

RAPPORT DE STAGE DE 4ÈME ANNÉE PRÉSENTÉ PAR

Sacha LHOPITAL

Filière Informatique

Année Universitaire 2016-2017

DÉVELOPPEMENT WEB DANS UNE ESN AGILE

Entre mission et amélioration en interne



Tuteur entreprise : Aude NEGRE

Tuteur académique : Samir AKNINE

Janvier 2017

POLYTECH Lyon - UNIVERSITE CLAUDE BERNARD LYON 1
Domaine Scientifique de La Doua – 15, Boulevard Latarjet

69622 VILLEURBANNE CEDEX

Tél. (33) 04.72.43.12.24 - Fax. (33) 04.72.43.12.25

<http://polytech.univ-lyon1.fr/>

Remerciements

Avant toute chose, il est pour moi indispensable de remercier tous ceux qui m'ont accompagnée dans le déroulement de ce stage.

Je commencerais donc par remercier Aude NEGRE, Responsable du centre de services Apollo SSC, de m'avoir poussée à approfondir mon travail sur les différents projets : ses conseils et son aide dans la mise en place de solutions m'ont motivée à donner le meilleur de moi-même. J'ai eu beaucoup de chance de travailler sur autant de projets différents (y compris une mission réelle) et pour cela, je lui suis extrêmement reconnaissante de sa confiance à mon égard.

Je souhaite également remercier Julien ROYER, Chef de projet et Rémy VILLAIN, Architecte Technique. Leur grande disponibilité d'écoute et leur implication ont permis à ce stage de se dérouler dans les meilleures conditions possibles. Je remercie également tous les autres collaborateurs d'Apollo SSC et d'Apollo Formation, pour leur accueil convivial, leur sympathie et leur bonne humeur au quotidien.

Enfin, je remercie aussi mon tuteur académique, Samir AKNINE pour l'intérêt qu'il a porté à mon stage, mais aussi pour le suivi de celui-ci.

Sommaire

Introduction	1
1 Apollo SSC : une ESN Agile	2
1.1 Création et évolution d'Apollo	2
1.2 Activités d'Apollo SSC	2
1.3 Marché et clients	2
1.4 Équipe de développement	2
1.5 Organisation et fonctionnement	3
2 Organisation & Méthodologie d'Apollo SSC	4
2.1 Stratégie de Développement	4
2.1.1 Gestionnaire de version de fichiers	4
2.1.2 Gestionnaire de suivi d'anomalies	4
2.1.3 Microsoft Azure	5
2.2 Stratégie de Test	6
2.3 Organisation d'Apollo SSC	6
3 Cartographie des projets	8
3.1 Présentation des intranets	9
3.1.1 Architecture orienté MVC*	9
3.1.2 Intranet Apollo Formation	10
3.1.3 Intranet Apollo SSC	10
3.2 Présentation des outils clients	11
3.2.1 Etude	11
3.2.2 Rapport d'Activité	12
3.3 Plannification	12
4 Présentation détaillée des développements : RAC & Etude	13
4.1 Rapport d'Activité	13
4.1.1 Périmètre Fonctionnel Détailé	13
4.1.2 Architecture	14
4.1.3 Tâches	16
4.2 Etude	22
4.2.1 Architecture	22
4.2.2 Tâches	22
5 Interprétation des résultats	25
5.1 Discussion	25
5.2 Perspectives	26
Conclusion	27
A Etude : Fonction generateAllDatas	28

Table des figures

1.1	Organigramme Apollo	3
2.1	Interface Mantis Bug Tracker	5
2.2	Interface de Microsoft Azure	5
2.3	Interface Trello	6
2.4	Meeting hebdomadaire	7
3.1	Cartographie des projets	8
3.2	Architecture des intranets	9
3.3	Visuel de l'intranet Apollo Formation	10
3.4	Visuel de l'intranet Apollo SSC	10
3.5	Exemple d'un visuel d'Etude	11
3.6	Planning des tâches effectuées pendant le stage	12
4.1	Diagramme de Cas d'Utilisation du RAC	14
4.2	Architecture du Rapport d'Activité	15
4.3	Exemple : Pierre effectue une demande du 5/12/16 au 8/12/16 pour 2,5 Congés Payés.	16
4.4	Formulaire de réalisation d'une Autorisation - hotline V1	22
4.5	Détail d'une Autorisation Réalisée	24

Glossaire & Abbréviations

Gironnage Principe de couverture de toiture où la pose des tuiles donne une forme arrondie (pour une tourelle par exemple)

Mock : En programmation orientée objet, les mocks sont des objets simulés qui reproduisent le comportement d'objets réels de manière contrôlée. Cette entité a pour seul but de permettre de tester le comportement d'autres objets, réels, mais liés à un objet inaccessible ou non implémenté. Ce dernier est alors remplacé par un mock.

MVC (*Abbr. Model View Controller*) : Patron d'architecture logicielle qui regroupe les fonctions nécessaires à une application en trois catégories : un modèle (modèle de données) ; une vue (présentation, interface utilisateur) et un contrôleur (logique de contrôle, gestion des événements, synchronisation).

RAC (*Abbr. Rapport d'Activité*) : Projet d'intranet interne à Apollo SSC pour gérer le temps de travail des collaborateurs.

ViewModel : Classe particulière qui nous permet de définir le format des données que l'on souhaite récupérer pour être affichées côté vue. Est appelé ViewModel un schéma de données singulièrement différent des modèles proposés par la base de données.

Introduction

De nos jours, le succès et la pérennité d'une entreprise se manifestent en grande partie à sa capacité à utiliser toutes les nouvelles technologies mises à sa disposition. En particulier pour les ESN* : il ne s'agit plus seulement de proposer des outils efficaces à ses clients pour répondre à leurs besoins, il s'agit d'impliquer au maximum le client dans le processus de création pour répondre au plus près à ses besoins.

Apollo SSC se propose de gérer l'information à l'ère du XXI^e siècle avec l'utilisation des méthodes agiles. Elle développe des outils autant pour elle-même que pour ses clients en essayant de respecter aux mieux ces méthodes de développement.

C'est dans ce contexte qu'Apollo SSC a pris la décision de s'agrandir. Mon stage a ainsi eu pour objectif de participer au développement de l'ESN en participant à l'amélioration de leurs logiciels internes, mais également en participant à un projet pour un client de la société car l'expansion interne va de pair avec l'expansion externe.

Après avoir présenté Apollo Solutions Services Conseil et ses filiales, je présenterai l'organisation et les méthodologies utilisées dans l'entreprise. Puis je présenterai rapidement une description des différents projets auxquels j'ai participé. Ensuite, je détaillerai le travail réalisé sur deux projets plus particulièrement au travers des tâches réalisées. Je réaliserais enfin une mise en perspective de ce que j'ai appris chez Apollo.

Chapitre 1

Apollo SSC : une ESN Agile

1.1 Création et évolution d'Apollo

Apollo SSC a été créé en 2004 par Yann SAMAMA-TIER qui la dirige toujours actuellement. Son activité première est de proposer ses services dans le domaine informatique.

Depuis plusieurs années, l'entreprise entreprend une extension importante. Ce choix de s'agrandir a pour but de s'installer durablement sur le marché dans lequel l'entreprise a déjà posé un pied.

L'entreprise compte à l'heure actuelle une cinquantaine de salariés (dont une quarantaine d'ingénieurs informatiques).

1.2 Activités d'Apollo SSC

Apollo SSC propose des services typiques d'une ESN permettant de répondre au besoin d'externalisation des expertises de leurs clients :

Le Développement d'applications , pour permettre aux clients de rester compétitifs dans leur domaine. Pour cela, Apollo favorise les technologies et méthodes de développements émergentes comme Microsoft .Net, les technologies Javascript (AngularJS par exemple), etc.

Un Services d'assistance Technique , les clients peuvent ainsi faire appel à Apollo pour poursuivre un développement, même si ce ne sont pas eux qui ont réalisé l'existant. Cette offre donne à l'entreprise une plus grande flexibilité sur le marché.

Enfin, en plus de ces services techniques, Apollo propose à ses clients un service de maintenance.

1.3 Marché et clients

Apollo SSC met à disposition ses services pour toutes les entreprises quelle que soient leur taille et leurs objectifs.

Parmi les principaux clients on retrouve : **April** (Assurances), **Vinci autoroute**, **Imerys** (Extraction et transformation de minéraux pour l'industrie), **Unilever** (Agro-alimentaire) ou encore **Thales** (Electronique).

1.4 Équipe de développement

En ce qui concerne le niveau de qualification du personnel, les équipes d'Apollo sont très variées. Néanmoins, la plupart des développeurs possèdent un niveau ingénieur.

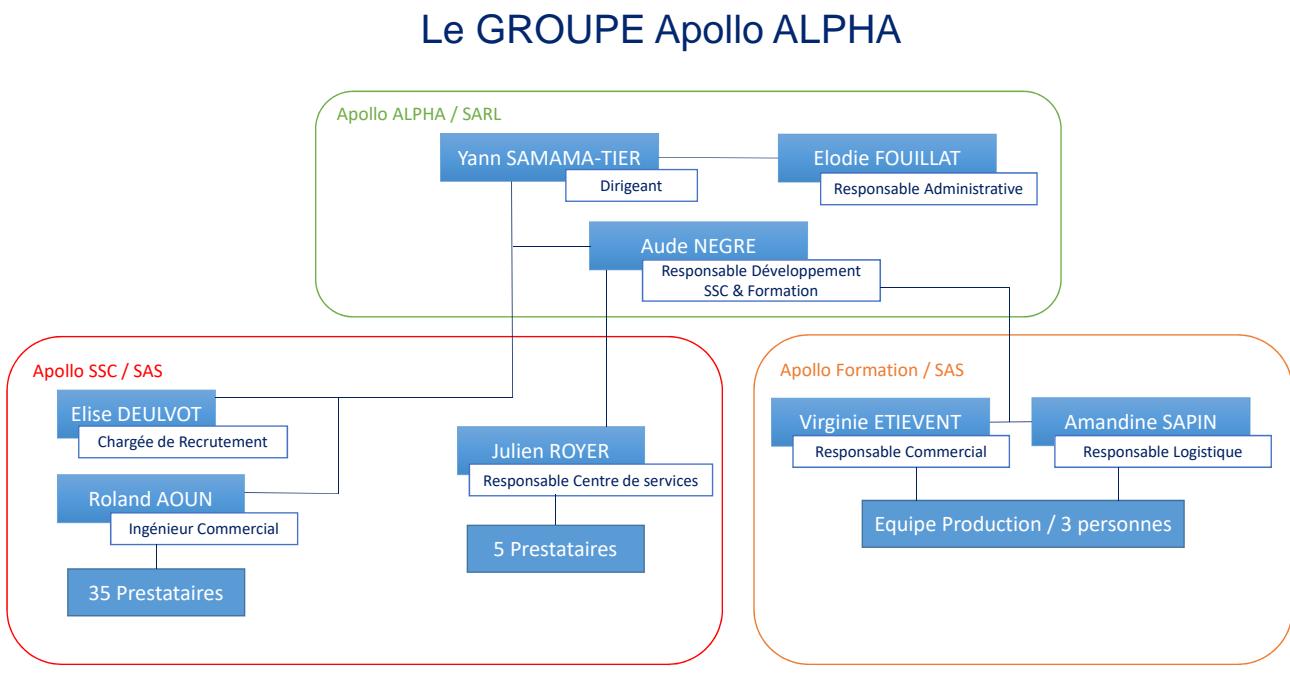
L'équipe de développement est divisée en deux : ceux qui travaillent à l'agence et ceux qui travaillent chez le client. Cette division permet à l'entreprise de travailler plus étroitement avec le client, mais aussi de conserver une certaine réactivité car l'équipe de l'agence est beaucoup plus disponible pour étudier les projets, et effectuer de la maintenance.

Il est important de noter que sur ces quelques quarante ingénieurs, ils sont rarement plus de deux ou trois à travailler à l'agence.

1.5 Organisation et fonctionnement

La figure suivante montre le fonctionnement interne de l'entreprise tel que j'ai pu le percevoir durant mon stage :

FIGURE 1.1 – Organigramme Apollo



Comme le montre cette figure, Apollo SSC fait partie d'un ensemble de trois sociétés différentes mais qui restent étroitement liées :

Apollo Alpha qui est une compagnie de gestion administrative du groupe.

Apollo Formation qui est une entreprise proposant des cours techniques ou théoriques (sur des sujets liés à l'informatique) à des entreprises ou des particuliers.

Chapitre 2

Organisation & Méthodologie d'Apollo SSC

2.1 Stratégie de Développement

Dans le cadre d'une démarche de qualité chez Apollo SSC, les projets développés suivent un processus de développement outillé et normé.

Pour développer un système, Apollo SSC utilise sa propre infrastructure de développement :

- **Visual Studio** est utilisé comme environnement de développement
- **Mantis** est le gestionnaire d'anomalies utilisé pendant les phases de développement et de recette
- **SVN Subversion** est utilisé comme gestionnaire de code source
- **Azure**, est utilisé afin de mettre à disposition régulièrement des versions intermédiaires

2.1.1 Gestionnaire de version de fichiers

Dans le cadre du développement d'une application, et quelle que soit son envergure, il est possible que plusieurs développeurs travaillent en même temps sur les mêmes fichiers. Pour ne pas perdre d'information d'une "version" à l'autre, on utilise alors un *Concurrent Versions System* (de nos jours on parle de SVN). Cet outil permet de gérer les versions d'un logiciel pour ne pas perdre de modifications d'un ordinateur à un autre.

Tortoise SVN est un des logiciels client de SVN les plus populaires car il est open source et s'intègre directement à l'interface Windows. C'est avec celui-ci qu'Apollo travail.

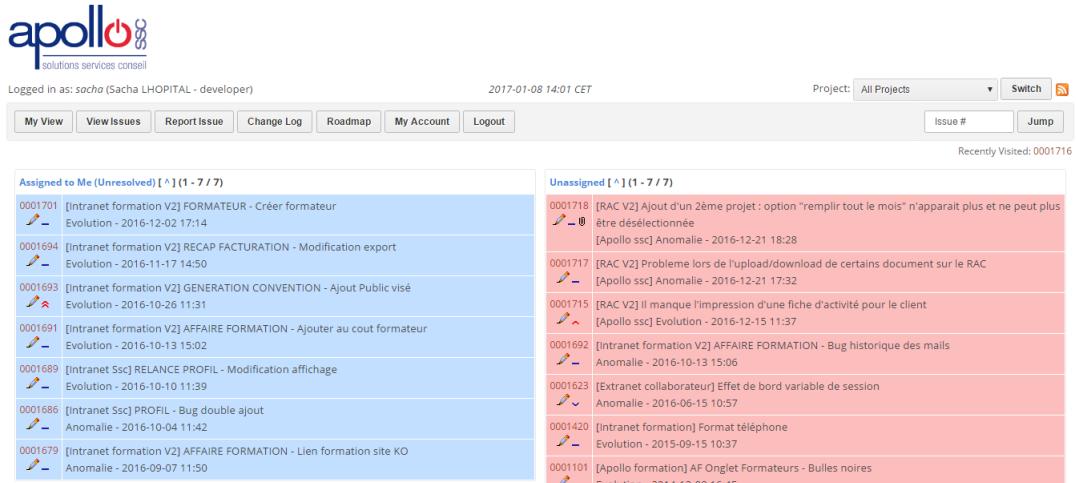
2.1.2 Gestionnaire de suivi d'anomalies

Le logiciel Mantis est un système de suivi de bugs basé sur une interface web.

Le principe de cet outil consiste à enregistrer la déclaration d'un bug, puis pour les développeurs concernés, à mettre à jour l'avancement de sa résolution, jusqu'à sa clôture. Le déclarant de l'anomalie peut s'informer à tout moment de l'avancement du traitement de son problème.

L'ensemble de l'équipe d'Apollo SSC utilise cet outil pour pouvoir facilement trouver les bugs, les traiter par ordre de priorité et les assigner à la personne la plus à même de les résoudre. Cet outil permet ainsi un gain de temps considérable et une communication rapide.

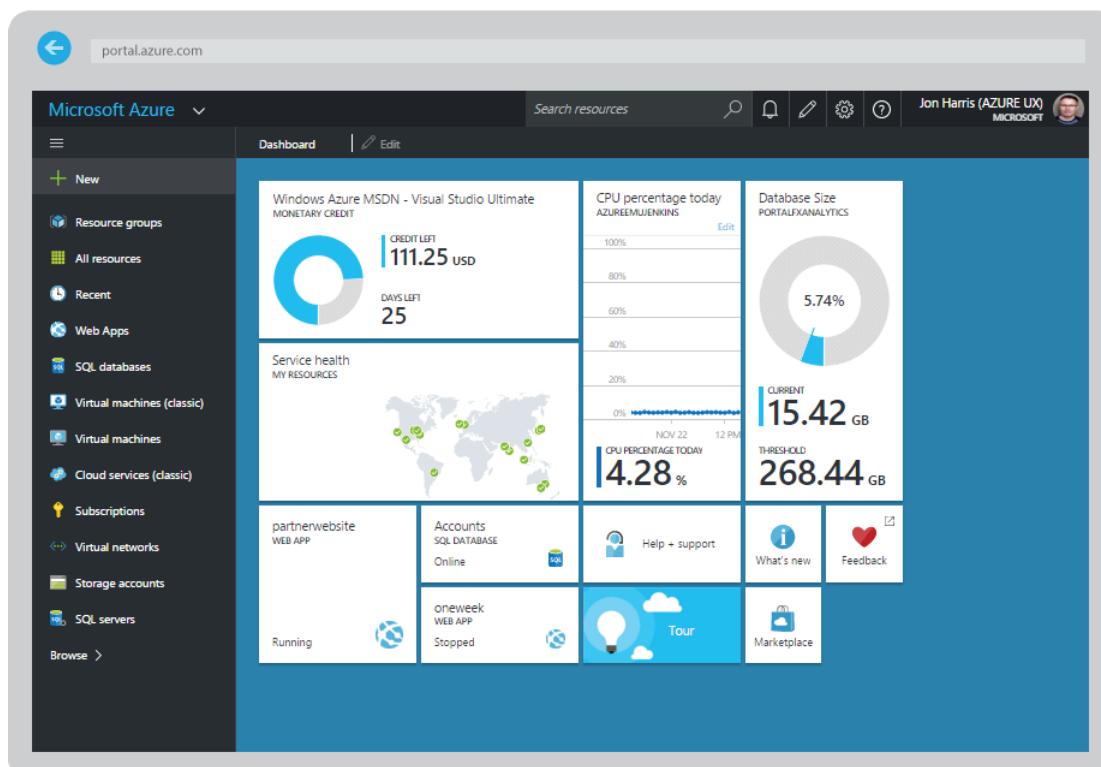
FIGURE 2.1 – Interface Mantis Bug Tracker



2.1.3 Microsoft Azure

Il s'agit d'une offre d'hébergement (applications et données) et de services (stockage de fichiers par exemple). Un ensemble d'API permet d'utiliser et d'accéder à cette plate-forme et aux services associés.

FIGURE 2.2 – Interface de Microsoft Azure



2.2 Stratégie de Test

La stratégie de test, formalisée lors de la phase de lancement d'un projet est déployée pour assurer des tests unitaires, des tests d'intégration et des tests de non-régression. Pendant mon stage, j'ai réalisé des tests avec Karma et Jasmine.

Les tests automatiques sont indispensables pour vérifier le bon fonctionnement d'une application rapidement. Karma et Jasmine sont des outils qui permettent d'automatiser des tests unitaires pour des applications AngularJS. Ils fonctionnent très bien ensemble mais ont des fonctions bien distinctes :

Karma : est un utilitaire Javascript qui permet de générer les codes sources d'une application et d'exécuter les tests qui y sont associés.

Jasmine : est un framework de tests pour Javascript qui est devenu très populaire dans les applications AngularJS. Son point fort est qu'il permet de structurer ses tests de façon très lisible, même si le test est imposant.

Pour plus de détails sur son fonctionnement, se référer à la section 4.1.3

2.3 Organisation d'Apollo SSC

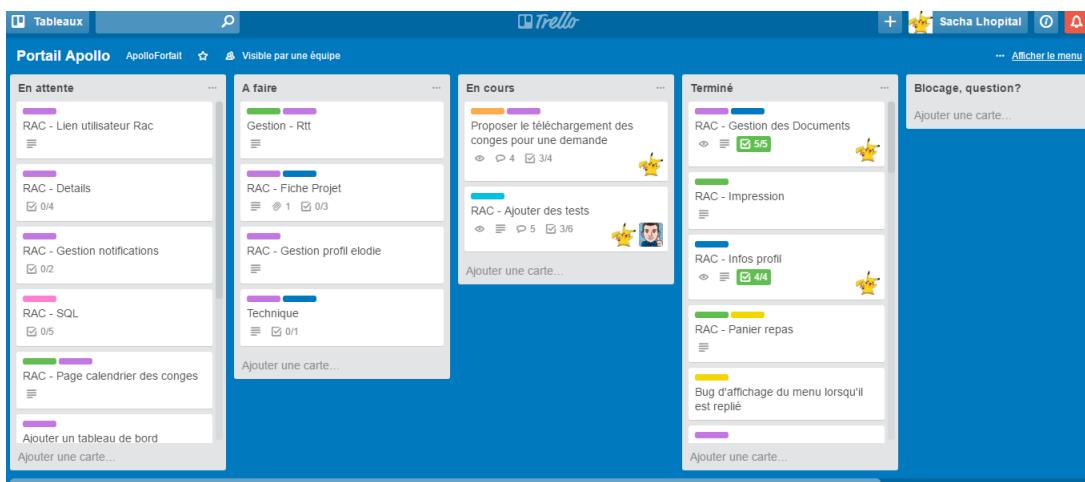
Apollo SSC a décidé d'utiliser l'agilité dans son processus de création d'applications. Ces méthodes agiles se veulent plus pragmatiques que les méthodes traditionnelles. Elles impliquent au maximum le demandeur (client) et permettent une grande réactivité à ses demandes.

Apollo met en place ces méthodes avec l'utilisation de Trello et de meetings hebdomadaires.

Trello

Trello est un outil de gestion de projet en ligne. Il est basé sur une organisation des projets en planches listant des cartes, chacune représentant des tâches. Les cartes sont assignables à des utilisateurs et sont mobiles d'une planche à l'autre, traduisant leur avancement.

FIGURE 2.3 – Interface Trello



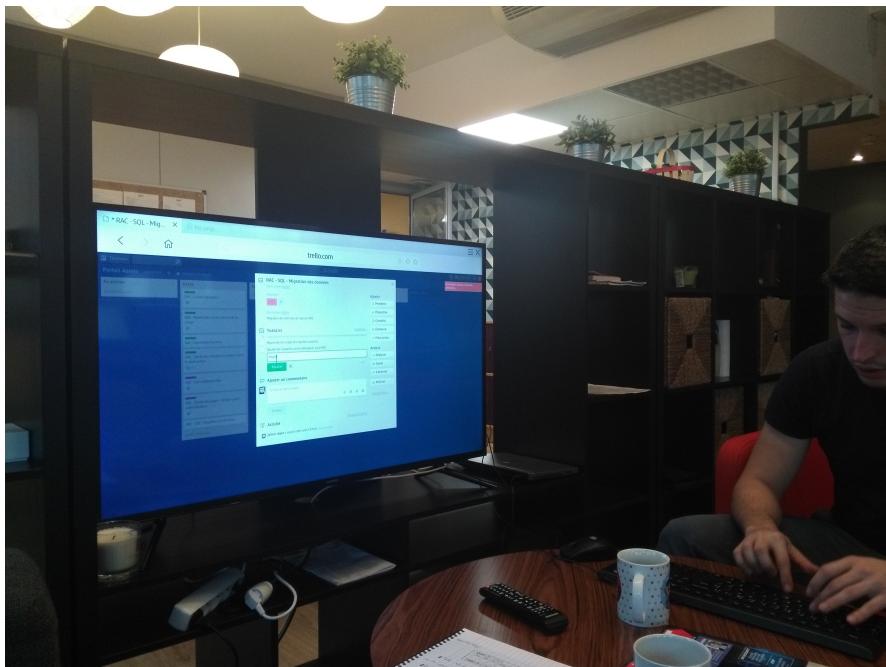
Cycle de développement

Les méthodes agiles proposent des cycles de développement sous forme d'*itérations* de durées variables. Une itération est un cycle dans le développement agile (plus court que dans une méthode classique) qui va se répéter autant de fois que nécessaire jusqu'à terminer l'ensemble des fonctionnalités.

Les cycles de développement des projets sont rythmés par des réunions en début de semaine pour discuter du projet et de son avancement. Chaque nouvelle réunion est l'occasion d'un nouvel échange entre développeurs et ce fonctionnement permet de mesurer l'évolution des projets pas à pas.

La figure 2.4 montre un exemple de meeting hebdomadaire avec Julien ROYER sur le Trello d'un des projets.

FIGURE 2.4 – Meeting hebdomadaire



Chapitre 3

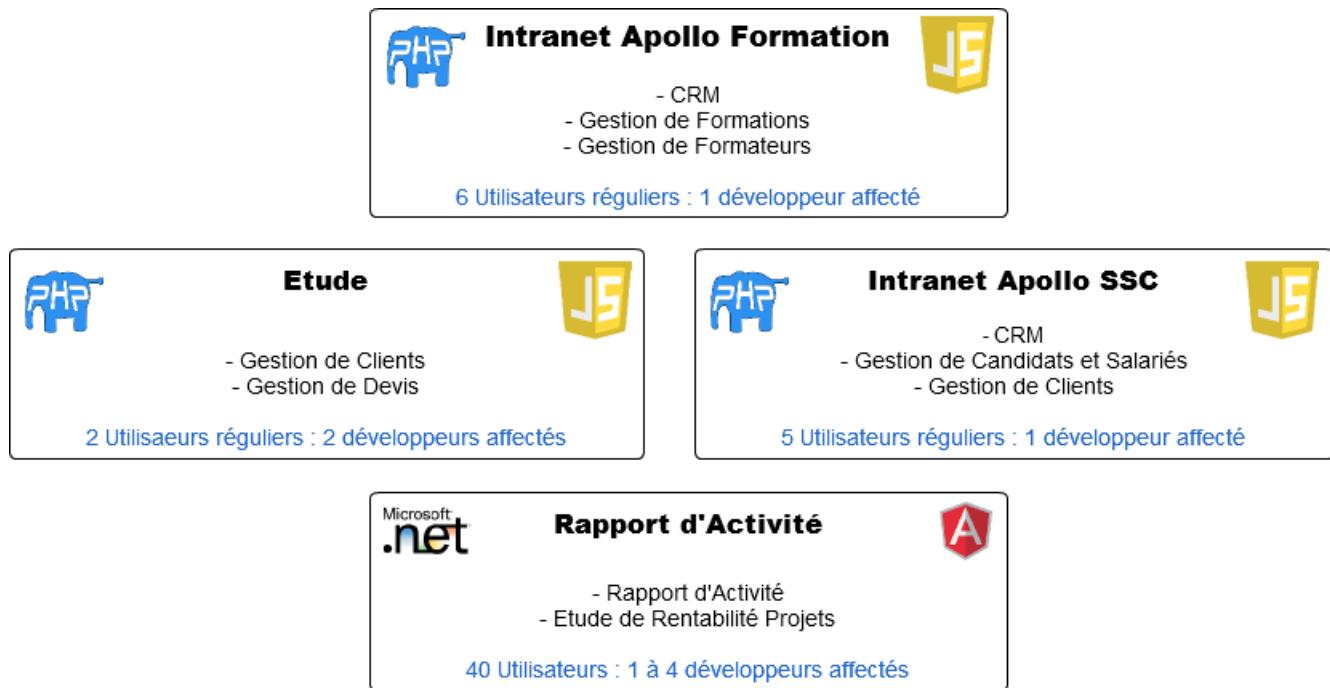
Cartographie des projets

Avant propos :

L'ensemble des logiciels sur lesquels je suis intervenue existent depuis plusieurs années et sont donc très complets.

Travailler dans une ESN, ce n'est pas toujours travailler pour un client. Il peut parfois arriver que l'on reste à l'agence et que l'on participe au développement interne de l'entreprise.

FIGURE 3.1 – Cartographie des projets



On distingue deux catégories de projets : des intranets internes à Apollo et des projets pour des clients . Ces projets sont décrits ci-après.

La cartographie ci-dessus résume en quelques points forts ces applications. En bleu est renseigné le nombre d'utilisateurs quasi-quotidien sur l'application au regard du nombre de développeurs travaillant sur le projet.

3.1 Présentation des intranets

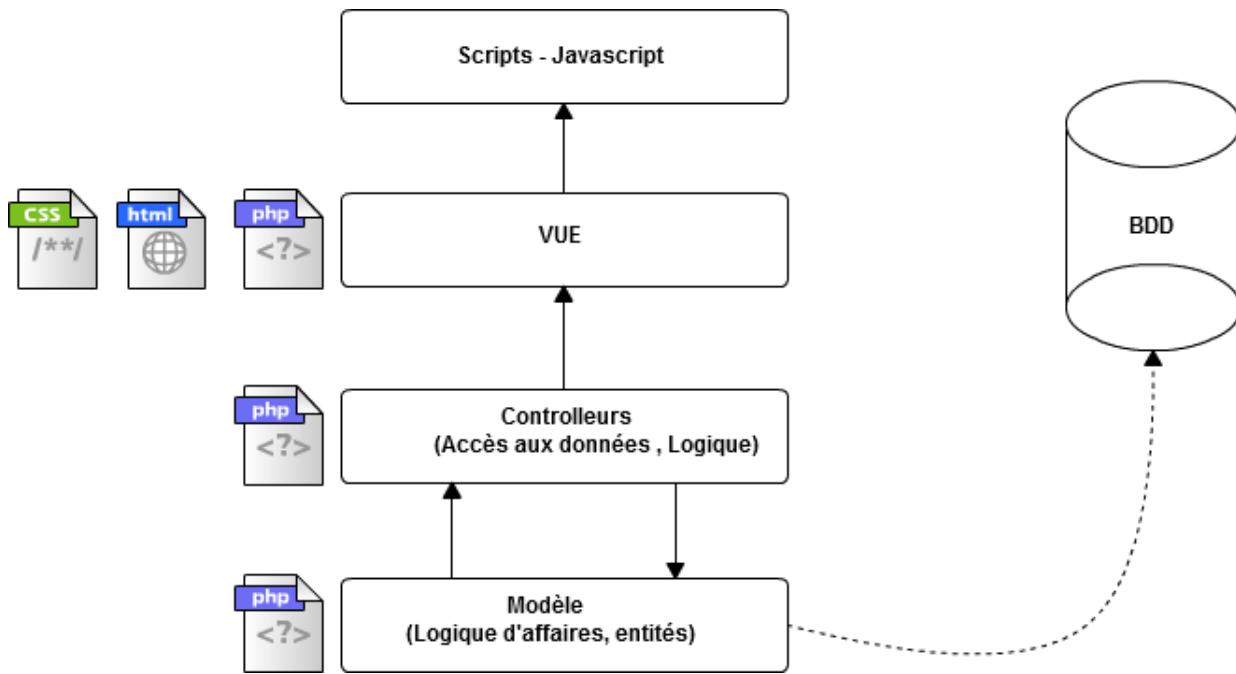
Ces deux intranets sont des projets sur lesquels je suis intervenue ponctuellement et surtout au début du stage.

3.1.1 Architecture orienté MVC*

Les deux intranets présentés ci-dessous sont développés en PHP, Javascript classique (en MVC*) et sont déployés sur un serveur OVH. Les bases de données sont enregistrées avec MySQL.

L'architecture des projets est très classique comme le montre le schéma ci-après : les fichiers sont organisés en MVC*.

FIGURE 3.2 – Architecture des intranets



Vue contient l'ensemble des vues du système, les fichiers html, css et tout ce qui concerne l'affichage des données.

Modèle contient un ensemble de classes pour l'implantation du model de la base de donnée. C'est par cette couche que l'on effectue les requêtes sql et que l'on construit les objets utilisables par les contrôleurs. C'est également à cet endroit que l'on définit des classes de type ViewModel*.

Contrôleurs est un répertoire MVC* qui contient entre autres le code permettant de lier model et vue. C'est dans ces fichiers que l'on écrit toute la logique des applications.

Scripts enfin, l'ensemble des fichiers Javascripts sont enregistrés à part, dans le fichier Script.

3.1.2 Intranet Apollo Formation

Apollo Formation propose ses formations à des entreprises ou des particuliers. Pour cela, elle fait appel à différents formateurs externes à l'entreprise qu'elle rémunère en fonction de la prestation.

Son intranet très complet permet à l'équipe d'Apollo Formation de gérer son processus métier, c'est-à-dire de gérer : les formations, les formateurs, les clients, la comptabilité, etc.

FIGURE 3.3 – Visuel de l'intranet Apollo Formation

The screenshot shows a complex web-based application for managing training programs. At the top, there's a navigation bar with tabs for 'PROFILS', 'COURSES FORMATEURS', and 'SUPPLIERS'. Below the navigation, there's a search bar and a date range selector ('Date Todo' from 16/04/2013 to 16/04/2013). The main area contains several sections: 'Informations principales' (with tabs for 'Qualification commerciale', 'Formation', 'Devis', 'Demande', 'Statut', 'Type d'affaire', 'Session', and 'Logis'), 'Client' (with tabs for 'Client', 'Formation', 'Formateur', 'Cours', 'Facturation', 'Documents', and 'Historique des mets'), and 'Formulaire' (with tabs for 'Client', 'Formation', 'Formateur', 'Cours', 'Facturation', 'Documents', and 'Historique des mets'). There are also sections for 'Adresse 1' and 'Adresse 2' with fields for 'Nom', 'Pays', 'Ville', 'Code postal', 'Téléphone', 'Fax', 'TVA', 'SIRET', and 'Personne à auditorer'. A large green button at the bottom left says 'CREER CE COURS'.

En complément de cet intranet, réservé aux salariés, il existe aussi un intranet accessible aux formateurs avec un compte pour qu'ils puissent gérer leurs cours.

3.1.3 Intranet Apollo SSC

Apollo SSC possède également son propre intranet qui permet de gérer (entre autres) : les clients de l'ESN, les collaborateurs (*i.e.* : *les salariés de l'entreprise*), la partie Ressources Humaines de l'ensemble du groupe Apollo, etc.

FIGURE 3.4 – Visuel de l'intranet Apollo SSC

The screenshot shows the Apollo SSC intranet. At the top, there's a navigation bar with tabs for 'Profils', 'Clients', 'Prospection', 'Tableau de bord', and 'Gestion'. Below the navigation, there's a search bar with a placeholder 'Filtre de recherche (facultatif)'. The main area has two main sections: 'Source' (with dropdowns for 'Résultat' and 'Préférences sélectives') and 'Etat' (with dropdowns for 'Statut', 'Email', 'Avec email', 'Téléphone', 'Tous', and 'Afficher tous les profils / Sortie Apollo'). Below these sections is a table titled 'Envoyer un email' with columns for 'Nom', 'Prénom', 'Email', 'Compétences', 'Portable', 'Statut', 'Date', 'Réactivité', and 'Etat'. The table lists several employees, such as GRAND MURELLON, Emiliana, Adrien, Kader, ABDESS, Rayane, ABDOURAHMANE, Abdourahman, Yousra, Abdellah, and others. At the bottom of the table, there's a note: 'Transfert des données depuis intranet.apollo.com...'. The footer of the page includes a copyright notice: 'Copyright © 2014 Apollo Formation - Tous droits réservés'.

3.2 Présentation des outils clients

Rapidement durant mon stage, j'ai eu le loisir de découvrir et de participer à des projets (presque) de A à Z dans leur réécriture et/ou leur restructuration. J'ai ainsi participé au développement du projet **Etude** pour un client et du projet **Rapport d'Activité** (utilisé en interne par Apollo).

La principale différence entre ces projets et les projets énoncés dans le chapitre précédent, est qu'ils possèdent des exigences métiers conséquentes que l'on doit implémenter dans l'application.

Toute la difficulté consiste alors à comprendre et à intégrer ces exigences dans l'application de la manière la plus propre, fonctionnelle et simple possible.

3.2.1 Etude

Imerys est une société spécialisée dans la vente de toiture (tuiles, pose de tuiles) qui dispose d'un intranet spécifique lui permettant de gérer ses produits, d'étudier ses statistiques ou encore de gérer ses clients. L'application permet également de lier les salariés sur cette plateforme car l'entreprise est assez conséquente et elle possède un certain nombre de responsables dans toute la France. Elle permet également d'envoyer différents mails aux clients ou aux utilisateurs pour suivre l'avancement des différentes études.

Cette application est principalement utilisée par le service *Hotline* de l'entreprise qui gère les devis pour les commerciaux et/ou les clients. C'est un outil indispensable pour eux car ils utilisent l'application quotidiennement.

FIGURE 3.5 – Exemple d'un visuel d'Etude

Etudes									
N° d'étude	Type	Département	Chantier	Tuile	Envoyée à	Date d'enregistrement			
11000	Batiment	Commerceaux Régions	Professionnels						
copie de 11058	Gironnage	A réaliser	Monnet Alexis Test	LHOPITAL Sacha Couvreur / Charpentier	Ardoise	02/01/2017			
			RGA		0 m ²	0 ppm	02/01/2017		
11000	copie de 11058	Gironnage A réaliser	10 Gaché Aurélie	PONT D' AIGREFEUILLER	CARTIER	Plate 17/27 Doyettee	29/12/2016		
				1, Rue Louis Berthinel 10000 TROYES		0 m ²	29/12/2016		
				03 25 83 26 60		95.6333 ppm			
				03 25 83 26 61					
11000	copie de 11058	Gironnage A réaliser	10 Gaché Aurélie	PONT D' AIGREFEUILLER	CARTIER	Plate 17/27 Doyettee	29/12/2016		
				1, Rue Louis Berthinel 10000 TROYES		0 m ²	29/12/2016		
				03 25 83 26 60		95.6333 ppm			
				03 25 83 26 61					
11015	copie de 11052	Gironnage A réaliser	92 Arnaud Stéphane	SAGA ENTREPRISE Couvreur / Charpentier	Château de MÉNILLES	Plate 17/27 Doyettee	21/07/2016		
				12, boulevard Louise Michel - Bât. B 92238 GENNEVILLIERS		0 m ²	21/07/2016		
				0147916709		84.575 ppm			
				0147916709					

Etude est actuellement constitué de deux projets : hotline V1 et hotline V3. Hotline V1 constitue la première version de Etude et hotline V3 est la dernière en date. C'est sur cette dernière que j'ai travaillé. L'objectif d'Apollo est de passer progressivement de la V1 difficilement maintenable à la V3 plus propre.

La première version d'Etude est développée en PHP Javascript classique. La V3 utilise en plus le framework PHP Code Igniter pour faciliter la mise en place du MVC*.

3.2.2 Rapport d'Activité

Enfin, en complément de l'intranet Apollo SSC (cf. 2.1.2), il existe une autre plateforme appelée communément *Rapport d'Activité* (RAC) qui permet à chaque collaborateur de l'entreprise de gérer son temps chez les différents clients et ainsi à l'administration d'avoir une vue globale sur ce qu'il se passe dans l'entreprise à un instant T.

Ce projet est développé avec PHP, Javascript et une base de données MySQL.

Très vite, il a été décidé de reprendre ce projet de zéro car il devenait difficile d'assurer sa maintenabilité.

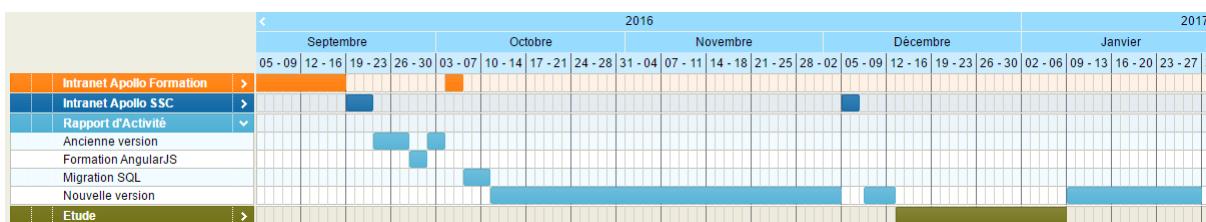
3.3 Plannification

Travailler sur ces quatre projets très différents n'était pas prévu au début de mon stage car six mois, c'est assez court. Ces changements réguliers étaient très enrichissants pour moi tant techniquement qu'intellectuellement.

Cependant, pour ne pas me perdre entre chaque projet, j'ai créé un *journal de bord* pour y noter les tâches à faire, et m'organiser. Grâce à celui-ci, j'ai réalisé un diagramme de Gantt dont un extrait se trouve ci-après (cf. figure 3.6). Il donne une idée du temps que j'ai passé sur chaque projet.

Les tâches ainsi réalisées étaient variées et de plus en plus complexes au fur et à mesure du temps si bien qu'elles m'ont permis de percevoir les différentes applications sous des angles différents.

FIGURE 3.6 – Planning des tâches effectuées pendant le stage



En bleu clair figurent les différentes tâches liées au **Rapport d'Activité** (soit environ treize semaines). En orange est représenté le temps de travail sur l'**intranet Apollo Formation** (soit un peu plus de deux semaines). Le bleu foncé représente ainsi l'**intranet Apollo SSC** (sur lequel j'ai passé une semaine). Enfin en vert est représenté ma contribution sur le projet **Etude**.

Pour clore cette section, je souhaite souligner le peu de temps que j'ai consacré aux intranets internes par rapport à mon temps de travail sur **Etude** et le **Rapport d'activité**. Il me semblait tout de même important de parler dans ce rapport de tous ces projets car ils m'ont permis de bien cerner les besoins de réécriture du **Rapport d'Activité** et de me concentrer sur la qualité de mon code lors de l'écriture d'**Etude**.

Chapitre 4

Présentation détaillée des développements : RAC & Etude

Dans cette partie on détaillera les différentes tâches que j'ai réalisées pendant le stage sur le RAC* et Etude (qui représentent environ 80% de mon stage). Mon travail a autant consisté en l'amélioration de leurs fonctionnalités qu'en leurs réécriture.

4.1 Rapport d'Activité

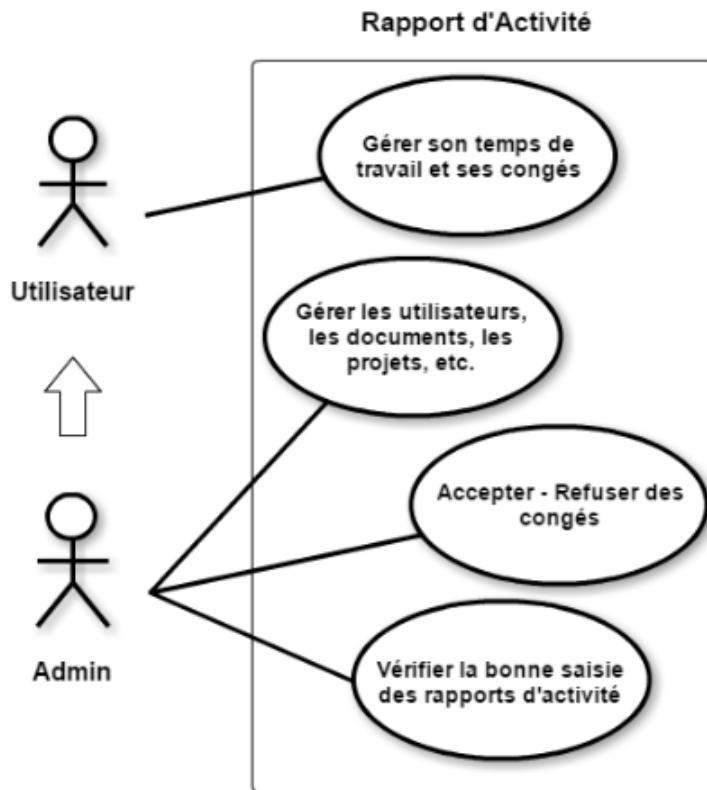
4.1.1 Périmètre Fonctionnel Détaillé

Le diagramme ci-après montre un peu plus en détail le besoin à (re)mettre à place.

La nouvelle application doit avant tout posséder les mêmes fonctionnalités qui sont déjà présentes dans l'ancienne version. Les plus importantes et urgentes sont décrites dans le point 4.1.3.

Pour repartir sur des bases saines, une nouvelle version du RAC* est développé en .Net (pour le back-end), avec le framework Entity, et AngularJS (pour le front-end). La version finale de l'application est déployée sur Azure avec une base de données SQL Server.

FIGURE 4.1 – Diagramme de Cas d’Utilisation du RAC

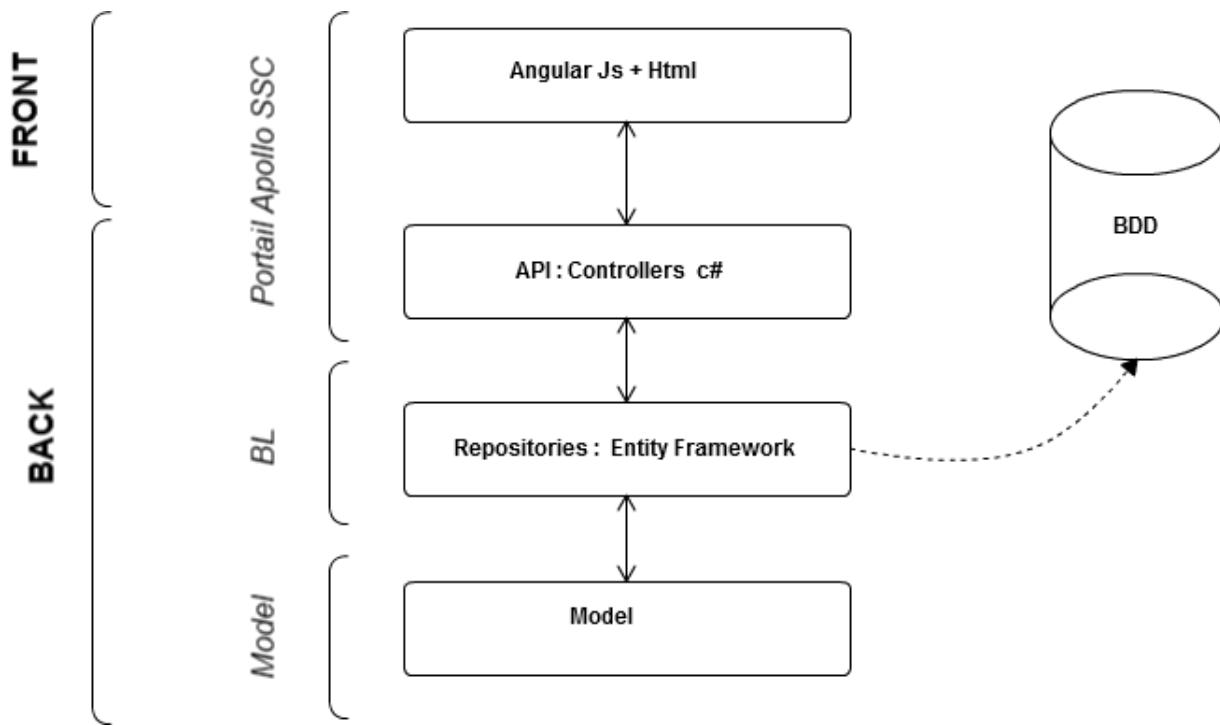


4.1.2 Architecture

La nouvelle version possède une architecture plus propre et plus claire que l’ancienne version (toujours en respectant le modèle MVC*).

L’application est divisée en trois couches logicielles principales (qui constituent trois projets différents) :

FIGURE 4.2 – Architecture du Rapport d’Activité



Couche Principale : PortailApolloSSC

Cette couche comprend l'ensemble du code nécessaire au front-end. Les fichiers sont regroupés comme suit :

- Controllers : Les controllers .Net qui permettent de faire le lien entre back-end et front-end. Ce sont ces fichiers qui contiennent toute la logique de l'application.
- Scripts : qui contient 100% du code AngularJs (les controllers, les vues, les directives, les fichiers de test, les librairies). Les fichiers sont organisés par composant.
- Models : qui contient (éventuellement) quelques models de données particuliers pour faciliter leur utilisation dans les vues.

Couche BL

Elle contient l'ensemble des Repositories qui permettent d'effectuer les requêtes sur la base de données avec Entity.

Couche Model

Cette couche traduit le model de la base de données pour chaque table de la base.

4.1.3 Tâches

Gérer les demande de congés

Rappel du contexte : Développer une application pour permettre de gérer les congés et le temps de travail des collaborateurs d'Apollo.

Objectif : Mettre en place le système de demandes de congé. Avant de partir en congé, un utilisateur doit soumettre à l'administration une demande pour la période.

Nous détaillerons plus particulièrement l'action d'effectuer une demande de congé côté utilisateur.

Un utilisateur peut effectuer un certain nombre de demandes de congés. Ce nombre varie en fonction de différents facteurs. Sur la base d'**1** journée, un utilisateur peut décider de poser **1** ou **0,5** congés (0,5 correspondant à une demi-journée).

Une demande de congé peut alors être confirmée ou infirmée par un administrateur.

Je me suis plus particulièrement intéressée à trois types de congés : les **Rtt**, les **Congés Payés** et les **Congés Payés Anticipés** :

Rtt : Chaque utilisateur qui dépasse les 35h de travail hebdomadaire acquiert 0,75 Rtt en fin de mois.

Ces Rtt ne sont pas cumulables d'année en année : au 31 décembre, tous les Rtt qui n'ont pas été posés sont perdus.

Congés Payés : Chaque utilisateur acquiert 25 jours de congés payés le 1er mai. Un utilisateur gagne 1 congé payé supplémentaire par tranche de cinq ans dans l'entreprise (donc un salarié dans l'entreprise depuis dix ans acquiert 27 congés par an).

Congés Payés Anticipés : Les congés payés anticipés correspondent aux futurs congés payés que l'utilisateur acquerra au 1er mai de l'année suivante. Ce compteur permet aux utilisateurs de disposer de leurs congés payés en avance. Au 1er mai, le nombre de congés anticipés restant se cumulent aux congés payés disponibles.

FIGURE 4.3 – Exemple : Pierre effectue une demande du 5/12/16 au 8/12/16 pour 2,5 Congés Payés.

décembre 2016	jeu.	ven.	sam.	dim.	lun.	mar.	mer.	jeu.	ven.
APOLLO SSC ✕	1	1						1	1
Total Clients	1	1						1	1
Congés payés					1	1	0.5		
RTT									
Congé temps partiel									
Maladie									
Congés payés anticipés									
Congés sans solde									
Congés exceptionnels									
Congés maternités									
Total Congés					1	1	0.5		
Total	1	1			1	1	0.5	1	1

Une demande de congé concerne ainsi un type de congé particulier pour une certaine période pour un utilisateur donné et pour un nombre précis. Un exemple est illustré ci-après par la figure 4.3. Pour gérer le nombre de congés disponible pour chaque utilisateur, leur nombre est enregistré en base de données et éventuellement mis à jour par un administrateur.

Compte tenu des règles de gestion des congés, les fonctions de demande de congés à développer sont énumérées telles que :

1. Si je ne possède pas suffisamment de congés pour le type que je souhaite soumettre, je ne peux pas effectuer la demande.
2. Au mois de décembre (et uniquement), j'ai le droit d'avoir jusqu'à **-0.75** rtt disponible (car on peut poser le Rtt de décembre en avance). Sinon, aucun compteur ne peut être négatif.
1. Si je n'ai plus de congés pour un type, je ne peux pas faire de demande pour ce type.

Par défaut lorsque l'utilisateur saisit des congés dans le tableau associé (extrait figure 4.3), un bouton apparaît dans le menu. Ce bouton permet à l'utilisateur d'envoyer sa demande de congé à l'administrateur.

La solution la plus simple consiste à camoufler ce bouton si on n'est pas en mesure de poser des congés. Avec AngularJs, il est possible d'utiliser une directive officielle *ng-if* qui crée une portion du DOM UNIQUEMENT si la condition passé en paramètre retourne vrai. Détaillons cette condition ci-dessous :

```

1 <li ng-if="racCtrl.service.racValidForSave
2   && racCtrl.haveDemandeToSend()
3   && (racCtrl.dataSaved || racCtrl.dataUnchanged)">
4
5   <button uib-tooltip="Envoyer les demandes de conges"
6     tooltip-placement="left" [...]>
7     [...]
8   </button>
9 </li>
```

La première condition (ligne 1) vérifie si le Rapport d'activité est valide pour une sauvegarde. Ce booléen (disponible dans l'instance service du controller racCtrl) retourne *false* dès que le Rapport d'activité possède des données incohérentes. Cela pour éviter une tentative de sauvegarde qui ne fonctionnerait pas de toute façon. La condition ligne 3 vérifie avec les booléens que les données affichées ne sont pas différentes de celles enregistrées en base de données (car le Rapport sauvegarde automatiquement les données lorsqu'on les modifie).

La seconde condition (ligne 2) est une méthode de racCtrl qui vérifie si on peut envoyer une demande. Cette méthode est détaillée ci-après. Le racCtrl est le controller AngularJS principal du RAC.

```

1  racCtrl.haveDemandeToSend = function () {
2
3      var haveDemande = false;
4
5      /* Si les donnees existent */
6      if (RacService.data && RacService.data.TableConges) {
7
8          /* Pour chaque ligne du tableau (i.e : chaque type de conge) */
9          angular.forEach(RacService.data.TableConges, function (line,
10              keyLine) {
11
12
13             /* Pour chaque case */
14             angular.forEach(line.Saisies, function (saisieConge, keyCol) {
15
16                 /* Si la valeur saisie dans la case est plus grande que 0
17                 et que cette demande n'a pas encore ete soumise aux
18                 administrateurs */
19                 if ((saisieConge.Value > 0 && !saisieConge.Demande) ||
20                     saisieConge.ForceDemande) {
21
22                     /* On affiche le bouton en retournant true */
23                     haveDemande = true;
24                     return;
25                 }
26             });
27         });
28     }
29
30     return haveDemande;
31 }
```

Ces vérifications sont dupliquées dans la partie back-end de l'application au cas où quelqu'un arriverait à contourner le front-end.

2. Si on est en décembre, je peux effectuer une demande pour -0,75 Rtt.

Pour vérifier cette condition particulière, il n'est pas nécessaire de modifier le code précédent qui est dans le front-end. On pourrait le faire mais ce serait très lourd pour par grand chose.

Par contre il faut impérativement l'implémenter dans le controller qui communique avec le back-end :

```

1 public IHttpActionResult SendDemandeConges(SaisieActiviteListViewModel
2   data){
3
4     var listTypeConges = repoTypeConge.Get();
5     List<DemandeConge> newDemandesConges = data.GetNewDemandeConges();
6
7     double countConges = 0;
8     double countCongesAnticipes = 0;
9     double countRtt = 0;
10    bool hasRTT = false;
11
12    foreach (DemandeConge dc in newDemandesConges){
13      var typeConge = listTypeConges.FirstOrDefault(tc => tc.Id == dc.
14        TypeCongeId).Nom;
15
16      if (typeConge == Constantes.CONGES_PAYES){
17        countConges += dc.Value;
18      }
19      if (typeConge == Constantes.CONGES_ANTICIPES){
20        countCongesAnticipes += dc.Value;
21      }
22      if (typeConge == Constantes.RTT){
23        hasRTT = true;
24        countRtt += dc.Value;
25      }
26    }
27
28    double rttAvailableMax = (data.Mois == 12) ? -0.75 : 0;
29
30    if (user.CongesDispo - countConges >= 0
31      && user.CongesAnticipesDispo - countCongesAnticipes >= 0
32      && (user.RttDispo - countRtt >= rttAvailableMax || !hasRTT)){
33
34      repoDemandeConge.UpdateByUserIdAndMonth(user.Id, data.Mois, data.
35        Annee, newDemandesConges, assistantes);
36      return Ok();
37    }
38
39    return BadRequest("Vous ne possédez pas assez de congés pour cette
40      demande");
41  }

```

La méthode précédente est disponible dans le controller du Rapport d'activité et envoie une demande au back-end.

Après avoir récupéré les demandes de congés que l'utilisateur souhaite envoyer (ligne 4), on en déduit le nombre de congés de chaque type qui sont concernés (dans le *foreach* ligne 11).

Puis ligne 26, on définit un nombre de *rttAvailableMax* qui prend la valeur -0.75 si on est au mois de décembre ou 0 sinon. Puis, ligne 28, on vérifie si aucun compteur n'est négatif. Petite spécificité, ligne 30 : en décembre, si on pose un congé pour janvier, le solde RTT sera toujours négatif. Il faut donc vérifier que l'utilisateur ne saisisse pas de RTT pour Janvier. Ceci est assuré par le booléen *hasRtt*.

Si tout est bon, on enregistre les demandes en base de données (ligne 32).

Tester le bon fonctionnement du Rapport d'activité

Rappel du contexte : Développer une application pour permettre de gérer les congés et le temps de travail des collaborateurs d'Apollo. Le RAC est constitué de trois tableaux édroitement liés permettant de préciser jour par jour : son temps de travail, son temps de congé, et ses notes de frais.

Objectif : Tester automatiquement le fonctionnement des tableaux du RAC pour éviter les régressions. Réaliser les tests côté AngularJS dans un premier temps.

Pour réaliser l'ensemble de ces tests, j'ai utilisé Karma en association avec Jasmine (cf. 2.2). Avec ces outils, il est possible de tester tous les composants d'une application AngularJS : Controllers, Services, Factories, Directives, Providers et Filtres. Néanmoins je n'ai pas eu le temps de réaliser des tests qui couvrent 100% de l'application, donc j'ai principalement testé la directive *apSpreadsheetDirective* qui génère le tableau du Rapport d'activité, le controller *RacController* et le service *RacService*. Est détaillé ici les tests concernant la directive.

Générer une instance du Controller

Une directive est un élément un peu particulier d'AngularJS car il rassemble à lui seul : un controller, un scope et un template html. Pour pouvoir tester son controller (qui contient une partie de la logique) il faut simuler son template et son scope. Nous allons en fait les compiler "manuellement" pour pouvoir garder l'indépendance de chaque test entre eux. J'ai ainsi défini deux méthodes qui permettent cette génération manuelle *generateDirective()* et *generateScope()*. Ces deux méthodes sont appelées dans un code qui s'exécute avant chaque test et crée une instance du controller :

```

1 beforeEach(inject(function (_$compile_, _$rootScope_, _$controller_, 
2   _apSpreadsheetService_, _DateService_) {
3 
4   $compile = _$compile_;
5   $scope = _$rootScope_.$new();
6   apSpreadsheetService = _apSpreadsheetService_;
7   dateService = _DateService_;
8 
9   generateScope();
10  generateDirective();
11 
12  apSpreadsheetController = directiveElem.controller('apSpreadsheet', {
13    $scope: isolatedScope
14  });
15 }));

```

Pour créer l'instance du controller (ligne 11), il faut définir tous les autres modules dont nous avons besoin pour exécuter les méthodes : *\$compile* pour compiler le template et le scope à la main ; *\$rootScope.new()* qui nous retourne un nouveau scope propre ; et des mocks* des différents services (ou les services réels !) que le contrôleur utilise (cf. le prochain point).

Mocker* les Services

Sur le code précédent (ligne 5 et 6), *apSpreadsheetService* et *DateService* correspondent à des services utilisés dans la directive, mais ils ne sont pas du tout utilisés de la même manière pour les tests. *apSpreadsheetService* est mocké* alors que *DateService* est réellement généré par l'application.

Cela signifie que par moment, le controller peut être amené à appeler des méthodes de ces services. Pour ce qui est de *DateService*, c'est un service très simple qui permet de manipuler les dates, donc on n'a pas besoin de le simuler. Par contre pour *apSpreadsheetService* on ne veut surtout pas le tester en même temps (car il pourrait y avoir des conflits), mais on souhaite tout de même être sûr que les méthodes dudit service fonctionnent correctement pour ne pas fausser nos tests. La solution revient alors à *Mocker** le module. De cette façon on "simule" le service et c'est nous qui choisissons quelles valeurs on retourne lorsqu'on l'appelle :

```

1  beforeEach(module('angularApp', function (_$provide_) {
2
3      apSpreadsheetService = {
4          createScopeVariables: function (scope) {
5              scope.isJourFerie = function (day) {
6                  return dateService.isDateInJsonArray(day, scope.
7                      joursFeries);
8              }
9          },
10     }
11
12     spyOn(apSpreadsheetService, 'createScopeVariables').and.callThrough();
13
14     _$provide_.value('apSpreadsheetService', apSpreadsheetService);
15 });

```

Ainsi, on redéfinit la sous-fonction *apSpreadsheetService.createScopeVariables* dans laquelle on définit à nouveau *scope.isJourFerie* qui retourne un booléen. Ensuite, ligne 13, on utilise *\$provide* pour signifier à l'application d'utiliser notre objet à la place du module qu'il connaît comme étant *apSpreadSheetService*.

Par la suite (ligne 11) on implémente un "espion" sur notre méthode pour être sûr que c'est cette mock* méthode qui est appelée dans nos tests.

Une fois tous ces préparatifs correctement réalisés, on peut ainsi réaliser nos tests dont un exemple est donné ci-dessous :

```

1  describe('isolatedScope.initScope()', function () {
2
3      beforeEach(function () {
4          spyOn(isolatedScope, 'isJourFerie').and.callThrough();
5
6          isolatedScope.joursFeries = [VALID_DATE];
7
8          isolatedScope.initScope();
9          directiveElem.scope().$apply();
10     });
11
12     it('should create one column for titles + one column by day + one
13        total column', function () {
14         expect(isolatedScope.columns.length).toBe(1 + moment(VALID_DATE).
15             daysInMonth() + 1);
16     });
17 });

```

4.2 Etude

4.2.1 Architecture

Concernant le projet Etude, son architecture est très proche du MVC* classique (cf. figure 3.2) à la différence que les fichiers sont gérés par Code Igniter.

CodeIgniter est un framework libre écrit en PHP. Il suit le motif de conception MVC* et s'inspire du fonctionnement de Ruby on Rails. Il permet de faciliter l'utilisation de certaines fonctionnalités du PHP (comme l'envoi de mails par exemple).

4.2.2 Tâches

Migration du module Autorisation

Rappel du contexte : Améliorer l'intranet de l'entreprise pour gérer la vente de toiture (tuiles, pose de tuiles). Apollo migre progressivement d'une V1 à une V3.

Objectif : Une étude (i.e : une affaire pour l'entreprise) peut avoir plusieurs statuts : En attente d'éléments, A réaliser, Réalisée, En Cours, Abandonnée. De plus, elle peut être : une Autorisation, un Gironnage*, un Courrier ou une étude Quantitative. Il existe en V1 un module permettant de transformer une étude de type "Autorisation" du statut "A Réaliser" à "Réalisée". Migrer ce module de la V1 à la V3. Prévoir également une vue de détail pour une Autorisation Réalisée.

Le visuel suivant, tiré de l'application hotline V1, montre l'interface qui permet de passer une étude Autorisation de l'état "A réaliser" à "Réalisée".

FIGURE 4.4 – Formulaire de réalisation d'une Autorisation - hotline V1

Autorisation:	<input style="width: 100px; height: 20px;" type="button" value="Ok"/>
Zone:	<input style="width: 100px; height: 20px;" type="button" value="Zone 1"/>
Site d'exposition:	<input style="width: 100px; height: 20px;" type="button" value="Normal"/>
Zone de neige:	<input style="width: 100px; height: 20px;" type="button" value="A1"/>
Charge de neige (SO) en kN/m ² :	<input type="text"/>
Pente principale en mpm:	<input type="text"/>
Pente annexe (si présence de coyau) en mpm:	<input type="text"/>
Surface de la toiture en m ² :	<input type="text"/>
Altitude du chantier:	<input type="text"/>
Ecran de sous-toiture (facultatif):	<input type="text"/>
Avec étanchéité complémentaire:	<input style="width: 100px; height: 20px;" type="button" value="Non"/>
Précisez le recouvrement de l'écran (facultatif) en cm:	<input type="text"/>
Longueur maximal du rampant (à l'horizontal) en m:	<input type="text"/>
Le pureau autorisé (si différent du minimum) Ex: "compris entre 310 mm et 315 mm":	<input type="text"/>
N'oubliez l'exemplaire du courrier en format PDF (si autorisation spéciale):	
<input type="button" value="Parcourir..."/> Aucun fichier sélectionné.	
<input type="button" value="Poursuivre l'enregistrement"/>	

Ce qui nous intéresse dans cet écran c'est avant tout de conserver les fonctionnalités du formulaire à savoir : s'assurer que tous les champs sont remplis et enregistrer ces données dans la base de données et envoyer un mail généré automatiquement par l'application.

La vue de détail à implémenter une fois l'étude "réalisée" doit être créée entièrement et résumer le mail précédemment envoyé.

Une étude de type Autorisation peut être de deux statuts différents : classique ou particulière. Dans le cas d'une autorisation classique, l'application génère un mail automatique à partir des données de l'application. Dans le cas d'une autorisation particulière, l'application envoi un mail générique très simple avec en pièce jointe le fichier pdf fourni dans le formulaire.

Migrer le formulaire

Toute la difficulté de cette partie réside en la migration du module sans perte de données. En effet, c'est ma première expérience dans le monde de la toiture et j'ai donc beaucoup de mal à comprendre les données qui sont saisies.

Néanmoins la migration est assez "simple" car le code ne nécessite pas énormément de changement entre la V1 à la V3. Comme la V3 est plus structurée que hotline V1, il faut par la suite "ranger" le code au bon endroit dans l'application (La vue doit contenir le minimum de code PHP ; Le model doit contenir les requêtes SQL ; Le controller doit contenir la logique et le traitement des données pour faire le lien Vue - Model). Pour gérer le formulaire plus facilement j'utilise une aide Code Igniter : *form_validation* dans le controller.

Une particularité de ce formulaire concerne la possibilité de fournir un fichier au format pdf ou non.

Dans un premier temps si ce fichier est fourni, il faut s'assurer qu'il est au bon format. Si c'est le cas on l'enregistre dans l'application. Il faut également s'assurer que dans le cas d'une autorisation particulière, le fichier est obligatoire. C'est ce que vérifie la fonction ci-après :

```

1 public function isFileValidThenUpload($str, $id) {
2     [...]
3     if (isset($_FILES['fichier']) AND $_FILES['fichier']['error'] == 0) {
4
5         $infosfichier = pathinfo($_FILES['fichier']['name']);
6         $extension_upload = $infosfichier['extension'];
7         $extensions_authorized = array('pdf');
8
9         if (in_array($extension_upload, $extensions_authorized)) {
10             $fichier = '.' . $id . '.pdf';
11             move_uploaded_file($_FILES['fichier']['tmp_name'], UPLOAD_PJ .
12                 $fichier);
13             $this->etudeManager->ajoutPieceJointe($id, $fichier);
14             return true;
15         }
16         [...]
17         return false;
18     }
19     /* Cas d'une autorisation spéciale */
20     if ($this->input->post()['isAuthorizationParticuliere'] == 'particuliere'
21         ) {
22         $this->form_validation->set_message('isValidThenUpload', 'Pour
23             une autorisation particulière, merci de fournir un courrier au
24             format pdf');
25         return false;
26     }
27     return true;
28 }
```

Cette fonction PHP est exécutée au moment où l'utilisateur valide son formulaire.

Le code compris entre la ligne 9 et 14 vérifie si l'extension du fichier upload est bien un pdf. Si c'est le cas, on l'enregistre dans l'application avec `move_upload_file()` ligne 11 et on enregistre en plus le chemin d'accès en base de données (ligne suivante).

En revanche (ligne 19), si aucun fichier n'a été fourni et que l'on est dans le cas d'une autorisation particulière, on affiche un message d'erreur à l'utilisateur pour le forcer à fournir un fichier.

Générer le mail

Là encore, le code relatif à la génération du mail est assez simple à migrer (à base de copier-coller). Il suffit de récupérer toutes les données nécessaires à la génération dans la base de données et de tout envoyer à la vue. Puis, j'utilise une aide Code igniter pour générer le mail à partir de la vue HTML et l'envoyer.

Le plus complexe ici a été de récupérer toutes les données correctement et de générer un mail correct et équivalent à l'ancienne version. En effet l'e-mail utilise beaucoup de données. La fonction generateAllDatas qui s'occupe de gérer toutes ces données est fournis en annexe.

Créer la vue détail

Enfin, pour la vue de détail, on réutilise les vues développées pour les autres types d'études pour garder une certaine homogénéité dans l'application en l'adaptant pour une Autorisation.

FIGURE 4.5 – Détail d'une Autorisation Réalisée

Comme le montre la figure 4.5, la vue détail est composée de trois autres petites vues qui permettent de combiner le code déjà existant avec du nouveau code spécifique aux autorisations :

- La partie bleue permet de gérer les emails et d'en renvoyer. C'est un formulaire commun à toutes les vues détails.
- La partie orange est la vue envoyée par courrier lorsqu'on génère automatiquement le mail.
- Enfin la partie violette (le bouton à droite de l'écran) permet de passer l'autorisation de l'état classique à l'état particulier et inversement pour mettre à jour la vue orange.

A noter que pour se souvenir si une autorisation a été saisie comme "particulière" ou "classique" lors de sa réalisation, un champ en base de données a été ajouté pour enregistrer l'information.

Chapitre 5

Interprétation des résultats

5.1 Discussion

Apollo SSC a décidé de mettre en place un processus Agile pour traiter l'ensemble de ces projets car les méthodes agiles possèdent un certain nombre d'avantages.

L'avantage le plus important des méthodes agiles est la mise au premier plan des individus et leurs interactions plutôt que les processus et les outils. Pour cela, on privilégie la collaboration avec le client, la communication au sein des équipes pour permettre une adaptation rapide. En favorisant des feed-back réguliers de la part de tous les acteurs du projet, la méthode agile se donne pour objectif de fournir une application de meilleures qualités et qui satisfait tous les parties : utilisateurs, clients, concepteurs et développeurs.

En six mois, j'ai eu l'occasion d'apprendre et de me familiariser avec cette méthode, et si les avantages me semblent aujourd'hui une évidence, ce sont des méthodes loin d'être facile à mettre en place.

Par exemple, sur le Rapport d'Activité, nous n'avions aucun cahier des charges précis. Ne connaissant pas très bien la première version de l'application, il m'est souvent arrivé de devoir reprendre une même fonctionnalité plusieurs fois de suite car il me manquait des règles que je n'avais donc pas respectées. J'ai eu plus d'une fois la sensation de travailler à l'aveugle à cause de l'absence de ce document.

Autre problème que j'ai perçu sur le projet Etude : l'ensemble des besoins du client étaient très mal exprimés par celui-ci. J'ai eu l'occasion de discuter plusieurs fois par téléphone avec lui en compagnie de mon chef de projet et le constat était à chaque fois le même : nous n'avions pas plus de réponses à nos questions à la fin de la conversation. C'est un point particulièrement difficile à améliorer car dans l'informatique, il arrive souvent que les utilisateurs eux même ne sachent pas exactement de quoi ils ont besoin.

Enfin, concernant les intranets d'Apollo, les applications sont difficilement maintenables à cause d'un code très peu lisible. Ces intranets sont en place depuis des années et possèdent des centaines et des centaines d'heures de travail derrière chaque ligne de code. Le problème pour ces intranets réside principalement dans le manque de documentation et de structure.

Tous ces problèmes soulèvent ainsi quelques questions : Peut-on développer une application sans spécifications ? Comment réaliser une application qui répond à un besoin si il est difficile (voire impossible) de déterminer ce besoin ? L'agilité n'est-elle pas censée donner naissance à des projets *sains* avec les pratiques qu'elle implique, si ce n'est pas le cas, comment améliorer cette qualité ?

5.2 Perspectives

Ces six mois m'ont permis de prendre un certain recul sur ces méthodes agiles de plus en plus présentes dans le monde de l'informatique. Et je pense que les problèmes auxquels j'ai été confronté ne sont pas insurmontable.

Par exemple, pour bien identifier les besoins du client, on aurait par exemple pu faire plus de réunions avec celui-ci (ou en tout cas communiquer plus) pour lui montrer des maquettes visuelles (lui proposer du choix ou bien laisser le client dessiner lui-même les maquettes). On aurait aussi pu réaliser un site statique pour voir exactement où le client clique sur l'application et ce qu'il s'attend à voir à tel ou tel endroit. Tout cela pour améliorer notre compréhension du besoin avant de l'implémenter dans l'application.

Autre exemple, pour améliorer la qualité du code d'une application, on pourrait mettre en place des séances de revues de code (au sens technique) sur ces projets pour identifier les axes d'améliorations. A plus court terme, on pourrait rédiger une documentation à destination des développeurs pour clairement définir la structure et l'existant des applications.

Tous ces axes de réflexions mon fait prendre conscience que l'agilité est une excellente méthode de développement pour produire des logiciels de qualité. Cependant il faut adapter la méthode au projet et à l'équipe et non l'inverse. Dans mon futur travail, je souhaite être capable d'adapter ces techniques et méthodologies à des projets différents les uns des autres. Pour cela, je devrais développer ma capacité à vulgariser mes propos techniques pour faciliter ma communication avec les clients. Je devrais également être capable d'analyser un projet dans son ensemble pour pouvoir mettre en place les meilleures solutions possibles.

Conclusion

Ainsi, j'ai effectué mon stage au sein d'Apollo SSC. Lors de ce stage de vingt-deux semaines, j'ai pu mettre en pratique mes connaissances théoriques acquises durant ma formation, de plus, je me suis confronté aux difficultés réelles du monde du travail et plus particulièrement aux méthodes de travail d'une ESN.

Après ma rapide intégration dans l'équipe, j'ai eu l'occasion de réaliser un ensemble de plusieurs missions variées autour de projets encore plus variés, et cela constitue la réussite de mon stage.

Chacune de ces missions, utiles au bon déroulement de l'activité de l'entreprise, se sont inscrites dans la stratégie de celle-ci : faciliter la gestion de l'information en interne tout en répondant à des missions pour les clients avec professionnalisme. Je garde du stage un excellent souvenir, il constitue désormais une expérience professionnelle valorisante mais surtout une expérience humaine plus qu'enrichissante.

Je peux affirmer que j'ai réussi une bonne partie des objectifs que l'on m'a fixés : le **Rapport d'Activité** est fonctionnel et utilisé actuellement ; Et le projet **Etude** a été livré quelques semaine avant la fin de mon stage.

Je pense que cette expérience dans une ESN jeune m'a offert une bonne préparation à mon insertion professionnelle. En effet, j'ai appris de nouvelles méthodes de travail, sans aucun doute différentes d'une entreprise de plus grande envergure. Je suis aujourd'hui consciente qu'un projet, ce n'est pas juste du code fonctionnel : c'est tout un ensemble. Cette expérience conforte mon désir d'exercer mon futur métier d'ingénieur dans le domaine de l'informatique.

Annexe A

Etude : Fonction generateAllDatas

```
/* Recupere et calcule toutes les donnees necessaires pour le mail et les
   vues */
2 public function generateAllDatas($id) {

4     /* Divers */
5     $data['qui'] = $this->session->userdata ( "logged_in" )['name'];
6     $data['Mois'] = array("", "Janvier", "Fevrier", "Mars", "Avril", "Mai",
7         "Juin", "Juillet", "Aout", "Septembre", "Octobre", "Novembre", "
8             Decembre");
9     $data['id'] = $id;
10    $data['modelTuile'] = '';
11    $data['p'] = '';
12    $data['email'] = NULL;
13    $data['att'] = NULL;
14    $data['coment'] = NULL;
15    $data['faxnumb'] = NULL;
16    $articleNeige = '';
17    $ecranTxt = '';
18    $etudes_txt = '';

19     /* Etude */
20    $etude = $this->etudeManager->getEtudeByIdWithPerson($id);
21    $data['mail'] = $etude->mail;
22    $data['Ref']=$etude->tuile;
23    $data['coloris']=$etude->coloris;
24    $data['date1']=$etude->date1;
25    $data['date2']=$etude->date2;
26    $data['date3']=$etude->date3;
27    $data['projet']=$etude->projet;
28    $data['projet3']=$etude->projet3;
29    $data['projet2']=$etude->projet2;
30    $data['projet4']=$etude->projet4;
31    $data['ad1']=$etude->ad1;
32    $data['ad2']=$etude->ad2;
33    $data['ad3']=$etude->ad3;
34    $data['com']=$etude->com;
35    $data['surface']=$etude->surface;
36    $data['mpm']= ' ' . $etude->mpm * 100;
37    $data['mpm2']=$etude->mpm2;
```

```

38     $data['alt'] = $etude->alt;
39     $data['lr'] = ' ' . $etude->lr;
40     $data['so'] = $etude->so;
41     $data['neige'] = $etude->neige;
42     $data['zone'] = $etude->zone == '4' ? '3 (' . CLIMAT_MONTAGNE . ')' : $etude->zone;
43
44     /* Specificites juridiques */
45     $pose = $this->caracteristiqueManager->getByRef($data['Ref']);
46     $dtu = $pose->CDTU;
47     $data['pose'] = strtolower(' ' . ($pose->CP == JOINTS_DROITS_OU_CROISES ? JOINTS_CROISES : $pose->CP));
48
49     /* Mise a jour de certaines variables */
50     switch($dtu) {
51         case '40-21':
52             [...]
53             break;
54
55         case '40-22':
56             [...]
57             break;
58
59         case '40-23':
60             [...]
61             break;
62
63         case '40-211':
64             [...]
65             break;
66
67         default:
68             $nfp = '';
69             $date_dtu = '';
70             $article47 = '';
71             $articleNeige = '';
72             $ecranTxt = '';
73     }
74
75     /* Tuile*/
76     $tuile = $this->tuileManager->getByRef($data['Ref']);
77
78     /* Ecran */
79     $data['ecran'] = $etude->ecran == NULL ? ' ' : '<strong>ECRAN DE SOUS TOITURE:</strong> ' . $etude->ecran . ' ';
80
81     /* Site */
82     switch($etude->site) {
83         case '1':
84             $data['site'] = SITE_PROTEGE;
85             break;
86         case '2':
87             $data['site'] = SITE_NORMAL;
88             break;

```

```

88     case '3':
89         $data['site'] = SITE_EXPOSE;
90         break;
91     default:
92         $data['site'] = $etude->site;
93     }
94
95     /* Aut */
96     switch($etude->aut) {
97
98         case FAIBLE_PENTE:
99             $data['modelTuile'] = $tuile->Mod . ' en ' . FAIBLE_PENTE;
100            $ecranTxt = 'souple '. $ecranTxt;
101            if($etude->etanche != 1 && $etude->acceptation == '1') {
102                $etudes_txt = $this->etudesTxtManager->getByType(
103                    FAIBLE_PENTE);
104            } else {
105                $etudes_txt = $this->etudesTxtManager->getByType(
106                    ETANCHEITE_FB);
107            }
108            break;
109
110        case PROJECTION:
111            if($etude->etanche != 1) {
112                $data['modelTuile'] = $tuile->Mod . ' en ' . PROJECTION .
113                    ' hors DTU';
114                $etudes_txt = $this->etudesTxtManager->getByType(
115                    PROJECTION);
116            } else {
117                $data['modelTuile'] = $tuile->Mod . ' sur un rampant
118                    \ud83c\udcfdunelongueur superieure a 12 metres en ' .
119                    PROJECTION . ' horizontale';
120                $etudes_txt = $this->etudesTxtManager->getByType(
121                    ETANCHEITE_PRO);
122            }
123            break;
124
125        case CLIMAT_MONTAGNE:
126            $data['modelTuile'] = $tuile->Mod . ' en ' . CLIMAT_MONTAGNE;
127            $etudes_txt = $this->etudesTxtManager->getByType(
128                CLIMAT_MONTAGNE);
129            break;
130
131        if(is_object($etudes_txt)) {
132            $textes = $this->textesManager->getByTxt($etudes_txt->id_textes);
133        } else {
134            $textes = array();
135        }
136
137        /* Action de remplacement de textes */
138        foreach($textes as $text) {
139            [...]
140        }

```

```

134
135     /* Date */
136     $data['d']= date('d', $data['date3']);
137     $data['m']=$data['Mois'][date('n', $data['date3'])];
138     $data['a']=date('Y', $data['date3']);

139
140     /* Contact & interlocuteurs */
141     if ($data['ad1'] != NULL){
142         //Poseur
143         $data['contactId'] = $data['ad1'];
144     }else if ($data['ad2'] != NULL){
145         //Negoce
146         $data['contactId'] = $data['ad2'];
147     }else if ($data['ad3'] != NULL){
148         //Prescripteur
149         $data['contactId'] = $data['ad3'];
150     }else {
151         $data['contactId'] = '';
152     }

153
154     $interlocuteurs = $this->interlocuteurManager->getByIdContact($data['
155             contactId']);
156     $contact = $this->contactManager->getById($data['contactId']);

157
158     if(isset($contact) && !empty($contact)) {
159         $data['ad_1']=$contact->nom;
160         $data['ad_2']=$contact->type;
161         $data['ad_3']=$contact->adresse;
162         $data['ad_4']=$contact->cp;
163         $data['ad_5']=$contact->ville;
164         $data['ad_6']=$contact->mail;
165         $data['ad_7']=$contact->tel;
166         $data['ad_8']=$contact->fax;
167         $data['ad_9']=$contact->dpt;
168     }

169
170     $data['listInterlocuteurs'] = array();
171     foreach($interlocuteurs as $i) {
172         $data['listInterlocuteurs'][$i->id] = $i->prenom . " " . $i->nom .
173             " - " . $i->mail;
174     }
175     return $data;
176 }
```

Références

- **Ensemble de forums** [consultation pendant tout le projet] :

<http://www.commentcamarche.net/forum/>
<http://stackoverflow.com/>
<http://openclassrooms.com/forum/>

Sites associatifs : Ces forums m'ont servi pour résoudre un certain nombre de problèmes rencontrés.

- **Yammer** :

<https://www.yammer.com>

Sites de participatif : site internet permettant de communiquer au sein de l'entreprise. Un outil très intéressant pour communiquer avec l'ensemble des collaborateurs, y compris ceux qui sont en déplacement chez un client. Je m'en suis beaucoup servi une fois le Rapport d'Activité mis en place pour gérer les premiers retours.

- **Scotch.io** :

<https://scotch.io>

Blog : site internet qui propose beaucoup de tutoriels concernant AngularJS mais également d'autres langages informatiques. Je m'en suis beaucoup servi notamment leur tutoriel sur Karma et Jasmine.

Résumé

Apollo SSC est une jeune ESN Agile qui possède un certain nombre d'outils internes pour faciliter son processus métier. Mais les développeurs de l'entreprise sont également là pour s'occuper de projets concrets pour des clients. Ils jonglent ainsi entre développements internes et externes au gré des projets. J'ai pris part à la vie de cette ESN en participant au développement de moult projets développés en PHP, Javascript ou encore c# avec des méthodes Agiles.

Mots Clés

AngularJS, ESN, Karma & Jasmine, Méthodes Agiles, MVC, .Net

Apollo SSC, young IT company (IT Services), owns few internal applications to ease its business process. But Apollo's programmers are also hired to take care of customer's projects. They may switch from internal to external project. I participated actively in the company process by contributing to a variety applications using PHP, Javascript or c# with agile methods.

Key Words

Agile Software Development, AngularJS, IT Service, Karma & Jasmine, MVC, .Net