

UCLCampus

Une application mobile Open Source pour étudiants UCL

Mémoire présenté par
Corentin LAMY , Jérôme LEMAIRE

en vue de l'obtention du grade de Master
Sciences informatique & Ingénieur civil en Informatique

Promoteur
Yves DEVILLE

Lecteurs
Charles PECHEUR, Hildeberto MENDONÇA FILHO

Année académique 2016-2017

Remerciements

Ce mémoire est le résultat d'un travail de près d'un an. En préambule, nous adressons tous nos remerciements aux personnes avec lesquelles nous avons pu échanger et qui nous ont aidés pour la rédaction de ce mémoire.

Nous tenons, tout d'abord, à remercier Monsieur Yves Deville, promoteur de ce mémoire, pour le temps qu'il nous a consacré, ses remarques pertinentes qui nous ont incités à explorer de nouvelles pistes de solutions ainsi que pour sa relecture attentive.

Merci à Monsieur Hildeberto Mendonça Filho, qui nous a accordé de son temps pour discuter du fonctionnement des différents services informatiques de l'Université catholique de Louvain et, plus particulièrement du projet OSIS.

Merci à Monsieur Olivier Francq et le service des restaurants universitaires, qui ont accueilli avec enthousiasme notre projet et nous ont mis en contact avec la société responsable du développement de l'application Resto4U.

Merci à Monsieur Charles-Henri Nyns et le service informatique des bibliothèques pour leur implication dans notre projet à travers la mise en place d'une API permettant l'accès aux informations pratiques des bibliothèques universitaires.

Nous tenions également à remercier Monsieur Francis Somers, qui nous a transmis des références sur le fonctionnement de l'ESB WSO2 et nous en a autorisé l'accès.

Enfin, nous adressons nos plus sincères remerciements à nos familles, proches, collègues de la salle Parnas et amis, qui nous ont accompagnés, aidés, soutenus et encouragés tout au long de la réalisation de ce mémoire.

Table des matières

1	Introduction	15
2	Background	17
2.1	Open Source	17
2.1.1	Philosophie	17
2.1.2	Propriétés intellectuelles	19
2.1.3	Les principales licences	21
2.1.4	Projet Open Source	23
2.2	Ionic, Angular et Cordova	26
2.2.1	Ionic	27
2.2.2	Angular	27
2.2.3	Cordova	32
2.2.4	Services proposés par Ionic	34
2.3	Entreprise Service Bus	34
2.3.1	Concept	34
2.3.2	WSO2	36
2.3.3	WSO2 à l'UCL	36
2.4	Portail et Learning Management System	37
2.4.1	Portail	37
2.4.2	Moodle	38
2.5	Applications UCL	39
2.5.1	OSIS	39
2.5.2	Shibboleth	41
2.6	Application UCLCampus existante	43
2.6.1	Concept	43
2.6.2	Choix d'implémentation	43
2.6.3	Limitations	45
2.6.4	Décision	48
3	Cahier des charges	49
3.1	Contexte	49
3.1.1	Audience cible	49
3.1.2	Besoins	49
3.1.3	Objectifs	52
3.1.4	Applications existantes	52
3.1.5	Ressources UCL disponibles	58
3.2	Spécifications fonctionnelles	61
3.2.1	Front-end	61
3.2.2	Backend	65
3.2.3	Récits utilisateur	66
3.3	Spécifications non-fonctionnelles	68

3.3.1	Facilité d'utilisation	68
3.3.2	Langages de programmation	68
3.3.3	Fiabilité	68
3.3.4	Maintenabilité	68
3.3.5	Performances	69
3.3.6	Compatibilité	69
3.3.7	Légal	69
4	Développement de l'application UCLCampus	71
4.1	Approche Open Source	71
4.1.1	Choix d'une licence	71
4.1.2	Forge utilisée	72
4.1.3	Communauté	74
4.1.4	Données confidentielles	74
4.2	Tests	75
4.2.1	Tests unitaires	75
4.2.2	Tests bout en bout	76
4.2.3	Jasmine	76
4.3	Choix d'implémentation	77
4.3.1	Framework Ionic	77
4.3.2	Interface Utilisateur	78
4.3.3	Partie logique	79
4.4	Sécurité	80
4.4.1	Authentification sur les services UCL	80
4.4.2	Stockage de l'information	81
4.5	Modules implémentés	83
4.5.1	Utilisation de l'objet Promise de JavaScript	83
4.5.2	Récupération de l'information par flux RSS	83
4.5.3	Récupération de l'information par WSO2	88
4.5.4	Récupération de l'information par des services web internes à l'UCL	90
4.5.5	Récupération de l'information par des services web externes à l'UCL	93
4.5.6	Informations codées en dur	96
4.5.7	Lancement d'application	97
5	Évaluation	101
5.1	Open Source	101
5.2	Environnement UCL	102
5.3	Sécurité	104
5.4	Framework Ionic	106
5.4.1	Les différentes versions	106
5.4.2	Développement hybride	107
5.5	Application déployée	108
5.5.1	Cycle de développement	108
5.5.2	Le projet en chiffres	109
5.5.3	Comparaison avec le Cahier des charges	109
5.5.4	Évolution et possibilités futures	114
6	Conclusion	115
	Bibliographie	116
A	An Open Letter to Hobbyists	121

B	Enquête	123
C	Personnes ressources	125
D	Illustrations des requêtes	127
D.1	Requête ADE	127
D.1.1	ouverture de la session	127
D.1.2	Récupération des différents projets d'horaire disponibles	127
D.1.3	Sélectionner un projet	127
D.1.4	Récupération de l'identifiant ADE du cours LFSAB1101	127
D.1.5	Récupération des activités du cours LFSAB1101	128
D.2	Requêtes des flux RSS	129
D.2.1	Récupération du flux RSS des actualités UCL	129
D.2.2	Récupération du flux RSS des évènements	130
D.3	Requêtes WSO2	131
D.3.1	Récupération de la liste des bibliothèques	131
D.3.2	Récupération des détails de la bibliothèque de droit (BDRT)	131
E	Guide de développement	133
E.1	Installation de l'application	133
E.1.1	Prérequis	133
E.1.2	Installation et lancement	134
E.2	Convention de l'application	135
E.2.1	Convention de codage	135
E.2.2	Structure du projet	135
F	Guide utilisateur	137
F.1	Menu	137
F.2	Actualités	138
F.3	Évènements	139
F.4	Etudes	140
F.5	Bibliothèques	141
F.6	Service d'aide	141
F.7	Restaurants	142
F.8	Covoiturage	143

Table des figures

2.1	Répartition des licences	22
2.2	Illustration des compatibilités entre les licences	23
2.3	Évolution des parts de marché des systèmes d'exploitation des smartphones	26
2.4	Représentation de l'architecture d'une application avec le framework Ionic 3	28
2.5	Structure d'un module	30
2.6	Utilisation de modules personnalisés au sein d'un template	31
2.7	Interactions entre le Document Object Model (DOM) et un composant	31
2.8	Instanciation d'un composant avec services	32
2.9	Patron de conception adaptateur avec héritage	33
2.10	Patron de conception avec agrégation	33
2.11	Infrastructure informatique des applications de gestion de l'UCL	35
2.12	Situation initiale des interactions entre les applications de l'UCL	36
2.13	Situation actuelle des interactions entre les applications de l'UCL	36
2.14	Situation finale des interactions entre les applications de l'UCL	37
2.15	Abstraction des interactions entre les applications de l'UCL	37
2.16	Structure d'OSIS	40
2.17	Processus de Single Sign-On	42
2.18	Structure d'un Modèle-Vue-Contrôleur	44
2.19	Structure d'un Modèle-Vue-VueModèle	44
3.1	Répartition du niveau d'études des répondants	50
3.2	Répartition des répondants par faculté	50
3.3	Illustrations de l'application Quivr	53
3.4	Illustrations de l'application EPFL	54
3.5	Illustrations de l'application UCL Go!	54
3.6	Illustrations de l'application LLNCampus	55
3.7	Illustrations de l'application LLNMap	55
3.8	Illustrations de l'application Guindaille 2.0	56
3.9	Illustrations de l'application Resto4u	56
3.10	Illustrations de l'application Commuty	57
3.11	Illustrations de l'application Affluence	58
3.12	Arborescence des pages	62
3.13	Illustrations de l'application UCLCampus	63
3.14	Illustrations de l'application UCLCampus	64
3.15	Processus d'accès aux informations via le portail UCL	65
3.16	Logo officiel de l'UCL en blanc sur fond Pantone 282	69
4.1	Abstraction des interactions entre les modules d'une application développée sous Ionic	78
4.2	Abstraction des interactions pour le module News	85
4.3	Diagramme de séquence pour l'affichage des News	86
4.4	Abstraction des interactions pour le module Events	86

4.5	Diagramme de séquence pour l’affichage des Events	88
4.6	Abstraction des interactions pour le module Bibliothèques	89
4.7	Diagramme de séquence pour l’affichage des bibliothèques	90
4.8	Abstraction des interactions pour le module Etudes	91
4.9	Diagramme de séquence de la récupération de l’horaire ADE	93
4.10	Abstraction des interactions pour le module Cartes	94
4.11	Diagramme de séquence pour le module Cartes	96
4.12	Abstraction des interactions pour le module Desk	97
4.13	Abstraction des interactions pour le lancement d’application	98
5.1	Système de sécurité théorique future pour l’application UCLCampus	104
5.2	Système de sécurité théorique future pour l’application UCLCampus	105
5.3	Illustration du modèle en V	108
E.1	Structure des dossiers de l’application	136

Liste des tableaux

2.1	Tableaux récapitulatifs des principales licences Open Source	22
3.1	Résultats de la question : "Quelles fonctionnalités/outils veux-tu avoir dans le cadre de tes études ?"	51
3.2	Résultats de la question : Quelles fonctionnalités/outils veux-tu avoir dans le cadre de la vie sur le campus ?"	52
4.1	Tableau des licences utilisées par l'application UCLCampus	73
5.1	Fonctionnalités en rapport avec la vie sur le campus	110
5.2	Fonctionnalités d'utilité publique	111
5.3	Fonctionnalités en rapport aux études	112
5.4	Grille d'évaluation des critères non fonctionnels	113
C.1	Personnes contactées dans le cadre du mémoire	126

Glossaire

AES Advanced Encryption Standard. 82

API Application Programming Interface. 29, 30, 32, 34, 36, 37, 46, 57, 59, 74, 81, 88–90, 92–95, 103, 104, 112, 116

BSD Berkeley Software Distribution. 21, 22, 45

CLI Commande Line Interface. 34, 106

CMS Content Management System. 37, 58

CSS Cascading Style Sheet. 27, 29, 32, 59, 107

cURL client Uniform Resource Locator. 88

DOM Document Object Model. 9, 28, 31, 75, 76

ESB Entreprise Service Bus. 15, 34–37, 40, 46, 88, 103, 112, 114

FSF Free Software Foundation. 18, 21

GNU GPL GNU General Public Licence. 18, 21–23, 38, 39, 72

GPL General Public Licence. 21, 22, 39, 54, 61, 69, 72

HTML HyperText Markup Language. 28–31, 33, 45–47, 59, 79, 95, 97, 99

HTTP Hypertext Transfer Protocol. 83, 84, 88, 89

IdP Identity Provider. 41, 42, 80

JSON JavaScript Object Notation. 40, 59, 82, 84, 85, 87, 89–92, 95, 96, 111

LDAP Lightweight Directory Access Protocol. 36

LMS Learning Management System. 37

MOOC Massive Open Online Course. 15

MVC Model-View-Controller. 43, 44

MVVM Model-View-ViewModel. 43, 44

OSI Open Source Initiative. 18, 19, 21

OSIS Open Student Information System. 39, 40, 60, 61, 102, 114, 116

OSRM Open Source Routing Machine. 45

REST Representational State Transfer. 36

RSS Rich Site Summary. 44, 58, 59, 83–87, 110, 111, 116

SAML Security Assertion Markup Language. 41, 42, 80
SDK Software Development Kit. 27
SOA Service Oriented Architecture. 35, 36
SOAP Simple Object Access Protocol. 36
SP Service Provider. 41, 42, 80
SSO Single Sing On. 38, 41, 80, 104
TDD Test Driven Development. 77
URI Uniform Resource Identifier. 99
URL Uniform Resource Locator. 45, 46, 81, 97, 103, 104
VPN Virtual Private Network. 88
W3C World Wide Web Consortium. 28, 29
XML Extensible Markup Language. 28, 29, 83, 84, 92
YQL Yahoo Query Language. 84

Chapitre 1

Introduction

A l'heure du numérique, des réseaux sociaux et des smartphones, il est essentiel pour les universités de s'adapter et d'utiliser les réseaux de communication privilégiés par leur public cible, à savoir essentiellement des jeunes de 18 à 25 ans. Pour ce faire, de nombreuses universités disposent déjà de leur application officielle pour aider les nouveaux venus et faciliter la vie de l'ensemble de ses étudiants sur le campus.

L'Université catholique de Louvain (UCL), consciente de l'évolution de la société vers le numérique, a élaboré un plan stratégique nommé "Université numérique"[1]. La vision qui y est proposée est celle d'une université où le numérique favorise la création, la diffusion et l'acquisition de connaissances. Ce projet s'articule autour de trois axes :

Open Education Approche proposant de favoriser le développement et l'exploitation de ressources éducatives libres telles que les Massive Open Online Course (MOOC).

Open Publication Approche visant à donner accès librement et gratuitement aux résultats des recherches.

Open Source Approche permettant à tous les utilisateurs d'un logiciel de l'analyser, le modifier et le redistribuer en vue de l'améliorer.

Dans cette perspective, l'Université catholique de Louvain souhaite proposer à ses étudiants une application Open Source répondant à leurs besoins. La conception de l'application UCLCampus s'insère dans un contexte de renouvellement et de refonte technologique : déploiement du nouveau portail internet, développement Open Source d'un système de gestion des ressources pédagogiques pour les établissements scolaires supérieurs. Il y a aussi l'apparition d'un Entreprise Service Bus (ESB) dont les objectifs sont de coordonner et standardiser les interactions entre les différentes ressources et services informatiques de l'Université catholique de Louvain. Pour l'ensemble de ces changements, l'UCL a fait le choix d'utiliser des outils et des technologies Open Source.

Le travail réalisé est la poursuite d'un projet débuté l'an passé par deux étudiants en master informatique à l'Université catholique de Louvain. L'objectif est de développer une application multiplateforme à destination des étudiants de l'Université catholique de Louvain. Dans ce contexte de transition et d'évolution, l'approche de développement en Open Source permet d'une part, aux étudiants de définir eux-même leurs besoins, de contribuer à la réalisation d'une application leur correspondant et, d'autre part, d'assurer la survie du projet dans le futur.

Le déploiement de l'application multiplateforme en septembre 2017 constitue la partie visible du travail fourni. La partie écrite synthétise les expériences acquises, les décisions prises et les difficultés rencontrées lors du développement de l'application en Open Source. En vue de faciliter la poursuite du projet et son évolution dans le futur, une attention particulière a été donnée aux points suivants : cahier des charges précis et détaillé, code de qualité et mise en place d'un

environnement propice au développement Open Source.

Le rapport écrit se divise en quatre grandes parties.

La première donne un aperçu des environnements et contextes au sein desquels l'application UCLCampus a été conceptualisée. Tout d'abord, le domaine de l'Open Source (sa philosophie, son histoire et ses règles), qui a impacté la conception du projet, est présenté. Cette partie décrit, ensuite, les aspects techniques et les outils de développement liés à l'environnement de programmation et se poursuit avec la présentation du paysage informatique de l'Université catholique de Louvain au sein duquel le projet doit évoluer. Pour terminer, une analyse du travail précédent est effectuée.

Le cahier des charges de l'application constitue la seconde partie de cet écrit. Il commence par une analyse du contexte dans lequel le projet s'inscrit, par la définition du public cible et de ses besoins. Ces éléments ont permis de se doter d'un cadre pour comparer les différentes applications universitaires et leurs fonctionnalités ainsi que pour déterminer l'offre des services informatiques utiles de l'UCL. Suite à cette analyse, les spécifications fonctionnelles et non fonctionnelles de l'application ont pu être établies.

La troisième section décrit les choix de développement et d'implémentation en tenant compte des contraintes liées à la réalité, aux technologies utilisées et à l'approche de développement choisie. Pour commencer, la construction d'un environnement de développement Open Source est abordée ainsi que la question de la mise en place de tests. Ensuite, les choix d'implémentation généraux et l'architecture principale de l'application, éléments qui sont les conséquences des technologies de développement utilisées, sont exposés. Pour terminer cette partie, une description détaillée de l'architecture et du fonctionnement de chaque fonctionnalité de l'application est faite.

La dernière partie est un temps d'arrêt sur le travail, une analyse critique de l'environnement, du processus de développement et des choix d'implémentations posés. Une comparaison est réalisée entre le cahier des charges et l'application déployée, elle permet de voir certains manquements, mais également des pistes d'améliorations possibles pour le futur.

Nous espérons que la lecture de cet écrit vous permettra d'appréhender les étapes de notre projet¹, les choix posés à certains moments et découvrir l'environnement dans lequel l'application s'est développée ainsi que celui dans lequel elle devrait continuer à évoluer.

1. <https://github.com/jerlemaire/UCLCampus>

Chapitre 2

Background

La participation au projet UCLCampus nécessite préalablement de s'intéresser au contexte dans lequel celui-ci s'insère. UCLCampus étant développé en Open Source, la première section du chapitre est donc consacrée à la définition de ce mode de développement et à ses impacts. La section suivante détaille le fonctionnement des technologies et frameworks utilisés pour le développement de ce projet. Ensuite, l'application interagissant avec les services de l'Université catholique de Louvain, les sections 2.2 à 2.5 présentent les différents outils et services informatiques mis à disposition par l'université pour la réalisation d'UCLCampus. Finalement, une analyse du projet existant a été effectuée pour prendre connaissance de l'avancement du projet, des choix d'implémentation et du travail restant à fournir.

2.1 Open Source

Avant de développer un projet dit Open Source, il est important de comprendre l'intérêt et les enjeux qui se cachent derrière cette notion. Tout d'abord, la genèse et les événements marquants de ce mouvement sont abordés afin d'en comprendre les idéologies. Ensuite, le statut juridique des licences Open Source est étudié. Pour finir, un aspect plus pratique est abordé concernant la mise en place, le fonctionnement et les avantages d'un projet Open Source.

2.1.1 Philosophie

La philosophie de l'Open Source peut être perçue à travers la métaphore : "Nous sommes des nains sur les épaules de géants". Cela signifie que les intellectuels s'appuient sur le travail fourni et l'expérience acquise par leurs prédécesseurs pour atteindre de nouveaux sommets. En informatique, et dans le domaine de l'Open Source en particulier, cette pensée a une signification toute particulière car elle incite les programmeurs à se baser sur des codes existants pour améliorer ou réaliser de nouveaux projets. En partageant les codes sources de projets, la quantité d'expériences informatiques augmente. Les programmeurs sont donc autorisés à prendre connaissance et à réutiliser une partie d'un projet existant et ne doivent pas coder leur nouveau projet en partant de zéro. Ce partage de code facilite également l'amélioration de projets existants par d'autres programmeurs.

2.1.1.1 Historique

En 1976, Bill Gates écrivit une lettre intitulée "An Open Letter to Hobbyists", se trouvant en annexe, dans laquelle il défend l'idée que le travail des développeurs doit être justement rémunéré, si ce n'est pas le cas, l'innovation sera étouffée. C'est le début de la différenciation entre projets propriétaires et projets en Open Source.

Dès 1983, Richard Matthew Stallman désire un système d'exploitation et des outils libres d'utilisation, en lançant le projet GNU qui vise à créer le premier système d'exploitation libre. En 1985, Stallman fonde la Free Software Foundation (FSF) qui est l'entité chargée du projet GNU, mais aussi un lieu de réflexion, un vecteur de promotion et de défense du logiciel libre. La FSF a créé la licence GNU General Public Licence (GNU GPL). En 1991, Linus Torvalds travaille à développer le noyau d'un système d'exploitation, Linux. Fin 1991, Linux passe sous licence GPL, ce qui lance une importante dynamique de développement communautaire et conduit à une nouvelle version de Linux en 1994. Le projet Linux est aujourd'hui devenu une référence dans le monde l'Open Source.

Eric Steven Raymond, un des avocats célèbres de l'Open Source, est en désaccord avec la vision de l'Open Source défendue par la FSF. Selon lui, la supériorité du modèle de développement et la qualité des applications sont, davantage que les questions morales et humanistes, les atouts pour l'extension du modèle Open Source. Il est un des fondateurs de l'Open Source Initiative (OSI) créé en 1998 et a écrit 'La Cathédrale et le Bazar', texte considéré comme une des bases du mouvement. L'OSI est un phare de l'Open Source à travers son site opensource.org qui est garant de la définition officielle d'une licence Open Source.

Si les deux grands courants de pensées, logiciel libre (FSF) et logiciel Open Source (OSI), sont tous les deux pour un partage du code source entre les programmeurs, ils divergent au niveau de l'idéologie mise derrière le terme Open Source. Ces deux paradigmes sont détaillés dans la suite de la sous-section.

Les deux appellations logiciel libre et logiciel Open Source sont pratiquement équivalentes mais diffèrent par leurs pensées. Aucune n'accepte d'être englobée par l'autre. Pour éviter toute ambiguïté, la littérature utilise également les acronymes FOSS pour "Free and Open Source Software", ou FLOSS pour "Free/Libre and Open Source Software".

2.1.1.2 Logiciel libre

Pour Richard Stallman, le logiciel libre est une question de liberté. C'est la liberté que chaque individu doit avoir d'utiliser, modifier et redistribuer n'importe quel programme. Cette liberté est aussi fondamentale que la liberté d'expression et indissociable des valeurs d'éthique et de responsabilité sociale. En suivant cette logique, un logiciel propriétaire porte atteinte à cette liberté fondamentale, il nous en prive. Quatre conditions sont énoncées pour garantir un logiciel libre[2] :

- Permettre d'exécuter le programme
- Permettre d'étudier le programme et l'adapter en fonction de ses besoins
- Redistribuer le programme pour aider son prochain
- Améliorer le programme et distribuer ces améliorations au public

L'accès au code source n'est pas une finalité d'un logiciel libre mais un prérequis pour respecter les quatre conditions et, par conséquent cette liberté fondamentale. L'accès au code source est une condition nécessaire mais pas suffisante.

2.1.1.3 Licence Open Source

L'Open Source Initiative[3] a défini de manière précise la notion d'Open Source à l'aide de dix points :

- 1. Redistribution libre** La licence doit permettre à quiconque de vendre ou de donner le logiciel en tant que composant d'une distribution de logiciels constitués de programmes provenant de différentes sources. La licence ne doit pas exiger de droits d'auteur ou d'autres commissions sur une telle vente.
- 2. Code source** Le programme doit inclure le code source et autoriser sa distribution sous forme compilée ou sous forme de code source. Lorsqu'un produit n'est pas distribué avec

son code source, il doit exister un moyen bien indiqué pour l'obtenir sans autres frais qu'un coût raisonnable de reproduction, avec une préférence pour le téléchargement gratuit depuis l'Internet. Le code source doit être la forme privilégiée afin qu'un programmeur puisse modifier le programme. Il est interdit de proposer un code source rendu volontairement difficile à comprendre. Il est également interdit de soumettre des formes intermédiaires, comme le résultat d'un préprocesseur ou d'un traducteur automatique.

- 3. Oeuvres dérivées** La licence doit autoriser les modifications et les applications dérivées, et doit permettre leurs distributions sous les mêmes termes que ceux de la licence du logiciel original.
- 4. Intégrité du code source de l'auteur** La licence peut restreindre la redistribution d'un code source sous forme modifiée uniquement si elle permet la distribution de fichiers de correction (patch) avec le code source, dont le but est de modifier le programme au moment du développement. La licence doit explicitement permettre la distribution de logiciels développés à partir de codes sources modifiés. La licence peut exiger que les applications dérivées portent un nom différent ou un numéro de version distinct de celui du logiciel original.
- 5. Non-discrimination contre des personnes ou groupes** La licence ne doit pas discriminer des personnes ou des groupes de personnes.
- 6. Non-discrimination contre des champs d'application** La licence ne doit pas limiter l'utilisation du logiciel à un champ d'application particulier. Par exemple, elle ne doit pas interdire l'utilisation du logiciel dans le cadre d'une entreprise ou pour la recherche génétique.
- 7. Distribution de licence** Les droits attachés au programme doivent s'appliquer à tous ceux à qui il est redistribué, sans obligation pour ces parties d'obtenir une licence supplémentaire.
- 8. La licence ne doit pas être spécifique à un produit** Les droits attachés au programme ne doivent pas dépendre du fait qu'il fasse partie d'une quelconque distribution de logiciels. Si le programme est extrait de cette distribution et est utilisé ou distribué sous les termes de sa propre licence, toutes les parties auxquelles il est redistribué doivent bénéficier des mêmes droits que ceux accordés par la distribution originelle de logiciels.
- 9. La licence ne doit pas restreindre d'autres logiciels** La licence ne doit pas imposer de restrictions sur d'autres logiciels distribués avec le logiciel licencié. Par exemple, la licence ne doit pas exiger que tous les programmes distribués sur le même support soient des logiciels Open Source.
- 10. La licence doit être neutre sur le plan technologique** Aucune disposition de la licence ne peut aller à l'encontre d'une quelconque technologie ou style d'interface.

A travers cette définition, il est aisé de comprendre que l'Open Source prôné par l'OSI n'est pas qu'une notion de liberté et d'éthique mais également un modèle de développement qui peut mener à un modèle économique viable pour les sociétés et les programmeurs.

2.1.2 Propriétés intellectuelles

En informatique, il est courant de ne pas tout réinventer, de réutiliser du code déjà existant afin de concentrer ses efforts sur ce qui est spécifique au projet. Internet a permis de faciliter les échanges de problèmes, de solutions, de codes et d'idées au travers de forums et de sites spécialisés. Cela a décuplé la productivité et l'efficacité des développeurs. Il est donc important de se questionner sur les propriétés intellectuelles des codes se trouvant sur les sites et forums. C'est dans le but de répondre à cette question que cette sous-section a été rédigée.

2.1.2.1 Aspect général

Les programmes Open Source ne sont pas des programmes sans licence. Au contraire, c'est leur licence qui les rend Open Source. Ils ne sont pas non plus dans le domaine public, c'est-à-dire sans auteur ou propriétaire particulier, ou exempts de droits. Lorsqu'un développeur écrit un programme, il en détient les droits d'auteur, c'est le copyright. Dans certaines situations, c'est l'entreprise qui emploie l'auteur du code qui en détient les droits. Un copyright peut être vendu, comme bien immatériel, d'un tiers à un autre.

Le détenteur du copyright est libre de définir l'utilisation qui peut être faite de son programme comme :

- Interdire son utilisation
- Vendre ses droits à une personne physique ou morale
- Préciser les conditions d'utilisation de son programme. Ces conditions sont décrites dans les termes de la licence d'utilisation.

La diffusion du code source d'un projet n'en fait donc pas un projet Open Source. Ce sont les droits inscrits dans la licence qui rendent le projet Open Source ou non. Pour résumer, à la base de l'Open Source, il y a une licence et cette licence ne peut exister qu'à partir du droit d'auteur. Les notions de droit d'auteur et de licence sont inaliénables.

Celui qui désire utiliser le programme n'est pas libre contrairement à l'auteur. Il est lié par les termes de la licence qui stipule les conditions d'utilisation du programme. Si ces conditions ne sont pas respectées, il n'a pas le droit d'utiliser le programme.

2.1.2.2 Les clauses communes des licences

Toutes les licences Open Source ont un ensemble de clauses communes :

- L'identification claire du propriétaire du copyright du programme mais également pour les copies ou travaux dérivés.
- L'obligation de conserver la licence sur le programme.
- La protection de l'auteur vis-à-vis des utilisateurs de son programme, ses éventuels défauts et les conséquences de ces défauts.

Une licence ne peut être contraire au droit national. Cela implique qu'il faut respecter les lois nationales et la licence pour pouvoir distribuer le programme.

2.1.2.3 Copyleft

Une licence est dite copyleft si elle impose à toutes ses oeuvres dérivées d'être distribuées sous la même licence. C'est le caractère viral d'une licence car elle se transmet aux travaux dérivés. L'idée derrière une telle notion est que c'est du donnant-donnant. En effet, un programmeur ne peut pas distribuer sous une licence propriétaire un programme sous licence Open Source qu'il a amélioré.

La définition du terme "oeuvre dérivée" est cruciale et complexe car elle a un impact important sur le choix de licence et la réutilisation ou non de codes sources existants. Si un morceau de code d'un programme A est modifié ou que quelques lignes sont ajoutées pour obtenir un programme B, le programme B est une oeuvre dérivée. Si le programme B appelle des fonctions définies dans le programme A, le programme B est également une oeuvre dérivée. Par contre, si le programme B appelle des services du programme A au moyen d'un protocole d'échange standard, celui-ci n'est pas une oeuvre dérivée. La notion d'oeuvre dérivée peut être résumée comme suit : Un programme B est une oeuvre dérivée du programme A si B ne peut pas fonctionner sans A indépendamment des modalités techniques d'échange.

Le terme "distribué" demande également une précision. Si un programme est commercialisé en tant que logiciel, le programme est distribué. Mais si celui-ci est utilisé et déployé en interne dans une organisation, il n'est pas distribué. Cela implique qu'une organisation peut utiliser et modifier un projet Open Source en interne sans être tenue de diffuser son projet. Si un prestataire développe une application à l'aide de composants sous licence copyleft, le prestataire doit livrer l'ensemble du code source mais l'obligation de distribution du code source ne concerne que le client qui peut décider soit de garder le programme pour lui, soit de le distribuer sous la licence initiale.

Un autre cas juridiquement complexe est la cohabitation de programmes Open Source et propriétaires au sein d'un même projet. Si les deux types de programmes sont interfacés par des protocoles d'échanges standards, il n'y a pas de souci. Par contre, si les deux sont intégrés au sein d'un même programme, il y a des conséquences juridiques. Les deux licences doivent être respectées et compatibles sinon les deux programmes ne peuvent pas cohabiter en cas de diffusion du projet.

2.1.2.4 Brevets

Une autre notion du droit à la propriété intellectuelle est le brevet qui fournit à un inventeur le monopole sur son invention. En informatique, le brevet peut porter sur un procédé tel que, par exemple, le double clic sur un bouton de souris ou un algorithme. Cet outil juridique a pour mission de protéger durant un temps déterminé la propriété de l'inventeur et d'encourager l'innovation en la protégeant, en empêchant toute personne de reproduire cette invention. Si un programmeur développe de son côté un programme avec ses connaissances personnelles, il ne viole aucun copyright et il est titulaire des droits d'auteur de son code. Mais, il peut potentiellement violer un brevet. En effet, il est impossible pour un développeur de vérifier s'il enfreint un brevet à chaque ligne de code écrite contrairement au droit d'auteur. De ce point de vue, le brevet nuit au développement des projets informatiques car, en cas de violation d'un brevet, le développeur peut être poursuivi pénalement par le détenteur du brevet.

2.1.3 Les principales licences

Il existe de nombreuses licences Open Source avec des conditions complexes et parfois saugrenues dues aux différentes philosophies et situations rencontrées par les concepteurs de ces licences. Pour qu'une licence soit une licence Open Source, elle doit être reconnue par l'OSI. Les licences qui vont être étudiées, reprises dans le tableau 2.1, sont les licences courantes et standards régulièrement rencontrées par les informaticiens, comme l'illustre la figure 2.1[4]. La majorité des plugins et projets Open Source les plus courants se retrouvent sous ces catégories de licences. Deux grandes familles de licences peuvent être distinguées : la famille Berkeley Software Distribution (BSD) et la famille GNU GPL.

La famille de licence BSD[5] est la famille de licences la plus libérale et la moins contraignante, c'est-à-dire qu'elle autorise n'importe quelle utilisation du programme, de son code source et des travaux dérivés. Le code sous licence BSD peut être intégré à des logiciels sous licence non Open Source. Les licences MIT et Apache sont toutes les deux proches de la famille BSD. Cela implique que les développeurs peuvent utiliser les projets sous ces licences pour réaliser des logiciels propriétaires. Elles sont régulièrement utilisées pour la réalisation de plugins et de framework.

La famille de licences GNU General Public Licence[6] est une famille, par contre, très restrictive qui est fournie par la FSF. Cette famille de licences se caractérise par son article sur le copyleft qui stipule le droit de modifier le programme et de redistribuer les modifications à la condition que ce soit sous la même licence. Cela respecte la philosophie de liberté défendue par la FSF. Une telle condition divise évidemment l'ensemble des licences en licence General Public Licence (GPL)

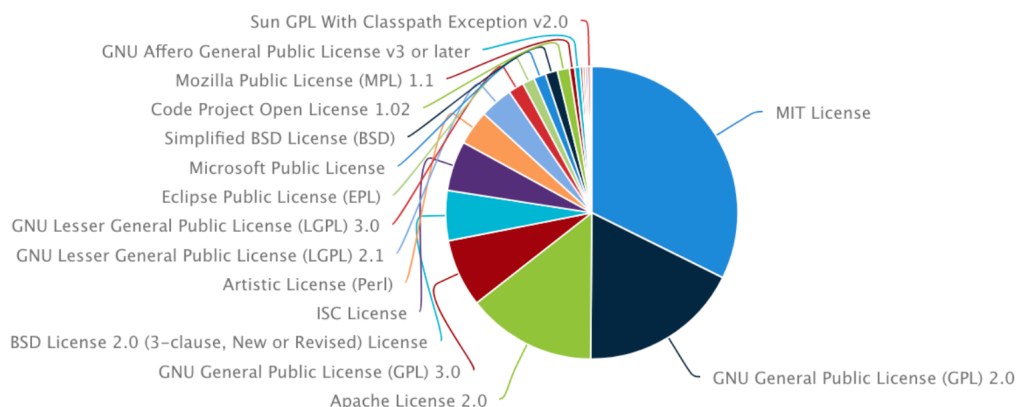


FIGURE 2.1 – Répartition des licences

et licence non GPL. En effet, si un développeur ne travaille pas sous la licence GPL, il n'a pas accès aux programmes sous licence GPL. La famille GNU GPL possède de nombreuses licences, les deux plus connues et répandues sont la GPL-v2 et la GPL-v3. La GPL-v3 a pour objectif d'améliorer la GPL-v2 en l'adaptant à l'évolution du domaine Open Source. Les améliorations les plus significatives sont :

- Une définition juridique plus stricte laissant moins de place à l'interprétation
- Une meilleure protection contre les revendications de brevets logiciels
- Plus de flexibilité pour résoudre les problèmes de compatibilité entre licences

Nom	Type	Informations
GNU GPL	Copyleft	— Licence populaire — Licence restrictive et contaminante
MIT	Non copyleft	— Licence permissive — Licence également nommée X11 — Licence académique
BSD	Non copyleft	— Licence permissive — Licence originale contenait une clause de publicité qui a été retirée
Apache	Non copyleft	— Initialement destinée aux projets Apache — Incompatible avec la GNU GPL-v2 — Compatible avec la GNU GPL-v3

TABLE 2.1 – Tableaux récapitulatifs des principales licences Open Source

2.1.3.1 Compatibilité des licences

Comme vu précédemment, le domaine des licences Open Source est fortement morcelé. Certaines licences sont compatibles entre elles, d'autres ne le sont pas car certaines de leurs clauses sont trop restrictives ou en inadéquation philosophique. Un projet sous licence plus restrictive peut intégrer des programmes avec des licences moins restrictives mais non l'inverse. Les licences de la famille GNU GPL sont les plus restrictives de par leur clause de copyleft qui impose au programmeur désireux de reprendre un programme sous cette licence de passer à celle-ci. Mais cette famille a l'avantage de permettre d'intégrer des nombreuses licences moins restrictives au projet sous sa licence. La figure 2.2 illustre les compatibilités possibles entre les licences les plus communes et standards. Une licence A est compatible avec une licence B si un programme avec la licence A peut intégrer un programme avec la licence B sans modifier les termes de sa licence. Si un programme C utilise un programme A sous licence LA et un programme B sous licence LB, le programme C héritera des exigences des licences LA et LB. Mais, s'il existe une contradiction entre les licences LA et LB, il faudra renoncer à utiliser un des programmes.

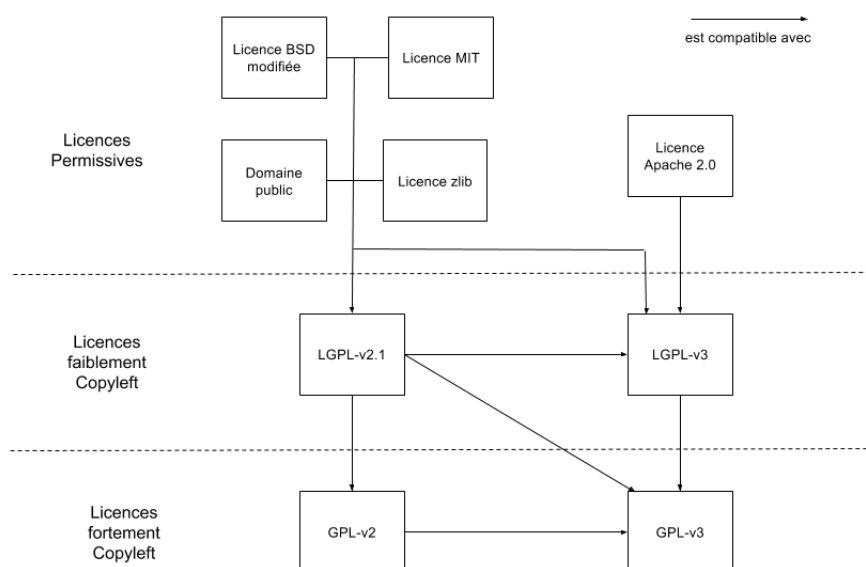


FIGURE 2.2 – Illustration des compatibilités entre les licences

2.1.4 Projet Open Source

Le cadre légal, conceptuel et philosophique étant posé, il faut comprendre les intérêts pour un projet d'être sous licence Open Source. Cette sous-section aborde donc les avantages à travailler en Open Source et les éléments clés pour une mise en place correcte d'un tel projet.

2.1.4.1 Avantages

Le développement Open Source possède de nombreux avantages qui ont permis sa propagation au sein d'entreprises multinationales ou PME, d'universités et des projets portés par des passionnés. Les avantages[7][8] détaillés ci-dessous sont à la fois des arguments pour utiliser et réutiliser des

projets Open Sources déjà existants ainsi que des arguments pour développer un projet Open Source.

Coopération/Mutualisation Le développement d'un projet en Open Source permet une coopération et un échange de savoir entre les différents développeurs. Ceux-ci peuvent mutualiser leurs connaissances en vue d'atteindre un même objectif. Un exemple d'une telle coopération est la réalisation d'un système d'exploitation par des grands constructeurs automobiles. Utiliser un système propriétaire serait trop coûteux et non adapté aux besoins spécifiques. En mettant leurs ressources en commun, ils ont pu développer un système d'exploitation performant et répondant à leurs besoins. En étant partie prenante du projet, les constructeurs pouvaient donner la direction qui les intéressait au projet et profiter des connaissances des autres constructeurs. Mais, un constructeur n'ayant pas participé au projet depuis son lancement devra franchir une barrière à l'entrée pour comprendre et s'approprier le projet afin de pouvoir y participer et y ajouter sa contribution. La concurrence entre les constructeurs automobiles participant au projet Open Source s'effectuera sous d'autres aspects.[9]

Pérennité Un projet Open Source de grande ampleur est soutenu par une communauté active d'informaticiens. Ce soutien assure une certaine pérennité au projet car, même si une technologie novatrice apparaît, l'ensemble de la communauté ne va pas abandonner l'ancienne technologie pour la nouvelle du jour au lendemain. Cela donne un temps pour s'adapter et effectuer le changement de technologie en douceur.

Flexibilité Un participant à un projet Open Source peut donner une nouvelle direction au projet en soumettant ses contributions au projet initial. Participer à un projet Open Source offre aussi une forme de flexibilité car si le programmeur considère que le projet prend une mauvaise direction, il peut décider de repartir d'une ancienne version du projet pour lui donner la direction désirée et amener avec lui d'autres contributeurs du projet initial et en devenir une nouvelle référence.

Sécurité En développant un projet Open Source, le développeur autorise ses pairs à regarder son travail, à le tester, à l'interroger sur les aspects sécuritaires. En mettant les connaissances de chacun en commun, des failles peuvent être détectées et corrigées. Ce point rend le programme plus robuste et plus sécurisé.

Qualité du code Le développement Open Source impose une certaine approche et rigueur aux développeurs afin de permettre à d'autres développeurs d'entrer facilement dans le projet. Cette rigueur a pour conséquence d'améliorer la qualité du code fourni. De plus, les critiques, remarques et discussions effectuées avec les pairs permettent d'améliorer constamment la qualité du code.

Aspect économique Les programmes Open Source sont généralement moins coûteux que les projets propriétaires. Un autre avantage des programmes Open Source pour une société est d'être moins liée par l'utilisation d'un programme Open Source que par l'utilisation d'un programme propriétaire. Lors d'un projet Open Source, la société peut donner son avis et une direction au programme, selon ses envies et besoins. Si elle utilise, par exemple, un programme propriétaire, la société devra faire une requête à la société responsable du programme pour qu'elle effectue les changements voulus ou intègre les changements effectués dans le noyau du logiciel. La coopération dans la réalisation d'un projet Open Source permet également de réduire les coûts en utilisant le savoir d'autres programmeurs qui ne sont pas employés par la société. Le développement de projet Open Source peut être un modèle économique viable pour de nombreuses sociétés qui créent un modèle de service et de soutien informatique aux sociétés demandeuses.

2.1.4.2 Éléments clés

Lors du lancement d'un projet Open Source, il ne suffit pas simplement d'avoir une idée révolutionnaire. Il faut aussi être attentif à différents éléments[10] pour permettre au projet Open Source de survivre et de se développer malgré l'existence de nombreux autres projets.

Communauté La communauté est l'ensemble des personnes qui participent à la vie du projet. La communauté au sens large est constituée des développeurs disposés à participer au développement du projet mais, également de l'utilisateur qui s'informe du projet sur des forums, qui répertorie les erreurs et qui, par ce biais, tend à améliorer la robustesse du projet, contribue à la réalisation de la documentation, et parle du projet autour de lui pour le faire connaître. Une communauté forte est indispensable pour faire vivre et perdurer un projet Open Source. Dès que la communauté abandonne un projet, celui-ci va dépérir car il ne sera plus actif, ni mis à jour régulièrement. Il ne répondra donc plus aux attentes des utilisateurs. Les projets Open Source les plus prestigieux attirent les meilleurs programmeurs. Par exemple, devenir commiter, programmeur autorisé à soumettre directement les modifications, sur le projet Linux est une consécration pour bon nombre de programmeurs.

Leader Sur un projet Open Source, des choix doivent être posés et des décisions prises. Un leader, un architecte ou un groupe restreint de programmeurs doivent être capable de prendre les décisions fondamentales pour le projet afin d'éviter que l'énergie consacrée au projet ne s'éparpille. Il faut également prendre des décisions sur l'architecture du projet afin de permettre à tous les programmeurs de travailler ensemble.

Modularité Pour que plusieurs développeurs puissent travailler efficacement sur un même projet, il faut un découpage du projet en modules avec des frontières nettes et identifiables ainsi que des modules de taille gérable. Diviser pour mieux régner est déjà un principe bien répandu en informatique mais, il est vital dans le cadre de projets Open Source. Il est donc nécessaire de découper le programme en modules indépendants possédant des interfaces bien définies. Ces interfaces permettent l'utilisation d'autres modules sans devoir nécessairement en comprendre le code. Les développeurs peuvent donc se concentrer exclusivement sur le développement de leur propre module.

Modèle de développement Les modèles de développement en cascade ou cycle en V ne sont pas propices aux projets communautaires. L'enchaînement linéaire de phases distinctes suppose une planification globale et une affectation centralisée des tâches. Ceci est irréalisable dans le cadre d'un projet Open Source, vu que toute la communauté peut apporter sa vision et sa pierre à l'édifice. Le modèle de développement communautaire prédominant est un modèle itératif où, à tout instant, un développeur peut donner naissance à une version nouvelle d'un module pouvant remplacer la version précédente si ce module est de qualité supérieure.

Forge Une forge est un système de gestion de développement collaboratif de logiciels. Elle permet de rassembler des développeurs autour d'un même projet. Pour ce faire, elle offre généralement différents services tels que :

- Un système de gestion des versions
- Un outil de communication entre développeurs
- Un outil de suivi des erreurs
- Un outil de gestion des tâches

Intégration continue Développer une série de tests permettant une intégration continue est crucial pour générer et contrôler, à chaque modification, l'ensemble de l'application afin d'identifier le plus en amont possible d'éventuelles régressions, erreurs ou incompatibilités entre modules. C'est un gage de qualité du projet qui augmente la confiance des utilisateurs.

Licence Le choix de la licence est primordial pour le projet. Ce choix a un impact direct sur les programmes qui peuvent être intégrés ou non au projet et sur la finalité de celui-ci.

L'ensemble des connaissances théoriques et pratiques sur le domaine de l'Open Source énoncé dans cette section a alimenté la discussion sur le choix de la licence pour le projet UCLCampus (cf. la sous-section 4.1.1 Choix d'une licence) ainsi qu'orienter nos décisions pour développer le projet avec une approche Open Source.

2.2 Ionic, Angular et Cordova

Le marché des systèmes d'exploitation mobile est dominé par Android qui possède plus de 75% des parts de marché, comme l'illustre la figure 2.3[11]. Les autres systèmes d'exploitation se répartissent un peu moins du quart restant avec IOS qui domine largement la meute des poursuivants. Les développeurs d'application ne peuvent pas se permettre de laisser sur le carreau un peu moins d'un quart de leur utilisateurs potentiels s'ils décident de développer pour Android. Par ailleurs, ils ne peuvent pas non plus développer une application spécifique à une plateforme "mineure" vu le faible pourcentage d'utilisateurs de celle-ci.



FIGURE 2.3 – Évolution des parts de marché des systèmes d'exploitation des smartphones

Dans ce contexte, les développeurs doivent être capables de développer et de maintenir des applications pour plusieurs plateformes en même temps pour toucher un maximum de clients. Le développement natif d'une application spécifique pour chaque plateforme mobile demande un investissement conséquent car il impose de connaître et de programmer sous différents langages de programmation : Objective-C pour IOS, Java pour Android. C'est la raison pour laquelle le développement cross platform, développement d'une application unique avec la même base de code pouvant s'exécuter indifféremment sur les plateformes présentes sur le marché, a connu un essor important. Dans ce domaine, les outils de développement et frameworks Open Source proposent des solutions pertinentes. Ces outils peuvent être distingués en deux catégories :

Application native générée Les outils générant à partir d'un développement unique des applications natives pour différentes plateformes. Par exemple, l'outil Titanium[12] transforme une application écrite en JavaScript en application native. Pour ce faire, le code JavaScript est compilé en bytecode et ensuite interprété par le moteur d'exécution JavaScript embarqué dans l'application. Le code JavaScript est interprété par le mobile mais l'exécution des opérations natives est déléguée aux parties compilées du framework.

Application hybride Les outils de développement manipulant la plateforme HTML5[13] tel que, par exemple, Cordova[14]. Outre la portabilité et le respect des standards, le HTML5

a l'avantage de rapprocher le développement web et le développement d'applications. Une application hybride est une application mobile qui contient une instance de navigateur isolé, appelée `WebView`, pour exécuter une application web dans une application native. Ce type d'application utilise un wrapper d'application native qui peut communiquer entre la plateforme native et la `WebView`. Cela signifie que l'application peut tourner sur un mobile et avoir accès aux fonctionnalités natives de celui-ci.

L'application UCLCampus est une application multiplateforme hybride. Son développement est réalisé à l'aide du framework Ionic. Le projet UCLCampus a suivi l'évolution d'Ionic et est, actuellement, développé avec la version 3 du framework.

Ci-après se trouve, une analyse du fonctionnement du framework Ionic suivie d'une description séparée des différentes technologies : Angular, TypeScript, Cordova, HTML5, qui composent le framework Ionic. Et finalement, les services supplémentaires proposés par Ionic, afin de faciliter le développement et le déploiement d'applications, sont présentés.

2.2.1 Ionic

Ionic[15][16] est un Software Development Kit (SDK) sous licence MIT. L'objectif d'Ionic[17] est de faciliter le développement d'applications mobiles avec HTML5, c'est-à-dire d'applications hybrides. Ionic vise à combler l'écart entre HTML5 et le développement d'applications natives. Ionic se concentre principalement sur l'apparence et l'interaction avec l'interface utilisateur d'une application. Cela signifie qu'il ne vise pas à remplacer PhoneGap ou d'autre framework JavaScript. Il vise à simplifier le développement d'application pour la partie front-end. Mais Ionic n'est pas seulement un framework avec un balisage bien documenté et du Cascading Style Sheet (CSS), c'est aussi un modèle de conception JavaScript permettant la réalisation d'applications plus complexes similaires à des applications natives IOS et Android. De ce fait, Ionic nécessite l'utilisation des frameworks Angular et Cordova. La partie web et interface utilisateur sont gérées grâce au framework Angular et les fonctionnalités natives de l'application sont exécutées à l'aide des plugins du framework Cordova. La figure 2.4[18] illustre les interactions entre les différents composants du framework Ionic. La `WebView`, qui est une instance d'un navigateur, affiche l'interface utilisateur codée à l'aide d'Angular et utilise un wrapper à travers le framework Cordova pour utiliser les fonctionnalités natives de la plateforme.

2.2.2 Angular

Angular[19][20] est un projet Open Source, sous licence MIT, de développement d'application web axé front-end et basé sur le langage TypeScript. Ce projet est dirigé par une équipe de Google et une communauté. Le framework Angular utilise une architecture basée sur les composants. La figure 2.4[18] montre qu'Angular manipule trois technologies de développement d'applications web : le langage TypeScript, HTML5 et CSS. Ces trois technologies sont brièvement présentées pour mieux comprendre le fonctionnement interne du framework Angular.

2.2.2.1 TypeScript

TypeScript[21][22] est un langage de programmation Open Source développé par Microsoft qui a pour objectif d'améliorer et faciliter la production de code JavaScript. Le code source TypeScript est compilé en code source JavaScript, ce qui lui permet d'être interprété par n'importe quel navigateur web ou moteur JavaScript. Les apports les plus importants de TypeScript par rapport à JavaScript sont les suivants :

- Ajout de la notion de classe
- Supporte la programmation orientée objet standard
- Le typage se fait statiquement en TypeScript contrairement à JavaScript qui a un typage dynamique

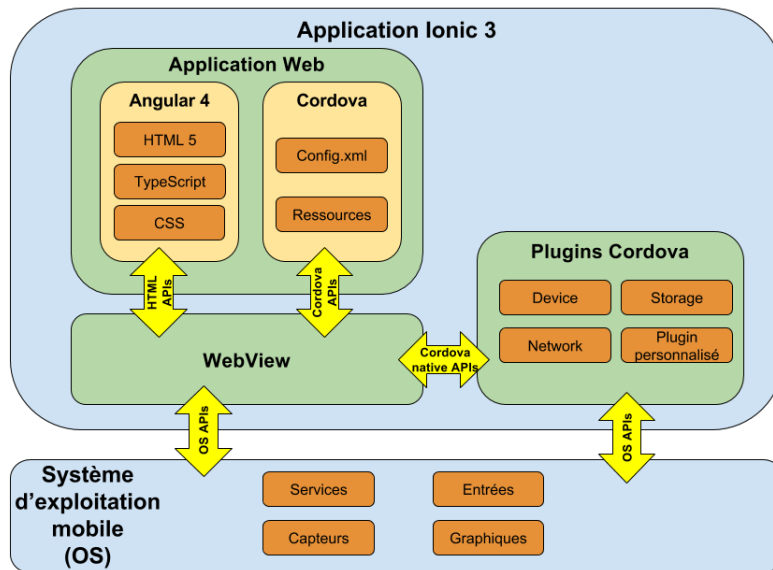


FIGURE 2.4 – Représentation de l'architecture d'une application avec le framework Ionic 3

Ces ajouts au langage JavaScript permettent aux développeurs d'applications web de mieux structurer leur code et d'avoir un meilleur retour sur les erreurs de codage grâce au typage statique.

JavaScript est le langage de script utilisé par tous les navigateurs web et permettant les interactions avec les serveurs. Dans les navigateurs, ce sont les scripts qui permettent de gérer les interactions et les événements se produisant sur la page web. JavaScript est un langage orienté objet de prototypage, c'est-à-dire que les objets ne sont pas des instances de classes mais possèdent leur propre constructeur et leurs propres propriétés.

Le DOM[23] est une interface de programmation normalisée par l'organisme World Wide Web Consortium (W3C) [24] qui permet à des scripts d'examiner et de modifier le contenu d'un navigateur web. Le W3C définit les objets, les méthodes et les propriétés dont un navigateur web doit disposer. Le DOM permet la représentation d'un document HyperText Markup Language (HTML) ou Extensible Markup Language (XML) sous forme d'objets reliés en structure d'arbre. Il y a trois grandes catégories d'objet DOM :

Document Informations concernant l'ensemble du document HTML compris dans la balise `<html>`

Élément Balises internes au document HTML pouvant posséder des attributs. Les éléments possèdent également des méthodes permettant d'accéder aux différents éléments de la structure en arbre du DOM.

Texte Contient le texte qui se trouve à l'intérieur d'une balise.

Chaque navigateur implémente, dans son propre langage de programmation, les normes du DOM. Ces spécifications sont alignées avec le langage JavaScript qui est reconnu par les navigateurs web. Cela permet à n'importe quelle partie d'une page web d'être modifiée par un programme JavaScript et au programme JavaScript de fonctionner sur n'importe quel navigateur web conforme aux normes de DOM.

2.2.2.2 HTML5

C'est la version du langage de balisage, finalisée en 2014, permettant de représenter des pages web sous le format de données HyperText Markup Language (HTML). Ce langage permet de structurer sémantiquement et logiquement le contenu des pages. HTML5 apporte une nouvelle sémantique pour une meilleure structuration, de nouveaux attributs, éléments graphiques et multimédias ainsi que des nouvelles Application Programming Interface (API) permettant le drag and drop, la géolocalisation, etc. De plus, il est doté d'un algorithme permettant de traiter des documents avec une syntaxe non conforme. HTML est conçu pour que ces documents soient interopérables, c'est-à-dire qu'ils puissent être supportés par le plus grand nombre de systèmes, appareils existants. HTML ne s'attache pas au rendu visuel final d'un document vu qu'il peut être consulté à l'aide de logiciels différents mais vise à donner du sens au document grâce aux différentes balises.

2.2.2.3 CSS

Le CSS, ou feuille de style, est un langage qui décrit la présentation visuelle des documents HTML et XML. Il se consacre exclusivement à la mise en forme du document et au rendu visuel contrairement au HTML qui s'occupe de la structure du document. Ces standards sont définis par la W3C.

2.2.2.4 Fonctionnement

La philosophie du framework Angular[25] est de considérer une application comme un ensemble de modules indépendants reliés, où chaque module représente une fonctionnalité de l'application. De ce fait, l'architecture utilisée par le framework est une architecture basée sur les modules. Angular propose de travailler avec des NgModule pour structurer l'application par fonctionnalité. Un module Angular ou NgModule est un bloc de code cohérent dédié à un domaine de l'application. Toutes les applications Angular sont composées d'au moins un module, le module racine, ensuite le développeur peut ajouter d'autres modules pour rendre l'application plus complexe. Un NgModule utilise une fonction décorateur qui prend les métadonnées du module. Ces métadonnées décrivent les propriétés du module. Une classe d'un module Angular est modifiée en attachant les métadonnées afin de connaître la signification et la fonction de la classe. Les propriétés définies par les métadonnées sont :

- declarations : Les classes Vues qui appartiennent au module.
- exports : Le sous-ensemble des déclarations qui doivent être visibles et utilisables dans d'autres modules.
- imports : Les modules dont la classe a besoin pour fonctionner.
- providers : Créateurs de services auquel le module contribue.
- bootstrap : La vue principale de l'application, appelée composant racine, qui héberge toutes les vues de l'application. Seul le module racine doit définir cette propriété.

Le code suivant illustre l'utilisation de la fonction décorateur de NgModule pour une classe AppModule :

```
@NgModule({
  imports:      [ BrowserModule ],
  providers:    [ Logger ],
  declarations: [ AppComponent ],
  exports:      [ AppComponent ],
  bootstrap:    [ AppComponent ]
})
export class AppModule { }
```

Les modules définis peuvent également être ajoutés au sein d'autres modules à l'aide de l'instruction :

```
import {Module} from 'path'
```

Angular possède une librairie de modules réutilisables mais les développeurs sont invités à développer leurs propres modules.

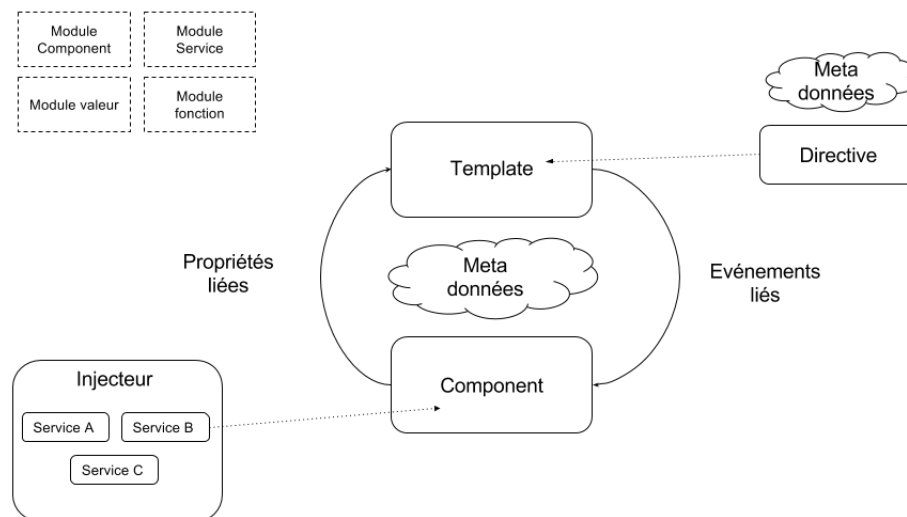


FIGURE 2.5 – Structure d'un module

La figure 2.5[25] illustre les interactions entre les différents éléments, modules pouvant interagir au sein d'un module Angular. Prenons le temps de détailler le rôle et le fonctionnement de chacun.

Composant Les composants contrôlent une partie de l'écran appelée Vue. Ils définissent le comportement de la Vue en fonction des événements se produisant sur celle-ci. Ils interagissent avec la Vue à travers une API de propriétés et de méthodes. Angular crée, met à jour et détruit les composants à mesure que l'utilisateur se déplace dans l'application. Les composants sont écrits en TypeScript. Le travail des composants consiste à permettre l'interaction et l'expérience de l'utilisateur, rien de plus. Les composants interviennent entre la Vue (implémentée par le template) et la logique de l'application qui comprend les modèles. Un composant standard présente les propriétés et les méthodes de liaisons de données avec la Vue, il délègue tout ce qui est plus complexe aux services. Les composants utilisent le décorateur `@Component` pour être identifiés et identifier leurs propriétés.

Template Définit la Vue d'un composant. Un template est codé à l'aide du langage HTML standard et d'autres fonctionnalités. Le HTML standard se mêle parfaitement avec les balises personnalisées et implémentées sous forme de modules comme l'illustre la figure 2.6[25].

Métadonnées Les métadonnées sont ajoutées pour indiquer comment doit être traitée une classe, quelles sont ses propriétés et son rôle. En TypeScript, les métadonnées sont attachées aux classes en utilisant le patron de conception du décorateur.

Data Binding Sans l'utilisation du framework Angular, les développeurs sont responsables de l'affichage des valeurs des données sur le contrôleur HTML et de la transformation des réponses, événements produits par l'utilisateur, en actions qui mettent à jour la partie logique. L'écriture d'un code spécifique en réponse à chaque événement est fastidieux, c'est pour cela que le framework Angular en est responsable. Le programme ajoute des marqueurs dans le template HTML pour indiquer les propriétés des balises. Ces marqueurs indiquent à Angular comment connecter le DOM et le composant responsable du HTML. La figure 2.7[25] illustre la syntaxe des différentes interactions entre le DOM et un composant.

- La première liaison est une interpolation qui affiche une valeur spécifique "value" du composant.
- Le second type de liaison est une liaison de propriété qui passe de la valeur "value" à la propriété "property".
- La troisième liaison d'événement appelle la méthode "handler" du composant lié lorsque l'événement "event" se produit.
- La quatrième liaison est le data binding qui combine les propriétés et les événements de manière bidirectionnelle. Les modifications effectuées par l'utilisateur sont directement répercutées sur le composant. Les changements survenus lors de la mise à jour du composant seront directement affichés sur le document HTML.

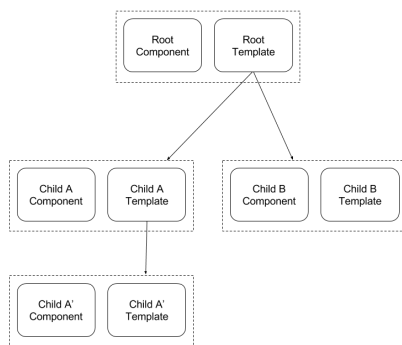


FIGURE 2.6 – Utilisation de modules personnalisés au sein d'un template

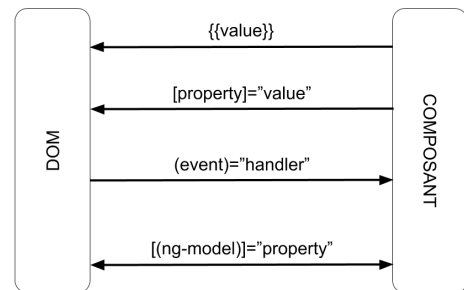


FIGURE 2.7 – Interactions entre le DOM et un composant

Directives Les templates Angular sont dynamiques. Lorsque Angular les interprète, il transforme le DOM selon les instructions données par les directives. Il y a deux types de directives :

Structurelle Directive qui modifie la mise en page en ajoutant, en supprimant ou en remplaçant des éléments du DOM. Angular possède des directives structurelles telles que `*ngFor`, `*ngIf`.

Attribute Directive qui modifie l'apparence ou le comportement d'un élément du DOM. Dans le template HTML, il ressemble à des attributs HTML réguliers tel que `ngModel`. Les directives sont des classes utilisant le décorateur `@Directive`. Un décorateur `@Component` est un décorateur `@Directive` qui est étendu et spécialisé avec des fonctions orientées template.

Services Les services sont un concept assez large qui englobe toutes les valeurs, méthodes, fonctionnalités dont une application a besoin. Les services représentent la partie logique et modèle de l'application. Ils s'occupent de récupérer les données du serveur, de la

base de données, valident les valeurs entrées par l'utilisateur lors d'une tentative de connexion, gèrent des calculs plus complexes, mettent à jour les modèles de données, etc. L'architecture d'Angular étant basée sur les composants, chaque service est une classe avec un but simple et bien défini. Les services étant indépendants, ils peuvent se combiner pour offrir des fonctionnalités plus complexes. Ce sont les composants qui font généralement appels aux services pour gérer la partie logique des événements perçus sur l'interface utilisateur.

Injection de dépendances C'est la manière dont Angular fournit une nouvelle instance de classe avec les dépendances nécessaires à son fonctionnement. La plupart des dépendances sont des services. Cela permet de fournir les instances d'un composant et les services qui lui sont liés. Comme l'illustre la figure 2.8[25], Angular connaît les services utilisés par le composant grâce à son constructeur. Les providers sont les éléments d'Angular qui fournissent et gèrent les services. L'enregistrement d'un provider au niveau d'un composant signifie qu'un nouveau service est instancié lors de chaque instanciation du composant, ce qui réduit les performances de l'application. La bonne pratique est d'ajouter les providers au module racine pour permettre l'accès à la même instance du service dans toutes les pages de l'application.

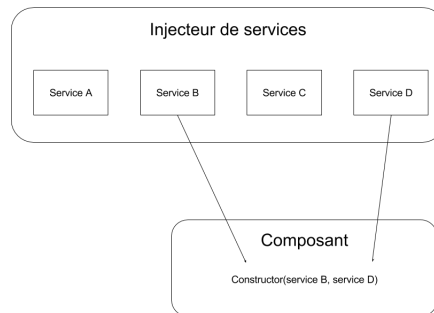


FIGURE 2.8 – Instanciation d'un composant avec services

2.2.3 Cordova

Apache Cordova est un framework de développement mobile. Il permet d'exploiter les technologies web courantes telles que HTML5, CSS et JavaScript, pour développer des applications cross platform hybrides. Cordova est le descendant de PhoneGap. Le projet Cordova est développé en Open Source par la fondation Apache sous la licence Apache. Cordova utilise la WebView, fenêtre de navigateur web, spécifique à chaque plateforme pour accéder aux API des fonctionnalités de la plateforme en question tout en ne manipulant que du code JavaScript. Pour faire correspondre les différentes interfaces, Cordova utilise le principe du patron de conception du wrapper ou de l'adaptateur. Cordova utilise des plugins natifs, permettant aux développeurs d'accéder à des fonctionnalités natives pouvant être appelées à partir de JavaScript, ce qui permet une communication entre la couche native et la page HTML5. Ces plugins permettent d'accéder à l'accéléromètre, à la caméra, à la boussole, au système de fichiers, au microphone et plus encore.

Comme expliqué précédemment, Cordova permet de créer une application qui ouvre une WebView disponible nativement sur les appareils mobiles. Une WebView est une fenêtre de navigateur web qui interprète et affiche le contenu des codes HTML5 et JavaScript. Les WebView, dont les possibilités sont étendues par Cordova, permettent d'accéder à un certain nombre de fonctionnalités natives de l'appareil mobile. Le rôle de Cordova est de faire correspondre les codes écrits en JavaScript pour le fonctionnement de l'application web avec les API de fonctionnalités

spécifiques à chaque appareil natif au travers d'une WebView. Vu que chaque WebView est spécifique à chaque appareil, elle a accès à l'interface des fonctionnalités natives de la plateforme mobile. Le framework Cordova ne compile pas les codes HTML et JavaScript, il ajoute une couche supplémentaire pour faire correspondre les langages entre eux. Pour ce faire, Cordova utilise le principe du patron de conception Adaptateur.

Patron de conception Adaptateur Le patron de conception Adaptateur, ou wrapper, est un patron de conception de type structure qui permet de résoudre le problème de réutilisation de fonctionnalités d'un composant extérieur à un logiciel qui n'est pas compatible avec la philosophie et l'architecture du logiciel. Pour résoudre ce problème, le patron de conception permet de convertir, envelopper l'interface d'une classe existante en une autre interface compatible et attendue par la classe cliente pour permettre aux deux classes de travailler ensemble. L'adaptateur consiste à créer une abstraction intermédiaire qui traduit, qui lie un composant existant avec le nouveau système. Ce problème d'adaptation peut être résolu à l'aide de l'une des deux architectures suivantes :

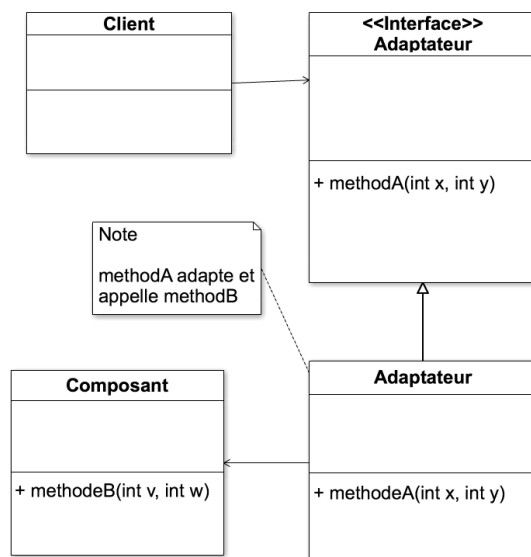


FIGURE 2.9 – Patron de conception adaptateur avec héritage

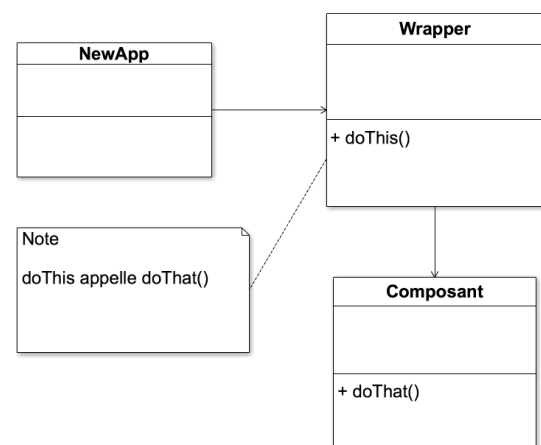


FIGURE 2.10 – Patron de conception avec agrégation

La figure 2.9 illustre une implémentation du patron de conception par l'utilisation de l'héritage. La classe Client peut faire appel à methodeB() via methodeA() malgré le fait que les paramètres des méthodes possèdent des spécifications différentes. L'implémentation de la méthode methodeB de la classe Adaptateur adapte ses paramètres à ceux de la méthode methodeA pour pouvoir l'appeler.

La seconde illustration, figure 2.10, représente l'implémentation du patron de conception en manipulant l'agrégation. La classe Wrapper permet d'envelopper la classe Composant et de permettre à la classe NewApp de faire appel à la méthode doThat() via la méthode doThis().

Limites Apache Cordova est limité au niveau performance. Les applications développées avec Cordova ne sont pas aussi fluides que les applications natives à cause de l'utilisation de la technologie web[26].

2.2.4 Services proposés par Ionic

En plus d'être un framework permettant le développement d'application hybride, Ionic fournit un ensemble d'outils et de services[27] pour accompagner les développeurs jusqu'au déploiement final de l'application. Les principaux services de la constellation Ionic sont présentés afin de mieux comprendre l'environnement existant autour du framework.

CLI Commande Line Interface (CLI) est le service principal du framework qui permet de générer automatiquement des nouveaux projets, pages, services. C'est le moyen d'interaction privilégié avec le framework Ionic et son environnement. Elle a subi une importante mise à jour lors du passage à la version 3 d'Ionic.

View App Service permettant de partager une application avec des utilisateurs en vue d'un test. L'application développée tourne à l'intérieur de l'application Ionic View ce qui permet aux testeurs de donner directement leur avis en navigant sur l'application.

Auth Service permettant l'authentification des utilisateurs avec une adresse mail et son mot de passe, ou par les identifiants des comptes des réseaux sociaux tels que Twitter, Facebook et d'autres systèmes d'authentification existants.

Ionic Deploy Service permettant la gestion des mises à jour des différentes versions de l'application.

Ionic Push Service permettant la création et gestion de notifications ciblées, via une interface graphique, qui peuvent être envoyés à une date précise ou lorsqu'un utilisateur de l'application satisfait à des critères spécifiques. Propose également une API pour fournir des notifications à partir de ses propres serveurs.

Ionic Package Service permettant de générer des versions natives de développement ou de production de l'application qui peuvent être déposées sur l'Apple Store ou le Google Play Store.

2.3 Entreprise Service Bus

Une organisation aussi importante que l'Université catholique de Louvain possède de nombreux services informatiques pour automatiser un maximum de tâches. La figure 2.11[28] illustre l'infrastructure informatique de la gestion de l'université. Cette section analyse le système d'interaction entre les principaux services et outils qui ont un impact sur le développement du projet UCLCampus. L'application UCLCampus doit interagir avec l'environnement qui l'entoure et avec les services de l'UCL. Dans le but d'harmoniser les interactions entre les différentes applications, l'UCL a investi dans l'outil WSO2.

Tout d'abord, le concept d'Entreprise Service Bus (ESB) est détaillé pour comprendre l'intérêt de son utilisation et, ensuite l'ESB WSO2 sélectionné par l'UCL est présenté. La dernière partie explique sa mise en oeuvre à l'UCL et ses impacts.

2.3.1 Concept

David A.Chappell, dans le livre "Entreprise Service Bus", définit l'ESB comme suit : "Un ESB est une plateforme d'intégration basée sur des normes combinant l'échange de messages, les services Web, la transformation de données et le routage intelligent pour connecter et coordonner de manière fiable l'interaction d'un nombre important d'applications diverses dans les entreprises étendues." [29] Tous les ESB possèdent des caractéristiques fondamentales et communes[30] :

Généralisation Un ESB doit être adaptable aux besoins des projets d'intégration générale dans toutes sortes de situations.

Connectivité et communication Un ESB doit permettre de communiquer avec de multiples protocoles et formats basés sur des standards reconnus.

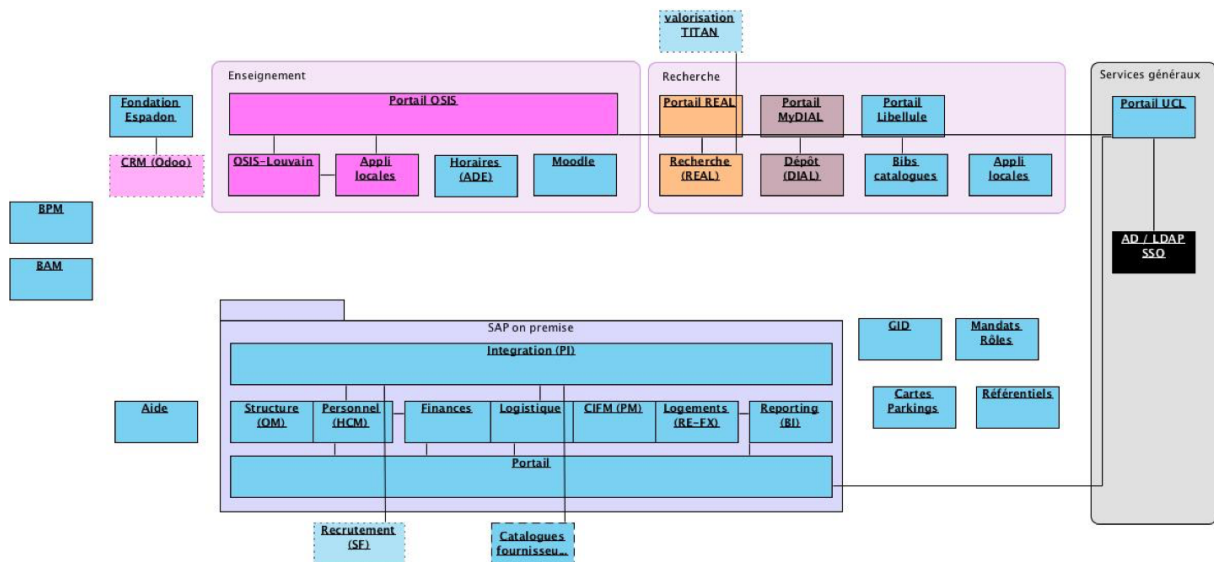


FIGURE 2.11 – Infrastructure informatique des applications de gestion de l’UCL

Routage Un ESB doit permettre de découpler fournisseurs et consommateurs de données. Il doit ainsi mettre à disposition un mécanisme permettant de distribuer les messages depuis un fournisseur vers un ou des consommateurs de manière transparente pour l’un et l’autre.

Transformation Un fournisseur et un consommateur ne dialoguant pas forcément dans un même langage, un ESB doit mettre à disposition des moyens de transformation des messages.

Orchestration Un ESB doit permettre aux données de circuler sur toutes les applications et services qui sont branchés sur le bus, qu’ils soient locaux ou distants.

Sécurité et fiabilité Tous les composants qui communiquent via le bus peuvent profiter des échanges de messages fiables, de l’intégrité des transactions et des communications sécurisées et authentifiées.

Pour résumer, un ESB est une plateforme qui permet l’interopérabilité d’environnements hétérogènes en utilisant une architecture orientée service, Service Oriented Architecture (SOA). Il permet d’abstraire un service et de l’isoler de son protocole de communication, de ses méthodes d’invocation, de ses standards d’échange de messages et de ses autres besoins d’infrastructure. Cela permet de coordonner et d’intégrer l’ensemble des services hétérogènes au travers d’un bus de messages et de faciliter l’utilisation de web services et de standards. Selon le site Smile[31], pour réaliser une telle tâche, un ESB est généralement composé d’un :

- framework d’intégration, qui réalise les opérations de médiation sur les flux de messages.
- framework de services, qui permet de concevoir les web services qui pourront être utilisés par les applications du système d’information.
- système de gestion de composants, qui permet de gérer dynamiquement l’installation, le démarrage, l’arrêt, la désinstallation des composants.
- serveur d’application qui gère toutes les connexions.
- Middleware Orienté Message, qui permet de gérer des files d’attente de messages.
- registre, qui permet de stocker et gérer des informations contextuelles nécessaires aux composants pour fonctionner.

2.3.2 WSO2

WSO2 ESB¹ est basé sur des composants sous licence Open Source Apache 2. Le projet WSO ESB est basé sur l'ESB Apache Synapse de la fondation Apache qui fournit deux frameworks : le framework d'intégration Apache Synapse et le framework de service Apache Axis2.

WSO2 ESB fait partie intégrante d'une plateforme middleware plus globale orientée SOA fournie par la société WSO2². Tous les produits de la plateforme sont basés sur le framework WSO2 Carbon. Le framework Carbon possède une importante bibliothèque de composants comprenant des composants de base tels que la gestion des logs, la gestion du déploiement des configurations ou l'interface d'administration. WSO2 possède également une série de produits dédiés au mobile.

Plus concrètement, WSO2 ESB permet d'exposer des web-services en Simple Object Access Protocol (SOAP) et en Representational State Transfer (REST) grâce à Apache Axis 2 et il supporte de nombreux protocoles et formats. WSO2 ESB repose sur Synapse pour la médiation et le routage. Apache Synapse met à disposition une quarantaine de composants de médiation utilisables. WSO2 ESB offre la possibilité de sécuriser les flux. Il met à disposition des médiateurs Oauth2 et Lightweight Directory Access Protocol (LDAP) pour authentifier les messages dans un autre format.

2.3.3 WSO2 à l'UCL

La situation initiale illustrée par la figure 2.12[28] expose la présence d'un fort couplage entre les différentes entités et la complexité des interactions entre les différentes applications des services de l'UCL. Les interactions vont dans tous les sens. L'instauration d'un ESB au sein des services informatiques de l'université apporte différents gains :

Découplage Chaque application connaît uniquement l'ESB.

Centralisation L'ESB sera l'élément central des services ce qui permet une meilleure transparence, visibilité des services disponibles ainsi qu'une normalisation des procédures.

Autorisations L'accès aux services se fera via des API ou des applications clientes de l'ESB.

La mise en place et le déplacement vers l'ESB WSO2 sélectionné par l'UCL se fait progressivement pour éviter une paralysie du système informatique en cas de problèmes techniques. La situation actuelle est illustrée par la figure 2.13[28]. Si l'UCL a choisi WSO2 c'est parce qu'il répond aux cahiers des charges d'un ESB et que c'est un projet Open Source qui a des références solides (en étant largement utilisé par des compagnies internationales).

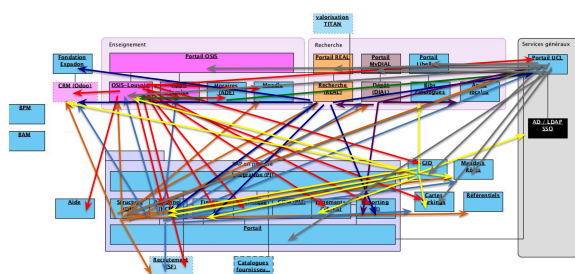


FIGURE 2.12 – Situation initiale des interactions entre les applications de l'UCL

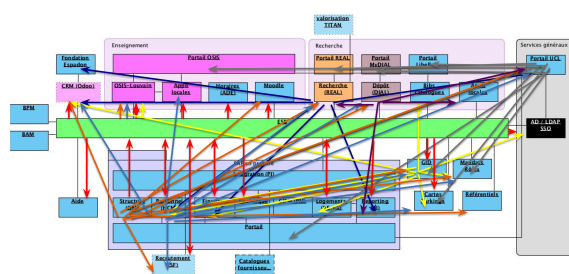


FIGURE 2.13 – Situation actuelle des interactions entre les applications de l'UCL

1. <http://wso2.com/products/enterprise-service-bus/>
2. <http://wso2.com>

Le système actuel est en pleine évolution, il tend à normaliser les interactions et atteindre l'état présenté par la figure 2.14[28].

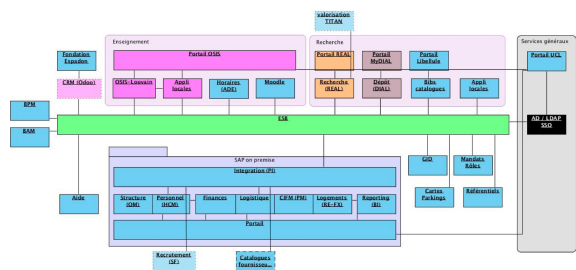


FIGURE 2.14 – Situation finale des interactions entre les applications de l’UCL

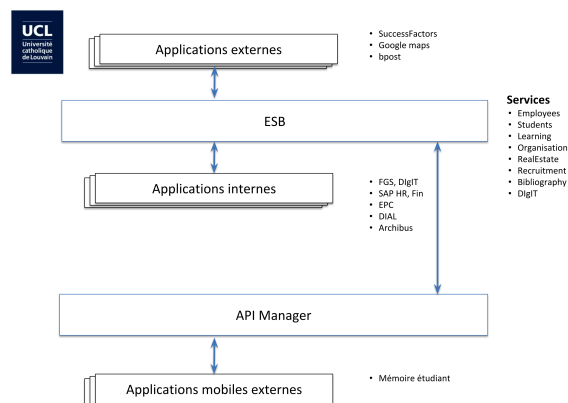


FIGURE 2.15 – Abstraction des interactions entre les applications de l’UCL

La figure 2.15[28] est une abstraction du fonctionnement du système de l’UCL. L’ESB regroupe tous les services disponibles au sein de l’UCL. Ces services sont accessibles par des applications internes et externes du système moyennant une intégration et une autorisation d’accès à l’ESB ou via un API manager. C’est par cet API manager que l’application mobile UCLCampus peut accéder aux différents services de l’UCL en fonction des API disponibles. L’ESB WSO2 permet également de visualiser et de gérer les différentes API disponibles à travers une interface graphique.

2.4 Portail et Learning Management System

L'Université catholique de Louvain gère un portail et un Learning Management System (LMS). Le premier est le portail principal, accessible à tout public, recensant l'ensemble des informations concernant la vie de l'université. Le portail est avant tout un outil de communication, mais également, un outil permettant aux membres de l'université d'accéder et de mettre à jour leurs informations personnelles. Moodle est un LMS diffusant et organisant les supports de cours de l'université. Les deux sous-sections ci-après présentent l'apport et les fonctionnalités de ces deux outils et un aperçu des caractéristiques techniques de leur fonctionnement.

2.4.1 Portail

Le portail³ de l'UCL est la vitrine de l'Université catholique de Louvain sur internet. Il contient une quantité astronomique d'informations. Chaque faculté, chaque service de l'UCL possède ses propres pages et contenus sur le portail. De plus, les étudiants et les membres du personnel de l'université peuvent accéder et mettre à jour leurs données personnelles depuis leur compte UCL accessible via le portail UCL. Il est très difficile de gérer, structurer, organiser et mettre à jour une telle quantité de contenus et d'informations. L'université utilise un système de gestion de contenu pour l'aider dans cette tâche.

Au cours de l'année académique 2016-2017, le site web de l'UCL a migré du Content Management System (CMS) Reddot vers Drupal⁴.

3. <https://uclouvain.be>

4. <https://www.drupal.org/docs/8/understanding-drupal-8/overview>

Un système de gestion de contenu est un outil logiciel destiné à la conception, édition et mise à jour dynamique du contenu d'un site internet. Les systèmes de gestion de contenu possèdent des caractéristiques et fonctionnalités communes[32] :

- Structurer le contenu : donner un sens au contenu en le structurant. Une structure générique serait : titre, sous-titre, accroche, corps de texte, ...
- Manipuler le contenu : gestion du contenu à travers sa modification, son enrichissement, sa publication.
- Exploiter le contenu : consiste à produire des pages pour un média donné en fonction du contenu.
- Séparation des opérations de gestion de forme et de contenu
- Politique de sécurité : l'identification unique Single Sign On (SSO) permet d'accéder à l'application sans devoir saisir à plusieurs reprises son identifiant et mot de passe.
- Attribution de rôles et de permissions aux différents utilisateurs.

La grande valeur ajoutée de l'utilisation d'un système de gestion de contenu est que, grâce à la séparation du contenu et de la présentation, il est possible pour des personnes non initiées aux bases de l'informatique d'écrire du contenu pour le site. Généralement, les systèmes de gestion de contenu propose une interface graphique de mise en forme pour simplifier l'ajout de contenu.

Drupal, système de gestion de contenu, est développé sous licence Open Source GNU GPL-v2 et écrit en langage PHP. Drupal se base sur une communauté importante et active qui développe de nouveaux modules ajoutant de nouvelles fonctionnalités utiles pour les utilisateurs de ce système. Drupal est très flexible et peut être utilisé de différentes manières :

- Standard : C'est une utilisation clé sur porte. Une fois installé et configuré, il est directement utilisable pour créer du contenu structuré.
- Personnalisé : L'affichage des menus et des outils Drupal peut être personnalisé pour offrir une meilleure expérience aux éditeurs de contenu.
- Extension : Drupal permet l'ajout de modules externes au noyau Drupal. Ces modules externes sont fournis par la communauté entourant Drupal ou en ajoutant des modules développés en interne.

L'Université catholique de Louvain a développé son nouveau portail avec le système de gestion de contenu Drupal qui offre de nombreux services permettant une meilleure gestion du contenu présent sur le portail. Les principaux services utilisés sont :

- La gestion des droits d'édition. Seul, un éditeur bénéficie des droits d'édition sur les pages du site. Les droits d'édition sont régulés au travers de groupes dont la gestion est assurée par l'Administration des relations extérieures et de la communication (AREC). En fonction du groupe dans lequel l'éditeur se trouve, il peut éditer certaines pages. Une telle approche permet de définir les éditeurs par service et de demander à chaque service de gérer ses propres pages du site.
- Drupal offre une interface d'édition personnalisable qui facilite l'édition de contenu par des éditeurs non experts en informatique. Un manuel d'utilisation du CMS-Drupal-UCL est disponible pour les éditeurs. Les éditeurs ont également accès à de nombreuses fonctions telles que la gestion des versions d'une page, le déplacement d'une page vers un autre site, le positionnement d'une page dans le site, la programmation de la publication d'une page, la modification de l'URL d'une page.
- L'utilisation de l'interface de Drupal permet une meilleure cohérence dans le visuel des différentes pages.

2.4.2 Moodle

Moodle⁵ est une plateforme d'apprentissage en ligne destinée à fournir aux enseignants, administrateurs et apprenants, un système pour créer des environnements d'apprentissages

5. <https://moodle.org>

personnalisés. Moodle est développé sous la licence GNU GPL-v3. La plateforme est actuellement à la version 3.3. Elle est écrite en JavaScript et PHP. Moodle est supporté et utilisé par une communauté active et de nombreuses institutions scolaires. Moodle étant une plateforme modulaire et personnalisable, la communauté développe de nombreux plugins pour y ajouter de nouvelles fonctionnalités.

En plus d'être un système de gestion du contenu pour les cours, Moodle offre un environnement propice à l'apprentissage en ligne avec de nombreux outils interactifs et pédagogiques ainsi qu'un moyen de communication. Voici quelques exemples de fonctionnalités disponibles :

- La création de cours à l'aide d'outils intégrés tels que des ressources multimédias, quizz, remise de travaux, évaluation
- Une organisation personnalisable des cours sous forme de filière
- Un calendrier recensant les événements spécifiques aux cours
- Une application mobile
- Service de messagerie

Par son approche très modulaire, Moodle incite chaque établissement scolaire à personnaliser sa plateforme en fonction de ses besoins, à l'aide des plugins mis à disposition par la communauté ou en développant ses propres plugins Open Source. L'Université catholique de Louvain utilise sa propre surcouche de Moodle afin de répondre aux besoins de ses enseignants et de ses étudiants. Par exemple, l'université a intégré son système d'authentification à son Moodle personnalisé.

2.5 Applications UCL

Cette section présente, de manière plus approfondie, deux applications présentes dans le paysage informatique de l'Université catholique de Louvain. Ces deux applications sont utiles dans le cadre du développement de l'application UCLCampus. La première est Open Student Information System (OSIS) qui permet, entre autres, aux étudiants de s'inscrire à leurs cours et examen ou d'accéder à leurs notes et d'interagir. La seconde application, Shibboleth, assure l'authentification des utilisateurs désirant accéder aux services offerts par l'UCL.

2.5.1 OSIS

OSIS⁶ est un système destiné à gérer la logistique des établissements d'enseignement supérieur. C'est un projet Open Source financé par l'Université catholique de Louvain développé sous licence GPL-v3. Le développement de ce système se fait dans un contexte de modernisation des systèmes d'informations et des technologies au sein de l'UCL. OSIS, projet en développement permanent, est développé à l'aide du framework Django 1.9 qui manipule Python 3.4 et HTML5. Le projet OSIS préconise l'utilisation de PostgreSQL comme base de données.

Le système OSIS permet à un établissement d'enseignement supérieur de contrôler et surveiller le parcours d'un étudiant de son inscription à l'obtention de son diplôme. Pour aider l'établissement dans ses tâches, le système fournit des fonctionnalités telles que la planification de l'année académique, la gestion des partenariats entre institutions, les programmes de mobilité, l'attribution des cours aux enseignants, l'inscription des étudiants aux cours et aux sessions d'examens, etc. OSIS va, à terme, remplacer l'ancien système de gestion des étudiants, EPC développé en Java.

Le projet OSIS possède deux entités distinctes et indépendantes : OSIS et OSIS portal comme illustré dans la figure 2.16. OSIS commun fournit les fonctionnalités communes aux deux entités. OSIS est la partie du système qui permet aux services administratifs de gérer la partie académique de l'établissement. OSIS portal est la partie du système qui permet aux étudiants d'interagir avec le système, par exemple en s'inscrivant aux cours et aux examens après leur authentification. Le

6. <http://uclouvain.github.io/osis/>

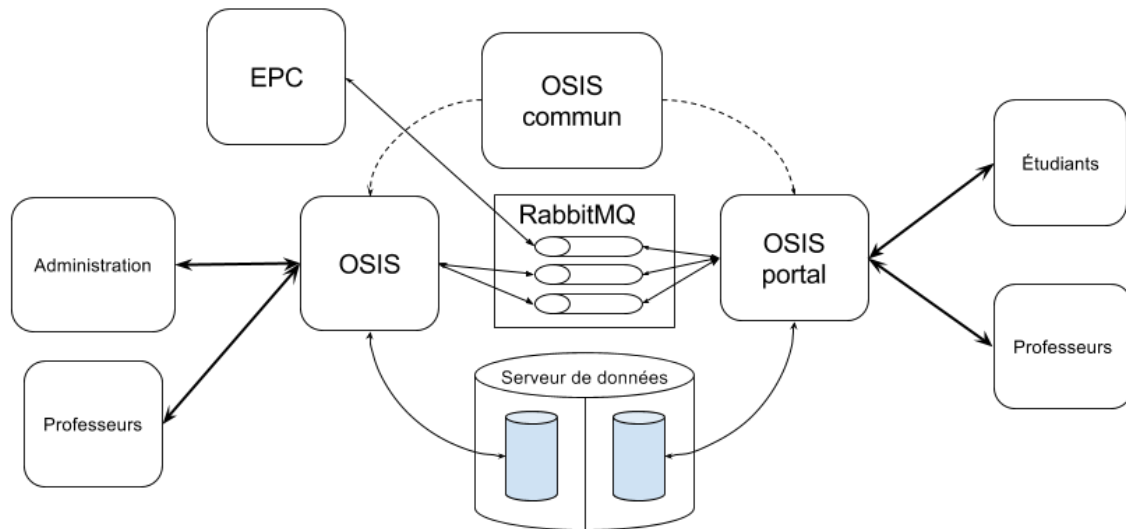


FIGURE 2.16 – Structure d’OSIS

système OSIS fait partie intégrante de la nouvelle infrastructure mise en place par l’université en interagissant avec les autres services informatiques via l’ESB WSO2. Chacune des deux parties principales du système, OSIS et OSIS portal, sont indépendantes et totalement découplées l’une de l’autre. Elles peuvent fonctionner l’une sans l’autre. Elles possèdent toutes les deux leur propre base de données logique présente sur un même serveur physique. La base de données avec laquelle OSIS interagit est une base de données structurée possédant l’ensemble des informations du système administratif de l’UCL. Les requêtes effectuées sur cette base de données sont complexes car elles font généralement interagir les multiples tables qui composent la base de données. De ce fait, les requêtes effectuées sur la base de données de la partie OSIS sont lourdes et lente, cela peut engendrer des problèmes de ressources et de performance lorsque de nombreuses requêtes sont effectuées simultanément, par exemple lors de la diffusion des résultats des examens. La solution proposée est de manipuler une seconde base de données plane avec OSIS portal. Cette seconde base de données offre un accès de lecture rapide aux données disponibles dans la base de données structurée sous forme de JavaScript Object Notation (JSON). Lors de la diffusion des résultats, la requête effectuée sélectionne dans une table le matricule de l’étudiant et retourne le JSON associé contenant les notes. Ce procédé permet de gérer un flux de requêtes important. Les différents JSON sont construits en amont de la diffusion des résultats, à partir des données disponibles sur la base de données OSIS suite à la validation des délibérations par le jury. La base de données d’OSIS portal peut également servir de cache pour stocker des informations qui vont être transférées vers la base de données d’OSIS suite au déclenchement d’un événement spécifique tel que la confirmation d’une inscription. La communication entre les bases de données des deux parties indépendantes du système est gérée par le mécanisme RabbitMQ qui implémente le concept de message broker. Le système RabbitMQ est la pierre angulaire du découplage des deux modules OSIS et OSIS portal. Il permet aux modules de recevoir et de manipuler des messages depuis différents canaux sans connaître la source du message. L’action effectuée avec le message reçu dépend du canal par lequel le message a été réceptionné. Donc, à chaque canal correspond une action spécifique généralement indiquée par son nom. La communication entre les deux modules est asynchrone, un module envoie un message sur un canal par RabbitMQ et peut, ensuite, passer à une autre activité. Le module destinataire doit régulièrement inspecter chacun des canaux pour relever les messages reçus, qui peuvent s’accumuler dans l’attente d’être

réceptionnés, et agir en fonction. Les actions internes aux modules se font de manière synchrone pour garantir la cohérence et la consistance de l'état des données. La communication asynchrone entre OSIS et OSIS portal se fait dans les deux sens. Le découplage induit par l'utilisation de RabbitMQ rend les modules indépendants les uns des autres ce qui permet une transition plus sereine des technologies. OSIS portal n'a pas besoin de savoir si les informations proviennent du système OSIS ou de l'ancien système EPC, il relève juste les messages en provenance d'un canal. OSIS portal ne remarquera pas le changement de système lui envoyant les messages.

2.5.2 Shibboleth

L'UCL utilise un processus d'authentification unique, SSO mis en place par le mécanisme Shibboleth. Ce processus permet aux utilisateurs de se connecter à un ensemble de services et d'applications nécessitant une authentification. L'authentification unique offre un contrôle d'accès pour un ensemble de services et d'applications indépendants les uns des autres mais appartenant à un même système. L'authentification unique permet à un utilisateur de se connecter une seule fois avec son identifiant et son mot de passe pour accéder à un système de services sans devoir réintroduire ses données personnelles lorsqu'il désire accéder à un autre service du système.

Shibboleth est un mécanisme de fédération d'identité utilisé principalement par des établissements universitaires ou de recherche. Ce mécanisme Open Source est basé sur les standards d'authentification unique sur le web ou au sein d'une organisation.

Le projet Shibboleth a débuté en tant qu'activité Middleware Internet2[33], en 2000 ensuite, le projet a été lié aux travaux du groupe de travail OASIS Security Assertion Markup Language (SAML). Shibboleth 1.0 a été publié en 2003 et a été rapidement adopté par les communautés de recherche et d'éducation dans le monde entier. Avec SAML 2.0 en 2005 et Shibboleth 2.0 l'année suivante, les normes SAML ont englobé toutes les approches multilatérales axées sur les métadonnées initiées par Shibboleth. Shibboleth est développé comme mécanisme Open Source et est publié sous la licence Apache.

2.5.2.1 Fonctionnement

Shibboleth[34] est composé de deux composants majeurs : le fournisseur d'identité (IdP) et le fournisseur de services (SP). Le fournisseur d'identité donne des informations sur les utilisateurs aux services et le fournisseur de services rassemble des informations sur les utilisateurs afin de protéger les ressources.

Afin de fournir un cadre fédéré d'authentification unique et d'échange d'attributs, le mécanisme Shibboleth implémente des standards d'identité fédérés largement utilisés, principalement le OASIS SAML. Un utilisateur s'authentifie via ses identifiants et le fournisseur d'identité transfère le minimum d'informations privées, à propos de l'utilisateur, nécessaires au fournisseur de services. Ce dernier est chargé de décider des autorisations à allouer à l'utilisateur en fonction des informations libérées par le fournisseur d'identité. Shibboleth fournit aussi une fonctionnalité de confidentialité étendue permettant à un utilisateur et à son site de contrôler les attributs à libérer pour chaque application.

Termes liés au Single Sign-On :

Fournisseur d'identité ou Identity Provider (IdP) : L'IdP vérifie les identifiants d'un utilisateur et renvoie les attributs nécessaires de l'utilisateur aux fournisseurs de services.

Fournisseur de services ou Service Provider (SP) : Le SP recueille les déclarations d'identification et d'attributs venant de l'IdP avant de les partager avec l'application appelante.

Ressource Protégée : Contenu d'un site web. Répertoire, fichiers, application web, ...

Attributs : Informations liées à un utilisateur individuel telles que : nom, affiliation, email, userid, ...

Message SAML : format de données standards pour l'échange de données d'authentification et d'autorisation. Shibboleth utilise des messages SAML pour communiquer entre ses IdP et ses SP.

2.5.2.2 Processus

1. L'utilisateur accède une ressource protégée

L'utilisateur tente d'accéder à une ressource protégée. Si l'utilisateur possède une session active sur son navigateur, la ressource est accessible. Dans le cas contraire, l'utilisateur est envoyé vers le SP afin de commencer le processus de Single Sign-On.

2. Le SP envoie une requête d'authentification

Le SP prépare une requête d'authentification et l'envoie avec l'utilisateur à l'IdP.

3. Utilisateur authentifié au niveau IdP

Si l'IdP possède dans son cache une session existante attribuée à l'utilisateur, le processus passe à l'étape suivante. Dans le cas contraire, l'IdP identifie l'utilisateur via un formulaire (par exemple en demandant et en vérifiant un nom d'utilisateur et un mot de passe) et l'utilisateur passe à l'étape suivante.

4. L'IdP envoie une réponse d'authentification

Après avoir identifié l'utilisateur, l'IdP prépare une réponse d'authentification et l'envoie avec l'utilisateur au SP. Dans la configuration de l'IdP, une liste d'attributs à libérer est associée à chaque SP. Chaque attribut correspond à une information du profil utilisateur. Cette liste est envoyée dans la réponse d'authentification au SP.

5. Le SP vérifie la réponse d'authentification

De retour au SP, celui-ci valide la réponse et crée une session pour l'utilisateur contenant les attributs à libérer (par exemple, l'identifiant de l'utilisateur). Le SP rend ensuite ces attributs disponibles pour la ressource et y renvoie l'utilisateur.

6. La ressource renvoie du contenu

Comme à l'étape 1, l'utilisateur tente d'accéder à la ressource protégée mais cette fois-ci avec une session identifiée. La ressource sert la requête de l'utilisateur et renvoie les informations demandées.

Ce processus est illustré par la figure 2.17.

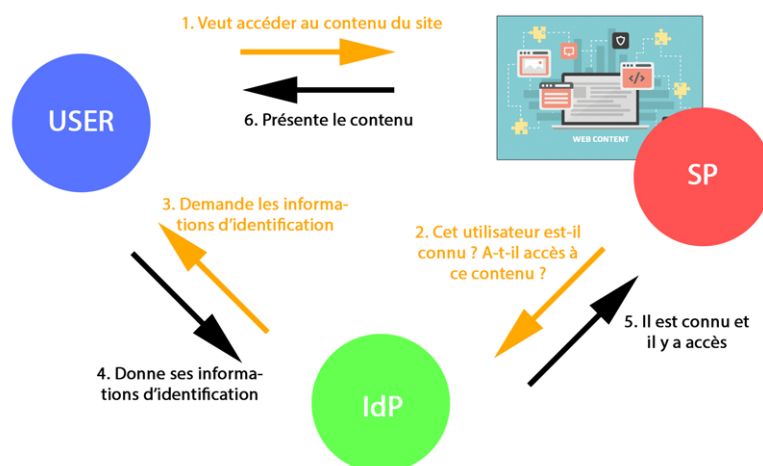


FIGURE 2.17 – Processus de Single Sign-On

2.6 Application UCLCampus existante

Le mémoire présenté se situe dans la poursuite du travail de fin de cycle effectué par deux étudiants en master en informatique à l’UCL durant l’année académique 2015-2016. Dans un premier temps, la partie écrite et le code du travail⁷ ont été lus et analysés pour établir un bilan de cette application. L’ensemble du bilan est détaillé dans les premières parties de cette section à travers une présentation de l’application existante et des choix d’implémentation posées. Ensuite, les limitations présentes dans le projet existant sont abordés. Pour finir, les éléments du projet existant réutilisés et exploités dans le cadre de ce mémoire sont présentés.

2.6.1 Concept

Le projet UCLCampus a été développée à destination de deux publics cibles : des utilisateurs lambda qui sont intéressés par les activités organisées sur les campus universitaires et les étudiants de l’Université catholique de Louvain. L’application propose des fonctionnalités qui sont générales et accessibles publiquement ainsi que des fonctionnalités spécifiques accessibles uniquement par les étudiants de l’UCL quand ils sont connectés avec leur identifiant et leur mot de passe UCL. Les fonctionnalités proposées par l’application peuvent être catégorisées comme suit :

Études Fonctionnalité concernant directement les études d’un étudiant de l’université comme l’horaire de ses cours, les auditoriums, les bibliothèques et l’accès aux sites internet de l’université.

Ville Fonctionnalité permettant d’accéder à des informations sur la ville dans laquelle se trouve le campus universitaire. Ces informations sont généralement d’ordre commercial avec les horaires de cinéma, les commerces, les bars et les restaurants.

Campus Fonctionnalité touchant à la vie étudiante sur le campus telles que les informations sur les kots à projets, les cercles ainsi que les événements organisés par ceux-ci, les horaires des activités sportives, l’horaire d’ouverture et menus des restaurants universitaires.

Outils Les fonctionnalités dites outils sont les fonctionnalités générales non spécifiques à un campus tel que l’accès à une carte interactive.

Paramètres Les paramètres permettent à l’utilisateur de définir la langue d’affichage des informations sur l’application, le choix d’un campus.

2.6.2 Choix d’implémentation

Les développeurs du projet précédent UCLCampus ont sélectionné le framework Ionic version 1 comme outil pour les accompagner dans le développement du projet. Ce choix a un impact sur l’architecture du projet. La manière dont l’application récupère les informations qu’elle affiche est également un choix d’implémentation posé par les développeurs. Ces deux éléments importants de la mise en oeuvre du projet UCLCampus sont présentés ci-dessous.

2.6.2.1 Architecture

En utilisant Ionic 1 pour la réalisation du projet UCLCampus, les développeurs utilisent l’architecture prônée par Angular JS qui est le framework de développement d’application web intégré à Ionic 1.

Angular JS adopte une architecture Model-View-ViewModel (MVVM) qui est une évolution du patron de conception Model-View-Controller (MVC). Cette architecture structure l’application d’une manière qui facilite la compréhension et l’amélioration du code, la réalisation de tests, l’ajout de bibliothèques, de fonctionnalités et la modularité du projet.

7. <https://github.com/amoyaux/UCLCampus>

MVC La structure MVC découpe l'application en trois parties connectées les unes aux autres.

Modèle communique avec la base de données, le serveur. Son rôle est de structurer les données récupérées avec pour objectif de les standardiser.

Vue affiche les informations désirées. C'est l'interface d'interaction entre l'utilisateur et la partie logique de l'application.

Contrôleur est la partie logique de l'application. Il gère les événements se produisant sur la Vue et met à jour le Modèle en fonction de ceux-ci.

Avec cette découpe illustrée dans la figure 2.18, les trois parties restent couplées et dépendantes de la structure de données utilisées par le Modèle. Si le Modèle venait à changer, des modifications devraient être effectuées sur les parties Contrôleur et Vue.

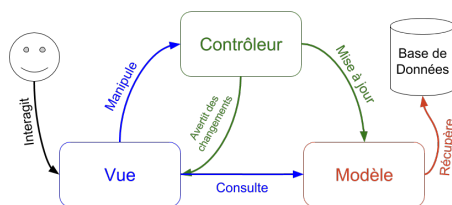


FIGURE 2.18 – Structure d'un Modèle-Vue-Contrôleur

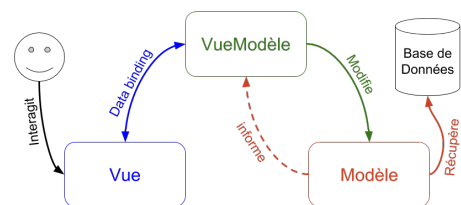


FIGURE 2.19 – Structure d'un Modèle-Vue-VueModèle

MVVM La structure MVVM[35] propose une solution à ce problème de couplage avec le Modèle. Comme le présente la figure 2.19, en introduisant une VueModèle à la place du Contrôleur, les interactions entre les trois parties ont évolué. Les rôles de la Vue et du Modèle restent les mêmes. Le Modèle possède un rôle supplémentaire, informer la VueModèle de son changement. La VueModèle possède un rôle central qui consiste à :

- présenter les données à la Vue
- recevoir les changements de données de la Vue
- demander au Modèle de se modifier

Les deux premiers rôles énoncés sont rassemblés sous la dénomination data binding. Dans cette structure, la Vue n'est plus dépendante du Modèle.

2.6.2.2 Récupération des informations

L'application ayant essentiellement pour objectif d'informer ses utilisateurs, la méthode de récupération et de questionnement des différentes sources d'informations est cruciale pour la durabilité et la maintenance de l'application. Les différentes méthodes utilisées par l'application UCLCampus sont :

Flux Rich Site Summary (RSS) L'application écoute un flux RSS. Elle structure les informations présentes sur le flux. Dès que l'information présente sur le flux est modifiée, l'information sur l'application l'est également. L'application UCLCampus écoute le flux du calendrier de louvainfo pour afficher les événements à venir sur le campus de Louvain-La-Neuve.

Redirection La redirection permet à l'application d'afficher directement la page web contenant toutes les informations nécessaires sur un sujet. Cette approche est utilisée pour afficher les menus des restaurants universitaires.

Bases de données locales L'application UCLCampus possède une base de données SQLite contenant 7 tables.

- "Auditorium" reprend l'ensemble des auditorios et des informations pratiques les concernant comme le nombre de places.
- "Bibliothèque" contient un id, le sigle et un lien Uniform Resource Locator (URL) vers la page web de la bibliothèque.
- "Bibliothèque_Horaire" représente pour chaque ID de bibliothèque, les informations sur les horaires d'ouverture, jours et heures.
- "Courses" ayant pour colonne le code du cours et son nom.
- "Horaire" est la table reprenant les informations sur les horaires des cours présents dans la table Courses.
- "Restaurant" est la table contenant les informations sur les restaurants universitaires de l'Université catholique de Louvain. Les informations reprises sont : nom, adresse, campus, une image, heures et jours d'ouverture ainsi qu'un lien vers la page web.
- "poi" est la table reprenant l'ensemble des points d'intérêt de l'application. Les informations présentes sont le nom, les coordonnées géographiques, le type, l'adresse, l'image, le campus et l'abréviation.

Parsing Cette approche consiste à récupérer et restructurer les informations présentes sur les pages internet en analysant le contenu des documents HTML envoyés en réponse à un chargement de page. Cette technique nécessite de connaître la structure HTML de la page à parser pour récupérer les bonnes informations. Ce processus est présent dans l'application UCLCampus pour la récupération des horaires de cours sur ADE, page internet contenant tous les horaires de cours et les horaires des activités sportives. Le processus de parsing est couplé avec un stockage en base de données afin d'améliorer les performances et d'accéder aux informations hors connexion.

2.6.2.3 Carte interactive

La partie carte interactive de l'application est inspirée de l'application LLN Maps développée par Mathieu Zen assistant à l'Université catholique de Louvain. Cette fonctionnalité est développée à l'aide de la librairie JavaScript de cartographie Open Source Leaflet sous licence BSD 2-clauses à destination des applications mobiles. Les cartes disponibles à partir de la librairie Leaflet sont les cartes d'OpenStreetMap. OpenStreetMap est un projet qui vise à constituer une base de données géographiques libre pour permettre la création de carte sous licence libre. Leaflet manipule des tuiles, parties de carte générées sur OpenStreetMap. Ces tuiles peuvent être enregistrées localement dans l'application pour permettre à l'utilisateur d'accéder aux fonctionnalités de la carte malgré qu'il soit hors réseau. L'enregistrement de nombreuses tuiles rend l'application très lourde, il est préférable d'utiliser cette approche uniquement pour des petites zones géographiques. Leaflet Routing Machine est une librairie sous licence ISC qui permet d'ajouter la fonctionnalité définition d'un itinéraire reliant deux points de la carte. Cette librairie exploite le projet Open Source Routing Machine (OSRM) implémenté en C++ sous licence BSD qui fournit des algorithmes de routage pour définir le plus court chemin entre deux points.

2.6.3 Limitations

Cette sous-section discute les choix posés pour la réalisation du projet UCLCampus par nos prédécesseurs afin d'avoir une meilleure connaissance des problèmes et des limitations qui peuvent être rencontrées dans le développement futur de l'application. Dans un premier temps,

les remarques porteront sur des aspects non techniques de l'application, tels que le concept et l'interface utilisateur, pour se diriger vers une discussion sur la technologie utilisée et finir par les impacts des choix d'implémentation effectués.

2.6.3.1 Concept

L'absence d'un cahier des charges complet ne facilite pas la continuité du développement de l'application. En effet, les développeurs souhaitant reprendre l'application ont juste une brève explication des fonctionnalités présentes dans l'application et des récits utilisateurs, ce qui rend difficile la compréhension de la philosophie de l'application. Le projet étant à destination des étudiants, une enquête auprès de ceux-ci pour connaître leurs attentes aurait pu orienter le choix des fonctionnalités de l'application. De plus, la catégorie de fonctionnalités Ville, dont le but est d'informer les utilisateurs des restaurants, bars, commerces et horaires des cinémas à proximité, se trouve en concurrence avec d'autres applications. Ces dernières possèdent, en outre, une notoriété et des ressources ne permettant pas au projet UCLCampus d'égaler les services proposés par celles-ci.

2.6.3.2 Interface utilisateur

La navigation dans les menus n'est pas toujours intuitive. La distinction entre les parties est parfois floue et il y a des recoupements. Par exemple, les restaurants universitaires peuvent se trouver dans la partie Campus mais également de la partie Ville étant donné que c'est un commerce. En cas d'erreur ou de fonctionnalité non opérationnelle, l'utilisateur est dirigé vers l'écran d'accueil sans message d'erreur ou explication. Par contre, l'utilisation d'une Tab bar accessible depuis toutes les pages facilite la navigation dans l'application.

2.6.3.3 Récupération de données

L'utilisation du processus de parsing et de redirection rend l'application dépendante de la manière dont les informations sont représentées. En effet, si le portail de l'université change, les adresses URL et les documents HTML vont changer. La vérification d'un tel changement n'étant pas testée, l'application fonctionne dans son ensemble mais certaines fonctionnalités ne sont plus disponibles pour l'utilisateur. L'utilisateur peut ainsi être confronté à des pages qui n'existent plus ou à des informations erronées car les balises du document HTML ont changé.

Le stockage d'information dans une base de données offre à l'utilisateur un accès performant aux informations. Mais, en cas de changement de l'information présente dans la base de données, la mise à jour de celle-ci est compliquée à effectuer. La base de données doit donc uniquement contenir des informations peu sujettes à un changement.

Le grand avantage de ces méthodes est que les développeurs peuvent avancer rapidement dans le développement de l'application sans devoir être en contact avec les services responsables des services web et des informations disponibles. Les développeurs sont créatifs et se fient à leurs compétences pour accéder de manière détournée aux informations désirées. Le revers de la médaille est que l'application demande une maintenance accrue car à chaque modification du côté de la source d'information, l'application doit être modifiée. Par contre, avec l'utilisation d'API, une forme de contrat est établi, les services s'engagent à retourner les données sous une forme standard en fonction de la requête effectuée pour interroger le service. Ce choix rend l'application plus maintenable et modulaire.

Nos prédécesseurs ont été contraints d'utiliser les techniques de récupération de données que sont le parsing, la redirection et le stockage d'information dans une base de données statique car le passage vers l'ESB WSO2 n'avait pas encore eu lieu et ces API standardisées étaient à leurs prémices.

2.6.3.4 Maintenance et modularité

Le code de l'application UCLCampus précédente nécessite une factorisation pour les raisons suivantes :

- Manque de commentaires
- Les noms des méthodes et des variables ne sont pas toujours judicieux ou explicites
- Certaines méthodes sont composées de plus de 150 lignes de code
- Duplication de code

Les méthodes les plus longues sont essentiellement des méthodes liées au parsing de fichiers HTML qui apparaissent dans chaque contrôleur de page récupérant les données présentes dans un fichier HTML. La généralisation du processus de parsing, par la création d'un module spécifique, rendrait le code plus clair, plus modulaire et plus réutilisable.

2.6.3.5 Mise à jour Ionic

Au moment de la création du premier projet UCLCampus, le framework Ionic était à la version 1 qui est une version du framework mature et stable. Durant le développement du projet, Ionic a déployé la version 2 du framework en beta. En septembre, lors de la reprise du projet, la seconde version du framework prenait le pas sur la première version. Ionic 2 attirait de plus en plus de développeurs et la communauté migrait vers cette version au détriment de la version précédente qui, dès lors, devenait plus figée. Le changement principal entre Ionic 1 et Ionic 2 est le passage d'Angular JS à Angular 2. Il a fait grand bruit au sein de la communauté car un changement aussi important de l'architecture de développement d'une application web a de nombreux impacts sur le développement d'une application et exige une phase d'apprentissage et d'adaptation aux développeurs. L'organisation des fichiers des projets est également impactée par le changement d'architecture.

2.6.3.6 Application remise

Au terme du mémoire précédent, l'application développée n'a pas été déployée sur les stores d'application. Les fonctionnalités non opérationnelles étaient disponibles sur l'application alors que, dans le cadre d'un développement agile, à la fin de chaque sprint, seules les fonctionnalités finies doivent être présentes dans l'application. Suite à la manipulation de l'application, les principaux défauts aperçus lorsque l'utilisateur se trouve sur le campus de Louvain-La-Neuve ont été listés :

- L'agenda n'est pas fonctionnel.
- L'ouverture de la carte depuis un auditoire ou une bibliothèque n'est pas opérationnelle malgré la présence du bouton.
- Les boutons Cercles et Kots à projets dirigent l'utilisateur à la page d'accueil.
- Les menus des restaurants universitaires ne sont plus disponibles suite au changement du portail de l'UCL.
- La fonctionnalité cinéma envoie l'utilisateur vers la page d'accueil.

Le campus de Louvain-La-Neuve est le seul campus pour lequel l'application contient une quantité d'informations suffisantes pour être intéressante aux yeux des étudiants. Par exemple, l'outil carte qui est intéressant et attrayant est, malheureusement, disponible uniquement pour le campus de Louvain-La-Neuve.

Une incohérence a été soulevée entre le désir de rendre une partie de l'application accessible à tout public et la demande d'une connexion à l'ouverture de l'application. Cette connexion n'est pas opérationnelle, elle n'est pas reliée aux services d'authentification de l'UCL.

L'inconvénient d'une application avec des fonctionnalités non finies visibles est qu'elle n'est pas déployable. En effet, les utilisateurs ne voient que ces fonctionnalités-là, alors qu'il y a

de nombreuses autres fonctionnalités utiles et opérationnelles qui peuvent les aider dans leur quotidien sur les campus de l’UCL.

2.6.4 Décision

Au démarrage du projet, nous avons réalisé un sondage auprès d’étudiants afin de déterminer les fonctionnalités qu’ils souhaiteraient avoir dans l’application. Pour établir le questionnaire, nous nous sommes basés sur le travail antérieur. Nous nous sommes également inspirés, dans le cadre de notre travail, de certaines idées et solutions développées par nos prédécesseurs.

Cependant, à la suite de notre décision de développer l’application en Ionic 2, et non plus en Ionic 1, nous avons dû recommencer l’entièreté du code. En effet, le passage d’Ionic 1 à Ionic 2 implique de trop nombreux changements pour pouvoir repartir du code développé par nos prédécesseurs. De plus, leur façon de récupérer les informations mises à disposition par l’UCL n’est pas suffisamment modulaire, ni maintenable à cause de l’emploi des mécanismes de parsing du portail de l’UCL et de base de données statiques. Ce fait a été confirmé lors du passage au nouveau portail UCL qui a rendu de nombreuses fonctionnalités obsolètes.

Chapitre 3

Cahier des charges

Ce chapitre va définir en détail ce qui est attendu de l'application d'un point de vue fonctionnel et qualitatif. Les exigences ont été définies en fonction du contexte dans lequel l'application est développée et des applications existantes dans des domaines similaires. L'application se place dans un environnement particulier : les services informatiques proposés par l'Université catholique de Louvain qui sont en évolution permanente. De plus, l'application, développée en Open Source, devra répondre à certaines exigences.

3.1 Contexte

L'Université catholique de Louvain souhaite se doter d'une application facilitant la vie des étudiants sur ses campus en leur fournissant les informations utiles pour leur vie académique et leur vie sur le campus.

3.1.1 Audience cible

Initialement, l'application a pour public cible l'ensemble des étudiants présents sur les campus de l'UCL. La tranche d'âge de ce public est comprise entre 18 et 25 ans. L'audience cible actuelle de l'application est limitée aux étudiants du campus de Louvain-La-Neuve. Celui-ci est, en effet, le plus grand campus de l'université en terme d'étudiants et d'infrastructures. Il sera donc l'objet du premier déploiement de l'application avec pour but, par la suite, de l'étendre aux autres campus de l'université.

3.1.2 Besoins

Pour rédiger notre cahier des charges, il nous fallait connaître les besoins et souhaits des étudiants de l'UCL. Pour les identifier, nous avons réalisé une enquête électronique et papier.

3.1.2.1 Réalisation de l'enquête

L'enquête, voulant toucher un maximum de personnes, a été conçue de manière à demander un temps de réponse le plus limité possible. Pour ce faire, nous avons décidé de formuler les réponses aux questions avec de simples cases à cocher tout en laissant, par la suite, l'occasion à ceux qui le désiraient de développer leurs réponses.

L'enquête est divisée en trois parties. La première est destinée à connaître le répondant c'est-à-dire sa faculté et son niveau d'étude, bachelier ou master. Une seconde partie est axée sur les besoins rencontrés par les étudiants dans le cadre de leurs études universitaires. La troisième partie interroge les étudiants sur les besoins auxquels l'application mobile devrait répondre

concernant la vie sur le campus universitaire. Pour rédiger les questions, nous nous sommes inspirés des différentes fonctionnalités observées lors de l'analyse de l'application développée par nos prédécesseurs et de notre vécu en tant qu'étudiant. Vous trouverez le questionnaire en annexe B.

Dans un premier temps, un simple formulaire Google a été placé sur les réseaux sociaux en espérant avoir de nombreuses réponses rapidement. Mais malheureusement, le formulaire n'a pas eu le succès attendu. Nous nous sommes donc rendus à plusieurs reprises dans les auditoriums et salles d'études du campus de Louvain-La-Neuve avec le formulaire en version papier, afin de rencontrer les étudiants. Ces initiatives furent fructueuses, le nombre final de répondants est de 189.

3.1.2.2 Analyse des résultats de l'enquête

Les résultats obtenus ont été analysés à travers des schémas et des données statistiques.

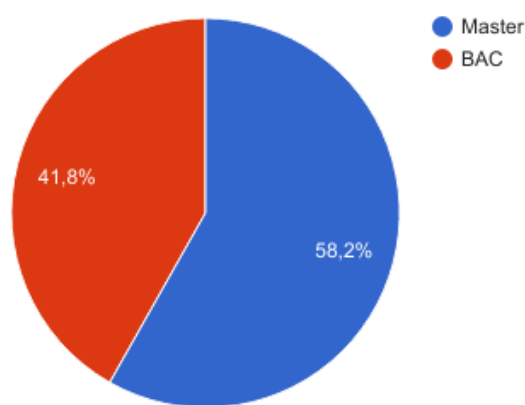


FIGURE 3.1 – Répartition du niveau d'études des répondants

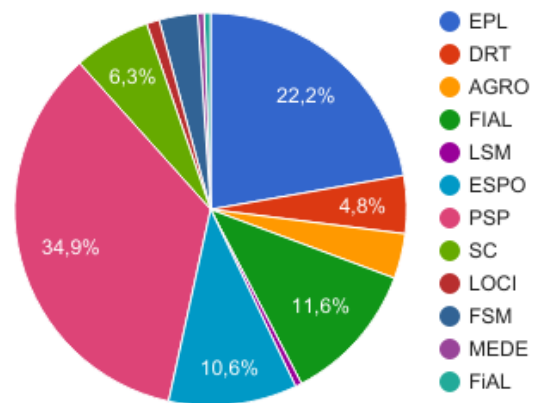


FIGURE 3.2 – Répartition des répondants par faculté

Suite à l'analyse de ces graphiques, quelques observations peuvent être faites. La première est que l'enquête a touché un large panel d'étudiants. La seconde constatation est que nous avons essentiellement touché des étudiants proches de nous soit par le choix d'études, soit par le niveau d'étude, ce qui peut biaiser les résultats. Ce biais peut s'expliquer par le fait que nous avons dans un premier temps interrogé des personnes dans notre entourage.

Ainsi, un taux de participation important des étudiants en master est constaté. Cet élément peut constituer un avantage dans le sens où ces derniers connaissent bien le système universitaire et son fonctionnement. Ils ont également une idée plus précise sur ce qu'une application devrait comporter comme informations pour leur faciliter la vie. L'inconvénient est qu'ils sont en fin de cycle et qu'une partie d'entre eux n'utilisera probablement jamais l'application, ils ne sont donc pas le public cible prioritaire.

L'analyse montre également que les étudiants de deux facultés l'EPL (Ecole Polytechnique de Louvain) et la PSP (Faculté de Psychologie et Sciences de l'éducation) représentent plus de 55% des sondés. L'inconvénient de cette représentation peu équitable des facultés est la difficulté de la prise en compte des besoins différents des étudiants en fonction de leurs études. En effet, l'enquête montre que les souhaits des étudiants diffèrent en fonction de leurs facultés qui utilisent différemment les ressources informatiques disponibles. Une illustration de ce fait est l'utilisation différente des valves informatiques selon les facultés. Pour certaines, c'est un canal privilégié de communication d'informations vers les étudiants. Dans d'autres facultés, les valves électroniques

ne sont jamais consultées par les étudiants car peu d'informations y sont affichées, l'information circule par mail.

Une autre limite est le fait que le sondage porte uniquement sur les étudiants du campus de Louvain-La-Neuve. Sachant que l'Université catholique de Louvain dispose de sept campus différents, cela peut poser problème dans le cadre de la réalisation d'une application plus large pour les étudiants de l'ensemble des campus de l'UCL.

Malgré ces remarques, nous avons pu mettre en évidence les principaux besoins, que nous pensons, communs à l'ensemble des étudiants de l'UCL. Ceux-ci sont repris dans deux tableaux en fonction du domaine sur lequel porte le besoin.

Fonctionnalités	Taux de demande (%)
Agenda reprenant mon horaire de cours et mes auditories	93,6
Notifications en cas de changement d'horaire ou d'auditoire	88,3
Accès à mes points	74,5
Accès à moodle	69,1
Horaire d'ouverture et de fermeture des bibliothèques	62,8
Accès à la messagerie universitaire	60,6
Localisation des salles d'étude et des salles informatique	55,9
Outils de gestion d'achat et de revente de livres en seconde main	53,7
Connaitre le nombre de places disponibles dans les bibliothèques	47,9
Forum étudiants permettant de centraliser les questions sur un cours	38,8
Informations sur la vie universitaire	37,2
Canaux d'interaction en temps réel pour poser des questions aux professeurs lors d'un cours magistral UCL	29,8
Accès aux informations sur les services offerts par l'UCL	26,6

TABLE 3.1 – Résultats de la question : "Quelles fonctionnalités/outils veux-tu avoir dans le cadre de tes études?"

Nous pouvons effectuer quatre constats :

- Les deux fonctionnalités les plus demandées dans le cadre académique mais également la fonctionnalité la plus prisée dans le cadre "vie sur le campus" sont des fonctionnalités liées à l'organisation et à la gestion du temps.
- La réalisation d'un réseau social interne à l'université n'est pas une priorité pour les étudiants au vu des faibles résultats obtenus par les fonctionnalités : Réseau social spécifique à l'UCL (Communauté d'étudiants par centre d'intérêt, sport, musique, ...), Création personnelle d'événements pour une communauté, un groupe d'amis, Inviter des personnes à discuter, partager sa table lors d'un temps de midi, Forum étudiants permettant de centraliser les questions sur un cours, Canaux d'interaction en temps réel pour poser ces questions aux professeurs lors d'un cours magistral UCL. Ce désintérêt est probablement dû aux plateformes de partage et d'échange ainsi qu'aux nombreux réseaux sociaux existants déjà et qui satisfont les étudiants.
- Un autre constat est le peu d'intérêt pour ce qui concerne la diffusion des informations et services utiles de l'UCL.
- Des résultats plus surprenants sont les fortes demandes concernant l'accès aux points, à moodle et à la messagerie universitaire alors que tous les smartphones sont dotés d'un navigateur web et d'une application mail pour accéder rapidement et facilement à ces contenus.

Fonctionnalités	Taux de demande (%)
Agenda de tous les événements, sorties, soirées, festivités	70,9
Menu, horaire et prix des restaurants universitaires	70,9
Partage de bons plans	61
Horaire du sport CSE + réservation de terrains	58,8
Réseau social spécifique à l'UCL (Communauté d'étudiants par centre d'intérêt, sport, musique, ...)	36,8
Se géolocaliser sur une carte	36,3
Description et géolocalisation des différents KAP's	33,5
Description et géolocalisation des différents cercles	27,5
Inviter des personnes à discuter, partager sa table lors d'un temps de midi	25,3
Création personnelle d'événements pour une communauté, un groupe d'amis	19,2

TABLE 3.2 – Résultats de la question : Quelles fonctionnalités/outils veux-tu avoir dans le cadre de la vie sur le campus ?"

3.1.3 Objectifs

En se basant sur les résultats récoltés lors de l'enquête, un premier concept d'application a été formulé. L'application UCLCampus doit être une application mobile permettant une gestion et une organisation intuitive et efficace de la vie d'un étudiant au sein d'un campus universitaire de l'UCL. L'application UCLCampus peut répondre à ces besoins en faisant l'interface entre l'agenda des smartphones et les événements que les étudiants choisissent de vivre quotidiennement. Une première partie de l'application sera consacrée aux études, elle comprendra tous les événements liés à la scolarité des étudiants tels que les horaires de cours, réunions de groupes, remise de travaux et les résultats. Une deuxième partie de l'application sera consacrée aux événements ayant lieu sur le campus universitaire tels que les conférences, activités estudiantines, soirées, activités sportives et les repas de cours. L'idée est de permettre aux étudiants d'être informés de tous les événements du campus en un seul mouvement de doigt. Ils pourront organiser et mettre à jour leur agenda électronique simplement et intuitivement. L'application permettra de centraliser l'ensemble des informations utiles à la vie sur le campus.

3.1.4 Applications existantes

Cette partie contient de brèves descriptions d'applications déjà existantes ainsi que les fonctionnalités qu'elles proposent à leurs utilisateurs. Les applications ont été regroupées en trois catégories : les applications développées par d'autres universités sur leurs campus, les applications présentes sur les campus de l'UCL, et les applications répondants à certaines fonctionnalités soulevées par l'enquête.

3.1.4.1 Applications universitaires

Les applications décrites ci-dessous sont des applications développées par des universités belges ou étrangères.

Quivr¹ Une application à destination des étudiants et du personnel académique de l'université de Leuven (KUL). L'application contient de nombreuses fonctionnalités telles que :

- La gestion d'agenda : horaire des cours, planification de réunions
- L'affichage de l'occupation des bibliothèques
- La diffusion des menus des restaurants universitaires
- La localisation sur une carte des points d'intérêt
- L'information des horaires des transports en commun
- La diffusion des nouvelles de l'université

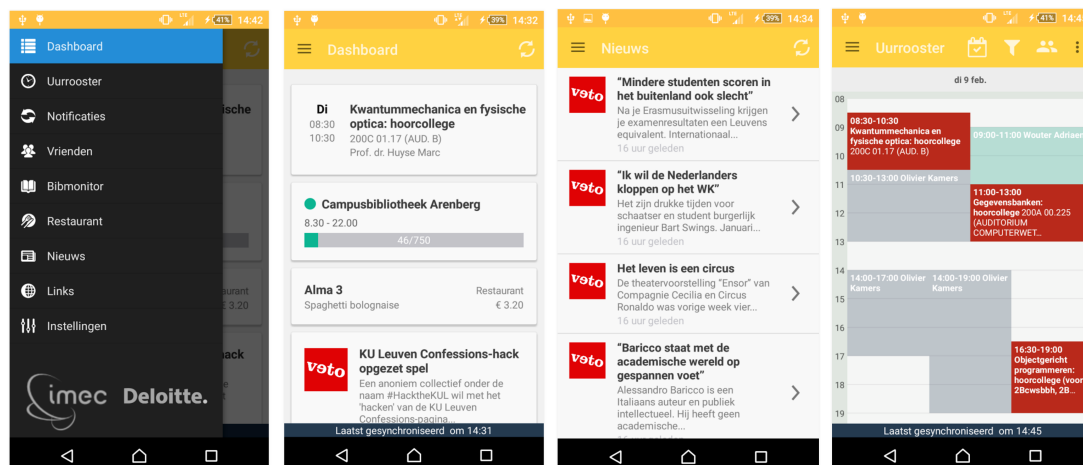


FIGURE 3.3 – Illustrations de l'application Quivr

EPFL Campus²

C'est l'application officielle pour les étudiants, collaborateurs et visiteurs de l'EPFL, université Suisse de Lausanne. Les fonctionnalités principales de l'application sont :

- IS-Academia : horaire personnel de cours et notes
- L'accès à Moodle
- Un service d'impression de PDF
- Une carte du campus
- L'accès à l'annuaire et aux numéros utiles
- Les menus des restaurants
- Les événement sur le campus et les nouvelles
- Camipro : Portefeuille électronique

UCL Go!³ Cette application est développée par l'University College London au Royaume-Uni. Cette application propose à ses utilisateurs de multiples fonctionnalités telles que :

- Le planning des cours
- Le répertoire des étudiants et du personnel
- Carte avec recherche et localisation des bibliothèques
- Diffusion des nouvelles et des événements sur le campus

1. <https://quivr.be>

2. <https://pocketcampus.org/home-en>

3. <http://www.ucl.ac.uk/isd/services/websites-apps/apps/ucl-go>

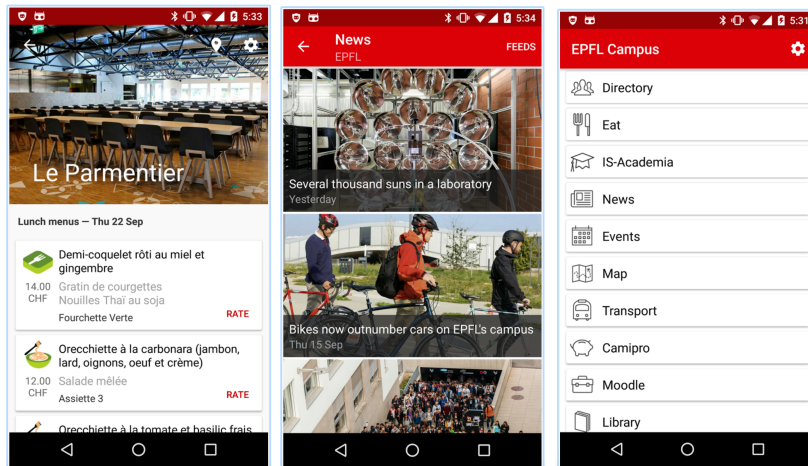


FIGURE 3.4 – Illustrations de l'application EPFL

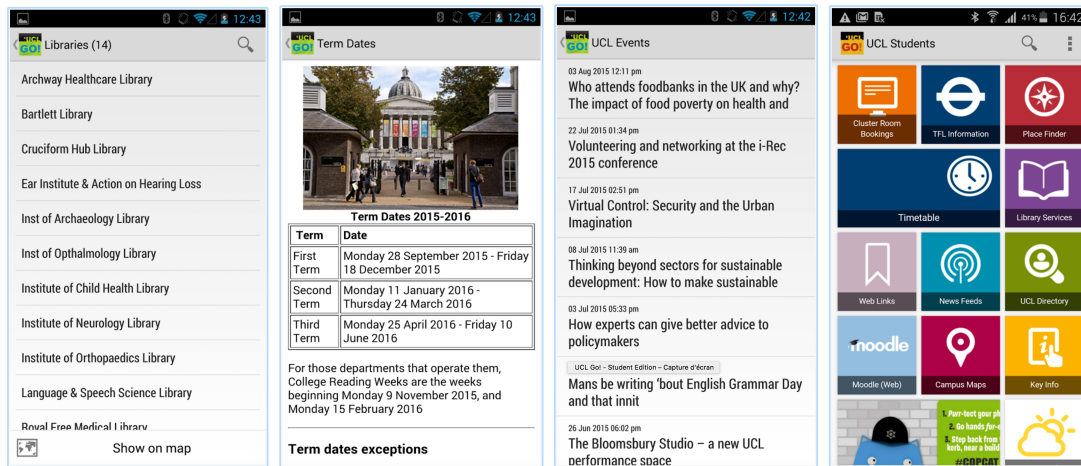


FIGURE 3.5 – Illustrations de l'application UCL Go!

3.1.4.2 Applications liées à l'UCL

Cette partie dresse le portrait des différentes applications présentes dans le paysage de l'Université catholique de Louvain.

LLNCampus⁴ LLNCampus est une application mobile pour Android développée en 2013 par des étudiants dans le cadre du cours FSAB1509 - Projet Informatique donnée par le professeur Yves Deville à l'Université catholique de Louvain. L'application est publiée sous licence GPL v3. Les fonctionnalités offertes par cette application sont :

- Une carte du campus et un guidage GPS
- L'horaire de cours personnalisable synchronisé avec ADE
- L'horaire des bibliothèques
- La liste des auditorios et leur représentation sur la carte
- L'horaire des activités sportives
- Les menus des restaurants universitaires
- Les événements du Carpe Studentem
- L'accès aux sites officiels de l'université

4. <https://sites.uclouvain.be/projetsBacEPL/projet/lln-campus/>

— L’horaire du cinéma

L’application se fait vieille, elle n’a plus subi de mise à jour majeure depuis le 11/10/2014.

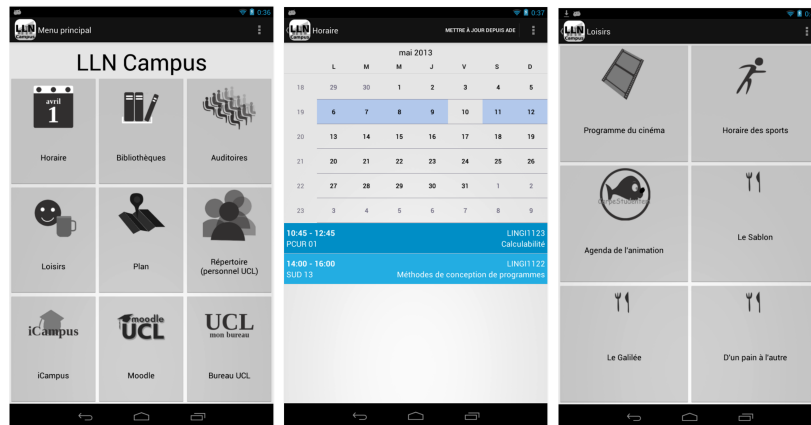


FIGURE 3.6 – Illustrations de l’application LLNCampus

LLNMap⁵ Comme son nom l’indique, cette application a pour but d’offrir une carte détaillée de Louvain-La-Neuve à ses utilisateurs. L’application a été conçue par Mathieu Zen, assistant à l’Université catholique de Louvain, à l’aide du framework ionic et des outils Leaflet, Leaflet Routing Machine, Open Source Routing Machine et OpenStreetMap.

Les fonctionnalités supportées par l’application sont les suivantes :

- Carte hors ligne de Louvain-la-Neuve
- Marqueurs indiquant les points d’intérêt
- Recherche intelligente
- Mode boussole
- Calcul d’itinéraires

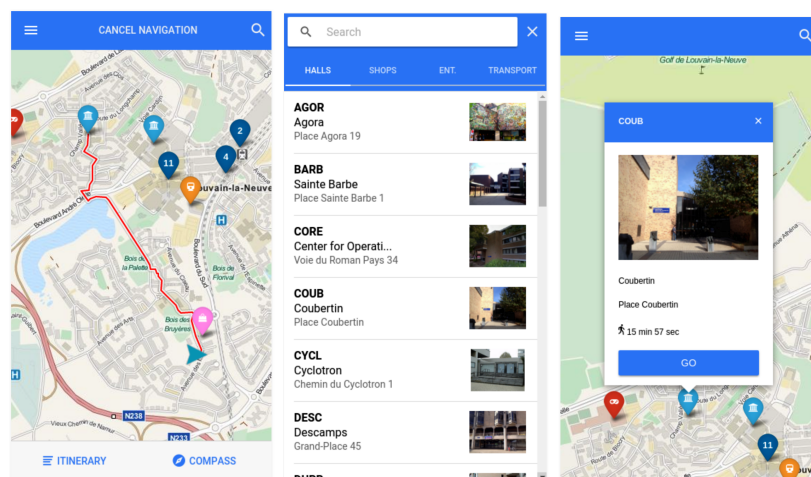


FIGURE 3.7 – Illustrations de l’application LLNMap

Guindaille 2.0⁶ Cette application permet aux étudiants du campus universitaire de Louvain-La-Neuve d’avoir accès aux informations sur les activités extra académiques. Elle a pour objectif

5. <https://play.google.com/store/apps/details?id=be.zen.llnmaps&hl=fr>

6. <http://www.guindaille2-0.be/application/>

de proposer des activités alternatives aux soirées étudiantes classiques. Elle prône également une guindaille consciente et responsable en accord avec la campagne Guindaille 2.0 au travers de pictogrammes et l'accès aux informations d'urgence. Le thème graphique de l'application respecte celui utilisé pour la campagne Guindaille 2.0. Cette application est développée par l'asbl Univers Santé.

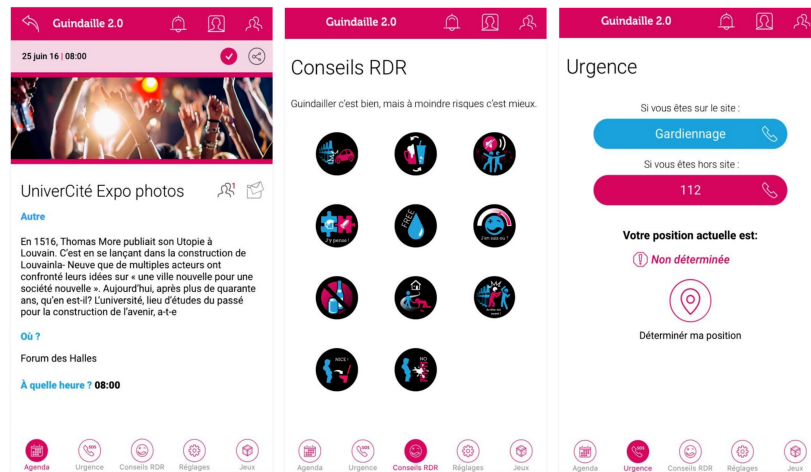


FIGURE 3.8 – Illustrations de l'application Guindaille 2.0

Resto4u⁷

Application développée par la société Apptree à la demande des restaurants universitaires. L'application permet à ses utilisateurs de :

- Afficher la liste des restaurants et leur horaire
- Consulter la carte de restauration
- Localiser les restaurants sur une carte
- Accéder aux informations pratiques et utiles diffusées par les restaurants universitaires

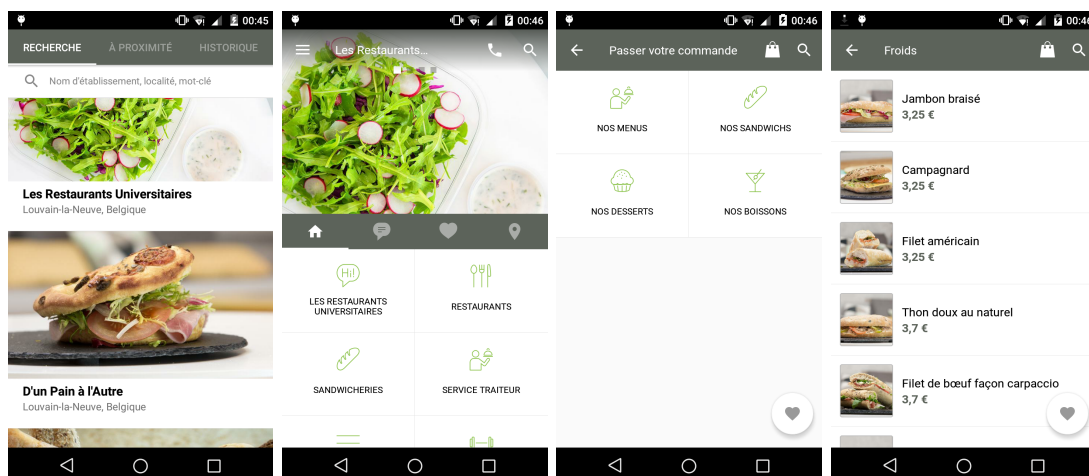


FIGURE 3.9 – Illustrations de l'application Resto4u

Commuty⁸ C'est avant tout une plateforme destinée à faciliter une mobilité durable et responsable. L'université est cliente de cette plateforme. Commuty propose une application mobile avec

7. <https://play.google.com/store/apps/details?id=com.apptree.resto4u>

8. <https://www.commuty.net>

les fonctionnalités suivantes :

- Trouver des co-voyageurs pour effectuer des trajets
- Contacter ces co-voyageurs à travers un chat pour s'organiser
- Remplir un journal de trajet pour récupérer les avantages fiscaux
- Réserver et partager des places de parking de l'entreprise

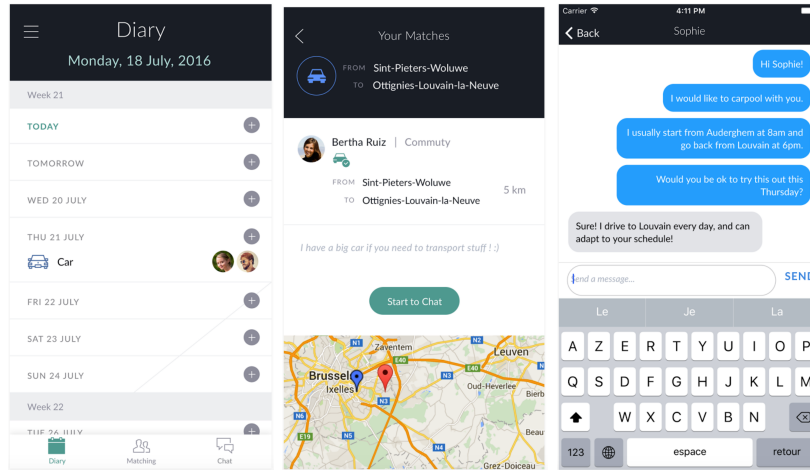


FIGURE 3.10 – Illustrations de l'application Commuty

3.1.4.3 Fonctionnalités utiles

La partie ci-dessous liste d'autres applications avec des fonctionnalités et services qu'il pourrait être intéressant d'intégrer à l'application.

Affluences⁹ L'application Affluences offre des services tels que la mesure de l'occupation, fiche détaillée sur mobile (gestion des horaires d'ouverture, communication avec les usagers), réservation de salle, analyse statistique, API de diffusion web. Un ensemble de services est proposé gratuitement :

- Modification en temps réel des horaires d'ouverture
- Saisie en avance du calendrier des horaires
- Affichage et gestion d'un ensemble d'informations pratiques telles que les conditions d'accès, les équipements, le nombre de places

Les services plus avancés tels que la mesure de l'occupation, la réservation de salle sont des services payants.

Seatfinder¹⁰ Le logiciel Seatfinder, développé par l'université de St Gallen, montre le taux d'occupation des espaces d'apprentissage au sein de l'université. Cet outil a été développé par le Karlsruhe Institute of Technology. Le nombre de places occupées ou libres est estimé toutes les 5 minutes à partir du nombre d'appareils connectés au réseau sans fil. La méthode d'estimation est basée sur des algorithmes de machine learning.

9. <https://www.affluences.com>

10. <https://www.seatfinder.de>

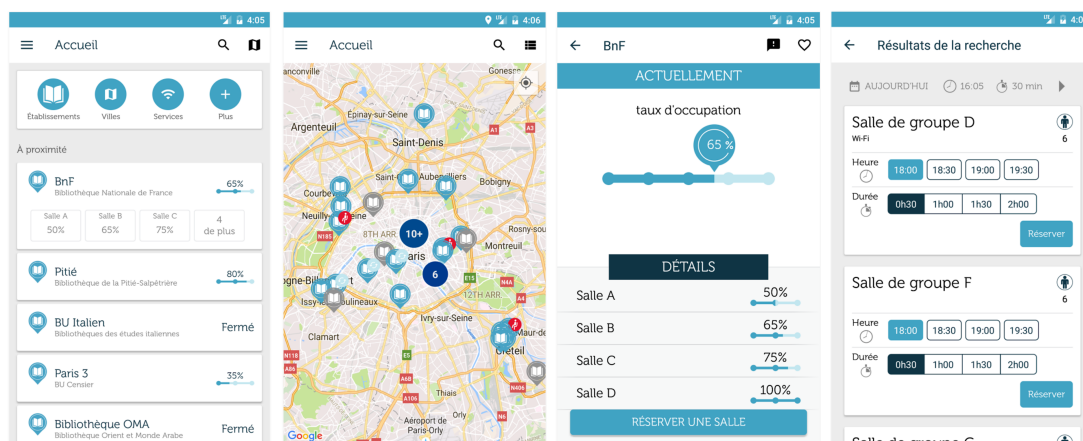


FIGURE 3.11 – Illustrations de l'application Affluence

3.1.5 Ressources UCL disponibles

L'application étant directement en rapport avec l'UCL, une analyse rapide de l'organisation et de ses entités a été réalisée. Lors de l'année 2015-2016, l'Université catholique de Louvain comprenait 29933 étudiants et 5854 employés¹¹. Elle est structurée en petites entités pour fournir différents services aux étudiants et aux membres du personnel. Les entités proposant des services en lien avec les besoins recensés lors de l'enquête ont été contactées afin de mieux connaître les ressources numériques et informatiques disponibles pour la réalisation de l'application.

3.1.5.1 Univers Santé

Univers Santé est une asbl créée par l'UCL dans le but de promouvoir la santé auprès des jeunes et des étudiants. L'asbl travaille en partenariat avec les services publics. Dans le cadre de la campagne Guindaille 2.0, l'asbl est à l'initiative du développement de l'application mobile Guindaille 2.0. Ce projet a été réalisé en partenariat avec d'autres organisations tels que le GCL, la Fédé, l'Organe, l'AGL, le Service d'aide aux étudiants, le Carpe Studentem et UCL Culture et le soutien de l'UCL, de la Province du Brabant Wallon et de la Fondation Louvain. Cette application reprend le calendrier des activités étudiantes sur le campus de Louvain-La-Neuve avec comme perspective d'élargir le service aux autres campus. Les événements repris dans son calendrier proviennent du calendrier en ligne du Carpe Studentem. De plus, le kot Carpe Studentem met à disposition un flux RSS contenant toutes les informations sur les événements disponibles sur le calendrier de leur site internet. Nous avons rencontré Monsieur Martin de Duve, directeur de l'asbl Univers Santé, pour lui présenter notre projet d'application. Au cours de cet échange, nous avons appris qu'Univers Santé était propriétaire du code source de l'application Guindaille 2.0 et qu'il était accessible en cas de besoin.

3.1.5.2 Restaurants Universitaires

Le service des restaurants universitaires est une entité extérieure à l'UCL faisant partie de l'unité technique d'exploitation. Ce service gère six établissements répartis sur trois sites universitaires : Louvain-La-Neuve, Mons et Woluwe. Chaque établissement propose une carte fixe ainsi que des menus et des soupes qui changent quotidiennement. Le menu est encodé manuellement chaque semaine dans le CMS Druppal. Les modifications se font uniquement en

11. Rapport Annuel 2015-2016, Éditeur responsable administration des relations extérieures et de la communication (AREC), Daniel Rahier, place de l'université 1 bte 10.01.08, 1348 louvain-la-neuve © uCl septembre 2016

terme de HTML et CSS à l'aide d'une interface d'édition présente sur le site. Il n'y a donc pas de serveur ou d'API permettant de récupérer les informations sur les restaurants universitaires et leur menu.

Nous avons présenté le projet de l'application et les résultats de l'enquête à Monsieur Garry Van Loo et Monsieur Olivier Francq respectivement directeur et adjoint de direction - opérationnel. Le projet a été très bien accueilli. Ils nous ont mis en contact avec Monsieur Jérôme Migeotte, Web design - F&B Analyst, pour toutes les questions techniques. Lors de notre rencontre, ils nous ont également présenté leur projet d'application Resto4u reprenant toutes les informations importantes sur les services offerts aux clients. Le développement de cette application a été confiée à la société Apptree. Apptree est une société belge qui développe des applications mobiles sous IOS et Android. A ce jour, l'application est disponible sur les stores mais est toujours en cours de développement afin de faire quelques ajustements.

3.1.5.3 Les bibliothèques de l'Université catholique de Louvain (BIUL)

Les 10 bibliothèques principales de l'Université catholique de Louvain sont coordonnées par le service central des bibliothèques (SCEB). Ce service administratif gère également les ressources centrales et représente les bibliothèques à l'extérieur. Sur le portail de l'université, chaque bibliothèque possède sa propre page contenant des informations spécifiques à chaque bibliothèque telles que les horaires, les contacts, les services proposés, les ressources documentaires, les aides et formations, des informations sur l'espace et l'équipement ainsi qu'un flux d'actualités et un agenda. Il existe une équipe informatique spécifique aux bibliothèques. Les ressources informatiques et numériques nécessaires pour satisfaire les services offerts par les bibliothèques sont diverses et variées. Ci-dessous se trouve une liste exhaustive des ressources utilisées :

- Boreal : Catalogue collectif de l'UCLouvain, UNamur et Saint Louis.
- Discovery est un outil de recherche donnant accès à de multiples ressources telles que le catalogue de l'UCL, des bases de données, des revues électroniques, des articles de presse, des articles scientifiques.
- Le portail Libellule accessible à partir du site web des bibliothèques permet aux utilisateurs d'accéder au catalogue informatisé UCL, aux fichiers numérisés de l'ancienne bibliothèque centrale, au catalogue de l'Académie Louvain et au catalogue collectif Unicat.
- DIAL.pr (Digital Access to Libraries – Publications de recherche) : dépôt institutionnel permettant la consultation, la diffusion et la conservation à long terme, en version électronique, des publications scientifiques des institutions partenaires à travers une interface web.
- Des flux RSS permettant de suivre les acquisitions effectuées par les différentes bibliothèques.

Une rencontre a eu lieu dans les locaux du SCEB en présence de Messieurs Charles-Henry Nyns, bibliothécaire en chef, Laurent Dubois informaticien dans l'équipe Bibsys et Fabrizio Tinti, Directeur – Bibliothèque des sciences économiques, sociales et politiques (BSPO). Différents sujets ont été abordés. Le premier point fut l'accès aux informations spécifiques à chaque bibliothèque présente sur le portail internet de l'université et, plus particulièrement leurs horaires d'ouverture. Chaque bibliothèque a trois types d'horaires possibles :

- horaire en période de cours
- horaire en période d'examen
- horaire en période de vacances

Il a été convenu que l'application pourra effectuer des requêtes sur deux services REST renvoyant des réponses sous forme de JSON. Ces services ont été réalisés par Laurent Dubois et l'équipe Bibsys. La structure des deux services est la suivante :

- Un service renvoyant une liste des bibliothèques
- Un service renvoyant en fonction de la bibliothèque, une note concernant les horaires, les jours de fermeture et les informations de contact (téléphone, email, ...)

Un second sujet de discussion a été le calcul de l'affluence et des places disponibles dans les bibliothèques. La tendance est à la hausse pour l'affluence dans les bibliothèques. Les étudiants cherchent des lieux propices à l'étude en période de cours ou de blocus. Afin de faciliter la recherche de ces lieux, l'application pourrait offrir un service affichant la fréquentation et le nombre de places disponibles dans chaque bibliothèque. Etant donné que les visiteurs de chaque bibliothèque doivent badger à l'entrée et à la sortie, une telle fonctionnalité semblait envisageable. Malheureusement, le système de comptage n'est actuellement pas adapté pour envoyer ces informations en temps réel. D'autres pistes ont été lancées telles que les applications et logiciels Affluence et Seatfinder.

3.1.5.4 Services des sports

Malheureusement et malgré de nombreuses tentatives, il n'a pas été possible de rencontrer la personne responsable de ce service. Le service des sports propose différentes activités sportives sur cinq campus de l'université à savoir Louvain-La-Neuve, Woluwe, Saint Gilles, Mons et Tournai. Sur le campus de Louvain-La-Neuve, le service des sports dispose de trois écrans diffusant en ville l'horaire des différentes séances de sport de la journée accessibles aux étudiants disposant de la carte sport. Ces écrans doivent probablement utiliser une base de données ou un serveur pour récupérer les informations affichées. Une partie du portail web de l'université est destinée au service des sports. Chaque campus dispose, également, de ses propres pages diffusant les informations suivantes :

- Les informations concernant l'affiliation
- L'offre sportive sur le campus et l'horaire
- Les informations pratiques : description du service, assurances, partenaire, ...
- Description des équipes universitaires

3.1.5.5 Service général du système d'information

Site internet de l'UCL Le portail web de l'UCL est l'outil de communication principal et central de diffusion d'informations et d'offre de services numériques fournis par l'université. Il est accessible à tous les visiteurs, étudiants, membres du personnel. Chaque utilisateur a accès à certaines parties du site en fonction de son statut. Le site a récemment migré du système de gestion de contenu reddot au système de gestion de contenu Druppall. Chaque département au sein de l'université est responsable de l'affichage et de la mise à jour du contenu des informations dans les parties du site qui lui sont attribuées. Le portail est le noeud central pour les étudiants car il permet de faire l'interface avec les différents services et outils fournis par l'université comme l'accès aux résultats, l'inscription à l'année académique, l'inscription aux cours ou aux examens à travers OSIS ainsi que l'accès aux horaires de cours à l'aide de l'outil ADE. Il propose également un lien vers le site Moodle. De plus, les utilisateurs connectés peuvent accéder à leur messagerie universitaire.

Moodle Moodle est un projet Open Source de plateforme d'apprentissage. Ce projet a pour objectif de fournir un système pour créer un environnement d'apprentissage personnalisé aux enseignants et étudiants. L'Université catholique de Louvain utilise actuellement la version 3.3 avec sa propre sur-couche. La plateforme permet aux professeurs de diffuser leurs contenus de cours, réaliser des tests en ligne, répondre à des questions ou diffuser des annonces sur les forums, gérer la remise des travaux et leur correction. Les étudiants peuvent s'inscrire aux cours qu'ils suivent, soumettre leurs travaux sur la plateforme. La plateforme offre un calendrier reprenant l'ensemble des dates de remise de travaux et des événements importants. De nombreuses autres fonctionnalités sont fournies par la plateforme Moodle qui propose également son application mobile.

OSIS OSIS + EPC est un système visant à faciliter la gestion de la partie administrative d'un établissement scolaire de degré supérieur. Le projet est Open Source et sous licence GPL v3. Les fonctionnalités offertes par le système sont la gestion du parcours scolaire d'un étudiant, la planification d'une année académique, la gestion des partenariats avec d'autres établissements, et l'attribution des cours aux enseignants. Actuellement le projet est axé sur le fonctionnement de l'Université catholique de Louvain. Le projet étant Open Source, il est en évolution permanente et souple.

ADE expert ADESOFT est un éditeur de logiciels dédiés à la planification des ressources et à la construction et l'optimisation d'emplois du temps complexes. Celui-ci est utilisé par l'université pour gérer les horaires de cours et la répartition des locaux.

Services Desk Le Service Desk est le centre d'assistance informatique de l'UCL. Il offre un point de contact privilégié avec les utilisateurs. Les demandes d'assistance peuvent être faites par mail ou par téléphone. L'objectif est d'aider les membres du personnel ainsi que les étudiants à résoudre leurs problèmes informatiques en les dirigeant, si besoin est, vers le bon interlocuteur.

3.2 Spécifications fonctionnelles

Cette section va décrire les spécifications fonctionnelles attendues de l'application. Celles-ci résultent de l'analyse du contexte et des besoins, énoncés précédemment, auxquels elle doit répondre. Les spécifications fonctionnelles vont décrire le comportement attendu de l'application en terme d'activités.

3.2.1 Front-end

Cette section se concentre exclusivement sur les fonctionnalités attendues de l'interface entre les utilisateurs et l'application.

3.2.1.1 Structure de l'information

L'arborescence des pages n'est pas profonde, afin que l'utilisateur ne se perde pas dans un dédale sans fin de pages pour trouver l'information qu'il désire. Pour ce faire, l'application est limitée à une profondeur de deux pages. Depuis toutes les pages racines se trouvant au deuxième étage de l'arbre, il est possible d'accéder directement aux autres pages racines se trouvant au même niveau, à l'aide du menu de navigation. Pour les pages enfants des pages racines, le bouton retour en arrière les ramène à la page parent précédente. Avec une telle structure horizontale, il est important de bien catégoriser les fonctionnalités afin qu'il n'y en ait pas trop en vue de minimiser le nombre pages racines différentes pour que l'utilisateur se retrouve facilement dans la structure.

3.2.1.2 Description des fonctionnalités par écran

News La page news est également la page d'accueil de l'application. Elle contiendra une liste des nouvelles diffusées. L'utilisateur peut s'abonner ou se désabonner des différents flux en fonction de ses envies.

Détails news Affiche l'ensemble détaillé des informations pour une nouvelle particulière.

Événements Affiche une liste des événements en cours ou à venir sur le campus. Cette liste peut être filtrée selon différents paramètres : le campus, le type d'activité, l'écart de temps avec l'événement à venir. L'événement peut également être ajouté à son agenda ou à ses événements favoris.

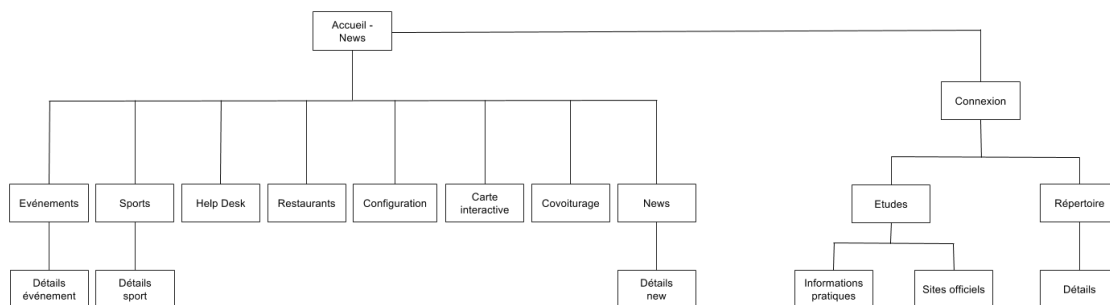


FIGURE 3.12 – Arborescence des pages

Détails événement Affiche de plus amples informations sur l'événement sélectionné. Les actions possibles sont les mêmes que pour la liste des événements. Une action supplémentaire est d'être dirigé directement sur la page de l'événement.

Sports Affiche une liste des activités sportives en cours ou à venir sur le campus. Cette liste peut être filtrée selon différents paramètres : le lieu, le type de sport, l'heure du début de l'activité. On peut ajouter une séance de sport à son agenda de manière ponctuelle ou de manière périodique. Une séance de sport peut être ajoutée comme favori.

Détails sport En sélectionnant, dans la liste, une séance de sport, l'utilisateur peut accéder aux informations détaillées de celle-ci.

Services Desk Affiche les informations utiles concernant la connexion au réseau eduroam, et d'autres informations utiles aux étudiants nouvellement arrivés.

Restaurants Lance l'application des restaurants universitaires Resto4u si celle-ci est déjà installée sur le smartphone. Si ce n'est pas le cas, l'application lance le store pour proposer de l'installer. Lorsque l'utilisateur a fini avec l'application Resto4u, il revient sur la page "restaurants" qui affiche des informations pratiques sur ceux-ci.

Configuration Page permettant à l'utilisateur de choisir la langue d'affichage et de se déconnecter.

Carte interactive Affiche une carte Google map avec une liste des différents points d'intérêt tels que les cercles étudiants, les kots à projet, les bibliothèques, les auditoriums, Les points d'intérêt sont également affichés sur la carte à l'aide de marqueur. L'utilisateur pourra afficher l'itinéraire jusqu'au point d'intérêt sélectionné.

Covoiturage Lance l'application de la plateforme Commuty si l'application est installée sur le smartphone. Si ce n'est pas le cas, l'application propose d'installer l'application depuis le store. Lorsque l'utilisation de l'application est terminée, l'utilisateur revient vers la page covoiturage qui affiche des informations concernant le covoiturage au sein de l'université.

Connexion Cette page permettra à l'utilisateur, faisant partie de la communauté UCL, de se connecter et d'accéder à certaines fonctionnalités sécurisées.

Études Affiche la liste des cours auxquels l'utilisateur est inscrit. La liste serait pré-remplie à l'aide des informations connues sur le bureau virtuel de l'étudiant. L'utilisateur peut

également ajouter des cours selon ses désirs. Sur cette page, l'utilisateur a également accès au lien vers les sites Internet officiels de l'université (moodle et portail UCL) . L'utilisateur peut également accéder aux résultats obtenus pour les cours auxquels il est inscrit.

Informations pratiques La page affiche les informations pratiques sur le cours sélectionné. Le terme "informations pratiques" sous-entend les dates de remise de travaux, les horaires des cours magistraux, des séances de travaux pratiques ainsi que les auditorios où les séances sont données.

Répertoire Page permettant d'effectuer une recherche dans le répertoire des étudiants ou le répertoire des membres du personnel de l'université en fonction d'un nom, d'un prénom ou d'une année d'étude.

Détails La page Détails permet d'afficher les informations spécifiques d'un étudiant ou d'un membre du personnel de l'université.

3.2.1.3 Maquettes

Dans cette section se trouvent les premières ébauches visuelles des différentes pages de l'application. Il y a un menu central fournissant l'accès à toutes les pages principales. Les pages ont toutes une structure similaire afin d'offrir à l'utilisateur une manipulation intuitive.

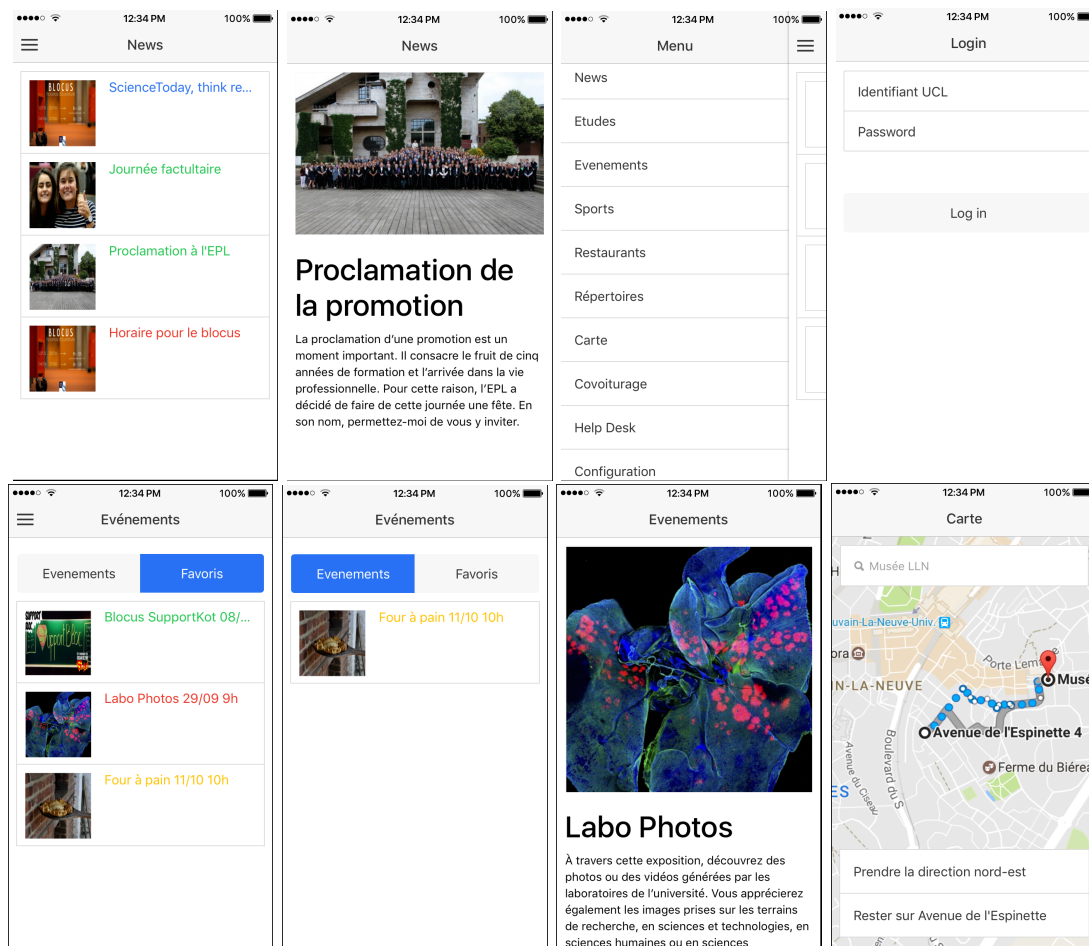


FIGURE 3.13 – Illustrations de l'application UCLCampus

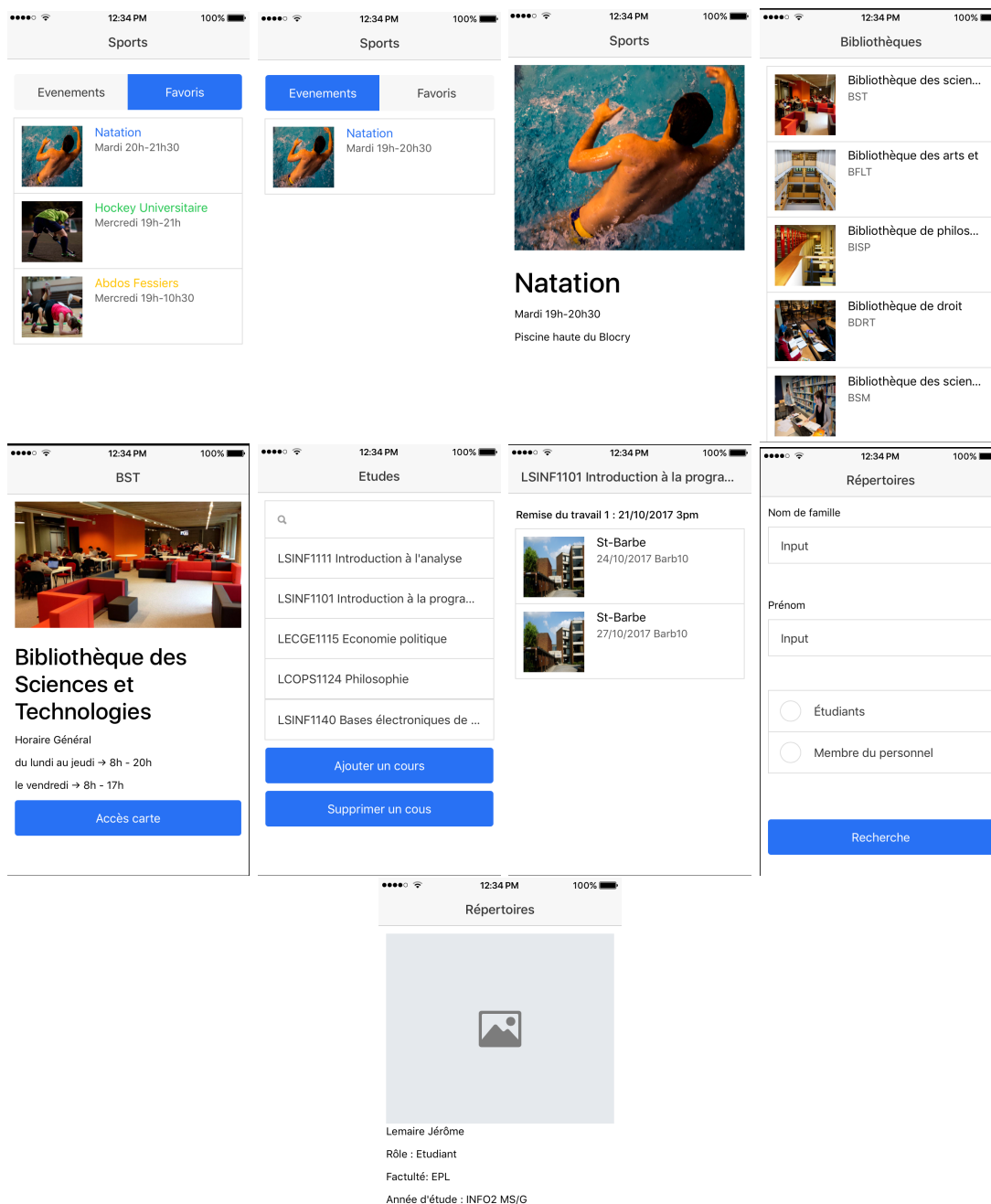
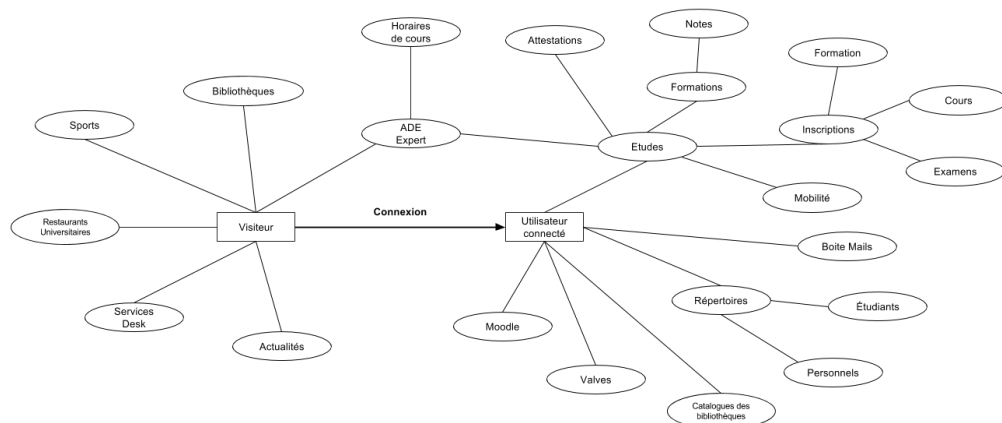


FIGURE 3.14 – Illustrations de l'application UCLCampus

Dans cette section, les fonctionnalités et les contraintes non visibles par l'utilisateur sont abordées.

Cette section décrit les processus d'accès aux informations pour toute personne se déplaçant sur le portail UCL. Un visiteur est, comme son nom l'indique, une personne visitant le site à la recherche d'informations. Un utilisateur connecté est un membre de l'Université catholique de Louvain qui possède un identifiant et un mot de passe lui permettant d'accéder à des informations supplémentaires et personnelles. Un utilisateur connecté n'a pas accès aux informations personnelles d'un autre utilisateur connecté. L'horaire des cours est accessible via plusieurs chemins, en tant qu'utilisateur connecté ou en tant que visiteur. En tant que visiteur, je peux accéder aux horaires de tous les cours soit en passant par la page descriptive du cours et en cliquant sur le bouton horaire, soit en me connectant directement sur la page ADE Expert de l'UCL et en m'identifiant avec l'utilisateur "etudiant" et le mot de passe "student" qui sont disponibles publiquement sur les pages du portail de l'UCL. L'utilisateur connecté peut, lui, accéder directement à son horaire de cours sur ADE Expert après avoir enregistré les sigles des cours suivis durant le quadrimestre.



3.2.2.2 Sécurité

Pour la partie non connectée de l'application, l'aspect sécurité n'est pas un aspect crucial car l'ensemble des informations échangées sont du domaine public. Par contre, pour la partie connectée, l'application doit garantir la sécurité des échanges d'informations entre un utilisateur connecté et les serveurs de données de l'université. Les utilisateurs devront se connecter à l'application avec leur identifiant global UCL et leur mot de passe : ces informations doivent donc être cryptées lorsqu'elles passent sur le réseau. De plus, ces informations ne peuvent pas être stockées de manière naïve sur le smartphone. Les informations échangées telles que notes de cours, données du répertoire des étudiants et du personnel ne sont pas du domaine public et

doivent également être échangées de manière sécurisée. Les cours suivis et les horaires de cours, quant à eux, peuvent être stockés sur le smartphone car il s'agit de données moins sensibles.

3.2.3 Récits utilisateur

Les récits utilisateur permettent de définir les fonctionnalités attendues en fonction du type d'utilisateur manipulant l'application. C'est un outil régulièrement utilisé dans la méthode de développement Agile.

News

- En tant qu'utilisateur, je veux afficher une liste de nouvelles afin de pouvoir me tenir informé des nouvelles.
- En tant qu'utilisateur, je veux m'abonner à un flux de nouvelles facultaire afin d'être tenu au courant des nouvelles qui m'intéressent.
- En tant qu'utilisateur, je veux me désabonner d'un flux de nouvelles afin de n'afficher que les nouvelles qui m'intéressent.
- En tant qu'utilisateur, je veux trier les nouvelles.
- En tant qu'utilisateur, je veux rechercher une nouvelle par mot clé.
- En tant qu'utilisateur, je veux afficher en détail toutes les informations liées à une nouvelle.
- En tant qu'utilisateur, je veux pouvoir être dirigé vers le site contenant la nouvelle affichée.

Évènements

- En tant qu'utilisateur, je veux afficher la liste des activités en cours et à venir.
- En tant qu'utilisateur, je veux afficher la description d'une activité en particulier.
- En tant qu'utilisateur, je veux afficher l'horaire d'une activité en particulier.
- En tant qu'utilisateur, je veux filtrer la liste des activités selon différents paramètres : heure, type d'activité, lieu.
- En tant qu'utilisateur, je veux ajouter une activité à mon agenda.

Sports

- En tant qu'utilisateur, je veux afficher la liste des sports en cours et à venir.
- En tant qu'utilisateur, je veux afficher la description d'un sport en particulier.
- En tant qu'utilisateur, je veux afficher l'horaire d'un sport en particulier.
- En tant qu'utilisateur, je veux filtrer la liste des sports selon différents paramètres : heure, type de sport, lieu.
- En tant qu'utilisateur, je veux ajouter une séance de sport à mon agenda.
- En tant qu'utilisateur, je veux ajouter une séance de sport de manière périodique à mon agenda.

Service Desk

- En tant qu'utilisateur, je veux afficher les informations d'aide informatique proposées par l'UCL pour les étudiants.

Configuration

- En tant qu'utilisateur, je veux changer la langue d'affichage de l'application.
- En tant qu'utilisateur, je veux me déconnecter de mon compte afin d'éviter qu'un autre utilisateur puisse y accéder.
- En tant qu'utilisateur, je veux me connecter à mon compte afin d'accéder à de nouvelles fonctionnalités.
- En tant qu'utilisateur, je veux accéder à mon adresse afin de pouvoir consulter mes mails.

- En tant qu'utilisateur, je veux activer ou désactiver la réception de notifications provenant de l'application.

Restaurants

- En tant qu'utilisateur, je veux afficher les menus des restaurants universitaires.
- En tant qu'utilisateur, je veux connaître l'horaire des restaurants universitaires.
- En tant qu'utilisateur, je veux pouvoir lancer l'application Resto4u afin de connaître toutes les informations diffusées par les restaurants universitaires.
- En tant qu'utilisateur, je veux pouvoir télécharger l'application Resto4u.

Covoiturage

- En tant qu'utilisateur, je veux lancer l'application commute afin d'avoir accès aux services de covoiturage de l'UCL.
- En tant qu'utilisateur, je veux pouvoir télécharger l'application Commute.

Carte interactive

- En tant qu'utilisateur, je veux connaître ma position sur la carte.
- En tant qu'utilisateur, je veux connaître la localisation des points d'intérêt sur la carte.
- En tant qu'utilisateur, je veux connaître l'itinéraire pour atteindre le point d'intérêt.
- En tant qu'utilisateur, je veux chercher un point d'intérêt par mots clés.
- En tant qu'utilisateur, je veux afficher la liste des points d'intérêt par catégorie.

Études

- En tant qu'utilisateur connecté, je veux synchroniser la liste des cours auxquels je suis inscrit sur mon bureau virtuel.
- En tant qu'utilisateur connecté, je veux synchroniser la liste des cours auxquels je suis inscrit sur Moodle.
- En tant qu'utilisateur connecté, je veux supprimer un cours de la liste des cours suivis.
- En tant qu'utilisateur connecté, je veux ajouter manuellement un cours à la liste des cours suivis.
- En tant qu'utilisateur connecté, je veux pouvoir accéder aux sites officiels de l'université.
- En tant qu'utilisateur connecté, je veux afficher les dates de remise de travaux d'un cours.
- En tant qu'utilisateur connecté, je veux afficher les horaires d'un cours présent dans ma liste de cours.
- En tant qu'utilisateur connecté, je veux afficher l'auditoire dans lequel j'ai cours pour un cours présent dans ma liste de cours.
- En tant qu'utilisateur connecté, je veux recevoir une notification 15 min avant le début d'un cours.
- En tant qu'utilisateur connecté, je veux recevoir une notification 1h avant l'heure de remise d'un travail.
- En tant qu'utilisateur connecté, je veux régler les délais des notifications.
- En tant qu'utilisateur connecté, je veux pouvoir sélectionner les travaux pratiques auxquels je suis inscrit.
- En tant qu'utilisateur connecté, je veux pouvoir ajouter mon horaire de cours à l'agenda de mon téléphone.
- En tant qu'utilisateur connecté, je veux pouvoir synchroniser mon horaire de cours à l'agenda de mon téléphone.
- En tant qu'utilisateur connecté, je veux pouvoir ajouter une remise de travail à l'agenda de mon téléphone.

Répertoires

- En tant qu'utilisateur connecté, je veux rechercher un étudiant par son nom de famille, son prénom ou son année d'étude.
- En tant qu'utilisateur connecté, je veux afficher le profil d'un étudiant.
- En tant qu'utilisateur connecté, je veux rechercher un membre du personnel de l'université par son nom de famille ou son prénom.
- En tant qu'utilisateur connecté, je veux afficher le profil d'un membre du personnel.

3.3 Spécifications non-fonctionnelles

Cette section précise les spécifications non-fonctionnelles de l'application telles que notamment la fiabilité, la modularité, les performances, la portabilité et l'aspect légal.

3.3.1 Facilité d'utilisation

C'est l'ensemble d'attributs qui tiennent compte de l'effort nécessaire à l'utilisation et à l'amélioration du logiciel par un ensemble d'utilisateurs.

Compréhension L'application devra avoir une manipulation intuitive. Un guide utilisateur sera rédigé ainsi qu'un tutoriel vidéo. De plus, lors du premier lancement de l'application, une visite guidée de l'application sera proposée à l'utilisateur.

Apprentissage Le projet étant Open Source, la présence d'un fichier readme et d'un guide à destination des développeurs sont primordiaux pour permettre à tout programmeur désireux de se joindre au projet facilement, sans avoir un coût élevé d'apprentissage. Les noms des fonctions et des variables devront être choisis judicieusement. Le code sera également commenté judicieusement.

Opérabilité L'application doit rester opérationnelle en absence de connexion internet. Pour ce faire, elle doit conserver en cache les informations récupérées lors des dernières utilisations. Lorsque les services de l'UCL évoluent ou sont en maintenance, l'application doit toujours être fonctionnelle.

3.3.2 Langages de programmation

L'application doit être développée avec le framework Ionic version 3. Ce framework est actuellement basé sur Angular 4.0, TypeScript, JavaScript et HTML5.

3.3.3 Fiabilité

L'application se doit d'être la plus fiable possible et éviter les problèmes techniques évidents.

3.3.4 Maintenabilité

Testabilité En plus de l'application, une batterie de tests doit être développée. Cette batterie de tests doit être composée de tests unitaires, de tests d'intégration et de tests fonctionnels de bout en bout.

Modularité L'application doit respecter le concept d'une architecture en trois couches : Données, Logique, Interface. De plus, il faut une indépendance entre les services développés dans l'application.

Interopérabilité D'autres universités doivent pouvoir adapter l'application à leur propres services informatiques. Cette demande a pour conséquence que le code de l'application doit être générique pour pouvoir permettre le changement/switch des services auxquels les requêtes sont adressées.

3.3.5 Performances

Temps de réponse Pour le confort, des utilisateurs le temps de chargement d’une page ne doit pas excéder une seconde. Si le chargement dépasse cette limite, une logo de chargement doit apparaître.

Utilisation des ressources L’application ne devra pas exploiter trop de mémoire cache. De plus, l’application ne doit pas être trop énergivore afin de préserver au mieux la batterie du smartphone.

Scalability L’objectif de l’application est d’atteindre les 20 000 utilisateurs, elle doit être capable de supporter de grosses affluences.

3.3.6 Compatibilité

Installation L’application doit tourner sur les smartphones supportant la version 4.4 d’Android ou plus récente. Pour les smartphones tournant sur le système d’exploitation IOS, les versions 8 ou plus récentes sont exigées. L’application doit être disponible sur le Playstore et l’App Store.

Mises à jour Les versions des mises à jour respecteront la convention du semantic versioning¹².

3.3.7 Légal

Licences L’application doit être sous licence GPL 3.0.

Charte graphique L’UCL a une entité visuelle qui lui est propre et qui doit être utilisée à bon escient. Pour ce faire, l’Université catholique de Louvain a défini certaines règles en termes de communication et d’image. Ces règles sont définies dans une charte graphique. L’application étant à destination des étudiants, elle devra respecter les règles présentes dans cette charte. Il n’existe qu’une seule version du logo de l’UCL.



FIGURE 3.16 – Logo officiel de l’UCL en blanc sur fond Pantone 282

12. <http://semver.org>

Chapitre 4

Développement de l'application UCLCampus

Ce chapitre présente la manière dont l'application a été développée. La première section décrit la mise en place d'un environnement propice au développement du projet Open Source. La mise en place des tests d'intégration continu, autre composant primordial de l'environnement de développement du projet, fait également l'objet d'une section. La section suivante discute des choix d'implémentation posés tels que les choix de l'architecture et les interactions entre les modules. L'aspect sécurité est également présenté pour ce qui concerne le stockage d'information sensible et l'authentification aux services de l'Université catholique de Louvain. Finalement, la dernière section détaille le fonctionnement de chaque fonctionnalité de l'application.

4.1 Approche Open Source

La réalisation d'un projet Open Source nécessite la mise en place d'un environnement propice au développement de celui-ci. Pour mettre en place un tel environnement, il a fallu choisir la licence et respecter ses termes, choisir la forge la plus adaptée pour le développement du projet et mettre en place des canaux de communication pour informer la communauté de l'état d'avancement du projet. La dernière section présente la problématique de conserver des données confidentielles dans un projet Open Source et la solution mise en place.

4.1.1 Choix d'une licence

Il faut protéger le projet contre tous les détournements qui ne sont pas en adéquation avec les valeurs et la philosophie du projet et de l'Université catholique de Louvain. Le projet UCLCampus est un produit fini s'adressant directement aux utilisateurs finaux, et non à d'autres développeurs, qui doit être protégée contre toute utilisation à des fins commerciales. L'objectif est de rendre l'application disponible gratuitement sur les stores d'application pour qu'un maximum de personnes présentes sur les campus universitaires puisse l'utiliser. Il ne faut pas que le projet puisse être repris par une société qui ajoute une interface ou une surcouche plus accrocheuse à l'application existante sans en faire profiter le projet soutenu par l'université. Une telle situation de concurrence serait néfaste. Sans licence restrictive sur l'utilisation du projet, il pourrait être détourné pour récolter des informations sur les étudiants de l'université. L'utilisation d'une licence restrictive obligeant la société à rendre son code open source sous la même licence va réduire la marge de manoeuvre de celle-ci et permettra à l'UCL d'être tenue au courant du contenu de l'application concurrente utilisant pour base le projet initial. L'université pourra également améliorer l'application officielle avec les fonctionnalités supplémentaires développées par les applications dérivées

L'application est Open Source dans le but de permettre à d'autres universités, ou étudiants, de pouvoir s'approprier le code existant afin de l'adapter pour qu'il convienne au mieux à leurs besoins. Il faut donc que la licence soit compatible avec d'autres licences pour permettre l'utilisation d'un large panel de plugins et de librairies.

La licence la plus adaptée au regard des critères énoncés précédemment est la GNU General Public License v3.0. Elle protège le projet de dérive commerciale en étant copyleft, ce qui oblige tout utilisateur du code source de mettre son projet également sous licence GNU GPL v3.0 s'il désire distribuer son projet. De plus, la licence est compatible de manière unidirectionnelle, c'est-à-dire qu'elle est très restrictive, ce qui permet d'autoriser l'intégration de code provenant d'autres licences moins restrictives et non contradictoires, telles que MIT, Apache 2.0, mais ne permet pas l'utilisation de code source sous des licences moins restrictives.

Afin de développer le projet UCLCampus sous licence, il faut que chaque fichier contienne l'avis suivant :

```
Copyright 2017 Universite catholique de Louvain
```

```
This file is part of UCLCampus.
```

```
UCLCampus is free software: you can redistribute it and/or modify  
it under the terms of the GNU General Public License as published by  
the Free Software Foundation, either version 3 of the License, or  
(at your option) any later version.
```

```
UCLCampus is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License  
along with UCLCampus. If not, see <http://www.gnu.org/licenses/>.
```

La première ligne est l'avis de copyright et, la suite est la déclaration de la licence. En plus, le projet contient un fichier LICENCE détaillant le contenu de la licence GNU General Public License v3.0. Le copyright appartient à l'Université catholique de Louvain afin de permettre une plus grande flexibilité dans la gestion du code source de l'application. En effet, avec un copyright nominatif pour les développeurs et avec la licence restrictive GPL-v3, l'université devrait contacter l'ensemble des contributeurs du projet pour leur demander l'accord en cas de changement de licence. Il est donc préférable que l'université soit propriétaire du code source et que les contributeurs acceptent de céder leurs droits d'auteur sur leur participation au code source du projet UCLCampus.

4.1.1.1 Licences des frameworks et plugins utilisés

Les frameworks Ionic et Angular sont sous licence MIT ce qui ne pose aucun problème avec la licence choisie. Il en est de même pour la licence Apache-2.0 sous laquelle se trouve le framework Cordova.

Une attention particulière a été portée aux différentes licences sous lesquelles se trouvaient les plugins utilisés dans l'application. Généralement, les plugins sont licenciés sous Apache-2.0 et MIT. Il n'y a donc pas de problème de compatibilité avec la licence GPL-v3 choisie pour l'application.

4.1.2 Forge utilisée

Une forge est un système de gestion de développement collaboratif de logiciel. Github¹ a été choisi comme forge pour le projet UCLCampus. Github est une plateforme propriétaire de projet

1. <https://github.com>

Plugins	Licences
cordova-plugin-market	MIT
cordova-plugin-appavailability	MIT
cordova-plugin-calendar	MIT
cordova-plugin-console	Apache-2.0
cordova-plugin-device	Apache-2.0
cordova-plugin-geolocation	Apache-2.0
cordova-plugin-inappbrowser	Apache-2.0
cordova-plugin-network-information	Apache-2.0
cordova-plugin-splashscreen	Apache-2.0
cordova-plugin-statusbar	Apache-2.0
cordova-plugin-whitelist	Apache-2.0
cordova-sqlite-storage	MIT ou Apache-2.0
ionic-plugin-keyboard	Apache-2.0
cordova-plugin-googlemaps	Apache-2.0

TABLE 4.1 – Tableau des licences utilisées par l’application UCLCampus

libre. Les raisons de ce choix sont les suivantes :

- Github utilise le logiciel de gestion de version Git.
- Github propose un service web d’hébergement et de gestion de développement de logiciel. Son interface est conviviale et intuitive.
- Github est une forge fortement utilisée avec plus de 14 millions d’utilisateurs et 35 millions de projets en Avril 2016 selon leur propre chiffre.
- Github est une forge "sociale". En effet, des nombreuses fonctionnalités offertes par Github se rapprochent de celles proposées par les réseaux sociaux. Tous les utilisateurs Github ont un compte personnel. Les utilisateurs peuvent suivre d’autres utilisateurs ou des projets. Les utilisateurs peuvent communiquer entre eux au sujet d’un projet. Tous les utilisateurs peuvent voir les activités (commit, fork, ...) des autres utilisateurs, ce qui augmente la visibilité des projets.
- Github propose également d’intégrer des services externes à ses projets pour faciliter le développement ou l’interaction avec les contributeurs et les utilisateurs du projet. Les services externes utilisés pour le projet UCLCampus sont détaillés dans les sous-sections suivantes.

4.1.2.1 Zenhub

ZenHub² est un outil de gestion de projet qui s’intègre à l’interface utilisateur de GitHub suite à l’ajout d’une extension au navigateur Chrome ou Firefox. L’apport principal de l’outil ZenHub est son tableau de bord qui utilise les données Github existantes pour assurer un meilleur suivi dans le processus de développement du projet. Les utilisateurs peuvent ajouter des issues, c’est-à-dire des sujets ou problèmes importants à débattre ou à régler, au tableau de bord. Les issues sont transférées d’une colonne à l’autre en fonction de leur état d’avancement. Cette fonctionnalité permet à la communauté d’interagir et de donner un retour sur le projet et les problèmes rencontrés. Une autre fonctionnalité disponible, qui n’a pas été utilisée dans le cadre de ce projet, est la possibilité de regrouper les issues ensemble dans le but de faire de la programmation agile et de définir des sprints. Des représentations graphiques sur l’avancement

2. <https://www.zenhub.com>

de chaque sprint sont possibles pour permettre à l'équipe de mieux estimer sa charge de travail pour les sprints suivants.

Github propose également un outil similaire avec "Projects" présent automatiquement sur tous les dépôts de projets. Le choix s'est cependant porté sur l'outil ZenHub car l'outil est plus complet et offre plus de perspective.

4.1.2.2 Travis CI

Travis CI³ est un logiciel libre et un service en ligne utilisé pour compiler, tester et déployer le code source des logiciels développés. Ce logiciel soutient l'intégration continue du projet UCLCampus. Le logiciel est publié sous licence MIT. La configuration du projet pour utiliser cet outil nécessite quelques manipulations et l'ajout d'un fichier de configuration spécifique à l'outil pour lui donner les informations sur l'environnement de compilation et de tests. Lorsqu'une modification est soumise sur le projet Github, le service Travis CI compile et exécute les tests.

4.1.2.3 Gitter

Gitter⁴ est un système de messagerie pour les développeurs et les utilisateurs des dépôts Github. Ce service offre des salles de discussion liées à des dépôts de projet GitHub. Une salle de discussion publique a été créée pour le projet UCLCampus. De plus, un badge graphique reliant le projet à la salle de discussion a été placé dans le fichier README du projet, pour attirer l'attention de tous les utilisateurs et développeurs du projet.

4.1.3 Communauté

La communauté est l'ensemble des personnes qui participent à la vie du projet allant du développeur à l'utilisateur parlant du projet à d'autres personnes. Un projet Open Source doit avoir une communauté forte et active pour pouvoir continuer à évoluer et survivre. Dans le projet UCLCampus, la communauté cible est composée d'étudiants en informatique en tant que développeur potentiel, d'étudiants de l'Université catholique de Louvain en tant qu'utilisateur. D'autres universités pourraient également être intéressées par un tel projet. La création d'une communauté demande un plus grand investissement que la simple réalisation d'un manuel utilisateur, d'un tutoriel vidéo et d'un manuel développement qui sont des éléments de base pour faciliter l'intégration au projet. En effet, le développement d'une communauté autour du projet UCLCampus doit se faire, en continu, de la sélection des différents outils utilisés à l'utilisation de divers canaux de communication durant la vie du projet. L'utilisation de Github, qui a une forte fréquentation, a été un premier choix dans ce sens. Le choix et l'utilisation des extensions proposées par Github a suivi la même logique. Comme vu précédemment, Gitter et ZenHub sont des outils facilitant les échanges et le partage au sein de la communauté entourant le projet. Github fournit également la possibilité de réaliser un Wiki spécifique à chaque projet. Le Wiki fourni sur le Github du projet définit la feuille de route du projet, montre l'état d'avancement actuel et la documentation du projet. L'avantage d'un Wiki est que tous les membres de la communauté peuvent le modifier et l'enrichir librement.

4.1.4 Données confidentielles

Le projet UCLCampus est développé en Open Source ce qui implique toutes les informations et ressources nécessaires au fonctionnement du projet soient accessibles à tous. Certaines de ces informations peuvent être sensibles et demander une attention particulière. Dans le cadre du développement de l'application UCLCampus, la question est apparue lors de l'utilisation des API

3. <https://travis-ci.org>

4. <https://gitter.im/>

Key pour l'accès aux services Google Maps. L'accès à ces services nécessite une identification de l'application les utilisant. Google fournit des clés uniques pour chaque application pour permettre l'accès aux services, cela permet à Google de faire payer ces services après un certain quota de connexion. Il faut éviter que ces clés soient visibles et n'apparaissent dans le code source sous peine d'être volées et détournées. Les développeurs souhaitant participer au projet doivent faire les démarches auprès de Google pour acquérir leur propre API Key afin de faire fonctionner l'application.

Afin de protéger les API Key des développeurs participants au projet, un fichier de configuration permettant de contenir les API Key a été mis en place sur le Github du projet. Ce fichier de configuration a, ensuite, été ajouté à la liste des fichiers à ignorer en cas de soumission de code sur le Github. Les fichiers utilisant ces API Key font appel aux valeurs et données présentes dans le fichier de configuration. Les développeurs sont invités à mettre localement le fichier de configuration avec leur propre API Key. Lorsque le fichier de configuration est correctement écrit, l'application fonctionne comme attendu et les données présentes dans le fichier ne sont pas visibles sur le Github du projet[36].

4.2 Tests

Les tests jouent un rôle important dans le développement d'une application. Ils permettent aux développeurs de s'assurer de la qualité du code fourni et de la bonne fonctionnalité des modules implémentés. Le projet étant Open Source, il est difficile d'imposer un mode de développement particulier aux développeurs extérieurs. Dès lors, dans le but d'éviter toute régression des fonctionnalités de l'application, un système d'intégration continue a été mis en place à l'aide de l'outil Travis CI. L'intégration continue consiste à vérifier, à chaque modification du code source, que l'application est toujours fonctionnelle en faisant passer une batterie de tests. C'est une sécurité supplémentaire lorsqu'une maintenance ou une factorisation du code est effectuée. En effet, le programmeur peut ainsi s'assurer qu'il n'a pas cassé la moindre fonctionnalité et, si c'est malheureusement le cas, il peut rapidement identifier la fonctionnalité défaillante.

Deux types de tests sont automatisés pour le projet UCLCampus : les tests unitaires et les tests bout en bout. L'application est également testée par la communauté et ses utilisateurs afin de fournir un retour sur les problèmes rencontrés lors de l'utilisation de l'application. Ces problèmes sont recensés sur le projet Github de l'application à travers les issues.

Cette section spécifie la mise en place et le fonctionnement des deux types de tests utilisés lors de l'intégration continue. Les outils utilisés pour développer ces tests sont ceux préconisés par le framework Angular.

4.2.1 Tests unitaires

Les tests unitaires ont pour but de tester un module, indépendamment du reste du programme, afin de s'assurer que le module réponde correctement aux spécifications fonctionnelles et qu'il est robuste en toutes circonstances. La réalisation de ces tests incite les développeurs à rendre leur code modulaire afin de pouvoir tester plus facilement les différents modules indépendamment.

JavaScript est un langage de programmation dynamiquement typé utilisé dans la partie logique de l'application. Étant un langage dynamiquement typé, la détection des erreurs du style faute de frappe dans le nom d'une variable ou appel d'une méthode inexistante n'est pas faisable sans exécuter le code, contrairement à des langages statiquement typés, comme Java, dont le compilateur détecte ce genre d'erreur. L'exécution de code d'application web se fait généralement sur un navigateur web, mais le souci est que chaque navigateur possède sa propre implémentation du DOM et donc certaines différences, malgré la standardisation existante, ce qui demande de tester le code sur plusieurs navigateurs. Une autre solution est d'exécuter du code JavaScript

sans navigateur en utilisant un environnement tel que Node.js mais, ces environnements n'ont pas accès au DOM et aux APIs spécifiques des navigateurs web. Tester manuellement un même code sur plusieurs navigateurs est long et fastidieux, c'est dans ce but d'automatisation du processus de tests unitaires que les environnements de tests ont été développés.

4.2.1.1 Karma

Karma⁵ est un outil Open Source qui a été initialement développé pour AngularJS. Par la suite, il a été étendu pour la réalisation de tests d'applications basées sur d'autres frameworks de développement. Karma a pour objectif de fournir un environnement de tests approprié à tout développeur web. La configuration de l'environnement est simple, afin d'écrire rapidement des tests et d'obtenir les résultats des tests lors de chaque changement du code.

Karma surveille tous les fichiers, spécifiés dans le fichier de configuration et, chaque fois qu'un fichier est modifié, il déclenche l'exécution du test en envoyant un signal au serveur de test pour informer tous les navigateurs d'exécuter le code de test. Chaque navigateur charge les fichiers source, exécute les tests et rapporte les résultats au serveur. Le serveur recueille les résultats de tous les navigateurs et les présente au développeur[37].

Les explications ci-dessus sur le fonctionnement sont suffisantes à la compréhension de l'intérêt de l'utilisation de cet outil pour le développement de tests unitaires. Il est possible d'approfondir le fonctionnement interne de Karma en lisant le mémoire réalisé par Vojtěch Jína en juin 2013[38].

4.2.2 Tests bout en bout

Les tests bout en bout (end to end ou e2e) sont des tests complémentaires aux tests unitaires. Ils permettent de tester les interactions entre les différents modules, là où les tests unitaires consistent à tester le bon fonctionnement de chaque méthode d'un module indépendamment. Les tests bout en bout permettent de tester l'application dans son ensemble, ils simulent la manipulation de l'application par un utilisateur à travers différents scénarios.

4.2.2.1 Protractor

Protractor⁶ est un framework de tests bout à bout développé sous licence MIT qui permet de tester les applications AngularJS. Protractor exécute les tests dans un navigateur web. Un premier processus exécute l'application sur le navigateur et un second processus exécute les tests qui simulent le comportement de l'utilisateur et vérifient que l'application réagit comme prévu. Protractor est un wrapper qui permet la communication entre les tests écrits en JavaScript à l'aide du Framework de test Jasmine et le framework Selenium qui permettent d'automatiser et de simuler les interactions sur les navigateurs. Les scripts de tests envoient les commandes au serveur Selenium par l'intermédiaire de Protractor et, ensuite le serveur Selenium communique à son tour les instructions au navigateur.

4.2.3 Jasmine

Jasmine⁷ est un framework d'écriture de tests en JavaScript sous licence Open Source MIT. C'est le framework utilisé pour rédiger et exécuter les tests pour le projet UCLCampus. Son avantage est qu'il peut exécuter les tests sur toutes les plateformes pouvant exécuter du code JavaScript. Jasmine est le framework utilisé pour écrire les tests unitaires qui sont lancés sur les différents navigateurs web par l'environnement de tests configurés avec Karma. Il permet également d'écrire les tests bout en bout pouvant être exécutés à partir du framework Protractor.

5. <https://karma-runner.github.io/1.0/index.html>

6. <http://www.protractortest.org/#/>

7. <https://jasmine.github.io>

Jasmine offre différentes méthodes permettant de structurer les tests et de vérifier les assertions. Les trois méthodes les plus couramment utilisées pour structurer les tests sont reprises ci-dessous :

describe(description, specDefinitions) permet de décrire un ensemble de tests.

it(description, testFunction) définit un test avec une description du test effectué et une fonction JavaScript testFunction qui contient le code de vérification.

beforeEach(function) Permet de configurer les éléments intervenant dans les tests avant chaque exécution de ceux-ci.

L'utilisation de l'ensemble de ces outils et de l'outil Travis sur le Github du projet offre une double exécution du code. La première exécution du code se fait localement avec le lancement des environnements de tests spécifiques à ce projet à l'aide des lignes de commande suivantes :

```
// Run karma unit tests :  
npm test  
  
// Run Protractor E2E tests :  
  
ionic serve  
npm run e2e
```

Les environnements de tests lancés, ils informent le développeur du passage ou non des tests par le code source modifié. L'outil Travis offre une seconde sécurité car il exécute la batterie de tests lors de chaque soumission de code sur le projet Github.

La prise en main des outils et des environnements de tests n'a pas été chose aisée. La rédaction des tests demande une grande maîtrise du langage et un recul sur le code réalisé pour réussir à tester le code source de manière efficace. Les tests réalisés dans ce projet ne couvrent pas l'ensemble des fonctionnalités de l'application. L'idéal aurait été de réaliser le projet en suivant l'approche Test Driven Development (TDD), prônant la réflexion et le développement des tests avant l'écriture du code permettant de les passer.

4.3 Choix d'implémentation

Cette section décrit et justifie les choix d'implémentation effectués durant la réalisation du projet UCLCampus. L'application est conçue à partir du framework Ionic, la structure et la conception générale de l'application découlant de son utilisation sont présentées dans la première sous-section afin de mieux comprendre les choix qui ont été posés lors du développement. Ensuite, le fonctionnement des différents modules jouant un rôle dans l'interface graphique sont détaillés. La dernière sous-section justifie les choix d'implémentation concernant la partie logique de l'application.

4.3.1 Framework Ionic

Le framework Ionic est le squelette de l'application, il impose la structure et l'architecture générale de l'application. Les programmeurs construisent le corps de l'application en développant les différentes fonctionnalités souhaitées en réalisant des composants et des modules. Le framework Ionic propose une interaction avec deux familles de modules :

Service Les services s'occupent de la partie modèle et logique de l'application.

Page Les pages gèrent les interactions avec l'utilisateur via l'affichage de l'interface utilisateur et la gestion des événements.

Ce ne sont pas les modules qui font appel au framework Ionic mais bien le framework qui fait appel aux différents modules. Les services sont importés par les pages selon leurs interactions avec la partie logique de l'application pour satisfaire les fonctionnalités qui leur sont attribuées.

Les développeurs sont libres de choisir la conception interne de chacun des composants d'un module. Ils définissent également la frontière entre la partie logique et la gestion des événements.

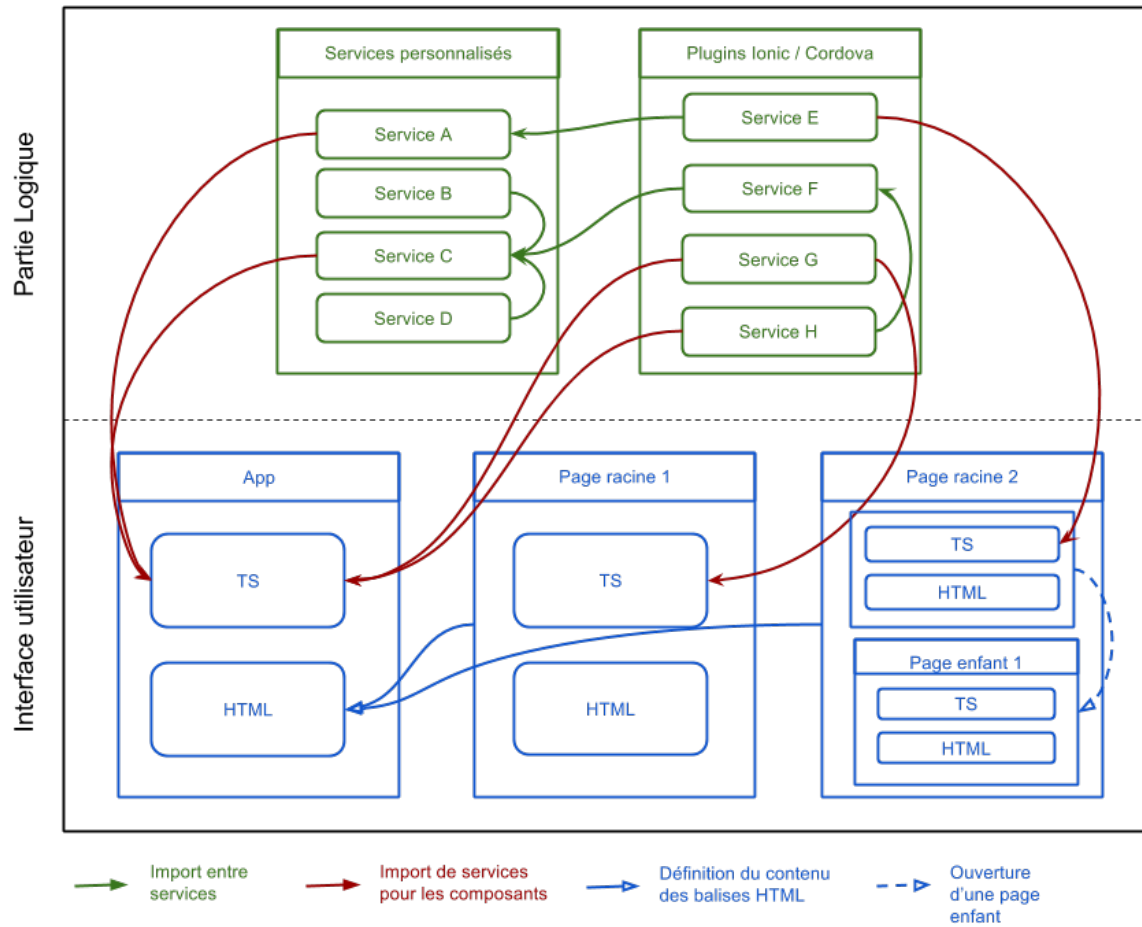


FIGURE 4.1 – Abstraction des interactions entre les modules d'une application développée sous Ionic

Cette architecture basée sur les composants est la conséquence de l'utilisation du framework Ionic d'Angular 4 comme framework de développement. L'architecture est intéressante pour les développeurs car elle réduit le couplage entre la partie logique et l'interface utilisateur. Les différents modules sont indépendants et possèdent chacun une fonctionnalité qui lui est propre. Un module peut faire appel aux services des autres modules implémentés en les important. Une telle approche permet d'effectuer des tests spécifiques sur chacun des modules et sur leurs interactions. Elle facilite le développement Open Source car chaque programmeur peut développer un module et le faire ensuite interagir avec les autres sans connaître, obligatoirement, le fonctionnement interne des modules utilisés. Un module peut rapidement être remplacé par un autre plus performant possédant les même fonctionnalités.

4.3.2 Interface Utilisateur

La partie interface utilisateur de l'application est composée de pages. Une page standard est composée de quatre éléments :

page.module.ts est responsable de l'importation des différents modules nécessaires au bon fonctionnement du module. Les modules importés peuvent être des services ou d'autres

modules de l'interface utilisateur tels que des pages enfants.

page.ts gère uniquement des réactions aux différents événements se produisant sur la page. Les réactions aux différents événements sous-entendent la gestion du data-binding, l'appel de méthode suite à une action, l'affichage de valeur. Il n'est pas responsable de la partie logique de l'application.

page.html est responsable de la structure de la page. Des balises HTML personnalisées peuvent être intégrées aux balises HTML standards. Ces balises HTML personnalisées sont composées de leurs propres modules possédant les fonctionnalités désirées.

page.scss est responsable du rendu visuel de la page.

Lors de l'ouverture de l'application, le premier module appelé par le framework Ionic est le module app. Il est le socle sur lequel les autres pages de l'application vont être appelées. Les services, qui sont importés par "app.module.ts", sont instanciés et accessibles pour l'ensemble des pages de l'application. "app.module.ts" importe également les pages racines de l'application.

La page app met en oeuvre le menu qui permet à l'utilisateur de passer d'une page racine à une autre. Ce menu est visible et accessible depuis toutes les pages racines de l'application. Cette page gère donc la navigation entre les différentes pages racines de l'application.

Comme l'illustre la figure 4.1, chaque page racine appelée par la page app utilise les services disponibles pour afficher et mettre à jour les informations de la partie logique. Une page racine peut avoir des pages enfants. Les pages enfants ont accès aux mêmes instances des services accessibles par la page racine parent. La navigation à travers les pages de l'application peut être vue comme l'ajout et la suppression d'éléments sur une pile. Lorsqu'une page enfant est ouverte, elle est ajoutée au sommet de la pile. Lorsque l'utilisateur appuie sur le bouton de retour en arrière, la page au sommet de la pile est retirée. La page app forme le socle de la pile et les pages racines sont le premier étage de la pile. En cas de navigation entre les pages racines, elles ne sont pas superposées mais interchangeables.

4.3.3 Partie logique

La figure 4.1 divise la partie logique en deux familles de services : les modules Ionic/Cordova et les modules personnalisés. La première famille de modules sont des modules développés par les communautés Ionic et Cordova. Ces modules ont principalement pour rôle de faire fonctionner l'application sur le mobile et de donner l'accès aux fonctionnalités natives du mobile. Ils demandent d'être importés dans les différentes pages ou services qui les utilisent, mais ils doivent également apparaître dans les fichiers de configuration de l'application vu que ce sont des modules extérieurs à l'application. La deuxième famille reprend les services développés spécifiquement pour l'application. Un service peut être la combinaison d'autres services pour offrir de nouvelles fonctionnalités.

L'application a pour objectif d'informer ses utilisateurs de la manière la plus efficace possible. Elle met donc en page les informations utiles à l'utilisateur, mais l'utilisateur n'a pas la possibilité de modifier les informations affichées. Les seules modifications que l'utilisateur est autorisé à effectuer sont des modifications concernant son profil et ses préférences. Chaque page de l'application représente une fonctionnalité, un flux d'informations différent qui est récupéré depuis de multiples sources d'informations. Les rôles principaux des services sont de récupérer les informations brutes auprès de ces sources. Ensuite, d'autres services doivent transformer et filtrer les informations brutes récupérées pour les transmettre de manière standard aux modules de la partie interface utilisateur qui se chargent de la mise en page de celles-ci.

La partie logique de l'application n'est pas une conception orientée objet car les objets n'interagissent pas entre eux, mais les services transmettent des objets JavaScript aux modules de la partie interface utilisateur pour qu'ils puissent les exploiter facilement.

4.4 Sécurité

L'application UCLCampus utilise différents services de l'UCL afin de fournir des informations à ses utilisateurs. Certaines informations sont uniquement accessibles aux étudiants de l'UCL telles que les cours auxquels ils sont inscrits et le répertoire des membres de l'UCL. Dans le futur, d'autres services nécessitant une authentification pourraient être ajoutés à l'application comme l'accès aux résultats des examens. L'authentification en tant que membre de l'Université catholique de Louvain doit pouvoir être effectuée à partir de l'application UCLCampus pour permettre une interaction sécurisée avec les services manipulant des données sensibles. La problématique d'authentification validée par le service sécurité de l'université est abordée dans la première sous-section. Lors du processus d'échanges d'informations sécurisées, certaines données privées sont stockées sur le mobile. Celles-ci doivent aussi être sécurisées au sein du mobile pour éviter tout acte malveillant. Le stockage sécurisé des données privées utilisé par l'application UCLCampus est expliqué dans la seconde sous-section. Le code permettant ces processus de sécurité ne fait pas partie de l'application déployée sur les stores d'application, vu que l'application dans son état actuel ne demande pas la sécurisation de données sensibles ni d'échanges sécurisés avec les services de l'UCL, mais est disponible sur une branche secondaire du projet Github ⁸.

4.4.1 Authentification sur les services UCL

L'UCL a implémenté un système d'authentification par Shibboleth, un mécanisme de fédération d'identité utilisé principalement par des établissements universitaires ou de recherches. Cette authentification se fait par un processus de Single Sign On (SSO) qui permet aux utilisateurs de rester connectés au travers de différentes applications proposées par l'UCL.

Le processus SSO utilise deux composants appelés Service Provider (SP) et Identity Provider (IdP). Lorsque l'utilisateur tente d'accéder à une ressource sécurisée, le Service Provider liée à cette ressource vérifie si l'utilisateur est connu et peut accéder à cette ressource. Si l'utilisateur n'est pas authentifié, le SP transmet la requête à l'Identity Provider. L'IdP vérifie si l'utilisateur a été récemment identifié sinon, il propose à l'utilisateur de se connecter par un formulaire d'identification. L'Identity Provider vérifie si cet utilisateur est connu du système. Si les informations d'identification de l'utilisateur sont correctes, l'Identity Provider renvoie les informations nécessaires au Service Provider pour permettre à l'utilisateur d'accéder au contenu protégé. L'ensemble des échanges entre Service Provider et Identity Provider se fait avec le protocole d'échange SAML. La figure 2.17 illustre ce processus.

Le projet initial était d'utiliser ce système d'authentification afin de permettre aux utilisateurs d'accéder de façon sécurisée aux fonctionnalités traitant des données sensibles. Malheureusement, la mise en place de ce processus d'authentification n'est pas réalisable avec l'infrastructure actuelle de l'UCL. En effet, l'application n'est pas capable de communiquer directement avec le Service Provider ou l'Identity Provider de l'UCL car la connexion à l'IdP est accessible uniquement par les redirections créées à partir des Services Provider de l'UCL. Le Service Provider communique directement avec les IdP ce qui implique que l'application UCLCampus n'est pas capable d'écouter et de récupérer les informations échangées. Pour permettre à l'application UCLCampus d'interagir directement avec l'IdP, il faudrait qu'elle soit elle-même un SP ce qui est contre nature vu que c'est un service externe à l'UCL.

Sans accès direct à un Service Provider ou Identity Provider et sans accès ni aux messages SAML transmis durant le processus d'authentification, ni à l'algorithme permettant leurs déchiffrements, l'application UCLCampus ne peut pas utiliser le processus de sécurité mis en place par l'UCL. Un travail conjoint avec le service de sécurité de l'UCL est nécessaire pour mettre en place un système autorisant les applications mobiles externes comme UCLCampus à effectuer des requêtes d'authentification et d'autorisation afin de connecter leurs utilisateurs. L'Université

8. <https://github.com/jerlemaire/UCLCampus>

catholique de Louvain prévoit de faire évoluer son processus de sécurité. Des discussions sont en cours concernant la mise en place du système OAuth⁹ mais elles n'en sont qu'à leurs prémices.

OAuth est un protocole libre qui permet d'autoriser une application client à utiliser l'API sécurisée d'une autre application pour le compte d'un utilisateur. L'authentification permet de vérifier l'identité d'un utilisateur alors que l'autorisation permet de vérifier les permissions d'accès de l'utilisateur. L'intérêt principal de l'utilisation OAuth est que l'application client ne doit ni connaître, ni demander à l'utilisateur ses informations d'identification.

En attendant la mise en place d'un système plus adapté, une solution alternative a été codée afin de permettre à un utilisateur de se connecter. Il n'est pas possible pour UCLCampus d'obtenir un profil utilisateur depuis les serveurs UCL. Néanmoins, il est possible de vérifier si un utilisateur a pu être authentifié par le processus Shibboleth de l'UCL. Notre solution alternative utilise le plugin InAppBrowser de Cordova¹⁰. Ce plugin ouvre un navigateur web depuis UCLCampus avec l'URL responsable du lancement du processus d'authentification Shibboleth. La page d'authentification du site de l'UCL est donc affichée sur le navigateur et l'utilisateur est invité à fournir ses informations d'identification en remplissant le formulaire. Si le processus Shibboleth authentifie correctement l'utilisateur, il est redirigé vers l'index du site internet de l'UCL. L'application ne peut pas récupérer les informations utiles d'authentification depuis le navigateur. Il est toutefois possible de vérifier les changements de pages effectués. L'application vérifie que l'utilisateur a atteint la page d'index du site de l'UCL. Comme l'utilisateur n'a pas d'accès à la barre d'adresse, il ne peut donc pas modifier lui-même la page sur laquelle il se trouve. L'unique moyen d'accéder à cette page d'index est de remplir et valider le formulaire d'identification présenté. Lorsque l'application identifie le chargement de la page d'index du site de l'UCL, elle a la confirmation que l'utilisateur a été vérifié par le système d'authentification. Aucune information n'est réellement échangée entre les serveurs UCL et l'application. L'unique information récupérée de manière détournée est la capacité d'un utilisateur à se connecter en tant que membre de l'UCL.

4.4.2 Stockage de l'information

L'application UCLCampus garde en mémoire des informations nécessaires à son fonctionnement. L'application stocke sur l'appareil différentes sortes de données :

Données statiques sont des données et textes directement écrits dans les fichiers HTML qui sont manipulés et affichés au sein de l'application.

Données non sensibles sont des données ne nécessitant pas d'accès authentifié qui sont modélisées sous forme d'objet. Les attributs de ces objets sont affichés sur différentes pages de l'application. Ils sont stockés de manière non sécurisée.

Données sensibles sont les données propres à l'utilisateur qui doivent être sécurisées. Par exemple, les résultats des examens de l'utilisateur qui sont stockés pour éviter de devoir les encoder à chaque ouverture de l'application

Différents mécanismes de stockage existent avec leur propre niveau de sécurité, des avantages et des inconvénients différents.

Stockage de l'information en clair Ce mécanisme de stockage signifie que l'information n'est pas chiffrée. Les données peuvent être récupérées et lues par un tiers si ce dernier accède au fichier où sont enregistrées les informations.

Stockage par chiffrement symétrique Ce mécanisme applique un algorithme, demandant une clé de chiffrement, sur les informations sauvegardées afin de les rendre incompréhensibles. La récupération de l'information compréhensible se fait à l'aide d'un algorithme

9. <https://oauth.net>

10. <https://github.com/apache/cordova-plugin-inappbrowser>

et de la même clé de chiffrement. La clé de chiffrement doit être tenue secrète, ne pas être récupérable et ne peut pas être devinée.

Stockage par chiffrement asymétrique Ce mécanisme de chiffrement est implémenté avec une paire de clés : une clé publique et une clé privée. La clé publique est disponible à quiconque souhaitant envoyer un message. La clé privée est maintenue secrète. Un message chiffré par un algorithme utilisant une certaine clé publique ne peut être déchiffré qu'en utilisant le même algorithme avec la clé privée associée. Vu que les clés dans la paire de clés sont dépendantes, il est quasiment impossible pour un tiers de décrypter un message sans connaître la paire de clés. Android et IOS proposent chacun des méthodes de stockage sécurisé avec chiffrement asymétrique.

Le framework Ionic met à disposition des modules et plugins implémentant de plusieurs manières les mécanismes de stockage présentés. Les modules et plugins de stockage sécurisé utilisés par l'application UCLCampus sont décrits ci-dessous :

Storage¹¹ est un plugin de stockage de paires de clés/valeurs. Il est possible d'enregistrer des chaînes de caractères et d'utiliser des objets JSON. Un objet JSON est un format de fichier contenant des données textuelles structurées. Storage choisit, parmi une variété de moteurs de stockage, le meilleur disponible selon la plateforme. Storage donne, si possible, la priorité à l'utilisation de SQLite qui est une base de données stable et communément utilisée. Cette implémentation d'un système de stockage n'offre pas de sécurité particulière et son utilisation est limitée au stockage de données non sensibles.

Secure Storage Il a été développé dans le plugin cordova-plugin-secure-storage¹². Secure Storage est un système de stockage sécurisé multiplateformes. Il est basé sur le système Storage qui permet d'enregistrer uniquement des chaînes de caractères sous forme de paires clés/valeurs. Ce plugin fonctionne différemment en fonction de la plateforme mobile sur laquelle il est exécuté[39].

- Sur la plateforme IOS, Secure Storage utilise la sécurité offerte par le KeyChain Storage. KeyChain Storage prend en charge le chiffrement des données avant de les enregistrer dans le système de fichiers. La keychain, objet où sont enregistrés les données, peut être verrouillée afin d'en bloquer l'accès. Il permet aussi le partage de données sensibles à travers un groupe d'applications et non la seule application qui les a enregistrées. A chaque élément enregistré sur la keychain sont associés des attributs d'accès. IOS KeyChain utilise deux niveaux de chiffrement des données. Le premier niveau utilise le mot de passe de l'écran de verrouillage de l'appareil comme clé de chiffrement. Le second niveau utilise une clé unique associée à l'appareil lui-même. Afin d'utiliser les services de l'IOS KeyChain, le module Secure Storage fait appel à une librairie externe appelée SAMKeychain. Cette librairie permet l'utilisation des méthodes et algorithmes de sécurité développés en C par Apple depuis une application hybride.
- Sur la plateforme Android, Secure Storage a une autre implémentation. D'après la documentation officielle du module, il n'existe pas d'équivalent à IOS KeyChain sur Android. L'implémentation se fait par deux chiffrements différents, Advanced Encryption Standard (AES) un algorithme de chiffrement symétrique et RSA, ainsi que par un espace de stockage par paire de clé/valeur qui est SharedPreferences. Le processus de sécurisation de l'information sur Android se déroule en quatre étapes :
 - Une clé 256-bits AES aléatoire est générée.
 - Cette clé AES chiffre l'information.
 - La clé AES est chiffrée via un algorithme de chiffrement RSA (RSA/ECB/PKCS1Padding) fourni par Android.
 - La combinaison de la clé AES chiffrée et de l'information chiffrée est enregistrée dans SharedPreferences.

12. <https://github.com/Crypho/cordova-plugin-secure-storage>

Pour récupérer l'information de manière compréhensible, il faut effectuer le processus dans l'ordre inverse. Pour pouvoir utiliser l'algorithme de chiffrement RSA, il est nécessaire que l'utilisateur ait configuré un écran de verrouillage sécurisé, c'est ce code de verrouillage qui est utilisé comme clé de chiffrement. Si l'utilisateur n'a pas d'écran de verrouillage paramétré, UCLCampus n'autorise pas l'utilisation des fonctionnalités nécessitant un stockage sécurisé et redirige l'utilisateur vers la page de paramétrage de l'écran de verrouillage afin qu'il le paramètre.

Il est à noter que l'utilisation de Secure Storage est dédiée au stockage d'informations sensibles. En effet, en cas de changement des paramètres de sécurité du mobile, les données sécurisées sont supprimées.

4.5 Modules implémentés

L'application UCLCampus affiche principalement des informations. Les différents modules de l'application récupèrent et mettent en forme les informations disponibles auprès de nombreuses sources. Les modules ont été catégorisés selon leur méthode de récupération des informations. L'intérêt de cette catégorisation est de mettre en évidence les services génériques développés qui sont réutilisables pour récupérer des informations auprès d'un même type de source. Les différentes sources d'information utilisées par l'application UCLCampus sont : les flux RSS, l'ESB WSO2, les bases de données locales, les applications existantes et les services web externes. A l'exception des services web externes, l'ensemble des sources d'information proviennent de l'UCL ce qui rend l'application dépendante du fonctionnement et de la structure des services informatiques de l'université. La méthode de récupération par parsing, récupération d'information sur les pages d'un site web à l'aide de scripts, a été écartée. En effet, cette méthode est difficilement généralisable et, une page web est sujette à des changements qui induisent l'obsolescence de la fonctionnalité utilisant le parsing. Cette section explique de manière détaillée le fonctionnement des différents modules et services mis en oeuvre pour récupérer les informations en fonction de chaque type de source. En plus d'explications sur la manière de récupérer l'information, le fonctionnement de chaque page utilisant ces modules pour afficher les informations est présenté.

4.5.1 Utilisation de l'objet Promise de JavaScript

L'application UCLCampus utilise des requêtes Hypertext Transfer Protocol (HTTP) GET pour récupérer les informations de diverses sources. Les requêtes HTTP exécutées étant asynchrones, l'utilisation d'objet Promise permet de traiter les réponses aux requêtes[40]. Une Promise est un objet qui représente une valeur qui n'est pas encore disponible. Cet objet a été créé avec pour objectif de gérer au mieux l'asynchronisme des réponses aux requêtes exécutées. L'objet Promise est associé à un objet deferred. L'objet deferred représente une action rendant disponible la valeur de Promise. Deux actions peuvent être réalisées par deferred : soit, il résout la Promise et, dans ce cas, il fournit à Promise une valeur de retour unique et fait appel à la fonction `resolve()` ; soit, il rejette la Promise et, dans ce cas, il fournit la raison du rejet et fait appel à la fonction `reject()`. Une Promise ne peut être satisfaite que par une seule action. Une Promise ne peut pas être résolue et rejetée en même temps. Le code de la gestion des réponses aux requêtes asynchrones est scindé en deux parties : une partie qui gère le cas du succès de la requête et une autre partie qui gère le cas où la requête renvoie une erreur.

4.5.2 Récupération de l'information par flux RSS

Un flux RSS est un fichier au format XML dont le contenu est produit automatiquement en fonction des mises à jour en ligne d'un site web. Ce fichier est diffusé vers tous les utilisateurs

abonnés au flux, d'où l'intérêt d'utiliser un format tel que XML qui est compatible avec différents programmes et machines. Des logiciels et services permettent d'écouter les flux RSS qui sont ajoutés par l'utilisateur voulant être informé des actualités. Les flux RSS sont composés d'informations publiques. Les utilisateurs ne doivent pas s'identifier pour les écouter.

L'application UCLCampus utilise le service Yahoo Query Language (YQL) de Yahoo pour lire les flux RSS diffusant des informations en lien avec l'UCL. YQL est un service gratuit qui permet de combiner et de filtrer les sources de données sur le web. YQL permet l'écoute et la transformation de flux RSS en format JSON plus efficace pour des manipulations ultérieures sous JavaScript. Le service responsable de la récupération des données provenant des flux RSS envoie une requête HTTP contenant l'adresse du flux à écouter au service YQL. YQL répond en envoyant les informations disponibles sur le flux RSS sous forme de JSON ainsi que les métadonnées associées à la requête. La lecture du flux RSS est gérée dans l'application par le service appelé RSSService. Il sert uniquement à lire un flux RSS. Il renvoie les éléments d'un flux RSS utiles aux services appelants.

Deux modules, News et Events, récupèrent les informations qu'ils affichent depuis des flux RSS. Chacun de ces modules possède un service de récupération d'information spécifique faisant appel à RSSService. Une fois les informations récupérées, elles sont transformées en objet par ce même service pour rendre les informations manipulées par les modules indépendantes de leur source. Le fonctionnement et les spécificités de ces deux modules sont approfondis dans les sous-sections suivantes.

4.5.2.1 News

Le module des actualités (ou news), illustré par la figure 4.2, s'articule en quatre composants :

- Une vue permettant d'afficher le détail d'une actualité, appelée NewsDetailsPage
- Une vue permettant d'afficher la liste des actualités récupérées, appelée NewsPage
- Un service associé à ces vues permettant de modéliser les informations provenant du flux RSS, appelé NewsService
- Le service générique RSSService qui gère la récupération des informations sur un flux RSS par le service YQL de Yahoo.

La NewsPage et la NewsDetailsPage appartiennent à la partie interface utilisateur de ce module tandis que le NewsService et le RSSService appartiennent à la partie logique. La partie logique récupère et traite des informations au format JSON tandis que la partie interface utilisateur récupère et traite des informations sous forme d'objet NewsItem. L'objet NewsItem est composé de plusieurs attributs décrivant la news associée : titre, date de publication, description, image, guide et le lien vers la source de l'actualité. Enfin, l'objet contient un attribut permettant de ne pas afficher l'objet dans la liste des actualités. Cet attribut est utilisé pour n'afficher que les objets répondant à la recherche de l'utilisateur dans la liste des actualités.

NewsPage est le premier composant chargé à l'appel du module News. La source du flux pour le module News provient de trois pages d'actualités venant de <http://uclouvain.be/actualites/\protect\T1\textbraceleftp1,p2oup3\protect\T1\textbraceright>, où p1, p2 et p3 représentent différentes pages d'actualités. NewsPage présente trois onglets correspondants aux trois flux RSS d'actualités de l'UCL. Au chargement d'un onglet, ce scénario est illustré par la figure 4.3, NewsPage demande à NewsService de récupérer la liste des actualités à afficher. NewsService gère cette requête en deux étapes.

Premièrement, le service fait appel à RSSService afin de récupérer la liste des actualités du flux RSS correspondant à l'onglet sélectionné. Le NewsService ne s'intéresse qu'aux résultats de la requête, c'est-à-dire aux actualités. Ainsi, le RSSService ne renvoie que cette partie de la réponse au NewsService sous format JSON.

Deuxièmement, NewsService crée, pour chaque news extraite de la réponse JSON, un objet NewsItem et l'ajoute à la liste des actualités. Cet objet permet de simplifier et d'unifier le contenu

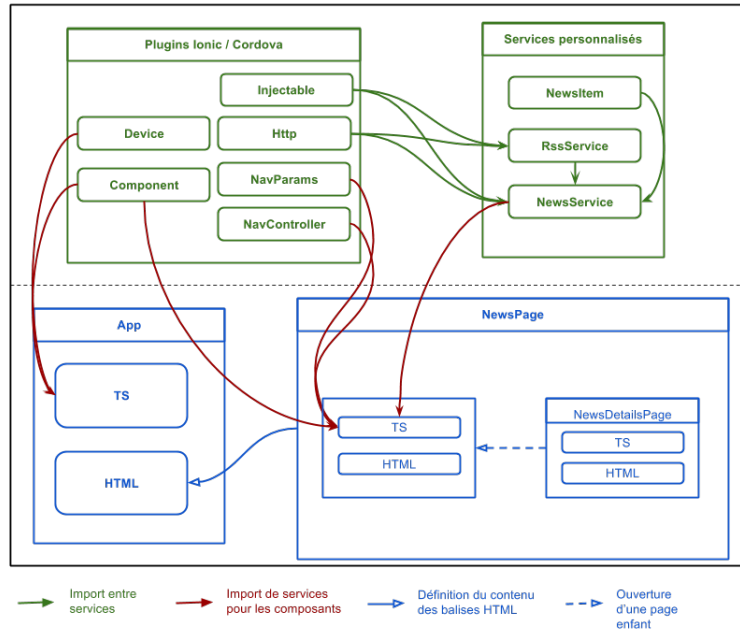


FIGURE 4.2 – Abstraction des interactions pour le module News

renvoyé par le backend au frontend. Cette liste est enfin renvoyée par le service à la `NewsPage` afin d'être affichée.

En haut de la vue `NewsPage` se trouve une barre de recherche permettant à l'utilisateur de filtrer la liste des événements sur base d'une chaîne de caractères introduite par l'utilisateur.

NewsDetailsPage est la page appelée lorsque l'utilisateur clique sur une actualité de la page `NewsPage`. La page est appelée avec en paramètre l'objet `NewsItem` associé à cette actualité. L'objet a été instancié durant la phase de récupération des actualités, toutes les informations nécessaires à l'affichage y sont déjà enregistrées.

Grâce à cette implémentation en couches, chaque composant gère uniquement les informations nécessaires à sa tâche. En effet, la `NewsPage` n'a pas besoin de savoir d'où viennent les actualités. Que ce soit un flux RSS, une base de données ou un fichier JSON quelconque, son seul but est d'afficher une liste de `NewsItem`. Le `NewsService` quant à lui ne doit pas savoir comment lire un flux RSS. Son but est uniquement de créer une liste de `NewsItem` à renvoyer à la `NewsPage` à partir d'un fichier JSON dont il connaît la structure. Ceci permet une meilleure modularité des composants. La standardisation des actualités en format `NewsItem` est l'objet sur lequel repose la modularité des composants. Tant que la `NewsPage` reçoit une liste de `NewsItem`, l'affichage fonctionne. Si l'origine des actualités devait changer et passer, par exemple, d'un flux RSS à une base de données, seul `NewsService` serait à modifier.

Cette modularité est un point important dans le développement d'une application Open Source qui aspire à être facilement prise en main par un tiers et maintenable par la suite.

4.5.2.2 Events

Les six composants principaux du module Events sont :

- Une vue affichant les détails d'un événement, appelée `EventsDetailsPage`
- Une vue affichant les événements récupérés, appelée `EventsPage`
- Une vue affichant les paramètres de filtrage des événements, appelée `EventsFilterPage`

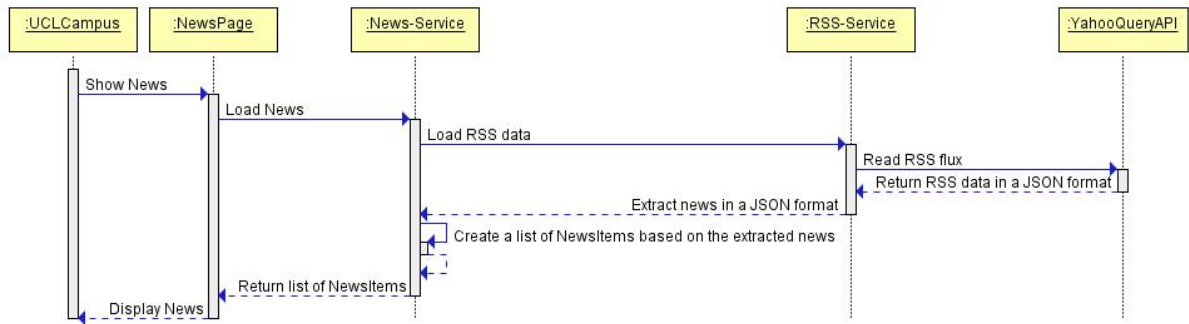


FIGURE 4.3 – Diagramme de séquence pour l’affichage des News

- Un service, appelé EventsService et associé à EventsPage, qui traite les informations provenant du flux RSS
- Un service générique RSSService qui gère la récupération des informations par un flux RSS
- Un service externe au module, appelé UserService, est utilisé pour gérer le stockage des événements favoris d’un utilisateur

La figure 4.4 illustre les interactions entre les différents composants du module. Le frontend est implémenté avec les pages EventsPage et EventsDetailsPage. Le backend est formé par EventsService et RSSService. Pour le frontend, le module manipule le format EventItem. L’objet EventItem est composé d’attributs décrivant l’évènement associé : titre, date de début et de fin, description, image, localisation, lien vers la source de l’article et catégorie. A cela s’ajoutent deux autres attributs utilisés pour la gestion de l’affichage de la liste des événements dans EventsPage : Un attribut permettant de masquer un objet dans la liste et, un autre permettant de marquer un objet comme favori de l’utilisateur.

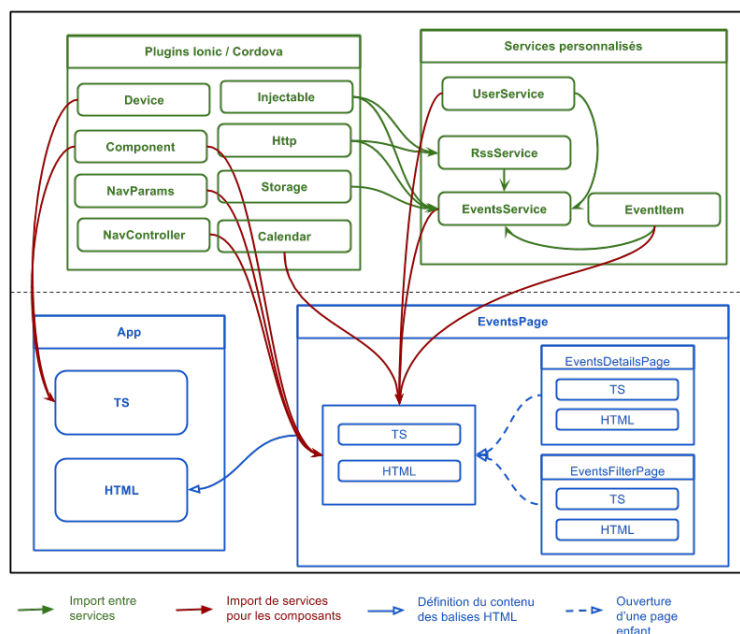


FIGURE 4.4 – Abstraction des interactions pour le module Events

EventsPage est le premier composant chargé à l'appel du module Events. La source du flux pour le module Events vient du site <http://louvainfo.be/evenements/feed/calendar/> utilisé par l'application Guindaille 2.0. Cette page présente deux onglets : la liste des événements et la liste des événements favoris de l'utilisateur. Dans ces deux onglets, l'utilisateur peut faire coulisser chaque événement d'un geste du doigt afin d'en présenter les actions. Deux actions sont possibles : ajouter ou supprimer l'événement de la liste des favoris et ajouter l'événement à son calendrier. Au chargement du premier onglet, EventsPage envoie une requête auprès de EventsService afin de récupérer la liste des événements à afficher. EventsService gère cette requête en deux étapes illustrées dans la figure 4.5 :

- Le service fait appel à RSSService afin de récupérer la liste des événements du flux RSS sous le format JSON.
- Dans un second temps, EventsService traite la réponse en créant un objet EventItem pour chaque élément du JSON. Chacun des objets est ajouté à la liste des événements à afficher qui est renvoyée à EventsPage.

Le second onglet ne présente que les événements marqués comme favoris par l'utilisateur. Cet onglet ne fait ni appel à l'EventsService, ni au RSSService. La liste des favoris est générée en filtrant la liste des événements. La liste des événements est une liste d'objets EventItem et chaque EventItem a un attribut "favori" prenant la valeur vrai ou faux. La liste des favoris est créée en filtrant la liste des objets pour n'afficher que ceux marqués comme favoris. Le stockage des favoris est géré par un service externe au module appelé UserService. Ce service permet d'enregistrer et de récupérer des données concernant l'utilisateur. Les données enregistrées sont les événements favoris de l'utilisateur. Pour ce faire, ce service utilise le plugin Ionic Storage. La liste d'événements favoris d'un utilisateur n'étant pas des données sensibles, l'utilisation du composant Storage est suffisant pour gérer sa liste de favoris et ne nécessite pas de sécurité supplémentaire.

EventsFilterPage est la page appelée lorsque l'utilisateur clique sur le bouton de paramétrage représenté par l'icône en haut à droite de la page EventsPage. Une page modale s'affiche présentant à l'utilisateur différents paramètres afin d'affiner la liste des événements affichés. Une page modale est un volet de contenu qui est présenté au-dessus de la page en cours d'utilisation. Ces paramètres sont le filtrage des événements pour n'afficher que ceux compris dans le(s) X mois, dont le X est défini par l'utilisateur, ainsi que le filtrage pour n'afficher que les événements dont la catégorie appartient à celle(s) sélectionnée(s) par l'utilisateur.

Un autre paramètre de filtrage est disponible directement sur la page EventsPage : la barre de recherche. Cette barre est située en haut de la vue EventsPage et permet de filtrer les événements sur base d'une chaîne de caractères introduite par l'utilisateur. Bien qu'étant un paramètre de filtrage, la décision a été prise d'implémenter cette barre de recherche dans la page EventsPage afin d'en faciliter l'accès à l'utilisateur.

EventsDetailsPage est la page appelée lorsque l'utilisateur clique sur un événement de la page EventsPage. L'objet EventItem associé à cet événement est envoyé en paramètre à la page EventsDetailsPage. Cet objet contient dans ses attributs l'ensemble des informations à afficher. Aucun appel à EventsService n'est nécessaire car l'ensemble des informations est déjà disponible grâce à l'objet EventItem.

L'ajout d'un événement au calendrier du smartphone de l'utilisateur est géré grâce au module Calendar. Cette fonctionnalité est implémentée dans le plugin cordova-plugin-calendar¹³. Ce plugin offre différentes fonctionnalités telles que : créer un nouveau calendrier, trouver un événement du calendrier natif et ajouter un événement au calendrier natif d'un appareil mobile. Calendar est un plugin hybride qui fonctionne sur iOS et sur Android. Néanmoins,

13. <https://github.com/EddyVerbruggen/Calendar-PhoneGap-Plugin>

certaines fonctionnalités ne sont pas encore disponibles sur les deux plateformes comme la fonction permettant de modifier un évènement du calendrier (disponible uniquement sur iOS). L'implémentation de cette fonctionnalité affiche à l'utilisateur la vue d'ajout d'un évènement au calendrier avec des options préremplies par les informations de l'évènement telles que le titre, la date, l'heure et le lieu. L'utilisateur peut, dès lors, enregistrer l'évènement tel quel dans le calendrier ou y apporter des modifications avant de l'enregistrer. Une fois enregistré l'utilisateur revient à la vue affichant la liste des évènements.

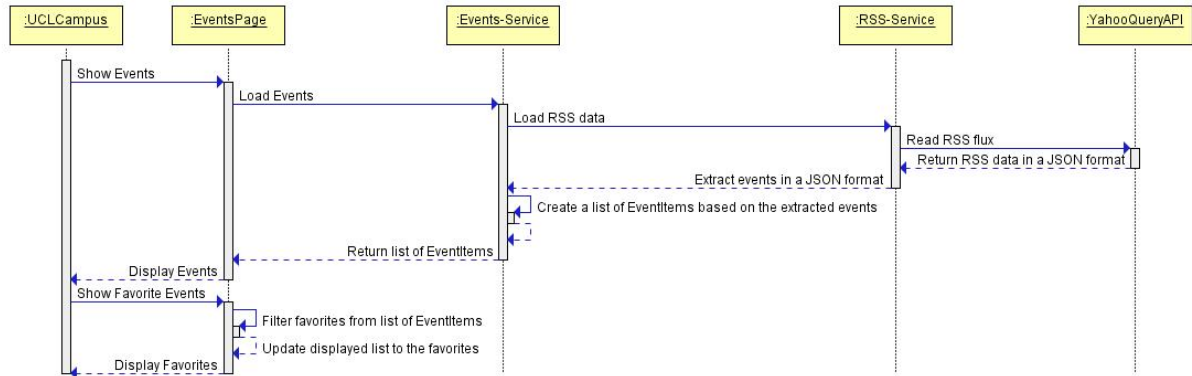


FIGURE 4.5 – Diagramme de séquence pour l'affichage des Events

4.5.3 Récupération de l'information par WSO2

WSO2, en plus d'être un ESB, propose un outil de gestion d'API, WSO2 API manager[41]. L'API manager permet aux développeurs de prendre connaissance des API existantes et disponibles des services interagissant avec ESB WSO2. Il se présente sous forme d'interface graphique. Les développeurs peuvent interagir avec les API pour connaître la structure des requêtes et des réponses fournies. WSO2 API manager étant intégré sur l'intranet de l'UCL, l'accès à l'interface de WSO2 se fait par le Virtual Private Network (VPN) fourni par l'UCL. Une fois sur l'interface, WSO2 affiche les différents services disponibles ainsi que leur statut. Les services affichés ne sont pas tous utilisables, certains sont en cours de production, d'autres sont publiés en interne et, d'autres encore sont publiés publiquement. Chaque service possède sa propre page contenant les différentes fonctions appelables ainsi qu'une documentation rédigée par l'auteur du service. L'interface WSO2 permet aussi de tester directement les fonctions offertes par l'API d'un service. Cette interface est pratique pour comprendre l'utilisation d'une fonction ainsi que les paramètres nécessaires à son bon fonctionnement. Le test de la fonction est effectué directement via l'interface WSO2 par un appel de client Uniform Resource Locator (cURL). Pour chaque test effectué, l'interface WSO2 affiche l'appel cURL correspondant. cURL est un outil permettant de récupérer du contenu venant d'un serveur.

Pour l'instant, notre application fait appel à un de ces services. Ce service permet de récupérer les informations liées aux bibliothèques. Afin d'appeler un service publié, un token d'accès associé à UCLCampus est nécessaire. Tous les utilisateurs de l'application utilisent le même token d'accès. Pour l'obtenir, l'application doit s'abonner au service via l'interface WSO2. Une fois abonné, l'application UCLCampus utilise le token d'accès disponible pour appeler le service. Un token d'accès est une chaîne de caractères qui est passée dans l'en-tête HTTP d'une requête. La forme de cette chaîne de caractères est, par exemple, "Autorisation : Bearer NtBQkXoKElu0H1a1fQ0DWfo6IX4a." Les tokens d'accès authentifient l'application utilisant l'API ce qui évite à l'API d'être sollicitée par des requêtes dont la source est inconnue. Le token aide notamment dans la gestion des attaques par déni de service en permettant à WSO2 de

rapidement rejeter la requête si le token n'est pas valide. Cette vérification est effectuée dès la première étape du traitement par WSO2.

L'application UCLCampus accède aux services WSO2 de l'UCL via une en-tête intitulée "Autorisation : Bearer". Pour ce genre d'autorisation il est nécessaire de sécuriser le token d'accès afin qu'il ne puisse pas être utilisé par un tiers. En effet, tout tiers en possession d'un token bearer peut l'utiliser pour accéder aux ressources associées. Toute personne ayant accès au token est donc en mesure d'appeler les services proposés par l'UCL par WSO2 avec les mêmes droits que l'application UCLCampus.

4.5.3.1 Bibliothèques

La logique du module des bibliothèques, illustrée par la figure 4.6, s'articule en quatre composants :

- Une vue affichant les détails d'une bibliothèque, appelée LibraryDetailsPage
- Une vue affichant la liste des bibliothèques, appelée LibrariesPage
- Un service traitant les informations récupérées, appelé LibrariesService
- Un service permettant de récupérer les informations collectées via WSO2, appelé WSO2Service

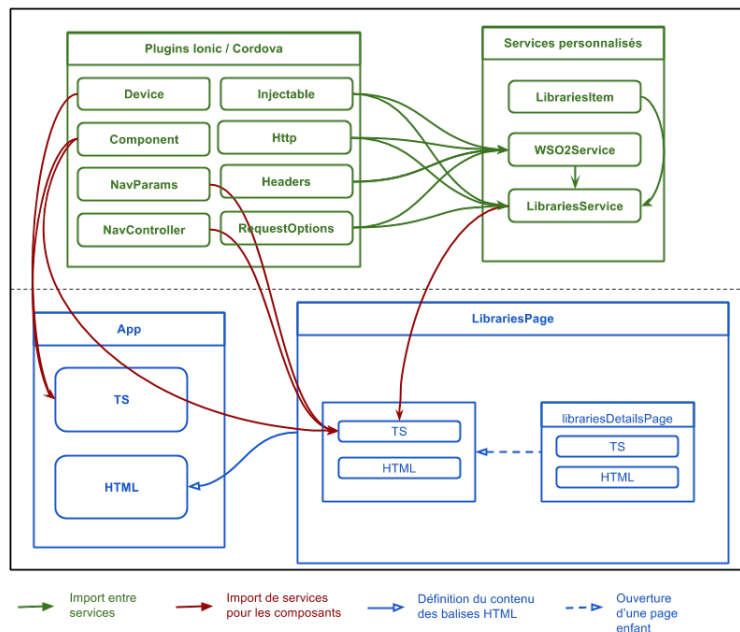


FIGURE 4.6 – Abstraction des interactions pour le module Bibliothèques

LibrariesPage est le premier composant utilisé lorsque le module des bibliothèques est appelé. Cette page affiche la liste des bibliothèques de l'UCL. La figure 4.7 représente les différentes requêtes effectuées pour récupérer les informations affichées. Pour récupérer la liste des bibliothèques, LibrariesPage envoie une requête au service LibrariesService. Ce dernier transmet la requête au WSO2Service qui récupère un fichier JSON contenant la liste des bibliothèques de l'UCL. WSO2Service exécute une requête HTTP GET appelant la méthode "list" de l'API WSO2 "Libraries V1". Chaque élément du fichier JSON renvoyé par l'API WSO2 possède un ID et un nom. WSO2Service renvoie ce fichier JSON à LibrariesService. LibrariesService crée un objet LibraryItem pour chaque entrée dans le fichier JSON et ajoute cet objet à une liste. La liste d'objets LibraryItem est ensuite renvoyée à LibrariesPage pour être affichée. Un objet LibraryItem contient les attributs d'une bibliothèque : nom, numéro de téléphone, site web, etc.

L'objet instancié par LibrariesService n'est cependant pas complet. Le fichier JSON récupéré contient uniquement la paire nom/id pour chaque bibliothèque. Les attributs manquants de l'objet sont mis à jour en fonction des besoins de l'utilisateur par l'exécution d'autres requêtes.

LibrariesDetailsPage est la page chargée lorsque l'utilisateur clique sur le nom d'une bibliothèque. Dans ce cas, la page est appelée avec en paramètre l'objet LibraryItem associé à cette bibliothèque. Cet objet est incomplet pour en afficher les détails puisqu'elle ne contient que deux variables : un ID et un nom. LibraryDetailsPage envoie dès lors une requête au LibrariesService afin de récupérer les détails manquants de la bibliothèque à afficher. LibrariesService transmet une requête contenant l'ID de la bibliothèque au service WSO2Service. La réponse de l'API WSO2 sous le format JSON contient toutes les informations concernant la bibliothèque demandée. LibrariesService met à jour les attributs de l'objet LibraryItem avec les données récupérées et renvoie l'objet mis à jour à LibraryDetailsPage. Si un utilisateur souhaite revoir les détails d'une bibliothèque déjà consultée, l'objet LibraryItem est déjà complété, et la requête n'est pas exécutée une seconde fois. L'objet LibraryItem contient également un objet MapLocationItem décrit dans le module Cartes. Cet objet MapLocationItem permet au module des Cartes d'afficher la bibliothèque sélectionnée par l'utilisateur sur la carte. Depuis la page LibrariesDetailsPage, l'utilisateur peut cliquer sur un bouton afin d'afficher la bibliothèque sur la carte. LibrariesDetailsPage appelle le module Cartes avec en paramètre l'objet MapLocationItem contenu dans l'objet LibraryItem associé à la bibliothèque sélectionnée par l'utilisateur. Le module des Cartes affiche ainsi la carte avec les coordonnées instanciées dans l'objet MapLocationItem transmis en paramètre.

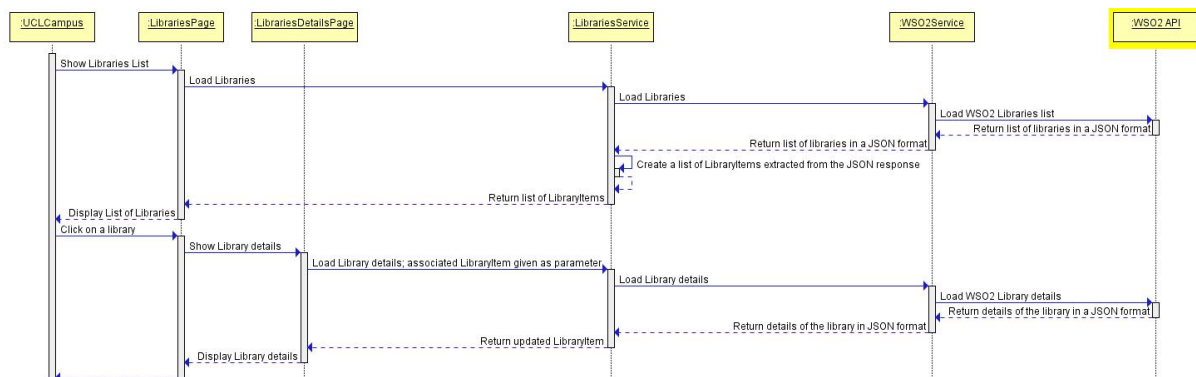


FIGURE 4.7 – Diagramme de séquence pour l'affichage des bibliothèques

4.5.4 Récupération de l'information par des services web internes à l'UCL

Actuellement, tous les services disponibles à l'UCL ne passent pas nécessairement par l'Entreprise Service Bus WSO2. L'application doit, donc, pouvoir interagir directement avec l'API de ces services. Les modules ont été conçus de manière modulaire pour tenir compte d'une migration future des services vers WSO2. Pour ce faire, la partie logique de récupération d'informations est scindée en deux types de services :

- Les services responsables de récupérer l'information brute, sous différents formats, auprès des API fournies par l'UCL et de la transformer en information sous forme de JSON. Ces services sont des services intermédiaires entre les API de l'UCL et la partie modèle de l'application et, ils seront remplacés en cas d'un transfert vers WSO2. Le format de sortie en JSON n'est pas fortuit car c'est le format des réponses aux requêtes effectuée sur WSO2.

- Les services récupérant l'information en format JSON pour la transformer en objet générique utilisable par l'ensemble des pages de l'application.

La seule fonctionnalité de l'application concernée par cette situation est la partie Etude de l'application dont le fonctionnement est expliqué dans la sous-section suivante.

4.5.4.1 Etudes

La fonctionnalité Etudes joue un rôle central dans l'application UCLCampus. Elle doit permettre à l'ensemble des étudiants d'organiser leurs vies académiques. Cette partie de l'application est vaste et peut être déclinée et mise en oeuvre de différentes manières, comme le cahier des charges l'exprime. Dans le cadre de ce travail, l'implémentation s'est concentrée principalement sur la récupération de l'horaire des cours d'un étudiant depuis une liste de cours. L'horaire des cours à l'Université catholique de Louvain est accessible depuis le logiciel ADE. Cette sous-section s'intéresse à la récupération et l'affichage de l'horaire de cours depuis ADE.

La figure 4.8 illustre le fonctionnement et les interactions entre les différents composants de la fonctionnalité Etudes telle qu'elle est mise en oeuvre dans l'application actuelle.

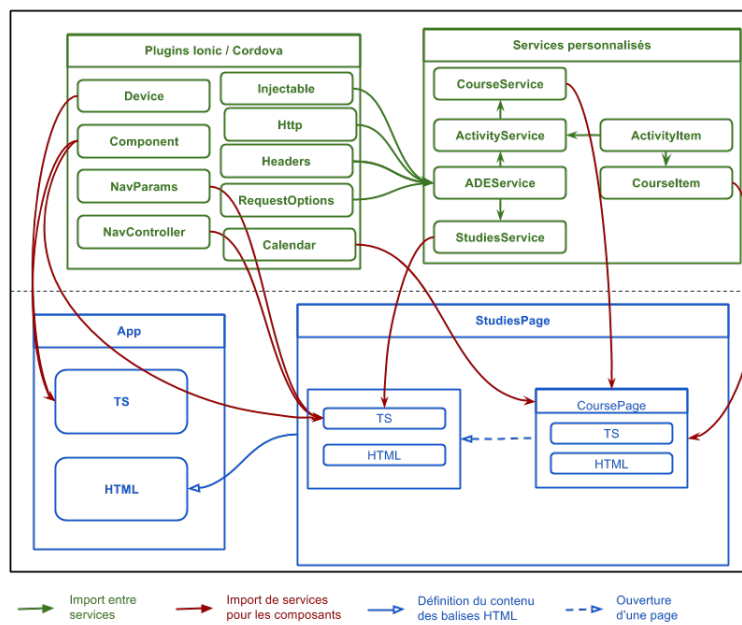


FIGURE 4.8 – Abstraction des interactions pour le module Etudes

La partie interface de la fonctionnalité Etudes comprend deux pages différentes :

StudiesPage est la page racine qui s'ouvre lorsque l'utilisateur accède à la fonctionnalité Etudes. Elle offre à l'utilisateur une interface affichant une liste de cours. Elle lui permet de mettre à jour manuellement cette liste de cours et de sélectionner l'année pour laquelle il désire connaître l'horaire.

CoursePage est une page enfant affichant une liste interactive des activités liées au cours sélectionné par l'utilisateur sur la StudiesPage.

Afin de manipuler de manière générique les différentes données affichées sur ces trois pages, la partie logique de la fonctionnalité utilise différents objets :

CourseItem représente les données inhérentes à un cours universitaire par les attributs suivants :

- Nom

- Sigle
- ActivitiesList : une liste d'objets ActivityItem

ActivityItem représente une activité liée à un cours universitaire à travers différents attributs :

- Type : Le type d'activité soit Cours Magistral soit Travaux Pratiques
- Start : Date et heure de début de l'activité
- End : Date et heure de fin de l'activité
- Instructor : Nom du professeur dispensant le cours
- Etudiant : Nom du groupe d'étudiants suivant le cours
- Local : lieu, auditoire où se déroule l'activité

ADE est le logiciel utilisé par l'Université catholique de Louvain pour organiser et diffuser les horaires de cours. Ce logiciel dispose de web API qui fournit un service permettant aux applications d'accéder aux données du serveur ADE quels que soient leurs langages de programmation. L'application UCLCampus interagit, donc, avec le serveur ADE par des requêtes vers ces web API. Les requêtes à effectuer, par exemple, pour la récupération de l'horaire du cours LFSAB1101 Mathématiques 1 sont les suivantes :

- Identification et ouverture de la session ADE à l'aide de l'identifiant "etudiant" et du mot de passe "student".
- Récupérer les différents projets d'horaire disponibles pour connaître leur identifiant et leur nom.
- Sélectionner un projet par son identifiant.
- Récupérer l'identifiant ADE d'un cours grâce à son code/sigle, ici LFSAB1101.
- Récupération des activités du cours LFSAB1101.

Une illustration plus détaillée des différentes requêtes et leurs réponses pour cet exemple se trouve en annexe. Les réponses récupérées sont en format XML.

L'interaction avec les web API d'ADE et la modélisation des données en objet se font par l'intermédiaire des trois services mis en oeuvre dans la partie logique de la fonctionnalité.

ADEService effectue toutes les requêtes vers les API ADE. Le service récupère les réponses sous format XML. Il est chargé de convertir les données récupérées en JSON.

StudieService est responsable de la partie logique de la page StudiesPage, c'est-à-dire de la mise à jour et stockage des cours présents dans la liste des cours de l'utilisateur et du choix du projet ADE.

CourseService est responsable de la construction de la liste des activités d'un cours par la gestion de l'enchaînement des requêtes pour récupérer les informations. Il transforme les données JSON des activités reçues de ADEService en objet ActivityItem pour faciliter la manipulation des données. Ce service est également capable de filtrer les activités à afficher en fonction des paramètres de l'utilisateur. Il représente la partie logique de page CoursePage.

La figure 4.9 est le diagramme de séquence représentant les différents appels de méthodes entre les services de l'application dans le cadre du scénario suivant : Un utilisateur ouvre la fonctionnalité Etudes et sélectionne un projet ADE. Ensuite, il clique sur un cours de la liste pour connaître les différentes activités de celui-ci.

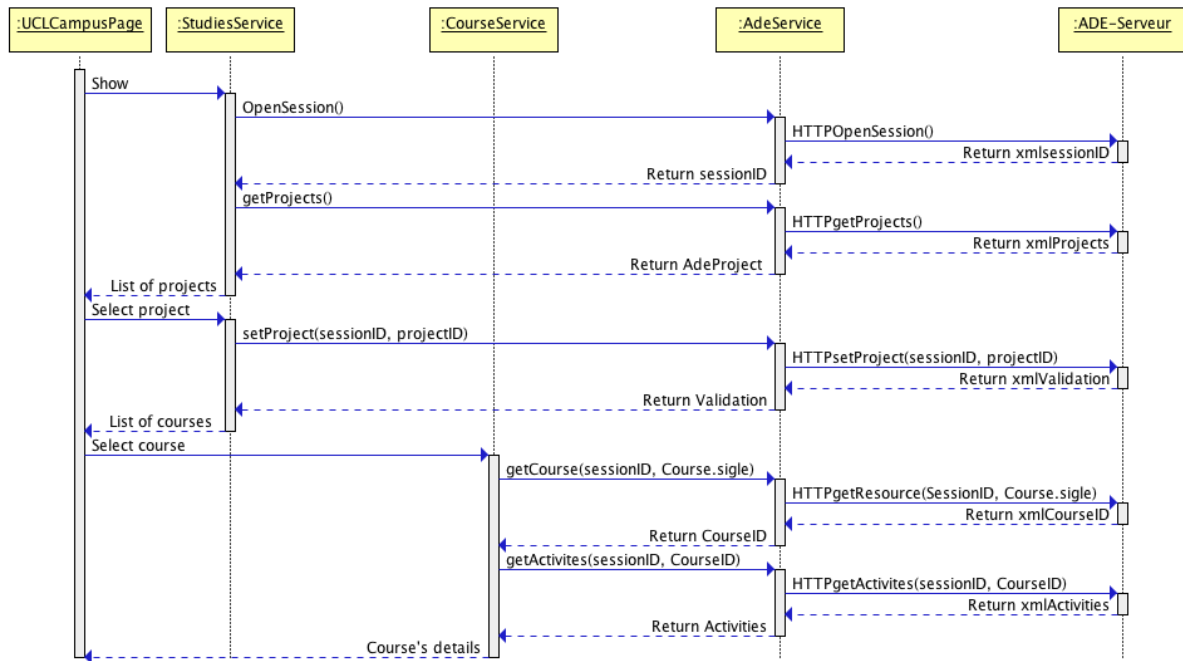


FIGURE 4.9 – Diagramme de séquence de la récupération de l’horaire ADE

4.5.5 Récupération de l’information par des services web externes à l’UCL

Certaines fonctionnalités proposées par l’application UCLCampus ne sont pas en lien direct avec l’université. L’application doit donc récupérer les informations et les services nécessaires au fonctionnement de ces fonctionnalités auprès de services web externes. La seule fonctionnalité utilisant des services web externes est la carte interactive de l’application permettant aux étudiants de se repérer sur les différents campus et de calculer l’itinéraire vers un point d’intérêt du campus. Le fonctionnement du module s’occupant de cette fonctionnalité est présenté ci-dessous.

4.5.5.1 Cartes

Deux solutions ont été envisagées pour l’implémentation de la fonctionnalité de la carte interactive.

La première est de continuer avec la librairie leaflet, qui a été utilisée par le projet précédent, et les outils qui la supporte. Les avantages de cette technologie sont que le projet est Open Source et qu’une partie des fonctionnalités est déjà implémentée et opérationnelle. Mais le défaut majeur est le manque de flexibilité de cette technologie. Comme observé dans le projet précédent et dans l’application LLNMaps développée par un assistant de l’UCL, elle nécessite de stocker des tuiles, bouts de cartes, pour fonctionner. Définir des zones cibles de carte à stocker pour l’ensemble des campus universitaires de l’UCL est fastidieux et contraignant.

La seconde alternative est d’utiliser les services Google Maps nécessitant une connexion au réseau internet mais permettant un accès efficace à l’ensemble des cartes du monde et à la géolocalisation. Le choix a été fait d’implémenter le module des cartes via les API Javascript et mobiles de Google Maps (Google Maps Android API et Google Maps SDK). De plus, les API proposées par Google facilitent le développement d’un module cartes pour une application multiplateforme. Les services sont développés par une compagnie active, ils sont donc améliorés de façon régulière. Le framework Ionic propose d’utiliser des plugins utilisant l’API mobile Google Maps. Ces plugins sont utilisés par de nombreux développeurs et possèdent une documentation fournie et détaillée. L’API Google Maps conçue pour les mobiles offre de meilleures performances

en termes de temps de réponse et d'utilisation du réseau que l'API Google Maps standard[42]. En effet, l'API Javascript ayant été pensée pour les ordinateurs, les performances mobile réseau ne sont pas bonnes et consomment trop de ressources. La raison est que l'API Javascript charge énormément d'images pour créer la carte. Chaque tuile composant la carte est une image téléchargée. Pour un ordinateur, cela ne pose aucun problème mais, pour un appareil mobile, ce sont de lourdes données à traiter. Les API mobiles chargent des vecteurs de données permettant de dessiner les tuiles de la carte à la place des tuiles elles-mêmes. L'utilisation d'un vecteur au lieu d'images permet de réduire la charge sur le réseau. Par exemple, pour une carte nécessitant neuf tuiles environ 113KB de données sont téléchargées, alors qu'avec un vecteur de données, on atteint moins de 12KB. De plus, si le poids des données est élevé, l'application mettra plus de temps à afficher les données à l'écran. L'utilisation des API mobiles permet de réduire les transferts sur le réseau et d'améliorer le temps d'exécution. Le plugin cordova-plugin-googlemaps¹⁴ est le plugin implémenté pour interagir avec les services Google Maps pour les plateformes IOS et Android. Un autre avantage des API mobiles de Google Maps est l'absence de quotas. En effet, l'API Javascript est soumis à un quota de 25 000 requêtes par tranche de 24h[43]. Dépasser ce nombre entraîne une facturation de la part de Google. Heureusement, les API mobiles ne sont soumises à aucune limitation et sont entièrement gratuites.

Les interactions avec la carte interactive pour un utilisateur sont les suivantes : lors de l'ouverture de la carte, si aucun accès au réseau est détecté, l'application prévient l'utilisateur que la connexion au réseau n'est pas active et la carte est gelée jusqu'au retour de la connexion. Quand la connexion est établie, la carte est affichée et centrée sur la position actuelle de l'utilisateur. Sous la carte, une liste de points d'intérêt est proposée à l'utilisateur. Chaque point d'intérêt peut être coché par l'utilisateur afin de l'ajouter à la liste des points affichables sur la carte. L'utilisateur coche les points d'intérêt qu'il désire et peut ensuite sélectionner, parmi cette liste, les points à afficher sur la carte via un marqueur. Un point d'intérêt est initialement décoché pour éviter de surcharger la carte avec les nombreux points d'intérêt disponibles. L'utilisateur a donc la possibilité de choisir quel point d'intérêt il désire afficher sur la carte.

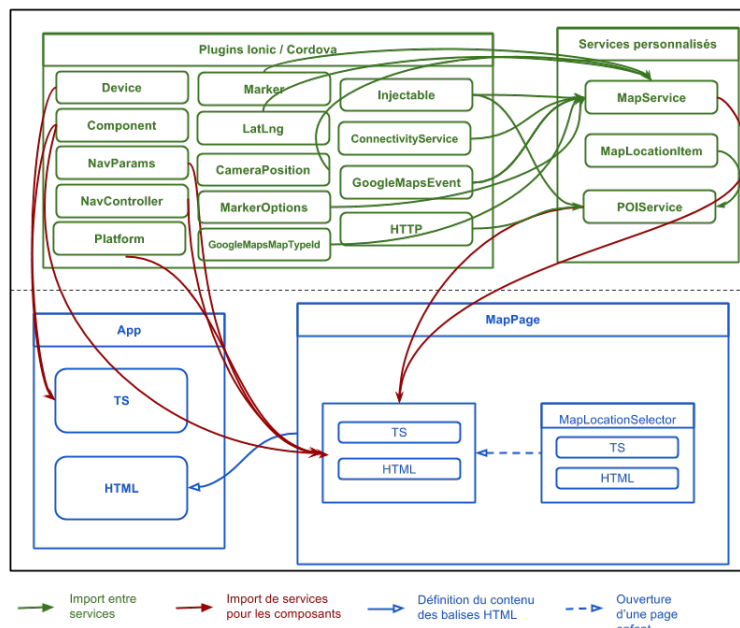


FIGURE 4.10 – Abstraction des interactions pour le module Cartes

14. <https://github.com/mapsplugin/cordova-plugin-googlemaps>

La logique du module des cartes est divisée en quatre composants illustré dans la figure 4.10 :

MapPage est le composant chargé d'afficher la carte, ses marqueurs ainsi que la liste des points d'intérêt du campus. Quand le module cartes est appelé, MapPage fait une requête auprès de MapService afin d'initialiser la carte à afficher. Une seconde requête est effectuée auprès de POIService pour charger la liste de points d'intérêt de la carte. Ces requêtes étant asynchrones, elles sont gérées par des Promises.

Les deux requêtes résolues, MapPage crée un objet MapLocationItem représentant la localisation de l'utilisateur et l'ajoute à la liste des localisations affichables. Un objet MapLocationItem est représenté par cinq attributs : un nom, une adresse, une latitude, une longitude et un sigle. Dans l'ensemble de l'application, chaque localisation est représentée par un objet MapLocationItem. La latitude et la longitude de l'utilisateur sont des variables instantiées par le MapService lors de l'initialisation de la carte. Cet objet créé, MapPage fait une nouvelle requête au MapService afin d'ajouter un marqueur sur la carte représentant la position de l'utilisateur.

MapService est un service qui crée et maintient l'objet map à afficher que ce soit pour le navigateur ou pour le mobile. Lorsque MapService est appelé, il appelle un service externe au module, ConnectivityService, afin de vérifier si l'utilisateur est bien connecté au réseau. Ensuite, ce service vérifie si la plateforme utilisée est mobile ou non et adapte l'origine du chargement de la carte en fonction. Si la plateforme est mobile c'est le plugin cordova qui sera appelée, sinon c'est l'API Javascript qui sera appelée. Une fois connecté, ce service fait appel au plugin Ionic Geolocation¹⁵ afin de récupérer la latitude et la longitude de l'utilisateur. Lorsque la position de l'utilisateur est déterminée, le service instancie un objet map avec deux paramètres : mapOptions et mapElement. MapOptions représente les options de la carte à instancier : sur quelle position centrer la carte (ici la position de l'utilisateur), le zoom initial et le type de carte à afficher. L'application propose une carte de type roadmap. MapElement est une référence vers le conteneur HTML dans lequel sera affichée la carte. La carte étant initialisée et affichée, MapService n'a plus qu'à gérer les marqueurs. Une liste de marqueurs est maintenue à jour durant l'utilisation de la carte. Lorsque l'utilisateur veut afficher une nouvelle position sur la carte, un objet Marker est créé et ajouté à la liste. Pour afficher un objet Marker sur la carte, il suffit de référencer l'objet map à ce Marker. Afin d'éviter de créer et détruire plusieurs fois le même objet Marker, la suppression d'un marqueur est gérée via la visibilité du marqueur. L'objet Marker n'est donc pas détruit même s'il n'est plus affiché sur la carte. L'objet reste présent dans la liste des marqueurs de MapService mais sa visibilité prend la valeur "false". Enfin, MapService vérifie que l'utilisateur reste bien connecté durant l'utilisation du module. Ceci se fait via l'ajout de deux EventListeners, 'online' et 'offline', au module grâce à la méthode javascript document.addEventListener(). Ainsi, lorsque l'évènement 'online' se produit, MapService instancie (si nécessaire) et présente la carte. Lorsque l'évènement 'offline' se produit, MapService prévient l'utilisateur qu'il n'est pas connecté au réseau et désactive l'utilisation de la carte.

POIService est un service permettant de charger les points d'intérêt à présenter à l'utilisateur. Les points d'intérêt sont chargés à partir d'un fichier JSON enregistré dans les assets de l'application. Une manipulation de fichier JSON a été préféré à une base de données car les points d'intérêt sont peu propices au changement. En effet un auditoire, un parking ou bibliothèque, ont peu de chance de changer d'adresse durant leur existence. Ces informations sont destinées à être lues et non modifiées. Un autre avantage de la manipulation de JSON par rapport aux bases de données est que le format JSON est lisible et compréhensible par l'humain et par la machine. Un humain non informaticien peut ajouter et modifier un JSON une fois qu'il a compris la structure de celui-ci. Tandis que l'ajout et la modification de données sur une base de données standard sont plus fastidieux pour des non-initiés sans l'aide d'un interface graphique adaptée. Si une

15. <https://github.com/apache/cordova-plugin-geolocation>

nouvelle zone de points d'intérêt doit être créée, le fichier JSON peut être mis à jour ou un autre fichier JSON suivant la même structure peut être ajouté par tous les membres de la communauté UCLCampus.

Une autre fonction associée à ce service est le calcul des distances entre deux points. Lorsque l'utilisateur sélectionne un point d'intérêt, la distance entre sa position et le point en question est affichée à l'écran. La formule "Haversine" est utilisée pour calculer la distance entre deux points exprimés par leurs coordonnées géographiques (latitude et longitude). Cette formule mathématique permet de calculer la plus petite distance entre deux points sur une sphère et est communément utilisée pour trouver la distance entre deux points.

MapLocationSelector est une page modale qui gère la sélection du point d'intérêt à afficher sur la carte. Une page modale est un volet de contenu qui est présenté au-dessus de la page en cours d'utilisation. L'utilisateur crée une liste de points d'intérêt affichables en les cochant via la MapPage. C'est par cette liste qu'il peut sélectionner les points à afficher sur la carte. La création de cette liste se fait grâce à la MapPage mais, la sélection dans cette liste est gérée par MapLocationSelector. MapPage appelle cette page modale avec deux paramètres : la liste des localisations à présenter et la position actuellement affichée sur la carte. Une fois que l'utilisateur a sélectionné le point qu'il désire afficher, la page modale retourne la valeur sélectionnée et renvoie l'utilisateur à la MapPage. L'implémentation de cette page modale est réalisée grâce au composant ionic ModalController.

La figure 4.11 reprend l'ensemble des interactions effectuées par la fonctionnalité Cartes.

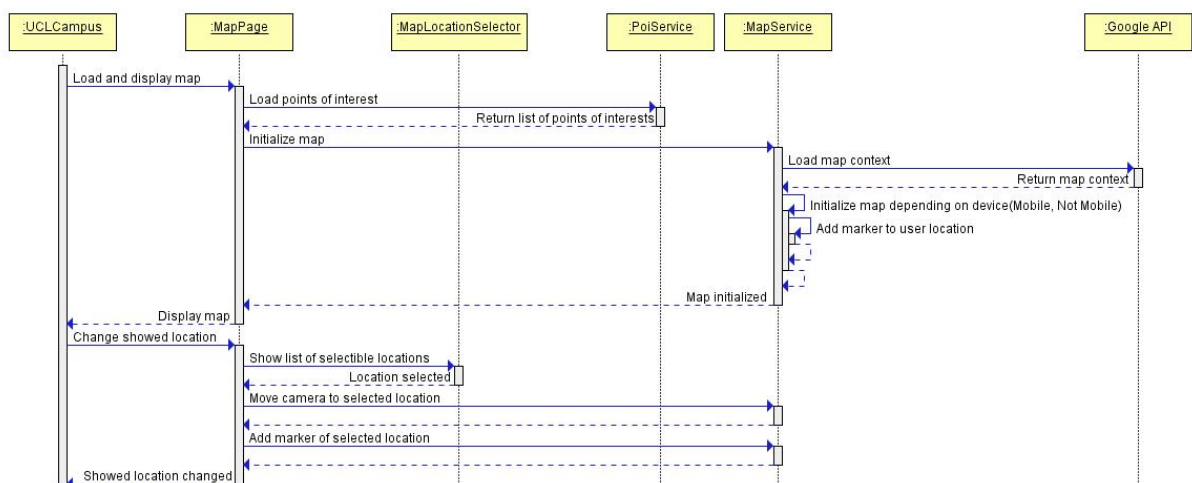


FIGURE 4.11 – Diagramme de séquence pour le module Cartes

4.5.6 Informations codées en dur

Le solution d'intégrer directement l'information dans l'application a été utilisée en dernier recours, quand l'information n'est pas disponible en dehors du portail de l'UCL et qu'aucune autre solution n'est envisageable. Cette méthode a été choisie parmi d'autres parce qu'elle offrait moins d'inconvénients. Elle est rapidement mise en place en créant une page statique mettant en forme les informations disponibles sur le portail. Évidemment, il est préférable que les informations disponibles ne soient pas sujettes à des changements réguliers pour éviter l'obsolescence de l'information s'y trouvant. Les autres solutions envisageables qui ont été refusées sont :

- Le parsing des pages du site internet possédant les information désirées. Cette approche nécessite une connexion internet et demande d'écrire des scripts permettant de récupérer

les informations depuis la structure HTML de la page. Le fonctionnement de cette méthode est dépendante de la manière dont les informations sont structurées or, la structure d'une page internet change régulièrement.

- Le chargement de la page internet contenant l'information dans un navigateur web. Cette approche demande à l'utilisateur d'être connecté au réseau et de quitter momentanément l'application. L'URL d'une page peut également être modifiée.

Un avantage de la méthode sélectionnée est qu'elle est simple et facile à modifier. Lorsque des sources génériques seront mises en place pour récupérer ces informations, il suffira au programmeur de récupérer les données à l'aide de services et de transformer ces données en objet pour les intégrer dans structure HTML existante de la page de l'application.

4.5.6.1 Service Desk

Le module Service Desk est un module, représenté par la figure 4.12 affichant des informations utiles du service Desk de l'UCL. Les informations ont été copiées depuis le site www.uclouvain.be¹⁶. L'utilisateur est informé du but du Service Desk ainsi que des informations de contact. Il y est aussi expliqué comment obtenir une SmartCard et son utilité. L'utilisateur trouve également des informations d'aide à la connexion au wifi de l'UCL et divers pdf explicatifs selon les systèmes d'opérations utilisés.

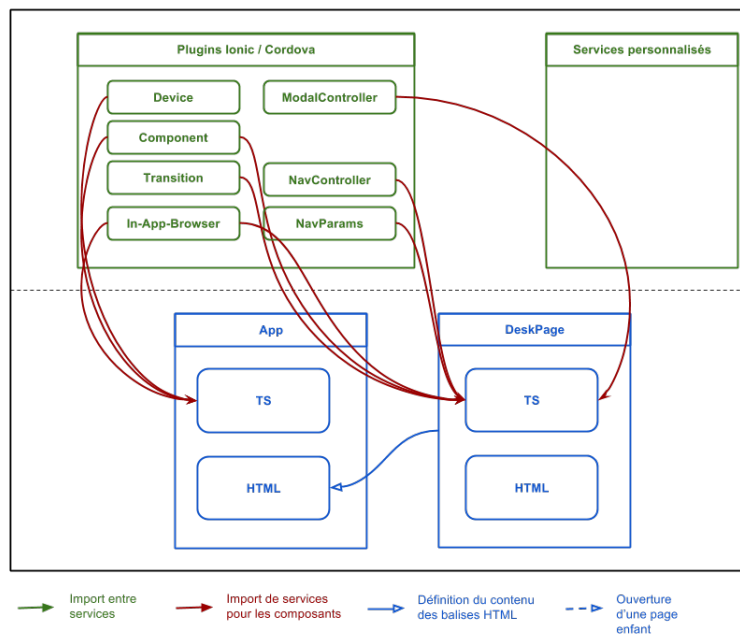


FIGURE 4.12 – Abstraction des interactions pour le module Desk

4.5.7 Lancement d'application

Les partenaires de l'Université catholique de Louvain ont déployé des applications spécifiques à leur domaine de compétence. Ces applications répondent à certaines fonctionnalités listées dans le cahier de charges. Il était envisageable d'exploiter les informations disponibles sur ces applications pour les intégrer au sein de l'application UCLCampus. Mais le choix le plus judicieux est de permettre le lancement de ces applications depuis l'application UCLCampus. Ce choix est justifiable pour les raisons suivantes :

16. <https://uclouvain.be/fr/decouvrir/service-desk.html>

- Cela permet d'éviter une duplication et un recouplement des fonctionnalités existantes dans les applications de l'environnement UCL et permet une réutilisation des fonctionnalités spécifiques existantes.
- Cela évite une concurrence inutile entre deux applications visant des objectifs similaires.
- Les partenaires de l'université restent les seuls responsables du contenu et de l'information disponible. Ils peuvent avoir un retour direct sur les utilisations de leur application.
- L'application UCLCampus va permettre de faire connaître et de centraliser les applications et les services existants.

L'inconvénient d'une telle approche est que l'utilisateur doit télécharger différentes applications sur son mobile pour accéder à l'ensemble des services. De plus, les applications se lancent depuis l'application UCLCampus et nécessite donc de quitter momentanément l'application UCLCampus. A ce moment-là, l'utilisateur est confronté à une nouvelle interface et un nouveau design car il n'existe pas de convention de développement entre les applications de l'UCL. Elles ont été, chacune, développées par des sociétés différentes.

L'application UCLCampus permet le lancement de l'application Resto4u développée par les restaurants universitaires et l'application Commuty, une application d'un partenaire de l'UCL, pour la mise en place de covoiturage.

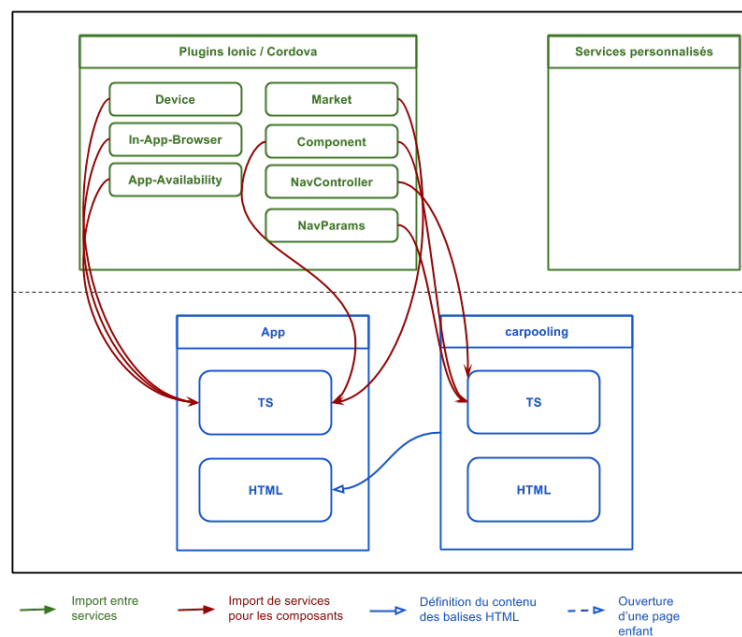


FIGURE 4.13 – Abstraction des interactions pour le lancement d'application

L'implémentation du lancement d'une application se fait dans la page principale de l'application. En effet, c'est lors de l'ouverture d'une page racine à partir du menu principal avec l'appel à la méthode d'ouverture `openRootPage()` qu'intervient la méthode de lancement de l'application. Dans un premier temps, la méthode `openRootPage()` vérifie si la page ouverte est une page impliquant le lancement d'une application externe. Si c'est le cas, la méthode appelle la méthode `launchExternalApp()`, sinon elle ouvre simplement la page racine désirée. La logique derrière l'implémentation de la méthode `launchExternalApp()` est la suivante :

- Prendre connaissance de la plateforme du mobile grâce au service Ionic Device.
- Vérifier si l'application est déjà présente sur le mobile à l'aide du service Ionic App-availability.

- Si l'application est présente, le service in-app-browser démarre l'application avec pour argument l'Uniform Resource Identifier (URI) de l'application désirée.
- Si l'application n'est pas présente, le service market ouvre l'application de téléchargement d'applications spécifiques au mobile sur la page de l'application désirée afin que l'utilisateur puisse prendre connaissance de son existence et l'installer.

Les services utilisés et repris dans la figure 4.13 dans la méthode sont :

- App Availability¹⁷ : plugin sous licence MIT. Les plateformes supportées sont Android et IOS.
- cordova-plugin-device¹⁸ : plugin sous licence Apache-2.0. Les plateformes supportées sont Android et IOS.
- cordova-plugin-inappbrowser¹⁹ : plugin sous licence Apache-2.0. Les plateformes supportées sont Android, IOS, Amazon Fire OS, Windows 8 et les navigateurs web.
- cordova-plugin-market²⁰ : plugin sous licence MIT. Les plateformes supportées sont Android et IOS.

Quand l'utilisation de l'application est finie, la WebView contenant l'application se referme et l'utilisateur revient sur l'application UCLCampus, plus précisément sur la page contenant des informations sur le partenaire de l'UCL fournissant l'application. Ces informations sont des informations générales présentes sur le portail de l'UCL, elles ont été écrites en dur dans le fichier HTML de la page.

L'ouverture de l'application Commuty n'a pas posé le moindre problème contrairement à l'application Resto4U. Suite à de nombreux contacts par mail avec la société AppTree, responsable du développement de l'application Resto4U, le problème rencontré à l'ouverture de l'application a pu être résolu. Le problème était que l'application UCLCampus ouvrait bel et bien une WebView pour permettre le lancement de l'application Resto4U mais, cette dernière refusait de s'ouvrir depuis une application externe. Les développeurs de la société AppTree ont du mettre à jour l'application en ajoutant quelques lignes afin de permettre l'ouverture externe de l'application. Cette mise à jour effectuée, l'ouverture Resto4U de l'application se déroule sans difficulté. Suite aux échanges de mails et de vidéos, des défauts d'affichage pour les écrans 5 pouces ont également été relevés par l'entreprise.

17. <https://github.com/ohh2ahh/AppAvailability>

18. <https://github.com/apache/cordova-plugin-device>

19. <https://github.com/apache/cordova-plugin-inappbrowser>

20. <https://github.com/xmartlabs/cordova-plugin-market>

Chapitre 5

Évaluation

Ce chapitre offre l'occasion de prendre du recul sur ce qui a été réalisé tout au long du développement du projet UCLCampus. La première section discute des difficultés rencontrées et de l'impact du développement du projet en Open Source. Les sections suivantes relatent les observations et difficultés rencontrées suite à l'interaction avec les différents services de l'UCL. La quatrième section jette un regard critique sur la technologie utilisée pour le développement du projet. Pour finir, une analyse détaillée de l'état d'avancement du projet et de l'application, au regard du cahier des charges, est réalisée. Cette analyse est suivie de pistes d'amélioration et d'idées pour les futures fonctionnalités à implémenter.

5.1 Open Source

UCLCampus est un projet développé en Open Source dans le but de permettre aux étudiants de l'université de pouvoir s'y impliquer. Lors de la reprise du projet, nous avons récupéré l'application de nos prédécesseurs depuis leur compte Github. Dans un premier temps, il nous semblait pertinent de développer le projet sur un Github privé pour nous permettre de prendre connaissance des bonnes pratiques de codage avec le framework Ionic. Nous voulions éviter de publier publiquement un projet qui ne tenait pas la route et qui n'avait pas de bases solides car cela risquait de faire une mauvaise publicité au projet. Avec le recul, cette approche fut une erreur. Dans un premier temps, il aurait fallu dès le départ mettre en place un écosystème autour de l'application UCLCampus. Les éléments de cet écosystème sont :

- Une communication efficace avec des outils de communication et l'utilisation des réseaux sociaux pour informer de l'avancement du projet.
- Un guide de développement et un wiki à jour
- Des outils d'intégration continue

Un écosystème bien construit permet d'attirer une communauté et de la faire vivre autour d'un projet en développement. En ayant développé le projet dans notre coin, nous n'avons pas pu construire une communauté autour de celui-ci. L'intégration au projet demande un investissement conséquent pour tous les développeurs qui peuvent être intéressés à y participer. Si le projet avait été construit dès le départ en Open Source, il aurait pu attirer, dès ses prémices, des développeurs. Ces derniers auraient généré des questionnements sur le projet, son orientation ainsi que des mises à jour régulières de l'écosystème.

Face à ce constat, nous avons changé d'optique. Notre projet Github est devenu public et les outils qui constituent l'écosystème du projet UCLCampus ont été mis en place. Ces outils sont :

Readme détaillé pour permettre l'initialisation du projet sur sa machine

Gitter pour offrir un espace de discussions

Zenhub pour définir les tâches à réaliser

Wiki pour informer de l'avancement du projet et des décisions stratégiques prises

La mise en place d'un tel écosystème est chronophage surtout qu'il a tout fallu reprendre depuis le début. Mais, ce changement d'attitude a été bénéfique : nous avons pu nous rendre compte de certaines incohérences et erreurs de compatibilité qui n'étaient pas apparues en travaillant en cercle fermé. Il faut de la rigueur pour maintenir à jour l'écosystème, mais c'est primordial.

Malgré cet investissement, nous n'avons pas réussi à créer une communauté, ni à attirer de nouveaux développeurs sur le projet. Actuellement, nous avons dix personnes qui suivent le projet, deux contributeurs et un fork. En analysant les autres projets Open Sources développés à l'UCL, OSIS et Inginius, un manque d'engouement et l'absence de réelles communautés soutenant ces projets sont également constatés. En chiffre, cela donne :

OSI 25 stars, 9 forks et 21 contributeurs

Inginius 44 stars, 25 forks et 11 contributeurs

Ionic 30 500 stars, 8 000 forks et 235 contributeurs

Angular 26 000 stars, 6600 forks et 487 contributeurs

Ces projets sont pourtant pertinents, bien structurés, soutenus et mis à jour quotidiennement par des professionnels employés par l'université. Nous émettons l'hypothèse que des projets finaux tels que OSIS, Inginius et UCLCampus ne peuvent s'adresser qu'à un nombre restreint de développeurs et d'utilisateurs qui rencontrent les mêmes besoins, malgré le souhait d'être le plus générique possible. Ces projets restent aussi fortement dépendants des technologies et de la structure de l'information présentes à l'UCL. Tandis que des projets Open Source proposant le développement de nouvelles technologies, de frameworks, de plugins mobilisent plus de développeurs car, par leur essence même, ils sont plus génériques et répondent potentiellement à un besoin plus large. De plus, les étudiants de l'UCL n'ont pas encore les réflexes de s'intéresser aux projets Open Source développés au sein de l'UCL, malgré l'existence de l'Open Week, semaine de développement par des étudiants de projets Open Source.

Pour offrir une plus grande visibilité et une plus grande crédibilité au projet, il nous semble important de transférer le projet UCLCampus sur le Github officiel de l'Université catholique de Louvain. De plus, pour pérenniser le projet, il nous semble essentiel d'avoir une personne permanente investie dans le projet pour permettre la transmission de l'information, de la philosophie, du cahier des charges ainsi que la justification des choix d'implémentation. En effet, malgré toutes les informations, les guides et les outils mis en oeuvre, il nous semble difficile, d'après notre expérience, de s'imprégner du projet sans avoir l'occasion de rencontrer une personne maîtrisant la genèse du projet.

5.2 Environnement UCL

Le projet UCLCampus est développé dans un environnement spécifique qui est l'environnement de l'UCL. L'application doit pouvoir interagir proprement avec les services disponibles à l'université. Ces interactions nous ont forcés à prendre connaissance de l'environnement et de l'infrastructure informatique présente à l'UCL.

La découverte de cet environnement fut laborieuse. Nous avons avancé par tâtonnements car nous n'avions aucune connaissance préalable des services existants et disponibles. Ce n'est qu'au fur et à mesure des réunions que nous avons pu créer un réseau de contacts et de personnes ressources. Afin que nos successeurs puissent bénéficier de ce travail, une liste des personnes contactées, se trouvant en annexe du travail, a été établie.

Au fil de nos rencontres avec les différents services et partenaires de l'Université catholique de Louvain, nous avons constaté un manque de communication et de vision globale. Chaque service et partenaire possède son propre pôle informatique avec des différences de ressources notables. Par exemple, le pôle informatique du service des restaurants universitaires possède un seul employé,

ayant suivi une formation dispensée par l'université, pour gérer la mise à jour des informations sur le portail de l'université. Le service des bibliothèques possède, lui, une équipe d'informaticiens pour gérer l'ensemble des données informatiques des bibliothèques. Les besoins des deux services sont évidemment différents mais il n'empêche qu'une harmonisation, une collaboration et une communication des projets informatiques réalisés au sein des différents services seraient utiles. Un autre exemple flagrant de manque de concertation est le développement de deux applications pour deux partenaires de l'UCL, Univers Santé et les restaurants universitaires, par des sociétés de développement différentes. Ce fait engendre un manque de cohérence entre les applications et des contrats différents (certains sont propriétaires du code et d'autres non). En se concertant, il aurait été possible de faire développer les deux applications par la même société, de réaliser des économies d'échelle et, peut-être, permettre à d'autres services de se doter, eux aussi, de leur application. Une autre possibilité aurait pu être de développer ces applications et d'en assurer la maintenance en interne. Selon nous, les services et partenaires devraient davantage communiquer sur leurs projets informatiques. Un organe de réflexion devrait exister pour permettre à chaque partenaire et service d'exprimer ses besoins numériques, multimédias et informatiques et réfléchir à des solutions communes. L'utilité d'un tel organe peut s'illustrer dans un cas bien concret : Au niveau du service des sports, la gestion des candidats et des horaires des moniteurs sportifs est réalisée manuellement par la secrétaire. Or, on pourrait pour ce faire adapter l'outil de gestion des candidatures pour les tutorats.

Constatons, toutefois, l'harmonisation technique des différents services de l'université actuellement en cours avec la mise en place de l'Entreprise Service Bus WSO2 qui est un réel plus pour l'université. En effet, WSO2 offre une clarification des interactions entre les services grâce à une standardisation de communication. De plus, le développement en Open Source des projets soutenus par l'université permet une meilleure visibilité et une plus grande transparence des projets en cours.

Il est certain que la mise en place de ces changements ne se réalise pas instantanément car elle nécessite l'apprentissage de nouvelles technologies, l'harmonisation et la concertation entre les services, la prise de bonnes décisions stratégiques. De plus, les services doivent pouvoir continuer à être opérationnels malgré ces changements. Si toutes ces transformations tendent vers un environnement informatique plus stable et plus simple, pour l'instant, l'environnement est fortement instable suite à ces changements. L'instabilité de l'environnement impacte directement le développement de l'application UCLCampus. Certaines APIs ne sont pas encore disponibles sur le WSO2 pour permettre des interactions standards. Par exemple, l'API pour l'horaire ADE est encore en cours de production. Une fois celui-ci mis en place, il nécessitera la création d'un nouveau service pour le module étude qui, actuellement, envoie des requêtes directement sur le serveur ADE sans passer par l'API fourni par WSO2. Cette méthode pose un problème si l'URL du serveur ADE change, comme cela a été le cas le 20/07/2017. Suite à nos échanges avec le service des bibliothèques, deux API ont pu être intégrées à l'ESB WSO2. Ces deux API donnent des informations sur les bibliothèques et leurs horaires d'ouverture.

Le développement au sein d'un tel environnement a, dans un premier temps, créé un sentiment de frustration car nous avons passé de nombreuses heures à écrire des mails, à rencontrer des personnes à la place de pouvoir coder l'application attendue. Nous avons également été confrontés à la difficulté d'être dépendants d'autres programmeurs, ce que nous n'avions jamais rencontrés lors des projets réalisés durant notre cursus universitaire. La découverte du processus de développement informatique, avec ses différentes phases, de l'université nous semblait également être un frein pour le développement de l'application qui était un service externe. Mais nous avons, au fil du projet pris conscience de son importance. Nous n'avions pas une vision globale et encore moins la main mise sur tous les éléments de l'environnement des services informatiques à l'UCL. Même si les employés que nous avons pu rencontrer dans le cadre de notre projet ont été patients et pédagogues avec nous pour nous expliquer au mieux l'organisation des services informatiques de l'Université catholique de Louvain, il se dégage un sentiment d'une machinerie complexe où tout

changement est difficile à opérer.

5.3 Sécurité

Le système de sécurité mis en oeuvre sur une branche secondaire du projet Github UCL-Campus est limité et possède des failles. C'est pour ces raisons que le répertoire des membres de l'UCL n'est pas disponible sur l'application malgré l'existence d'une API WSO2 opérationnelle. Les données contenues dans le répertoire ne peuvent pas être du domaine public.

La méthode de sécurité implémentée se résume à une approche détournée d'authentification de l'utilisateur de manière interne à l'application. Cette authentification se fait, lors d'un accès à une fonctionnalité exclusivement réservée aux membres de l'université, par l'ouverture d'un navigateur web sur la page contenant le formulaire d'authentification aux services de l'université. L'utilisateur encode son identifiant et son mot de passe en vue de s'identifier. L'application observe la réaction du navigateur : si l'utilisateur est reconnu par le service SSO de l'université, il est automatiquement dirigé vers une nouvelle page. L'application UCLCampus en est informée et a connaissance que l'utilisateur est capable d'être authentifié par les services de l'UCL. Elle peut donc lui laisser l'accès aux services disponibles aux membres de l'Université catholique de Louvain. La barre permettant la modification de l'URL doit être inaccessible à l'utilisateur pour éviter qu'il ne modifie lui-même l'URL.

Il est évident qu'avec le projet disponible en Open Source, un utilisateur peut modifier le code de l'application pour ne plus devoir passer par la page d'authentification du navigateur web avant d'accéder aux fonctionnalités réservées aux membres de l'UCL, ce qui rend cette sécurité inutile.

Pour intégrer les fonctionnalités nécessitant une authentification en tant que membre de l'université, une possibilité serait de mettre en place le protocole OAuth entre l'API Manager WSO2 et l'application UCLCampus. OAuth est un protocole qui permet à une application externe d'accéder à certaines informations ou ressources d'un utilisateur à partir d'un autre service Web, sans donner les informations d'identification de l'utilisateur pour le service Web à l'application externe[44]. L'outil WSO2 utilisé par l'Université catholique de Louvain propose déjà une version du protocole OAuth implémenté parmi ses services. La mise en place d'un tel mécanisme d'authentification ne modifierait pas fondamentalement la structure interne des services de l'UCL, comme l'illustre la figure 5.1. Les services de l'UCL interagissent déjà en grande partie au travers des outils WSO2. Ce procédé permettrait de conserver une certaine cohérence dans le système.

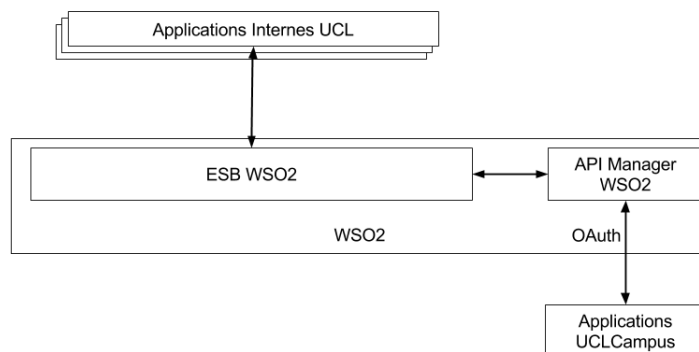


FIGURE 5.1 – Système de sécurité théorique future pour l'application UCLCampus

Le protocole OAuth fait intervenir trois acteurs : une application externe, l'utilisateur de

l'application et un système responsable de ressources sécurisées. Dans notre cas, l'application est l'application UCLCampus, l'utilisateur est une personne membre de l'UCL et l'outil WSO2 de l'UCL en tant responsable de ressources sécurisées. Le module UCL WSO2 comprend différents composants tels qu'un identity provider, des services providers, un API Manager et un ESB. La figure 5.2 donne un aperçu des différentes étapes du protocole OAuth[45].

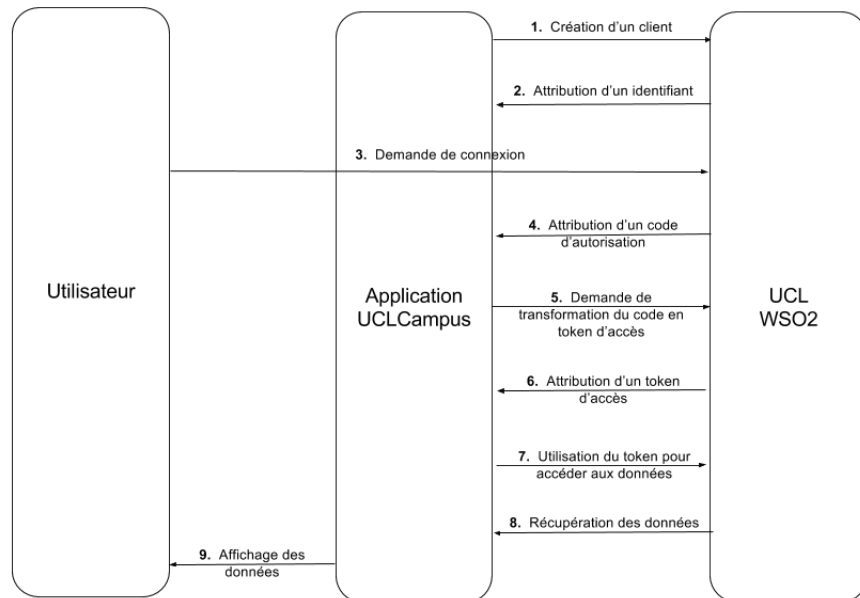


FIGURE 5.2 – Système de sécurité théorique future pour l'application UCLCampus

1. L'application UCLCampus est déclarée auprès des services UCL en créant un client et en listant les droits d'accès désirés.
2. UCL WSO2 attribue à l'application UCLCampus, un client ID et un client secret.
3. Lorsque l'utilisateur désire se connecter à un service sécurisé de l'UCL, l'application le redirige vers une adresse spécifique définie par UCL WSO2. Cette adresse demandera à l'utilisateur de s'identifier et d'accepter, ou de refuser, à l'application les accès aux services, ressources et informations utiles à son fonctionnement.
4. Si l'utilisateur accepte, il est renvoyé sur l'application avec un code d'autorisation.
5. Ce code d'autorisation permet à l'application UCLCampus de demander un token d'accès.
6. UCL WSO2 attribue un token d'accès permettant l'accès à l'API Manager WSO2 en fonction de l'identité de l'utilisateur.
7. L'application interroge l'API Manager WSO2 pour récupérer des données.
8. L'API Manager WSO2 transmet les données.
9. L'application affiche les données à l'utilisateur.

Le protocole OAuth propose de nombreux avantages :

- Il évite que les applications externes manipulent et prennent connaissance de l'identifiant et du mot de passe des membres de l'UCL, ce qui protège les membres de l'UCL.
- L'outil WSO2 est prévu pour intégrer le protocole d'authentification OAuth.
- Le protocole OAuth est un protocole standard qui peut être exploité pour tous les développements futurs d'applications externes à l'UCL.

5.4 Framework Ionic

La technologie Ionic a été choisie par nos prédécesseurs, il a donc fallu la découvrir et se l'approprier. Après une utilisation du framework Ionic, nous pouvons dire que c'est un framework possédant une large communauté dynamique et réactive. Durant la période de développement du projet UCLCampus, Ionic framework a subi différentes mutations pour devenir plus stable. La gestion de ces mutations et leurs impacts sont abordés dans la première sous-section. Ionic étant un framework de développement d'application cross-plateforme hybride, les applications de ce type possèdent des avantages mais également des inconvénients. Elles sont moins performantes que les applications natives et possèdent certaines limitations de fonctionnalités. Une discussion sur le choix d'un développement hybride est effectuée dans la deuxième sous-section.

5.4.1 Les différentes versions

Durant la période de développement de l'application UCLCampus, le framework Ionic a vécu le passage de la version 1 à la version 3. Les différentes versions ont existé simultanément et ont été développées parallèlement. Pour donner un aperçu de l'évolution rapide du framework, voici les événements marquants :

- Le 12 mai 2015, le blog officiel du framework annonce l'arrivée d'Ionic 1.0.¹
- Le 10 février 2016, la version beta du framework Ionic 2 est annoncée.² Le framework est modifié afin d'anticiper la révolution du passage de l'architecture Angular JS à l'architecture Angular 2 alors que celui-ci est toujours en phase de développement. La version 1 du framework continue de subir de nombreuses mises à jour.
- Le 28 septembre 2016³, Ionic lance officiellement Ionic 2 release candidate qui intègre la version finale et stable d'Angular 2 et de nouvelles fonctionnalités. Dans les mois qui ont suivi cette annonce, la documentation sur Ionic 2 est mise en avant au détriment de la documentation de la version 1. Cette version est encore instable et contient encore des erreurs mais est opérationnelle pour démarrer le développement d'une application.
- Le 25 janvier 2017 est le jour de la sortie de la version finale d'Ionic 2.⁴
- Le 9 mars 2017, Ionic annonce l'arrivée prochaine d'une nouvelle version du framework.⁵ Les apports de la version impactent principalement l'interface de ligne de commande.
- Le 7 avril 2017, la version beta d'Ionic 3.0 est lancée.⁶ Les changements principaux, autre qu'un lifting de l'interface de ligne de commandes, sont les mises à jour des technologies utilisées par Ionic, c'est-à-dire passage à Angular 4 et support de TypeScript 2.2. Il y a également l'apparition du concept de lazy loading pour améliorer la fluidité de la navigation sur les applications.
- Le 9 mai 2017, c'est la sortie officielle d'Ionic CLI v3 qui corrige quelques erreurs présentes sur la version beta.⁷

Initialement, en parallèle aux enquêtes et réunions pour définir le concept de l'application et son cahier de charges, nous avons suivi les tutoriels sur Ionic 1 et Angular JS pour acquérir une bonne maîtrise de ceux-ci lorsque la phase de développement de l'application démarre. Courant du mois de novembre, nous avons pris la décision de migrer le projet UCLCampus vers Ionic 2. Ce choix était risqué car la version 2 du framework était encore instable et la date de sortie de la version finale n'était pas encore annoncée mais, au vu de l'enthousiasme de la communauté, de l'apparition de projets d'application Ionic 2 développés par des sociétés ainsi que la régularité

1. <http://blog.ionic.io/announcing-ionic-1-0/>

2. <http://blog.ionic.io/announcing-ionic-framework-2-beta/>

3. <http://blog.ionic.io/announcing-the-ionic-2-release-candidate/>

4. <http://blog.ionic.io/announcing-ionic-2-0-0-final/>

5. <http://blog.ionic.io/ionic-cli-v3-beta/>

6. <http://blog.ionic.io/ionic-3-0-has-arrived/>

7. <http://blog.ionic.io/announcing-ionic-cli-v3/>

des postes sur le blog officiel d'Ionic, nous nous sommes dits que nous devions prendre ce risque. Cette décision eut pour conséquence de reprendre le projet UCLCampus à zéro car le passage d'Ionic 1 à Ionic 2 impliquait un changement total de la structure et de l'architecture du projet suite au passage vers Angular 2. L'instabilité et les mises à jour fréquentes du framework n'ont pas toujours été évidentes à gérer, cela nécessitait de revoir en permanence notre code et de l'adapter à celles-ci. Suite à ce choix, nous avons appris à faire partie d'une communauté, à suivre régulièrement les avancées dans le développement du framework et à faire part des problèmes rencontrés lors du développement de notre application. Au vu de la mort lente d'Ionic version 1, nous pensons avoir pris la bonne décision. Ionic 2 est plus facile à appréhender, offre une meilleure structure du projet et les services proposés comme l'interface de ligne de commande sont d'une grande aide pour le développement d'application. Si nous n'avions pas suivi le mouvement, nous serions restés à quai avec une technologie désuète. Le changement de technologie aurait été encore plus douloureux une fois le développement du projet bien avancé.

5.4.2 Développement hybride

Ionic est un framework de développement d'application hybride qui manipule des WebViews pour afficher le contenu de l'application et, en même temps, atteindre les fonctionnalités natives du mobile et, fonctionner hors connexion. L'avantage principal du développement d'application hybride est le fait que le développeur code une fois l'application pour un ensemble de mobiles contrairement au développement natif qui nécessite de développer une application pour chaque type de mobile. En développant nativement, le développeur doit maîtriser différents langages de programmation. Les applications hybrides s'exécutent à l'intérieur d'une WebView, qui est une instance d'un navigateur web, ce qui nécessite une maîtrise des technologies web telles que HTML5, CSS et JavaScript (ou un langage dérivé). Ces technologies sont plus courantes et moins complexes à maîtriser que les technologies spécifiques au développement d'applications. Contrairement à une application web, qui utilise les mêmes technologies, une application hybride offre l'accès aux fonctionnalités natives du mobile et peut fonctionner hors connexion. Un autre avantage d'un développement d'application unique est le fait que les applications restent cohérentes : peu importe le mobile utilisé, les interfaces seront identiques.

Le plus grand reproche vis-à-vis des applications hybrides est leur manque de performance et de fluidité. Il y a en effet, actuellement, un déficit de performance des applications hybrides par rapport aux applications natives suite à l'exécution de l'application dans un WebView. Mais ce déficit se résorbe au fur et à mesure que les navigateurs web deviennent de plus en plus performants. L'amélioration des performances des plateformes mobiles tend également à réduire l'impression de manque de fluidité perçue par les utilisateurs d'applications hybrides. En effet, une plateforme mobile plus performante est capable d'effectuer un plus grand nombre d'actions sur un même laps de temps.

Le framework Ionic rencontre ce problème de performance, les applications développées sous Ionic sont plus lentes que les applications natives. Dans le cadre de notre application, nous constatons que celle-ci met du temps à s'ouvrir. Mais, une fois ouverte, la fluidité de navigation entre les pages est acceptable. La principale raison de cette lenteur au démarrage est causée par le chargement et l'instanciation de tous les services utilisés par l'ensemble de l'application. Lors du passage à la version 3 du framework, un mécanisme a été mis en place pour solutionner ce problème en proposant un lazy loading[46]. Le principe du lazy loading est de charger une ressource seulement lorsque celle-ci est nécessaire. Dans le cas d'Ionic, toutes les pages et tous les services ne doivent pas être chargés à l'ouverture de l'application. Une autre raison de la lenteur des applications est due à un problème de compatibilité entre Ionic et Angular. Ce problème est en cours de réglage. Il reste encore des imperfections au niveau des performances des applications hybrides. Mais elles tendent à se résorber, ce qui incite de plus en plus de développeurs à développer des applications hybrides.

5.5 Application déployée

Cette section offre l'occasion de prendre du recul sur le travail réalisé pour obtenir l'application qui est délivrée. Dans un premier temps, l'analyse portera sur le processus de développement utilisé. Ensuite, quelques chiffres importants liés au projet UCLCampus sont présentés. Dans le point suivant, une comparaison entre l'application délivrée et le cahier de charge est effectuée pour, finalement, conclure sur les possibilités d'évolution et les fonctionnalités futures.

5.5.1 Cycle de développement

Avec du recul, un processus de développement clair aurait dû être défini pour le projet. Malheureusement, cette question ne s'est pas posée au début du projet. Nous étions trop absorbés par l'avancée de l'application. En nous posant cette question, nous aurions pu être plus efficaces dans la suite du projet.

Le processus de développement, qui s'est naturellement mis en place, est un processus proche du cycle de développement en V. Les étapes de développement de notre projet ont été les

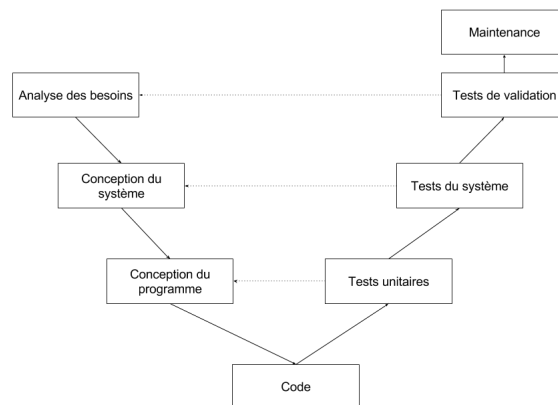


FIGURE 5.3 – Illustration du modèle en V

suivantes :

- Définition des besoins des utilisateurs et analyse de l'environnement de développement
- Établissement du cahier des charges
- Analyse des spécifications et contraintes techniques
- Décision sur l'architecture lors de la décision du passage d'Ionic 1 à Ionic 2.
- Écriture du code de l'application
- Écriture des tests unitaires
- Écriture des test d'intégration avec les tests bout à bout
- Analyse du produit final et des fonctionnalités opérationnelles

Étant donné l'objectif de déployer une application avec des fonctionnalités uniquement opérationnelles, nous aurions pu utiliser une approche agile de type Scrum pour le développement du projet. L'approche Scrum est une approche agile axée sur la satisfaction du client. Scrum est un processus itératif qui propose de découper la phase de développement en différents sprints, périodes de temps de travail. Au début de chaque sprint, les développeurs et le client définissent les objectifs du sprint à l'aide d'outils d'évaluation de charge de travail. A la fin de chaque sprint, une démonstration des fonctionnalités terminées est effectuée devant le client pour qu'il puisse donner son avis et demander des modifications à effectuer lors des prochains sprints. A la fin de chaque sprint a également lieu une analyse du fonctionnement de l'équipe et des difficultés rencontrées par celle-ci lors du sprint. Malgré tous les avantages d'une telle approche, dans notre situation certains problèmes apparaissaient.

- Nous possédions la double casquette de développeur et de client. En effet, nous avons nous même effectué et défini le cahier des charges et l'attente du client, ce qui rendait difficile la différenciation entre ces deux rôles cruciaux pour le développement sous l'approche Scrum.
- L'environnement de développement du projet était inconnu et instable, nous avons des difficultés pour définir les objectifs pour chaque sprint étant donné que nous ne savions pas quels services étaient déjà mis en place par l'UCL.

Maintenant que l'application possède suffisamment de fonctionnalités pour être déployée et présentée à des étudiants, il nous semble judicieux d'adopter une approche agile qui offre une flexibilité propice au développement Open Source. De plus, grâce à l'existence d'une application avec un nombre conséquent de fonctionnalités, les développeurs peuvent demander des retours réguliers aux utilisateurs de l'application à la fin de chaque sprint lorsque l'application est mise à jour.

5.5.2 Le projet en chiffres

Afin de mieux prendre conscience de l'ampleur du projet, voici quelques chiffres marquants :

- Le nombre de répondants à l'enquête réalisé est de 189.
- Nombre de fichiers de code créé
 - TypeScript : 47
 - Sass : 18
 - HTML : 18
- Nombre de lignes de code écrites
 - TypeScript : 3827
 - Sass : 557
 - HTML : 1191
- Nombre de contributeurs : 2
- Nombre de Fork : 1
- Nombre de suiveurs du projet sur github
 - Nombre de Stargazers : 7
 - Nombre de Watchers : 7

5.5.3 Comparaison avec le Cahier des charges

Cette sous-section reprend les différents éléments du cahier des charges établi comme base d'évaluation de l'application déployée. Dans un premier temps, les différentes fonctionnalités opérationnelles sont présentées. La deuxième partie évalue l'application selon les critères non fonctionnels énoncés dans le cahier des charges.

5.5.3.1 Partie fonctionnelle

L'ensemble des fonctionnalités présentées dans le cahier des charges sont divisées en trois catégories afin de faciliter l'analyse de l'application déployée. Chaque catégorie de fonctionnalités est reprise dans un tableau où pour chaque page appartenant à la catégorie, une évaluation des fonctionnalités devant être présentes sur cette page est réalisée.

Le tableau 5.1 reprend l'ensemble des pages liées à la vie sur le campus. Pour cette catégorie de fonctionnalités, le contrat est rempli excepté pour la page Sport qui n'apparaît pas dans l'application. En effet, aucune réunion n'a pu être organisée avec le service des sports pour permettre une récupération correcte des informations sur les séances de sports.

Le tableau 5.2 recense l'ensemble des pages contenant les fonctionnalités ne nécessitant pas une authentification UCL pour être utilisées. La majorité des fonctionnalités attendues dans cette catégorie a été implémentée.

Fonctionnalités attendues	Fonctionnalités réalisées	Remarques
Page Évènements		
Affichage de la liste des événements à venir	Oui	Depuis flux RSS de louvainfo
Affichage des détails d'un événement	Oui	
Connaitre l'horaire de l'événement	Oui	
Filtre des événements	Oui	Filtre par catégorie et la proximité temporelle
Recherche d'un événement par mot clé	Oui	
Ajout d'un événement à l'agenda	Oui	L'événement est ajouté uniquement sur l'agenda native du mobile
Ajout d'un événement en tant que favori	Oui	Fonctionnalité supplémentaire
Page Sports		
Affichage de la liste des séances de sports	Non	Pas contact avec le service des sports
Affichage de la description d'une séance de sport	Non	Pas contact avec le service des sports
Affichage de l'horaire d'une séance de sport	Non	Pas contact avec le service des sports
Filtre des séances de sport	Non	Pas contact avec le service des sports
Ajout d'une séance de sport à l'agenda	Non	Pas contact avec le service des sports
Ajout d'une séance de sport à l'agenda de manière périodique	Non	Pas contact avec le service des sports
Page Restaurants Universitaires		
Lancement de l'application Resto4U	Oui	
Proposition de téléchargement de l'application Resto4U	Oui	
Affichage des menus	Oui	Disponibles sur l'application Resto4U
Affichage des horaires	Oui	Disponibles sur l'application Resto4U

TABLE 5.1 – Fonctionnalités en rapport avec la vie sur le campus

Le tableau 5.3 liste les fonctionnalités en rapport avec les études des étudiants. La page principale de cette catégorie est la page Études. De nombreuses fonctionnalités de cette page n'ont pas été mises en oeuvre mais, la fonctionnalité principale, qui est d'offrir la possibilité aux étudiants d'organiser et d'afficher leur horaire de cours depuis l'application, est disponible. Les fonctionnalités supplémentaires de la page Études sont manquantes. Elles devraient apporter une plus value à cette page en centralisant l'ensemble des fonctionnalités liées à la vie académique.

Fonctionnalités attendues	Fonctionnalités réalisées	Remarques
Page News		
Affichage d'une liste d'actualité	Oui	Flux RSS disponible sur le portail UCL
Abonnement/Désabonnement au flux d'actualités	Non	Toutes les actualités ne sont pas accessibles via un flux RSS
Tri des actualité	Oui	
Recherche par mot clé	Oui	
Affichage des détails d'une actualité	Oui	
Redirection vers le site de l'actualité	Oui	
Page Configuration		
Choix de la langue d'affichage	Non	
Connexion/Déconnexion	Non	Fonctionnalité non déployée mais implémenté sur une branche du Github
Accès à l'adresse mail	Non	Existence d'applications spécifiques à la gestion des mails
Activation/Désactivation des notifications	Non	Aucune fonctionnalité ne nécessite des notifications
Page Covoiturage		
Proposition de téléchargement de l'application Commuty	Oui	
Lancement de l'application Commuty	Oui	
Page Carte interactive		
Affichage d'une carte	Oui	A l'aide des services Google Maps
Connaître sa position	Oui	A l'aide des services Google Maps
Connaître la localisation des points d'intérêt	Oui	
Afficher la liste des points d'intérêt	Oui	Depuis un fichier JSON
Filtrer les points d'intérêt	Oui	
Définir un itinéraire	Non	
Page Service Desk		
Affichage des informations d'aide informatique	Oui	Informations codées en dur

TABLE 5.2 – Fonctionnalités d'utilité publique

Fonctionnalités attendues	Fonctionnalités réalisées	Remarques
Page Études		
Synchronisation de la liste de cours avec le bureau virtuel	Non	
Synchronisation avec la liste de cours Moodle	Non	
Ajout/Suppression d'un cours	Oui	
Accès au portail UCL et site Moodle	Oui	
Affichage de l'horaire des cours selon la liste de cours	Oui	Effectue directement les requêtes par ADE, ne passe pas par l'API manager WSO2 en production
Affichage de la date de remise des travaux	Non	L'interaction avec le site Moodle n'est pas opérationnelle
Affichage de l'auditoire dans lequel le cours est dispensé	Oui	
Envoi de notifications avant les cours	Non	Le service de notification n'a pas été mis en place
Réglage des notifications	Non	Le service de notification n'a pas été mis en place
Sélection des séances de travaux pratiques	Non	Pas encore en place
Ajout d'un cours à l'agenda du mobile	Oui	Sur l'agenda natif
Synchronisation de l'horaire des cours sur l'agenda du mobile	Non	
Ajout d'une date de remise de travaux à l'agenda du mobile	Non	L'interaction avec le site Moodle n'est pas opérationnelle
Synchronisation des dates de remise des travaux sur l'agenda	Non	L'interaction avec le site Moodle n'est pas opérationnelle
Page Répertoire		
Recherche d'un étudiant	Non	API non disponible sur l'ESB WSO2
Affichage du profil d'un étudiant	Non	API non disponible sur l'ESB WSO2
Recherche d'un membre du personnel	Non	Problème au niveau de l'authentification et de la sécurité de l'information
Affichage du profil d'un membre du personnel	Non	Problème au niveau de l'authentification et de la sécurité de l'information
Page Bibliothèques		
Affichage de la liste des bibliothèques	Oui	A partir de l'ESB WSO2
Affichage des horaires des bibliothèques en fonction des périodes	Oui	A partir de l'ESB WSO2
Affichage des informations sur les bibliothèques	Oui	A partir de l'ESB WSO2

TABLE 5.3 – Fonctionnalités en rapport aux études

5.5.3.2 Partie non fonctionnelle

Les caractéristiques non fonctionnelles énoncées dans le cahier des charges sont reprises dans le tableau 5.4. Elles ont fait partie intégrante du processus de développement et des choix d'implémentation afin d'être satisfaites.

Caractéristiques	Remarques
Facilité d'utilisation	
Compréhension	Une vidéo explicative du fonctionnement de l'application est disponible. L'utilisation d'un menu unique rend l'utilisation de l'application intuitive
Apprentissage	Le projet est Open Source, le code source est accessible à tous. Un readme sur Github accompagne le développeur dans l'installation du projet. Un guide utilisateur reprend les choix d'implémentation du projet
Opérabilité	Les pages sont indépendantes les unes des autres ce qui permet le fonctionnement de l'ensemble de l'application malgré une source d'information défectueuse.
Maintenabilité	
Testabilité	Mise en place d'un environnement de tests facilitant l'intégration continue. Réalisation de tests unitaires et bout en bout.
Modularité	La modularité du projet est, dans un premier temps, assurée par l'architecture du framework Ionic. De plus, le respect de la scission entre la partie logique et l'interface utilisateur respecte la modularité. Le projet est structuré par des modules et des services indépendants qui peuvent être combinés et qui sont responsables d'une fonctionnalité précise.
Interopérabilité	Les services utilisés pour récupérer les informations auprès de leur source sont génériques et réutilisables pour interagir avec d'autres systèmes utilisant le même type de sources d'information.
Performances	
Temps de réponse	L'application, une fois lancée, est fluide. Les performances moindres des applications hybrides ne sont pas visible des utilisateurs.
Utilisation des ressources	L'espace mémoire du mobile utilisé par l'application est de 10,3 Mo.
Scalability	Non évalué
Compatibilité	
Installation	L'application est supportée par les mobiles tournant sous Android 4.4 et plus ou sous IOS version 8 et plus.
Mise à jour	Aucune mise à jour de l'application n'a encore été déployée.
Légal	
Licences	L'application est sous licence GPL-v3. Les plugins et outils utilisés sont compatibles avec cette licence.
Charte graphique	Utilisation du logo de l'université

TABLE 5.4 – Grille d'évaluation des critères non fonctionnels

5.5.4 Évolution et possibilités futures

Une première version de l'application UCLCampus a été déployée. Il reste cependant possible d'y apporter des améliorations et fonctionnalités supplémentaires afin de la rendre encore plus complète et attractive. Au cours du développement de l'application, des pistes d'améliorations et d'ajout de fonctionnalités sont apparues.

Voici les principales pistes de réflexion à évaluer pour le développement futur de l'application UCLCampus.

- Ouverture de l'application à l'ensemble des campus de l'Université catholique de Louvain. Actuellement, l'application est principalement centrée sur le campus de Louvain-La-Neuve. En effet, quelques fonctions de l'application sont spécifiques au campus sur lequel l'utilisateur se trouve. Ces fonctionnalités sont : la liste de points d'intérêt pour la carte interactive ne concerne que le campus de Louvain-La-Neuve. Par contre, les informations disponibles sur l'application Resto4U concernent l'ensemble des campus de l'UCL possédant un restaurant universitaire. Par une implémentation générique des services de récupération d'information, l'ouverture aux autres campus de l'université nécessitera des modifications moindres lorsque des sources d'informations similaires à celles accessibles sur le campus de Louvain-La-Neuve seront disponibles sur les autres campus. Les autres fonctionnalités ne sont pas spécifiques à un campus ce qui rend malgré tout l'application UCLCampus utilisable par l'ensemble des étudiants de l'UCL.
- Actuellement, l'application UCLCampus permet uniquement des interactions avec le calendrier natif des plateformes mobiles. Il serait intéressant d'élargir les interactions avec les autres calendriers couramment utilisés. Pour ce faire, une solution serait de permettre d'exporter un ensemble d'événements sélectionnés en format .ical.
- Offrir une application multilingue pour que les Erasmus inscrits à l'UCL puissent directement avoir accès aux différentes informations de l'université dans leur langue maternelle.
- La partie Études de l'application doit être étoffée à l'aide d'autres fonctionnalités dans le but de centraliser l'ensemble des informations et horaires liés au cursus académique des étudiants de l'UCL. Les fonctionnalités supplémentaires à intégrer à cette partie de l'application seraient :
 - Interagir avec les données disponibles sur le site de cours Moodle. Ces interactions permettraient d'offrir aux étudiants des fonctionnalités telles que : connaître la liste des cours pour lesquels l'étudiant est inscrit sur Moodle, la récupération des dates de remise de travaux, l'accès au forum des cours pour être informé des changements de locaux ou d'horaires, récupération des informations de l'agenda Moodle de l'étudiant.
 - Interagir avec OSIS et l'ESB WSO2 pour récupérer les informations personnelles sur le cursus de l'étudiant comme, par exemple, les cours auxquels il est inscrit sur son bureau virtuel, ou les résultats de ses examens.
- Un projet futur serait d'intégrer l'application Guindaille 2.0 à l'application UCLCampus en partenariat avec l'ASBL Univers Santé.
- L'application pourrait également prendre un aspect "réseau social" avec des fonctionnalités permettant aux étudiants de se rencontrer et partager des informations en fonction de leurs centres d'intérêts ou de leurs études. Ces fonctionnalités pourraient s'intégrer à des pages déjà existantes. Des fonctionnalités de gestion de travaux de groupe pourraient venir se greffer à la page Études. Les étudiants pourraient définir le lieu et la date de leur réunion de groupe ainsi que partager un rapport de réunion avec l'ensemble des membres de leur groupe. D'autres fonctionnalités dans le cadre de la vie sur le campus pourraient être de permettre aux kots à projets de créer et de partager directement leurs événements sur l'application.

Cette liste n'est pas exhaustive car il reste encore de nombreuses autres pistes à explorer pour faire évoluer cette application qui a pour but principal de faciliter la vie des étudiants sur les campus universitaires de l'UCL.

Chapitre 6

Conclusion

Développer une application qui soit Open Source, multiplateforme, utilisant la technologie Ionic, à destination des étudiants des différents campus de l'Université catholique de Louvain et qui réponde à leurs besoins et souhaits, telle est l'ambition de ce mémoire.

Afin d'y parvenir, différentes phases de travail et de réflexion ont été nécessaires pour mettre en place la solution la plus adéquate en fonction de divers critères et contraintes. La progression n'a pas été linéaire, de nombreux questionnements et remises en question ont eu lieu au cours du projet impliquant des ajustements et modifications.

La première phase a été une étape préparatoire consistant d'une part, en une appropriation des outils et éléments entourant le projet et, d'autre part en une collecte d'informations. Rassembler les données s'est effectué par différents biais : par la recherche d'articles, de sites internet et tutoriels de référence, mais également par un sondage auprès des étudiants pour déterminer leurs besoins et, enfin, par des réunions avec des personnes de différents services de l'Université catholique de Louvain. Cette étape préliminaire a permis d'établir, ultérieurement, un cahier des charges et une description du paysage informatique de l'UCL dans lequel le projet doit se déployer.

Une seconde étape a été le développement d'un premier « projet » de l'application sur un Github privé tout en multipliant les contacts avec des personnes ressources et des partenaires de l'université.

Suite à ces travaux, le cahier des charges de « UCLCampus » a été rédigé et a permis de cibler les éléments essentiels ainsi que les difficultés potentielles de l'application. Dans la continuité, l'environnement de développement Open Source a été élaboré et, une seconde phase de programmation, davantage en adéquation avec l'approche Open Source, s'en est suivie. En dernier lieu, les documents concernant le développement de l'application ont été rédigés et l'application déployée.

S'investir dans ce projet a permis de réaliser la complexité d'un développement en lien avec plusieurs services possédant, chacun, leurs propres standards. L'impact des décisions stratégiques prises par l'un ou l'autre est accentué dans le cadre d'un projet d'une portée aussi large qu'UCL-Campus. Ces faits ont eu plusieurs implications et ont amené des prises de conscience au niveau de la manière de travailler et ont orienté des décisions. Ainsi, les factorisations de code ont été régulières pour éviter le code spaghetti et conserver un code modulaire. Travailler en Open Source a été bénéfique pour le projet, car cela exige une rigueur dans l'écriture du code. De plus, force a été de constater qu'il est préférable, dans le cadre d'un développement de longue durée, de faire le choix d'une technologie stable et mature pour éviter de subir de nombreux changements qui, en provoquant des modifications structurelles du projet, impactent négativement la productivité.

L'apport de ce travail ne se limite pas au développement d'une application pour l'ensemble des étudiants de l'UCL, il a également établi un cahier des charges posant les bases de cette application ainsi qu'une description du paysage informatique de l'Université catholique de Louvain. De même, un environnement propice au développement Open Source a été constitué. Les outils de cet environnement sont : Github, Zenhub, Gitter et un Wiki pour informer et organiser l'avancement du projet tandis que Travis, Karma, Protractor et Jasmine sont utilisés pour tester l'application. Enfin, les contacts établis avec différents services pour la réalisation et mise en place de l'application UCLCampus ont été un incitant, pour certains, à développer et standardiser leurs informations disponibles grâce des interfaces. Ainsi, le service informatique des bibliothèques a décidé, à la suite de nos rencontres, de mettre en place un API sur WSO2 concernant les horaires et informations des bibliothèques. Les échanges avec le service responsable du portail ont accéléré le déploiement des flux RSS des actualités universitaires. De même, en testant l'application Resto4U, nous avons pu détecter et signaler la présence d'erreurs dans le code de l'application Resto4U lors du lancement externe de l'application. La société Apptree a découvert et résolu certains problèmes d'affichage sur les smartphones avec un écran de taille 5 pouces suite à nos échanges de mails.

Il reste cependant de nombreux challenges à relever. L'ensemble des fonctionnalités présentées dans le cahier des charges n'ont, malheureusement, pas toutes pu être développées suite à un manque de moyen technique. L'API manager WSO2 en est encore à ses balbutiements, de nombreux services de l'UCL n'y sont pas encore disponibles. Ce fait implique une limitation dans les fonctionnalités développées ou l'utilisation d'alternatives temporaires pour offrir un minimum de fonctionnalités aux utilisateurs de l'application. Dans le futur, l'API manager sera le moyen privilégié d'interactions entre l'application mobile UCLCampus et les fonctionnalités offertes par les services de l'université. De même, le système de sécurité utilisé par l'Université catholique de Louvain doit évoluer pour supporter, dans l'avenir, les interactions avec les applications mobiles externes.

L'attention tardive portée à la mise en place d'un environnement propice à la création d'une communauté autour du projet UCLCampus a pour corrolaire l'absence d'une véritable communauté étudiante portant le projet UCLCampus. En effet, plus que la réalisation annuelle d'un mémoire, seule une communauté active permettra de pérenniser et de faire vivre le projet. Les mémorants auront toujours un rôle à jouer en tant que leader et architecte du projet mais, ils devront s'enrichir des idées et du code réalisé par la communauté. A l'avenir, une plus grande énergie et attention devraient être portées aux outils de communication et de partage avec la communauté étudiante.

Au terme de ce travail, une première version de l'application UCLCampus est déployée et disponible sur les stores d'applications. Dans le futur, de nouvelles perspectives et l'ajout de fonctionnalités sont à envisager suite à l'évolution, la généralisation et standardisation des services informatiques de l'UCL qui passeront, dorénavant, par l'ESB WS02. Les priorités devraient être d'intégrer l'application Guindaille 2.0. et la mise en place d'un système d'authentification reconnu par les services de l'UCL pour autoriser un accès sécurisé à ceux-ci. Ce dernier élément permettrait l'ouverture de nouvelles perspectives de fonctionnalités particulièrement intéressantes comme l'accès aux données du système OSIS permettant la récupération des informations du bureau virtuel (liste officielle des cours et des résultats des examens) ou l'accès au répertoire des membres de l'université.

Bibliographie

- [1] Dominique Opfergelt and Yves Deville. *Louvain 2020 Projet stratégique pour l'UCL, L'université numérique*. Université catholique de Louvain, 2015.
- [2] Inc Free Software Foundation. What is free software ?, Avril 2017. <https://www.gnu.org/philosophy/free-sw.en.html> (Dernière consultation : 11/08/2017).
- [3] Open Source Initiative. The open source definition, Mars 2007. <https://opensource.org/osd> (Dernière consultation : 11/08/2017).
- [4] Inc. Black Duck Software. Top open source licenses, 2017. <https://www.blackducksoftware.com/top-open-source-licenses> (Dernière consultation : 11/08/2017).
- [5] Inc Free Software Foundation. Licences commentées, Février 2017. <https://www.gnu.org/licenses/license-list.html#OriginalBSD> (Dernière consultation : 11/08/2017).
- [6] Inc Free Software Foundation. Gnu general public license, Juin 2009. <https://www.gnu.org/licenses/gpl-3.0.fr.html> (Dernière consultation : 11/08/2017).
- [7] Smile Open Source Solutions. *Politique Open Source, Pourquoi les DSI doivent définir une politique Open Source et comment appréhender la question*. Smile - Open Source Solutions, 2010.
- [8] Smile Open Source Solutions. *Comprendre l'Open Souce et les logiciels libres*. Smile - Open Source Solutions, 2014.
- [9] Patrice Bertrand. L'open source dans sa logique mutualiste, un vecteur de compétitivité, Octobre 2009. <http://tribunes-libres.smile.fr/L-open-source-dans-sa-logique-mutualiste-un-vecteur-de-competitivite> (Dernière consultation : 11/08/2017).
- [10] Smile Open Source Solutions. *Point de vue sur l'Open Source, Analyse et décryptage*. Smile - Open Source Solutions, 2013.
- [11] Gartner. Global mobile os market share in sales to end users from 1st quarter 2009 to 1st quarter 2017, 2017. <https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/> (Dernière consultation : 11/08/2017).
- [12] Appcelerator Inc. The titanium command line (cli) project, 2012-2013. <https://github.com/appcelerator/titanium> (Dernière consultation : 11/08/2017).
- [13] W3C. Html5, 2014. <https://www.w3.org/TR/html5/> (Dernière consultation : 11/08/2017).
- [14] The Apache Software Foundation. Apache cordova, 2017. <https://cordova.apache.org> (Dernière consultation : 11/08/2017).
- [15] Drifty Co. Ionic, 2017. <https://ionicframework.com> (Dernière consultation : 11/08/2017).
- [16] ionic team. Ionic github, 2017. <https://github.com/ionic-team/ionic> (Dernière consultation : 11/08/2017).

- [17] Adam Bradley. Where does the ionic framework fit in?, Octobre 2013. http://blog.ionic.io/where-does-the-ionic-framework-fit-in/?_ga=2.168979684.414987698.1499183802-470528657.1497369702 (Dernière consultation : 11/08/2017).
- [18] Apache Cordova. Cordova overview, architecture, Avril 2016. <https://cordova.apache.org/docs/en/latest/guide/overview/> (Dernière consultation : 11/08/2017).
- [19] Google. Angular, 2017. <https://angular.io> (Dernière consultation : 11/08/2017).
- [20] Angular. Angular github, 2017. <https://github.com/angular/angular> (Dernière consultation : 11/08/2017).
- [21] Microsoft. Typescript javascript that scales, 2017. <http://www.typescriptlang.org> (Dernière consultation : 11/08/2017).
- [22] Microsoft. Typescript is a superset of javascript that compiles to clean javascript output., 2017. <https://github.com/Microsoft/TypeScript> (Dernière consultation : 11/08/2017).
- [23] W3C. W3c dom4, Novembre 2015. <https://www.w3.org/TR/2015/REC-dom-20151119/> (Dernière consultation : 11/08/2017).
- [24] W3C. W3c mission, 2016. <https://www.w3.org/Consortium/mission> (Dernière consultation : 11/08/2017).
- [25] Angular. Architecture overview, 2017. <https://angular.io/guide/architecture> (Dernière consultation : 11/08/2017).
- [26] Sapan Diwakar. Titanium vs phonegap vs native application development, Juin 2012. <http://www.sapandiwakar.in/api-research-study-iphone-and-android-applications/> (Dernière consultation : 11/08/2017).
- [27] Ionic Team. Ionic services, Juin 2017. <https://docs.ionic.io/services/> (Dernière consultation : 11/08/2017).
- [28] Service générale du système d'information (SGSI). *Présentation Infrastructure d'intégration, Entreprise Service Bus, WSO2 DSS/ESB/AM*. Université catholique de Louvain, 2017.
- [29] David Chappell. *Enterprise service bus*. O'Reilly Media, Inc, 2004.
- [30] Smile Open Source Solutions. *ESB Les meilleures solutions Open Source*. Smile - Open Source Solutions, 2012.
- [31] Smile Open Source Solutions. Esb : Entrprise service bus, 2015. <http://information-systems.smile.eu/Tout-savoir-sur/Enterprise-service-bus/Esb-entreprise-service-bus#> (Dernière consultation : 11/08/2017).
- [32] Vincent Maucorps. *Gestion de contenus : Le meilleur des solutions Open Source*. Smile - Open Source Solutions, 2013.
- [33] Shibboleth. What's shibboleth?, 2017. <https://www.shibboleth.net/index/> (Dernière consultation : 16/08/2017).
- [34] Shibboleth. How shibboleth works : Basic concepts, 2017. <https://shibboleth.net/about/basic.html> (Dernière consultation : 11/08/2017).
- [35] Monier Etienne. Architecturer ses applications js à l'aide du pattern mvvm, Février 2015. <https://www.docdoku.com/blog/2015/02/17/architecturer-ses-applications-js-pattern-mvvm/> (Dernière consultation : 11/08/2017).
- [36] Amy Conchie. Hide api keys, Mars 2016. <https://gist.github.com/derzorngottes/3b57edc1f996dddcab25> (Dernière consultation : 16/08/2017).
- [37] Friedel Ziegelmayer. Karma how it works, Février 2016. <https://karma-runner.github.io/1.0/intro/how-it-works.html> (Dernière consultation : 11/08/2017).
- [38] Vojtěch Jína. Javascript test runner. *examensarb., Czech Technical University in Prague*, 2013.

- [39] Crypho AS. Secure storage : Platform details, May 2017. <https://github.com/Crypho/cordova-plugin-secure-storage/blob/master/README.md#platform-details> (Dernière consultation : 16/08/2017).
- [40] Mozilla Developer Network. Promise, Juillet 2017. https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Promise (Dernière consultation : 16/08/2017).
- [41] WSO2. Wso2 api manager documentation : Quick start guide, 2017. <https://docs.wso2.com/display/AM210/Quick+Start+Guide> (Dernière consultation : 16/08/2017).
- [42] Masashi Katsumata. How different are the js apis and the mobile sdks, Avril 2017. <https://github.com/mapsplugin/cordova-plugin-googlemaps-doc/blob/master/v1.4.0/How-different-are-the-JS-APIs-and-the-mobile-SDKs/README.md> (Dernière consultation : 16/08/2017).
- [43] Google. Google maps javascript api limites d'utilisation, Avril 2017. <https://developers.google.com/maps/documentation/javascript/usage?hl=fr> (Dernière consultation : 16/08/2017).
- [44] Bhathiya Jayasekara. Securing your web service with oauth2 using wso2 identity server, Février 2014. <http://wso2.com/library/articles/2014/02/securing-your-web-service-with-oauth2-using-wso2-identity-server-1/> (Dernière consultation : 16/08/2017).
- [45] UTT Net Group. Accéder aux données étudiantes et associatives avec l'api : Oauth : qu'est ce que c'est, 2014. <http://etuutt.readthedocs.io/fr/latest/api/introduction/oauth.html> (Dernière consultation : 16/08/2017).
- [46] Mike Hartington. Ionic and lazy loading pt 1, Avril 2017. <http://blog.ionic.io/ionic-and-lazy-loading-pt-1/> (Dernière consultation : 14/08/2017).

Annexe A

An Open Letter to Hobbyists

-2-

February 3, 1976

An Open Letter to Hobbyists

To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds \$40,000.

The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however. 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent of Altair BASIC worth less than \$2 an hour.

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write me at 1180 Alvarado SE, #114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

Bill Gates
Bill Gates
General Partner, Micro-Soft

Annexe B

Enquête

UCL Campus

Formulez desli à découvrir vos envies, besoins et souhaits pour réaliser une application utile dans votre quotidien étudiants à l'UCL. Cette application sera réalisée par des étudiants en informatique dans le cadre de leur mémoire.

Merci pour votre participation

1. Dans quelle faculté étudies-tu

2. Année d'étude

Une seule réponse possible

☐ BAC
☐ Waste

UCL Campus Etude

3. Quelles fonctionnalités/outils veux-tu avoir dans le cadre de tes études

Plusieurs réponses possible

- ☐ Accès à mes points
- ☐ Un agenda reprenant mon horaire de cours et les auditoires
- ☐ Des notifications en cas de changements d'horaire de cours ou d'auditoires
- ☐ Accès à mon module
- ☐ Accès à la messagerie universitaire
- ☐ Horaire d'ouverture et de fermeture des bibliothèques
- ☐ Horaire d'ouverture et de fermeture des secrariats
- ☐ Localisation des salles d'études et des salles informatiques
- ☐ Savoir le nombre de places disponibles dans les bibliothèques
- ☐ Canaux d'orientation en temps réel pour poser ses questions aux professeurs lors d'un cours magistral
- ☐ Forum d'étudiants permettant de centraliser les questions sur un cours
- ☐ Accès aux informations sur les services offerts par l'UCL
- ☐ Accès au répertoire UCL
- ☐ Outil de gestion d'achat et de vente de livres en seconde main
- ☐ Informations sur la vie universitaire

UCLCampus Vie sur le campus

b. Quelles fonctionnalités/outils veux-tu avoir dans le cadre de la Vie sur le campus

- ☐ Agenda de tous les événements, sorties, soirées, festivités
- ☐ Horaire du sport CSE + réservation de terrains
- ☐ Réseau social spécifique à l'UCL (Communauté d'étudiants par exemple, sport mais aussi ...)
- ☐ Création personnelle d'événements pour une communauté, un groupe d'amis
- ☐ Savoir où il y a des places de parking dispo
- ☐ Partage de bons plans
- ☐ Description et géolocalisation des différents KAPs
- ☐ Description et géolocalisation des différents cercles
- ☐ Se géolocaliser sur une map
- ☐ Menu, horaire et prix des resto U
- ☐ Inviter des personnes à discuter, partager sa table lors d'un temps de midi

6. As-tu d'autres idées/remarques

Annexe C

Personnes ressources

Cette annexe reprend sous forme de tableau les coordonnées des personnes par service contactées dans le cadre du mémoire.

Nom	Adresse mail	Contacté au sujet de
Service des sports		
Liliane Anni	liliane.anni@uclouvain.be	Informations concernant le service des sports
Nicolas Gilson	nicolas.gilson@uclouvain.be	
Restaurant Universitaire		
Olivier Francq	olivier.francq@uclouvain.be	Informations sur les restaurants universitaires
Jerome Migeotte	migeotte.jerome@uclouvain.be	Informations Informatique des restaurants universitaires
Vincent Quoilin	vincent.quoilin@apptree.be	Contact pour la société responsable de l'application Resto4U
Garry Van Loo	garry.vanloo@uclouvain.be	Directeur
Univers Santé		
De Duve Martin	martin.deduve@uclouvain.be	l'application Guindaille 2.0 pour la fonctionnalité événements sur les campus
Bibliothèques de l'Université catholique de Louvain		
Charles-Henri Nyns	harles-henri.nyns@uclouvain.be	Informations concernant les bibliothèques universitaires
Laurent Dubois	laurent.dubois@uclouvain.be	API WSO2 pour les bibliothèques
Service générale du système d'information		
Francis Somers	francis.somers@uclouvain.be	ESB WSO2
Fabrice Degraux	fabrice.degraux@uclouvain.be	Horaire ADE
Frédéric Fervaille	frederic.fervaille@uclouvain.be	Service Web Moodle
Mike De Man	mike.deman@uclouvain.be	Flux RSS actualités et portail UCL
Alain Ninane	alain.ninane@uclouvain.be	Sécurité UCL
Infrastructures de production du système d'information		
Olivier Delcourt	olivier.delcourt@uclouvain.be	Shibboleth et sécurité UCL
Exploitation des produits et services du système d'information		
Hildeberto Mendonça Filho	hildeberto.mendonca@uclouvain.be	Système OSIS
Support aux utilisateurs du système d'information		
Frédéric Malengreau	frederic.malengreau@uclouvain.be	Service Desk UCL
Jeffrey Tillier	jeffrey.tillier@uclouvain.be	Service Desk UCL
Chercheurs		
Jorge Luis Perez Medina	jorge.perezmedina@uclouvain.be	Application UCLCampus 2015-16
Mathieu Zen	mathieu.zen@uclouvain.be	Application LLNMaps
Kots à Projets		
Carpestudentem	carpestudentem@kapuclouvain.com	site uclouvainfo.be

TABLE C.1 – Personnes contactées dans le cadre du mémoire

Annexe D

Illustrations des requêtes

D.1 Requête ADE

D.1.1 ouverture de la session

Requête :

```
http://horaire.uclouvain.be/jsp/webapi?function=connect&login=etudiant&password=student
```

Réponse :

```
<?xml version="1.0" encoding="UTF-8"?>
<session id="15dalcd6ad"/>
```

D.1.2 Récupération des différents projets d'horaire disponibles

Requête :

```
http://horaire.uclouvain.be/jsp/webapi?sessionId=15dalcd6ad&function=getProjects&detail=2
```

Réponse :

```
<?xml version="1.0" encoding="UTF-8"?>
<projects>
  <project id="21" name="2016-2017" uid="1454322060161"/>
  <project id="1" name="2017-2018" uid="1484208489430"/>
</projects>
```

D.1.3 Sélectionner un projet

Requête :

```
http://horaire.uclouvain.be/jsp/webapi?sessionId=15dalcd6ad&function=setProject&projectId=1
```

Réponse :

```
<?xml version="1.0" encoding="UTF-8"?>
<setProject sessionId="15dalcd6ad" projectId="1"/>
```

D.1.4 Récupération de l'identifiant ADE du cours LFSAB1101

Requête :

```
http://horaire.uclouvain.be/jsp/webapi?sessionId=15dalcd6ad&function=getResources&code=LFSAB1101
```

Réponse :

```
<?xml version="1.0" encoding="UTF-8"?>
<resources>
  <resource id="11832"/>
</resources>
```

D.1.5 Récupération des activités du cours LFSAB1101

Requête :

<http://horaire.uclouvain.be/jsp/webapi?sessionId=15da1cdd6ad&function=getActivities&resources=11832&detail=17>

Réponse :

```
<?xml version="1.0" encoding="UTF-8"?>
<activities>
  ....
  <activity id="3972" name="LFSAB1101-2" type="Cours magistral" folderId="922" url="" size="100"
    repetition="1" lastUpdate="06/09/2017 08:28" creation="03/23/2015 14:38" lastSlot="32"
    lastDay="2" lastWeek="13" firstSlot="24" firstDay="2" firstWeek="4" additionalResources
    ="0" durationInMinutes="840" nbEventsPlaced="7" nbEvents="7" duration
    ="120+120+120+120+120+120+120+120" email="" weight="0" seatsLeft="487/2 = 244" maxSeats="529"
    info="Data" codeZ="" codeY="2016" codeX="79832" timezone="Q1: 40.0 + 40.0" code="
    MATHEMATIQUES 1" color="255,255,255" isActive="true" isNotSameDay="false" isGrouped="false
    " isAligned="false" isSuccessiveDays="false" ownerId="46" owner="verheydenj">
    <events>
      <event id="34113" activityId="3972" session="1" repetition="0" name="LFSAB1101
        -2" endHour="16:00" startHour="14:00" date="25/10/2017" absoluteSlot
        ="2840" slot="24" day="2" week="6" additionalResources="0" duration="8"
        note="" color="255,255,255" isLockPosition="false" lastUpdate="06/09/2017
        08:28" creation="04/04/2017 10:22" isLockResources="false"
        isSoftKeepResources="false">
        <eventParticipants>
          <eventParticipant fromWorkflow="false" nodeId="6655" nodeOrId
            ="-1" quantity="1" category="trainee" name="FSA11B" id
            ="15354"/>
          <eventParticipant fromWorkflow="false" nodeId="37426" nodeOrId
            ="-1" quantity="1" category="category5" name="LFSAB1101"
            id="11832"/>
          <eventParticipant fromWorkflow="false" nodeId="1623" nodeOrId
            ="37870" quantity="1" category="classroom" name="SUD 08"
            id="56"/>
          <eventParticipant fromWorkflow="false" nodeId="37772" nodeOrId
            ="37762" quantity="1" category="instructor" name="Ben-
            Naoum Abdou Kouider" id="96"/>
          <eventParticipant fromWorkflow="false" nodeId="115457" nodeOrId
            ="37762" quantity="2" category="instructor" name="Wertz
            Vincent" id="1085"/>
          <eventParticipant fromWorkflow="false" nodeId="115485" nodeOrId
            ="37762" quantity="1" category="instructor" name="
            Verleysen Michel" id="1066"/>
        </eventParticipants>
        <additional/>
      </event>
      ....
    </events>
    <resources activityId="3972">
      <and load="1.0" quantity="1" nodeId="6655" id="15354" name="FSA11B" path="EPL.
        EPL Cycle 1.EPL-niveau 11.fsa11ba." category="trainee" isGroup="true"/>
      <and load="1.0" quantity="1" nodeId="37426" id="11832" name="LFSAB1101" path="
        SST / EPL.DEP_FSA." category="category5" isGroup="false"/>
      <or isContinuous="false" category="instructor" quantity="4" id="37762">
        <and load="1.0" quantity="1" nodeId="37772" id="96" name="Ben-Naoum
          Abdou Kouider" path="Personnel UCL.B." category="instructor"
          isGroup="false"/>
        <and load="1.0" quantity="1" nodeId="37800" id="1333" name="Pereira
          Olivier" path="Personnel UCL.P." category="instructor" isGroup="
          false"/>
        <and load="1.0" quantity="1" nodeId="115485" id="1066" name="Verleysen
          Michel" path="Personnel UCL.V." category="instructor" isGroup="
          false"/>
        <and load="1.0" quantity="1" nodeId="115457" id="1085" name="Wertz
          Vincent" path="Personnel UCL.W." category="instructor" isGroup="
          false"/>
      </or>
      <or isContinuous="false" category="classroom" quantity="1" id="37870">
        <and load="1.0" quantity="1" nodeId="1623" id="56" name="SUD 08" path="
          Zone Sciences et technologies.Auditoires Croix du Sud (SUD).Croix
          du Sud B (08-09-18-19)." category="classroom" isGroup="false"/>
      </or>
    </resources>
    <rights othersRights="read" groupRights="rw" userRights="none" group="root" user="root"
      profileName="W_ADE / U_Appariteurs / R_OTHER" profileId="81"/>
  </activity>
  ....
</activities>
```

D.2 Requêtes des flux RSS

D.2.1 Récupération du flux RSS des actualités UCL

Requête :

```
https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20feednormalizer%20where%20url%3D%27https%3A%2F%2Fuclovain.be%2Ffr%2Factualites%2Fp1%2Frss%27&format=json
```

Réponse (afin de ne pas surcharger cette annexe, la présentation de la réponse ci-après est limitée à deux actualités) :

```
{
  "query": {
    "count": 1,
    "created": "2017-07-31T13:05:14Z",
    "lang": "en-US",
    "diagnostics": {
      ...
    },
  },
  "meta": {
    "url": {
      "id": "https://uclovain.be/fr/actualites/p1/rss",
      "status": "200",
      "headers": {
        "header": [
          {
            ...
          }
        ]
      }
    }
  },
  "results": {
    "rss": {
      "version": "2.0",
      "atom": "http://www.w3.org/2005/Atom",
      "channel": {
        "link": [
          ...
        ],
        "title": "Actualités du site p1",
        "description": "List of news of p1 site",
        "language": "fr-be",
        "copyright": "Copyright (C) 2017 https://uclovain.be",
        "item": [
          {
            "title": "Inscrivez-vous avant vos vacances !",
            "description": "Quel que soit votre choix d'études, votre inscription à l'UCL commence toujours par une procédure en ligne.\n\nN'attendez pas le 30 septembre, date de clôture des inscriptions pour effectuer cette démarche !\n\nVoici un aperçu de la procédure : \n\nEtape 1\n\nRendez-vous à ...",
            "link": "https://uclovain.be/fr/etudier/actualites/inscrivez-vous-avant-vos-vacances.html",
            "pubDate": "Thu, 27 Jul 2017 10:33:20 +0200",
            "guid": "https://uclovain.be/fr/etudier/actualites/inscrivez-vous-avant-vos-vacances.html",
            "enclosure": {
              "length": "8335372",
              "type": "image/jpeg",
              "url": "https://cdn.uclovain.be/styles/rectangle_vertical/groups/cms-editors-p1/promotion/inscription.jpg?itok=OSmgBkjA"
            }
          },
          {
            "title": "87 étudiants participent cet été au programme...",
            "description": "Cet été, pour la 5ème année consécutive, 87 étudiants de l'UCL (ingénieurs civils, bioingénieurs, vétérinaires, et architectes partent en Afrique, en Amérique Latine et en Asie, ...",
            "link": "https://uclovain.be/fr/etudier/actualites/87-etudiants-participent-cet-ete-au-programme-ingenieursud.html",
            "pubDate": "Tue, 25 Jul 2017 14:58:52 +0200",
            "guid": "https://uclovain.be/fr/etudier/actualites/87-etudiants-participent-cet-ete-au-programme-ingenieursud.html",
            "enclosure": {
              "length": "1936751",
              "type": "image/jpeg",
              "url": "https://cdn.uclovain.be/styles/rectangle_vertical/groups/cms-editors-p1/Ing%C3%A9nieux%20sud%20groupe%2013%20Cambodge.JPG?itok=XnTNBzr3"
            }
          }
        ]
      }
    }
  }
}
```

```
}
}
}
}
}
}
}
```

D.2.2 Récupération du flux RSS des évènements

Requête :

```
https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20feednormalizer%20where%20url%20%3D%20%27http%3A%2F%2Flouvainfo.be%2Fevenements%2Ffeed%2Fcalendar%2F%27&format=json
```

Réponse (afin de ne pas surcharger cette annexe, la présentation de la réponse ci-après est limitée à deux évènements) :

```
{
  "query": {
    "count": 1,
    "created": "2017-08-13T09:29:27Z",
    "lang": "en-US",
    "meta": {
      "url": {
        "id": "http://louvainfo.be/evenements/feed/calendar/",
        "status": "200",
        "headers": {
          "header": [
            {
              ...
            }
          ]
        }
      }
    }
  },
  "results": {
    "rss": {
      "version": "2.0",
      "atom": "http://www.w3.org/2005/Atom",
      "channel": {
        "title": null,
        "link": [
          ...
        ],
        "description": null,
        "language": "fr-FR",
        "lastBuildDate": "Sun, 13 Aug 2017 09:29:15 +0000",
        "item": [
          {
            "title": "Four à pain",
            "link": "http://louvainfo.be/evenements/1906/",
            "description": "<p>Portes ouvertes, venez pr  parer avec nous pizzas, pains, fougasses, .... dans une ambiance conviviale</p>",
            "guid": "event-3774",
            "location": "3 scav  e du Bi  reau",
            "date_begin": "13/08/2017 10:00",
            "date_end": "13/08/2017 12:30",
            "photo": "http://louvainfo.be/media/events/14_1_NgEJljk.jpg",
            "category": "Animation",
            "organisateur": "association",
            "prix": "0"
          },
          {
            "title": "Salon du testament philanthropique",
            "link": "http://louvainfo.be/evenements/2179/",
            "description": "<p>Bonjour,</p>\n\n<p>Nous allons organiser au Foyer du Lac de l  #39;Aula Magna   aggrave; Louvain-la-Neuve un   quot;salon du testament pour les bonnes causes  quot;; dont voici un programme succinct : accueil, animation et pr  sentation par Jacques Mercier, stands occup  s par nos bonnes causes, ...",
            "guid": "event-4466",
            "location": "AULA MAGNA, Foyer du Lac",
            "date_begin": "14/09/2017 00:00",
            "date_end": "14/09/2017 23:59",
            "photo": "http://louvainfo.be/static/img/events/sensibilisation.jpg",

```


$$\left. \begin{array}{l} \{ \\ \{ \\ \{ \\ \{ \\ \{ \end{array} \right\}]$$

{

{

```

        "endHour": "18:45"
      },
      {
        "day": "Mardi",
        "startHour": "09:00",
        "endHour": "18:45"
      },
      {
        ...
      }
    ],
    "openingExaminationHours": [
      {
        "day": "Lundi",
        "startHour": "10:00",
        "endHour": "17:45"
      },
      {
        "day": "Mardi",
        "startHour": "09:00",
        "endHour": "17:45"
      },
      {
        ...
      }
    ],
    "openingSummerHours": [
      {
        "day": "Lundi",
        "startHour": "10:00",
        "endHour": "16:45"
      },
      {
        "day": "Mardi",
        "startHour": "09:00",
        "endHour": "16:40"
      },
      {
        ...
      }
    ],
    "openingHoursNote": " L'horaire d'été s'étale de la fin de la session
d'examens de juin à la rentrée académique (mi-septembre).",
    "closedDates": [
      {
        "from": "Mardi, 27 septembre 2016",
        "description": "Fête de la Fédération Wallonie-Bruxelles"
      },
      {
        "from": "Mardi, 1 novembre 2016",
        "description": "Toussaint"
      },
      {
        ...
      }
    ]
  }
}
}

```

Annexe E

Guide de développement

Ce guide a pour but de permettre aux développeurs intéressés par le projet de participer au développement de celui-ci. Cette annexe débute par une présentation des étapes nécessaires à l'installation et au lancement de l'application et des gestionnaires de tests en local. Ensuite, il y a une présentation de la convention de codage et de structure décidée au cours du développement de l'application.

E.1 Installation de l'application

Cette section présente les étapes nécessaires à l'installation et la mise en place du projet sur une machine locale.

E.1.1 Prérequis

Le téléchargement et l'installation de Nodejs est nécessaire au fonctionnement du projet. Nodejs est téléchargeable depuis l'adresse <https://nodejs.org/en/download/current/>. Ceci installera node et npm sur la machine. Utiliser les commandes suivantes dans un terminal afin de vérifier que la version installée soit conforme aux exigences :

```
$ node -v
```

Node doit être à une version supérieure ou égale à 6.0.0.

```
$ npm -v
```

Npm doit être à une version supérieure ou égale à 3.0.0.

Enfin, pour iOS, XCode doit être mis à jour à la version 8.0 ou plus.

Afin d'utiliser le plugin Google Maps présent dans ce projet il est nécessaire d'obtenir des clés API. Voici les étapes à suivre :

1. Rendez-vous sur la Google API Console via l'adresse : https://console.developers.google.com/flows/enableapi?apiid=maps_backend,geocoding_backend,directions_backend,distance_matrix_backend,elevation_backend,places_backend&reusekey=true
2. Sur cette page, créez ou sélectionnez un projet
3. Cliquez sur continuer pour activer l'API
4. Une fois la page Identifiants atteinte, sélectionnez l'API Javascript. Cliquez sur le bouton "De quels identifiants ai-je besoin ?" afin de créer la clé relative à l'API sélectionnée.
5. Cliquez sur la "Bibliothèque dans le panneau latéral gauche de la page

6. Sélectionnez l'API Google Maps Android puis cliquez sur le bouton d'activation du service. Répétez cette étape pour l'API Google Maps iOS
7. Cliquez sur "Identifiants" dans le panneau latéral gauche de la page et créez deux API supplémentaires.
8. Cliquez sur les clés générées et configurez les restrictions en fonction de l'usage de la clé (Android, iOS, ...)

E.1.2 Installation et lancement

La première étape de l'installation est le clonage du projet git à l'adresse <https://github.com/jerlemaire/UCLCampus>.

Ensuite, il est nécessaire d'installer Ionic et cordova grâce aux commandes :

```
$ npm install -g ionic cordova
```

Si Ionic et cordova sont déjà installés sur votre machine mais qu'une mise à jour est nécessaire :

```
$ npm uninstall -g ionic cordova
$ npm install -g ionic cordova
```

Installez node_modules :

```
$ npm install
```

Installez le plugin Google Maps en remplaçant "YOUR_ANDROID_API_KEY_IS_HERE" et "YOUR_IOS_API_KEY_IS_HERE" par vos clés API mobiles obtenues précédemment grâce aux commandes suivantes :

```
$ ionic cordova plugin add
https://github.com/mapsplugin/cordova-plugin-googlemaps
--variable API_KEY_FOR_ANDROID="YOUR_ANDROID_API_KEY_IS_HERE"
--variable API_KEY_FOR_IOS="YOUR_IOS_API_KEY_IS_HERE"
$ npm install --save @ionic-native/google-maps
```

Déplacez-vous dans le dossier UCLCampus/src/app/ et créez un fichier "variables-config.ts". Copiez ensuite le bloc de code suivant dans ce fichier :

```
export var jsApiKey:string = "YOUR_API_KEY";
```

Enfin, modifiez "YOUR_API_KEY" par la valeur de la clé API Javascript obtenue précédemment.

Pour lancer l'application, placez-vous dans la racine du projet depuis votre terminal et exécutez les commandes suivantes :

— Pour navigateur

```
$ ionic serve
```

— Pour Android

```
$ ionic cordova platform add android
$ ionic cordova run android
```

— Pour iOS

```
$ ionic cordova platform add ios
$ ionic cordova build ios
```

Pour utiliser les questionnaires de tests, exécutez les commandes :

— Pour Karma (Unit Tests)

```
$ npm test
```

— Pour Protractor (E2E Tests)

```
$ ionic serve  
$ npm run e2e
```

E.2 Convention de l'application

Cette section définit les règles à appliquer dans le développement du code source ainsi que la structure du projet. Le but étant d'uniformiser les pratiques de développement des différents contributeurs du projet Open Source afin d'en simplifier la lecture et la maintenance.

E.2.1 Convention de codage

Ce projet pratique le camel case. Cette pratique consiste à écrire un ensemble de mots en les liant sans espace ni ponctuation et, en mettant en majuscule la première lettre de chaque mot excepté le premier. Voici les règles de codage liées au projet UCLCampus :

- Les identificateurs des variables doivent être écrits en camel case en commençant par une minuscule (ex : myNewVariable)
- Les identificateurs des constantes doivent être écrits en majuscules, si l'identificateur est composé de plusieurs mots, ceux-ci doivent être séparés par un tiret bas (ex : MY_CONSTANT)
- Les identificateurs des classes doivent être écrits en camel case en commençant par une majuscule (ex : MyClass)
- Les identificateurs des méthodes doivent être écrits en camel case commençant par une minuscule (ex : myVariable)
- Les identificateurs des variables et méthodes privées doivent être précédés d'un tiret bas (ex : _myPrivateVariable)
- Les modificateurs d'accès des variables et méthodes privées doivent être écrits (ex : private _myPrivateVariable)
- Les modificateurs d'accès des variables et méthodes publiques ne doivent pas être écrits (ex : myNewVariable)
- Les types des variables de classe doivent être précisés dans leur définition
- Les identificateurs de classes objets doivent terminer par Item (ex : EventItem)
- Les identificateurs de pages doivent terminer par Page (ex : EventsPage)
- Les identificateurs de providers doivent terminer par Service (ex : EventsService)
- Les blocs de code if et else doivent être entre accolades
- La lisibilité du code prime sur la concision du code

E.2.2 Structure du projet

La structure des sources du projet suit l'architecture de base d'un projet Ionic avec quelques modifications. Les sources d'un projet Ionic de base sont divisées en cinq dossiers : app, assets, pages, providers et theme. Chaque page d'un projet Ionic possède son propre dossier dans le dossier "pages". Dans ce dossier sont présents les fichiers .html, .scss, .module.ts et .ts associés à la page. Chaque service d'un projet Ionic possède son fichier .ts dans le dossier "provider". Le dossier "assets" d'un projet Ionic contient les ressources numériques réutilisables dans l'application.

Dans le projet UCLCampus, la même structure est suivie mais les pages associées à la même fonctionnalité sont regroupées dans un dossier représentant la fonctionnalité. Par exemple, les pages EventsFilterPage, EventsDetailsPage et EventsPage sont, de base, respectivement contenues dans les dossiers "events-filter", "events-details" et "events" du dossier "pages". Le projet UCLCampus regroupe ces dossiers dans le dossier "events" représentant la fonctionnalité associée

à ces pages. Ceci permet de faciliter la navigation dans le projet et de ne pas avoir un dossier "pages" surchargé de sous-dossiers.

Le même principe est appliqué au dossier "providers". Si une fonctionnalité de l'application utilise différents services, les fichiers de ces services sont regroupés dans un dossier associé à la fonctionnalité. Par exemple, les services AdeService, CourseService et StudiesService sont, de base, respectivement implémentés dans les fichiers "ade-service.ts", "course-service.ts" et "studies-service.ts" du dossier "providers". UCLCampus regroupe ces fichiers dans un dossier "studies-services" dans le dossier "providers". Ce regroupement permet de faciliter la navigation dans le projet et la lisibilité du dossier "providers".

Enfin, le projet UCLCampus utilise des classes objets (par exemple EventItem) à travers ses pages. Chacune de ces classes doit être implémentée dans un fichier .ts et placée dans le dossier "entity" du dossier "app".

La figure E.2.2 présente la structure des dossiers de l'application.

FIGURE E.1 – Structure des dossiers de l'application



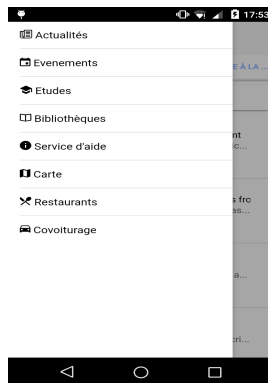
Annexe F

Guide utilisateur

Ce guide a pour but de vous faire découvrir l'ensemble des fonctionnalités offertes par l'application UCLCampus. L'application UCLCampus est une application mobile permettant une gestion et une organisation intuitive et efficace de la vie d'un étudiant au sein d'un campus universitaire de l'UCL. Une fonctionnalité de l'application est consacrée aux études. L'idée est de permettre aux étudiants d'être informés de tous les événements du campus en un seul mouvement de doigt. Ils pourront organiser et mettre à jour leur agenda électronique simplement et intuitivement. L'application permettra de centraliser l'ensemble des informations utiles à la vie sur le campus.

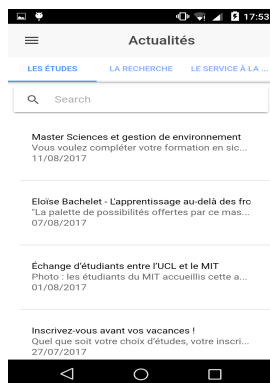
L'application UCLCampus est supportée par les smartphones Android 4.4 et plus ainsi que par IOS version 8 et plus.

F.1 Menu

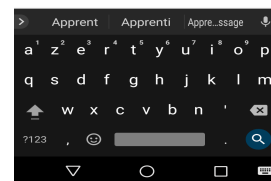


L'application est dotée d'un menu principal vous permettant d'accéder à l'ensemble des fonctionnalités de l'application. Il contient actuellement huit fonctionnalités. Dans la suite de ce manuel, nous naviguerons entre chacune d'elles afin de les découvrir.

F.2 Actualités

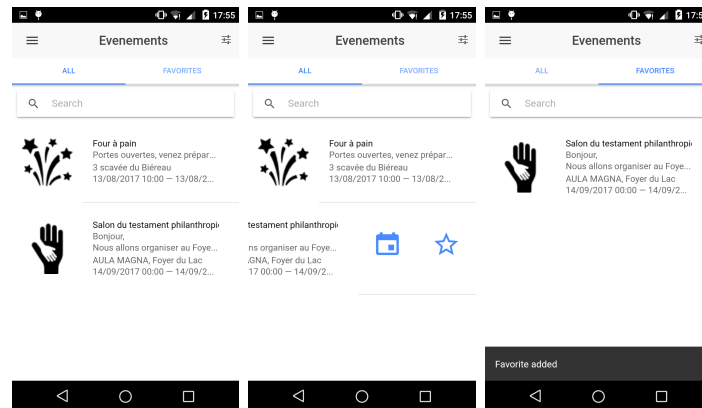


Sur la page se trouve une barre de recherche pour effectuer des recherches par mots clés. Vous encodez les mots clés à l'aide du clavier de votre smartphone. La recherche est automatique : au fur et à mesure que vous encodez les lettres du mot recherché, les actualités sont filtrées. Vous ne devez donc pas appuyer sur le moindre bouton pour activer la recherche.

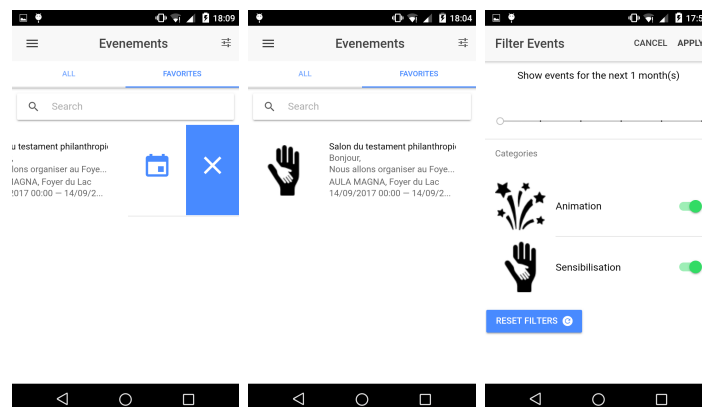


Lorsque vous appuyez sur une actualité, une nouvelle page s'ouvre. Celle-ci affiche l'actualité de manière plus détaillée. Elle se compose d'une image illustrant la nouvelle et d'un texte explicatif. Le bouton "view in browser" permet d'ouvrir un navigateur web affichant directement la page avec la nouvelle désirée. Vous pouvez évidemment appuyer sur les liens hypertextes de la page pour continuer votre navigation et votre recherche sur cette actualité. La flèche en haut à gauche de l'écran vous permet de revenir à la page principale contenant la liste des actualités. Une autre alternative pour revenir à la page précédente est d'appuyer sur le bouton retour de votre smartphone en bas à gauche de l'écran.

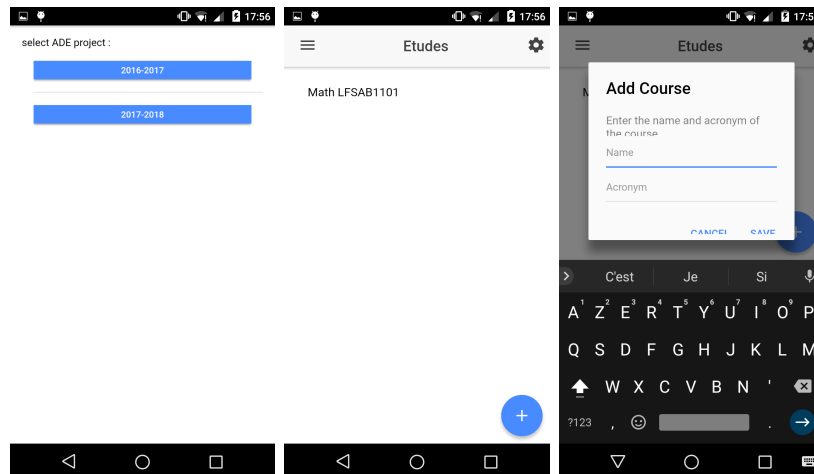
F.3 Évènements



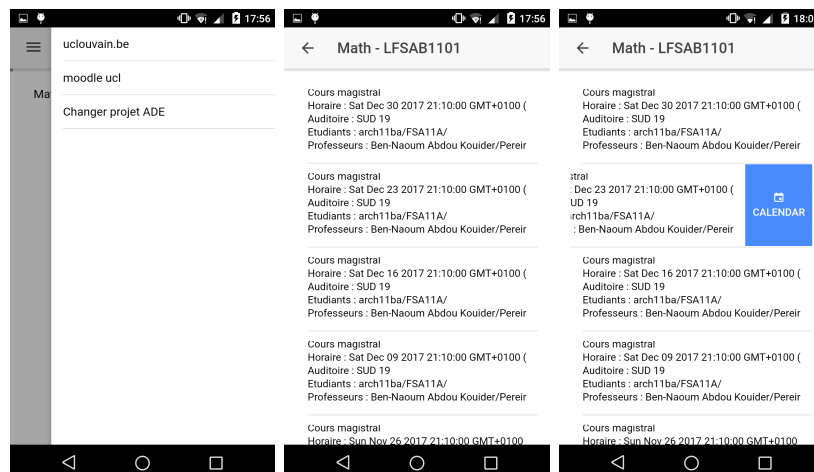
La fonctionnalité "Évènements" de l'application recense l'ensemble des événements en cours et à venir sur le campus de Louvain-La-Neuve. Ces événements sont repris du calendrier tenu par le Kot à projet CarpeStudentem sur le site louvainfo.be. Sur le premier écran (illustré en haut à gauche), se trouve la liste de l'ensemble des activités à venir dans le mois. Vous pouvez faire glisser votre doigt de droite à gauche sur une activité pour faire apparaître deux nouveaux boutons, comme l'illustre le second écran ci-dessus. Le premier bouton en partant de la gauche, avec un logo d'agenda, vous permet d'ajouter l'événement au calendrier natif de votre smartphone. Le second bouton représentant une étoile permet d'ajouter l'événement à votre liste d'événements favoris. Vous pouvez accéder à cette liste d'événements favoris en appuyant sur l'onglet "Favorites". Après l'ajout d'un événement, vous pourrez observer un écran similaire à l'écran illustré en haut à droite. Vous pouvez également faire glisser les événements présents dans la liste des événements favoris pour faire apparaître deux boutons (illustration du bas à gauche). Le premier bouton agenda permet d'ajouter l'événement à l'agenda natif de votre smartphone. En appuyant sur le second bouton, vous supprimez l'événement de la liste des événements favoris. Le bouton situé tout en haut à droite du titre de la fonctionnalité est présent sur l'ensemble des écrans de la fonctionnalité "Évènement" et vous permet de modifier les paramètres d'affichage de la liste des événements. En appuyant sur ce bouton, vous découvrirez un écran similaire à l'écran en bas à droite de la page. Il vous permet de définir la durée de la période (en mois) pour laquelle vous souhaitez afficher des événements. Vous pouvez également sélectionner ou désélectionner certaines catégories d'événements. Une fois vos paramètres encodés, il vous suffit d'appuyer sur le bouton "Apply". Vous pouvez également appuyer sur le bouton "Reset Filters" pour réinitialiser les filtres.



F.4 Etudes



La fonctionnalité "Etudes" vous permet de gérer votre horaire de cours. L'horaire de cours disponible sur l'application UCLCampus est directement lié au service ADE officiel de l'université. A l'ouverture de cette fonctionnalité, il vous est demandé de choisir le projet dont vous voulez connaître l'horaire (écran en haut à gauche). Ensuite, vous arrivez sur la page reprenant la liste des cours dont vous voulez connaître l'horaire (écran en haut au milieu). Si vous faites glissez votre doigt de droite à gauche sur un cours de la liste, vous verrez apparaître un bouton avec une icône de poubelle vous permettant de supprimer le cours de la liste. Si vous voulez ajouter un nouveau cours à votre liste, il vous suffit d'appuyer sur le bouton bleu et rond avec un plus au milieu. Une fenêtre d'alerte s'affichera (écran en haut à droite). Vous pouvez y encoder les informations concernant le nouveau cours. Ensuite, appuyez sur le bouton Save pour ajouter le cours à la liste. Le bouton en haut à droite de l'écran ayant la forme d'un engrenage vous permet d'ouvrir un menu spécifique à cette fonctionnalité. Ce menu est illustré sur l'écran en bas à gauche. Il vous permet d'être redirigé vers le site moodle ou le portail UCL. Vous pouvez également modifier le projet de cours sélectionné préalablement. En partant de la page contenant la liste des cours, si vous appuyez sur un cours, vous ouvrirez une nouvelle page contenant tous les événements (travaux pratiques, consultations du copie et cours magistraux) liés à ce cours sous forme de liste (écran du bas au milieu). Vous pouvez également faire glisser un événement de droite à gauche pour faire apparaître le bouton agenda permettant d'ajouter cet événement à votre agenda (écran du bas à droite).

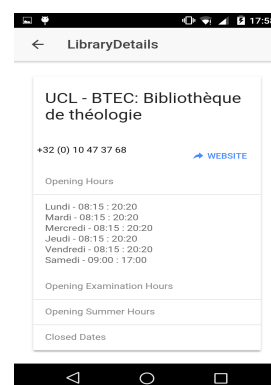


F.5 Bibliothèques

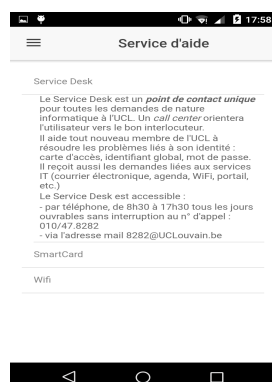


Cette fonctionnalité liste simplement l'ensemble des bibliothèques de l'Université catholique de Louvain. Il suffit de cliquer sur une des bibliothèques présentes pour avoir de plus amples informations sur celle-ci.

La page détaillée affiche différentes informations sur la bibliothèque. La première est le nom complet et l'acronyme. Ensuite vous pouvez trouver le numéro de téléphone et parfois d'autres informations pratiques. Les différents horaires de la bibliothèque sont également disponibles. Il y a trois périodes différentes d'ouverture des bibliothèques : L'horaire standard, l'horaire en période d'examen et l'horaire en période de vacances d'été. Il est également possible d'afficher les jours de fermeture des bibliothèques.



F.6 Service d'aide

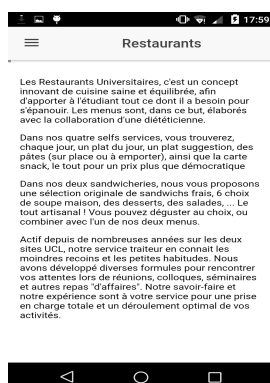
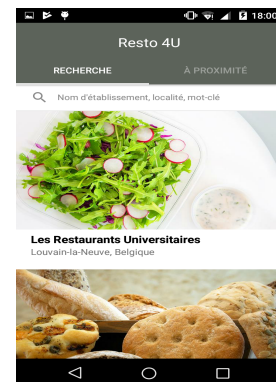


La fonctionnalité "Service d'aide" reprend l'ensemble des informations disponibles sur le portail internet de l'Université catholique de Louvain concernant le support informatique et numérique aux étudiants. Le premier sujet abordé est les informations utiles à connaître sur le service Desk et les moyens de le contacter. Le second concerne les SmartCard et, le dernier sujet explique les différentes démarches à effectuer pour se connecter au réseau internet universitaire Edurom avec ses identifiants. L'ensemble de ces sujets sont agrémentés de liens interactifs vous redirigeant vers les informations et les documents utiles pour répondre à l'ensemble de vos questions concernant le support informatique de l'université.

F.7 Restaurants



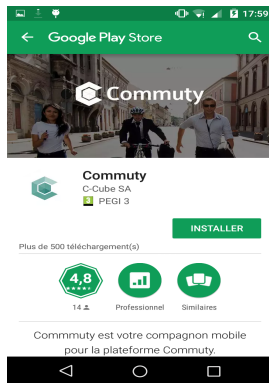
Une fois l'application téléchargée, lorsque vous appuyerez sur le bouton restaurants depuis le menu principal, l'application Resto4U sera automatiquement lancée. L'application Resto4U offre de nombreux avantages : Découvrez la carte restauration, commandez directement depuis l'application, recevez les informations pratiques et utiles, donnez votre avis grâce aux enquêtes de satisfaction.



Lorsque depuis le menu vous appuyez sur le bouton "restaurants" et si vous ne possédez pas l'application des restaurants universitaires Resto4U, vous serez automatiquement redirigé vers la page de téléchargement de l'application sur le store d'application natif de votre smartphone. Vous êtes invité à télécharger cette application pour profiter de la fonctionnalité.

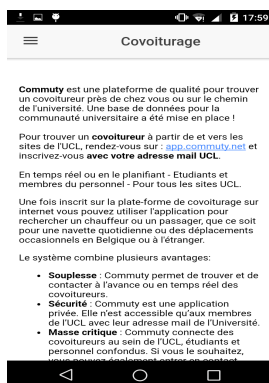
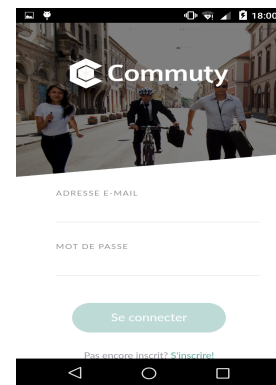
Pour revenir à l'application UCLCampus, il vous suffit d'appuyer sur le bouton retour de votre smartphone. Vous serez accueilli par une page de l'application UCLCampus contenant différentes informations concernant les restaurants universitaires et leur philosophie. Pour information, les Restaurants universitaires, c'est un concept de cuisine saine dans le respect de la pyramide alimentaire. Tous nos menus sont composés en étroite collaboration avec une diététicienne. Nous avons également participé à l'élaboration d'un projet Santé pour toute la communauté universitaire.

F.8 Covoiturage



Lorsque depuis le menu vous appuyez sur le bouton "Covoiturage" et si vous ne possédez pas l'application des Commuty, vous serez automatiquement redirigé vers la page de téléchargement de l'application sur le store d'application natif de votre smartphone. Vous êtes invité à télécharger cette application pour profiter de la fonctionnalité.

Une fois l'application téléchargée, lorsque vous appuyerez sur le bouton covoiturage depuis le menu principal, l'application Commuty sera automatiquement lancée. Sur l'application Commuty vous pouvez : Trouver des co-voyageurs pour la voiture et le vélo dans votre entreprise/université ou dans les organisations voisines. Contacter ceux-ci afin de vous organiser via un chat en temps réel. Remplir votre journal pour recevoir des récompenses. Remplir votre journal pour toucher les remboursements kilométriques vélo. Remplir votre journal pour recevoir les avantages fiscaux covoiturage. Remplir votre journal pour réserver ou partager des places sur le parking de votre entreprise.



Pour revenir à l'application UCLCampus, il vous suffit d'appuyer sur le bouton retour de votre smartphone. Vous serez accueilli par une page de l'application UCLCampus contenant différentes informations concernant le covoiturage à l'Université catholique de Louvain et le partenariat avec l'application Commuty.

