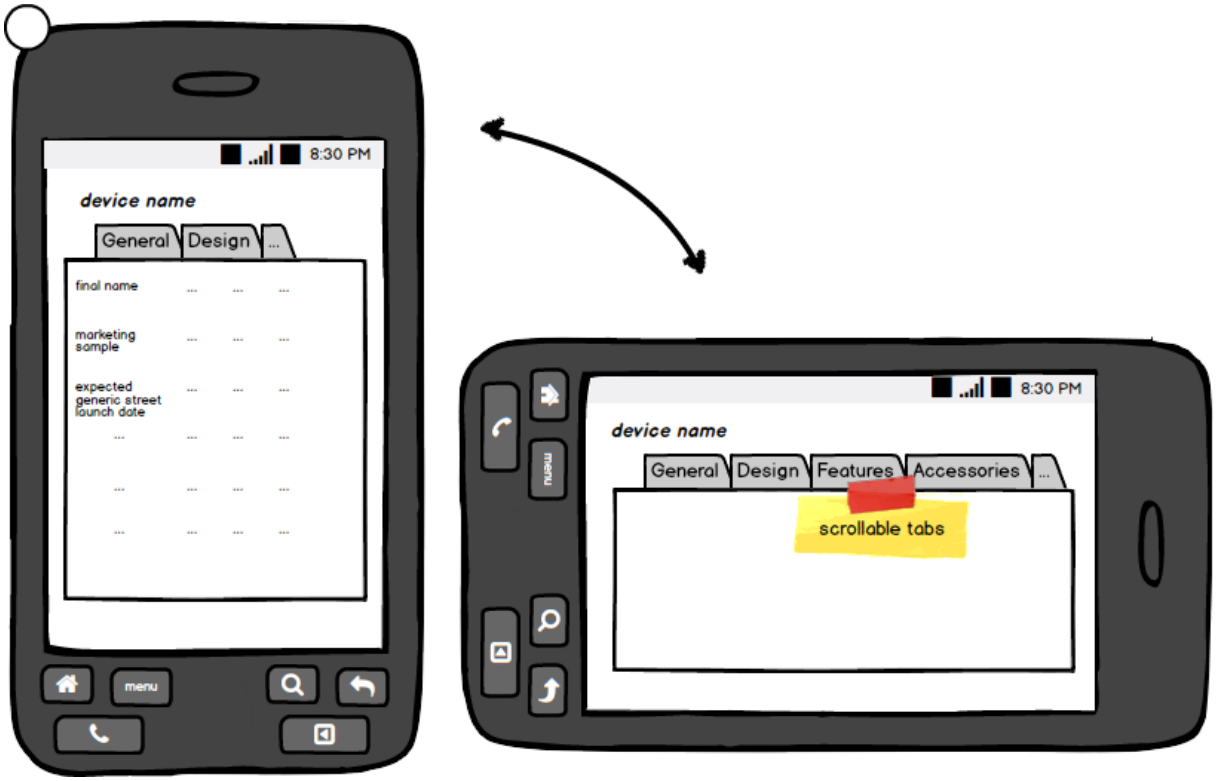

Figure 28 : IHM – Details

Cette vue donne tous les détails possibles sur le device. Ces attributs sont regroupés dans cinq différents onglets :

- **L'onglet « General »** : contient les informations générales sur le device.
- **L'onglet « Design »** : contient les informations à caractère graphique sur le device, comme la largeur et la hauteur de son écran (si c'est un Smartphone par exemple).
- **L'onglet « Features »** : précise les caractéristiques supplémentaires du device, par exemple s'il a une caméra ou non.
- **L'onglet « Accessories »** : contient les informations sur les accessoires qu'un device peut avoir.
- **L'onglet « Additional Information »** : contient des informations complémentaire sur un device s'il y en a.

Conclusion : Le chapitre « réalisation » a porté sur l'approche théorique suivie pour réaliser le projet, l'architecture adoptée qui est naturellement l'architecture 3-tiers pour une application mobile, puis le chapitre a inclus une brève démonstration sur les tests effectués sur les web services et finalement il a présenté les IHMs de l'application.

Conclusion

Mon projet de fin d'études effectué au sein de l'entité Sofrecom Service Maroc était une occasion d'acquérir de nouvelles connaissances dans le domaine du développement, et plus précisément du développement mobile. Ce projet était également une opportunité qui m'a permis d'intégrer une équipe professionnelle et d'interagir avec différentes personnes.

Ce projet consistait à concevoir et créer un prototype mobile Android permettant la recherche et la visualisation des devices, propres au groupe Orange. Ce prototype devrait dans un futur proche compléter l'application web déjà existante et opérationnelle depuis des années. La question qui se posait était de comprendre le fonctionnement de Device-DB web, pour pouvoir y rapprocher le plus possible le fonctionnement de l'application mobile.

Prenant ce point comme un point de départ, j'ai –à l'aide des orientations et conseils de mes maîtres de stage- structuré le projet en plusieurs phases, reposant sur la méthode agile RAD. Dans un premier temps, il fallait définir le contexte de développement mobile et étudier brièvement ses limites comparé au développement web. Par la suite, j'ai attaqué la phase du cadrage qui consistait à définir les fonctions principales du premier prototype. Puis il fallait passer à la phase design où j'ai eu à créer les IHMs qui s'adapteront au mieux aux fonctionnalités précisées durant la phase précédente. Une fois les vues nécessaires validées, j'ai commencé la phase la plus lourde du projet qui est la construction, et finalement, le projet débouchera sur la phase finalisation qui sert à tester le produit final.

Les difficultés que j'ai rencontrées durant mon stage étaient de natures variées mais, pour ne citer que les principales je dirais que la difficulté majeure

était pour moi de respecter au mieux les plannings établit, et aussi, essayer de passer outre les difficultés techniques (aussi bien physiques que logiques) que j'ai rencontré tout au long du projet.

Comme perspective à ce projet, je souhaiterai pouvoir y ajouter l'ensemble, ou du moins une majeure partie, des fonctionnalités proposées dans la version web, pour le rendre plus complet. Je souhaiterai également pouvoir élever le niveau de sécurité des web services et y implémenter des couches supplémentaires qui garantiraient ce but. Et finalement, je souhaiterai que ce prototype soit amélioré et adopté car le mobile devient de plus en plus une nécessité indispensable dans un monde où la technologie évolue sans cesse.

Bibliographie

Documents internes à Sofrecom :

OUK-SPEC-fonct_D-BD : document des spécifications fonctionnelles.

OUK-SPEC-technique_D-BD : document des spécifications techniques.

Carte_Fonctionnelle : document de la carte fonctionnelle.

Ouvrages :

ANDROID, de Florent GARIN. Dunod, Édition 2, Octobre 2012, 231 pages.

L'Art du développement Android, de Mark MURPHY. Pearson, Édition 4, Décembre 2012, 626 pages.

RAPID Application Development, de James MARTIN. Macmillan Coll Div, Mai 1991, 736 pages.

Sites Web :

Android :

http://android.developpez.com/cours/

https://developer.android.com/guide/

http://openclassrooms.com/courses/creez-des-applications-pour-android

Méthodologie Agile :

http://www.commentcamarche.net/contents/477-methodes-agiles-rad-xp

http://www.piloter.org/projet/methode/rad.htm

Autre :

http://www.infoq.com/articles/rest-soap-when-to-use-each

http://www.journaldunet.com/

http://www.lemagit.fr/

http://www.lesaffaires.com/

http://www.uml.org/

Annexe : Plan réel du projet

Planning **DeviceDB Mobile**
09/03/15 – 17/07/15

Rapport Gantt Project

Dates de début/fin du projet : 9 Mars 2015 - 24 Juillet 2015
Avancée : 81%
Tâches : 18

Tâches :

Nom	Durée	Date de début	Date de fin
Phase 1 - Initialisation	5	09/03/15	13/03/15
Étude du contexte générale	5	09/03/15	13/03/15
Phase 2 - Cadrage	16	16/03/15	06/04/15
Documentation initiale	7	16/03/15	24/03/15
Rédaction du CDC	15	16/03/15	03/04/15
Validation du CDC	1	06/04/15	06/04/15
Phase 3 - Design	23	07/04/15	07/05/15
Documentation sur les outils de maquettage	2	07/04/15	08/04/15
Création des mockups des vues/interfaces	20	09/04/15	06/05/15
Validation des mockups	1	07/05/15	07/05/15
Phase 4 - Construction	49	08/05/15	15/07/15
Conception	44	08/05/15	08/07/15
Développement des services REST	14	08/05/15	27/05/15
Test des services REST	2	28/05/15	29/05/15

Développement mobile	30	01/06/15	10/07/15
Test de l'application mobile	3	13/07/15	15/07/15
Phase 5 - Finalisation	7	16/07/15	24/07/15
Déploiement	7	16/07/15	24/07/15

Tableau 4 : Les tâches

Diagramme de Gantt

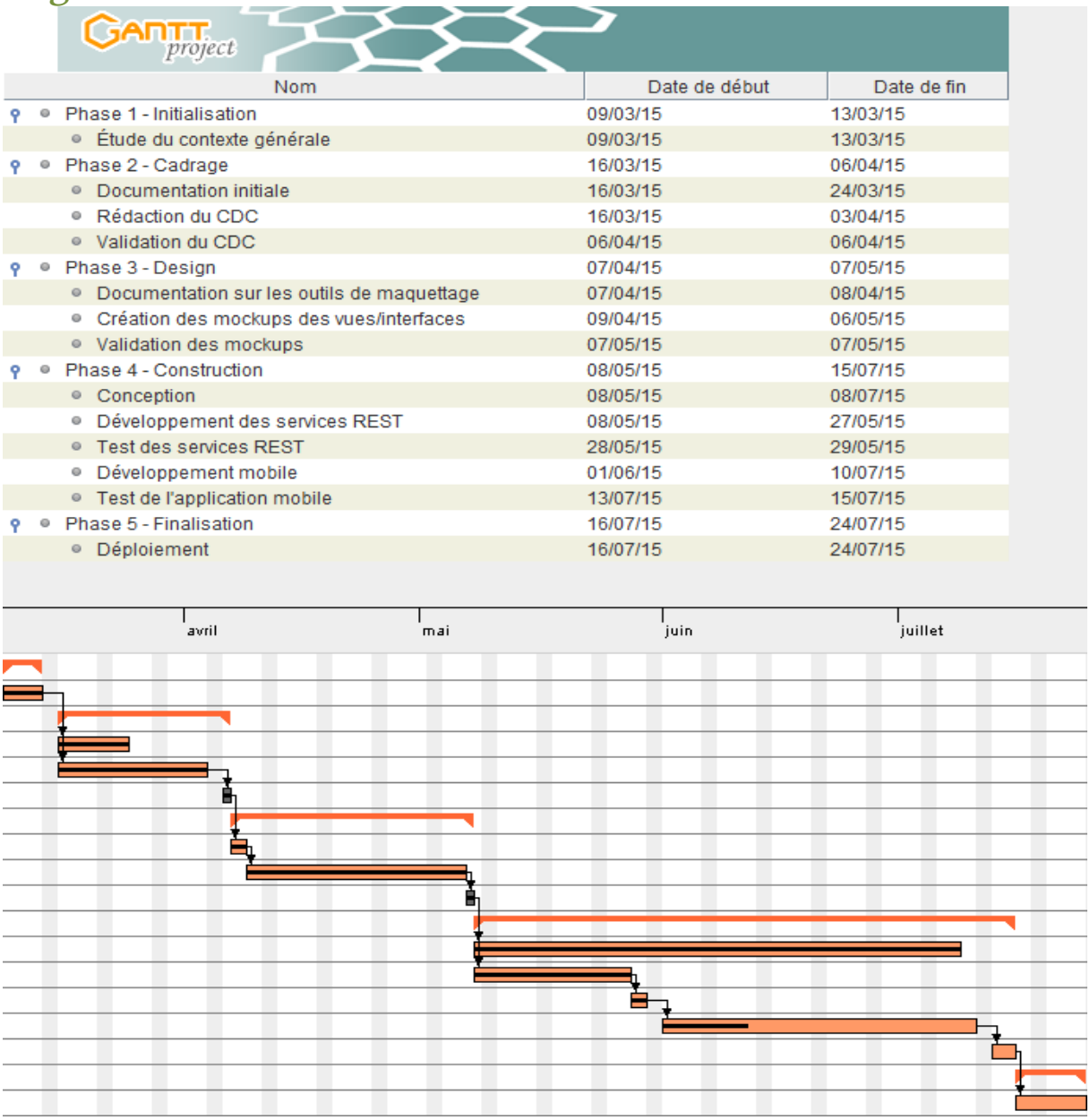

Figure 29 : Diagramme de GANTT réel