Mémoire de fin d'étude

Ingénieur Informatique
spécialité Systèmes et Réseaux

Conception et réalisation d'un outil de validation d'équipements CWMP

Alternant : Alexis BATTAGLI

*Maitre d'apprentissage :*Marc DOUET

Tuteur académique : Yan MORET

École : IMT Mines Alès

Entreprise : Orange

Promotion : INFRES 7

Septembre 2014 — Septembre 2017

Table des matières

Table des figures

Liste des tableaux

Acronymes

ACS Auto Configuratione Server. *Glossary :* ACS, 11, 12, 13, 15, 16, 17, 18, 19, 21, 22, 24, 26, 31, 70, 71

ADSL Asymmetric Digital Subscriber Line. *Glossary :* ADSL, 15

API Application Programming Interface. *Glossary :* API, 46

BBF BroadBandForum. *Glossary :* BBF, 12, 20, 71

CARE Cloud enablers for Administration of Residential Equipement. *Glossary :* CARE, 10, 11, 12, 13, 15

CPE Customer Premise Equipment. *Glossary :* CPE, 11, 12, 13, 17, 18, 20, 39, 40

CWMP CPE Wide Area Network (WAN) Management Protocol. *Glossary :* CWMP, 12, 13, 14, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 28, 29, 30, 31, 32, 36, 37, 38, 39, 40, 42, 43, 44, 46, 48, 49, 50, 51, 52, 63, 71

DHCP Dynamic Host Configuration Protocol. *Glossary :* DHCP, 15, 19, 29

DNS Domain Name Service. *Glossary :* DNS, 15, 17, 19, 23, 29

DSLAM Digital Subscriber Line Access Multiplexer. *Glossary :* DSLAM, 15, 16, 19

FAI Fournisseur Accès Internet. 10

FTTH Fiber To The Home. 9

HTTP HyperText Transfer Protocol. 13

HTTPS HyperText Transfer Protocol Secure. 16

IHM Interface Homme Machine. *Glossary :* IHM, 14, 33, 34, 46, 48, 49, 50

IP Internet Protocol. 11, 15, 17, 34

LAN Local Area Network. *Glossary :* LAN, 11, 15

MVC Model View Controller. 59

OLPS Orange Labs Products and Services. 10

OLS Orange Labs Services. 10, 23

PaaS Platform-as-a-Service. 33, 34

PoC Prove of Concept. 2, 49

SGBD Système de Gestion de Base de Données. *Glossary :* SGBD, 61, 62

SOAP Simple Object Access Protocol. 13

SSL Secure Sockets Layer. 19

TINK TINK Is Not Karma. *Glossary :* TINK, 2, 4, 31, 32, 33, 35, 36, 38, 39, 44, 46, 47, 48, 49, 50, 51, 52, 55, 57, 59, 61, 62

URL Uniform Resource Locator. 34, 52

WAN Wide Area Network. 6, 12

Remerciments

1 Introduction

1.1 L'entreprise

Orange est à l'origine une entreprise anglaise de télécommunication. Elle a été rachetée par France Télécom en 2000, entreprise française fondée en 1975, devenant par la suite de ce rachat une société internationale. Au 1er juillet 2013, France Télécom change de nom est devient Orange, société française qui est alors la 121ème entreprise mondiale avec un chiffre d'affaire de 41 milliards d'euros fin 2016. Actuellement, Orange emploie 155 000 personnes mondialement, dont 96 000 en France et possède plus de 263 millions de clients dans le monde répartis dans 29 pays dont 11 pays d'Europe. (Voir carte ci-dessous)

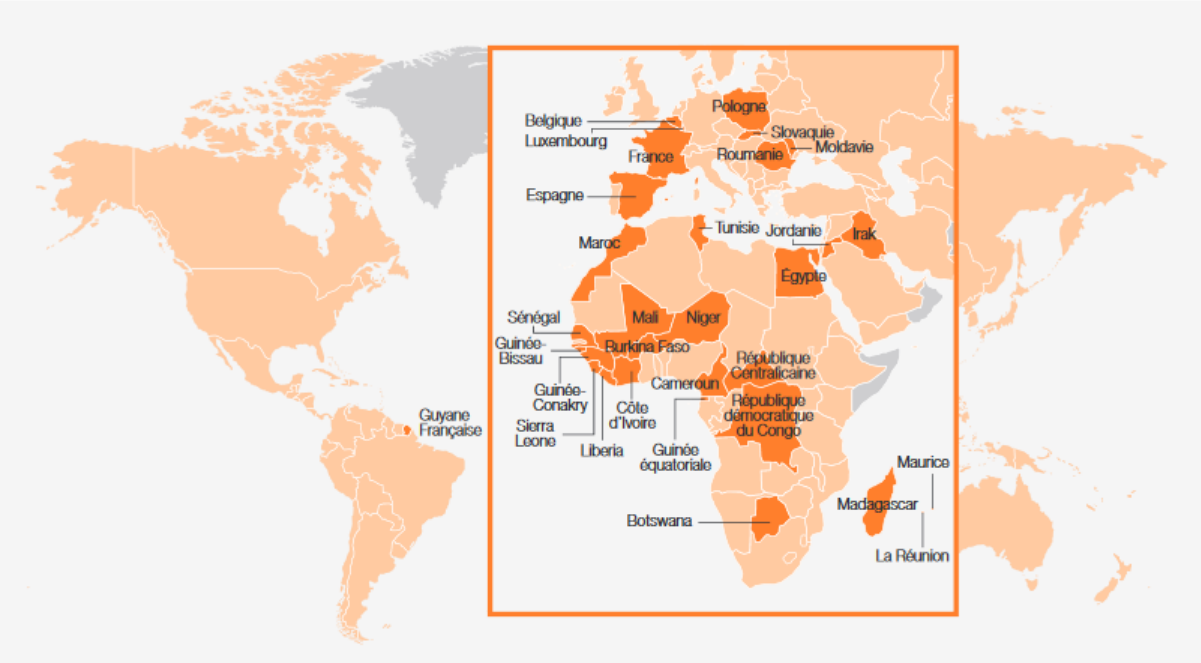

FIGURE 1 – Carte des pays où est présent Orange en 2016

Le groupe Orange est majoritairement présent en Europe et Afrique. Il est avant tout un leader de la téléphonie mobile avec un total de 202 millions de clients mobile en 2016 au niveau mondial. Orange est aussi leader dans le domaine de l'accès à Internet avec 18 millions de clients Internet haut débit fin 2016, 265 000 clients Fiber To The Home (FTTH) et 42 millions de clients sur la téléphonie fixe fin 2014 en France. Les pays où le groupe est le plus implanté sont la France, l'Espagne, la Pologne et la Roumanie. Depuis plusieurs années maintenant Orange essaie de se développer également en Afrique dans le

domaine de la téléphonie mobile.

Le secteur d'activité principal du groupe Orange reste les Télécommunications, en étant un opérateur téléphonique majeur en France et dans bien d'autres pays tels que la Pologne, l'Espagne, la Roumanie, Côte d'Ivoire, Égypte etc. Orange est également un fournisseur d'accès Internet et depuis quelques années élargit ses activités à la domotique, vente de contenus cinématographique et musical, médical, applications bancaires et automobiles etc.

Les principaux concurrents d'Orange en France dans le domaine Fournisseur Accès Internet (FAI) sont principalement Free, Numéricâble, OVH, Nerim, Wifirst et Bouygues Télécoms. Et pour la téléphonie mobile ses principaux concurrents sont SFR, Free et Bouygues Télécom. Tandis qu'au niveau européen sur le domaine téléphonique et FAI, les principaux concurrents sont Deutsche Telekom, Vodafone et O2 en grande majorité.

La branche où j'effectue mon alternance depuis 3 ans est Orange Labs Services (OLS). Cette branche concerne tous ce qui touche à la recherche et au développement des produits Orange. Anciennement nommé France Télécom R&D, puis Orange Labs Products and Services (OLPS) en 2007, et enfin rebaptisé OLS en 2017. Cette branche destinée à la recherche de l'ensemble du groupe Orange emploie 3500 personnes exclusivement en France. Fin 2012, le nombre de brevets déposés par Orange Labs s'élevaient à 7493. La R&D est très importante pour Orange qui investit chaque année près de 900 millions d'euros dans ce secteur.

1.2 Le contexte

1.2.1 Le Device Mangement à Orange

Mon alternance se déroule plus précisément au sein de l'équipe Cloud enablers for Administration of Residential Equipement (CARE) qui s'occupe de la gestion des équipements client, c'est-à-dire du « Device Management ».

Le concept de « Device Management » possède plusieurs définitions selon les objets ou équipements gérés, et les équipes qui le mettent en place. Pour nous, le Device Manage-

ment s'articule autour de quatre axes :

— Provsioning : Active ou désactive un service pour le client sur l'équipement adéquate ; Applique le bon Firmware selon le service souscrit ; Paramètre de manière personnalisé la configuration d'un équipement donné en fonction des services.
— Assistance : Permet de diagnostiquer à distance un incident sur l'équipement ; Déclencher à distance l'exécution d'action permettant de corriger un incident.
— Tracking : Collecte et stocke des informations sur l'ensemble du parc client.
— Maintenance : Permet la mise à jour de Firmware à différent temps souhaité.

L'architecture du Device Management est découpé en deux zones détaillées comme suis :

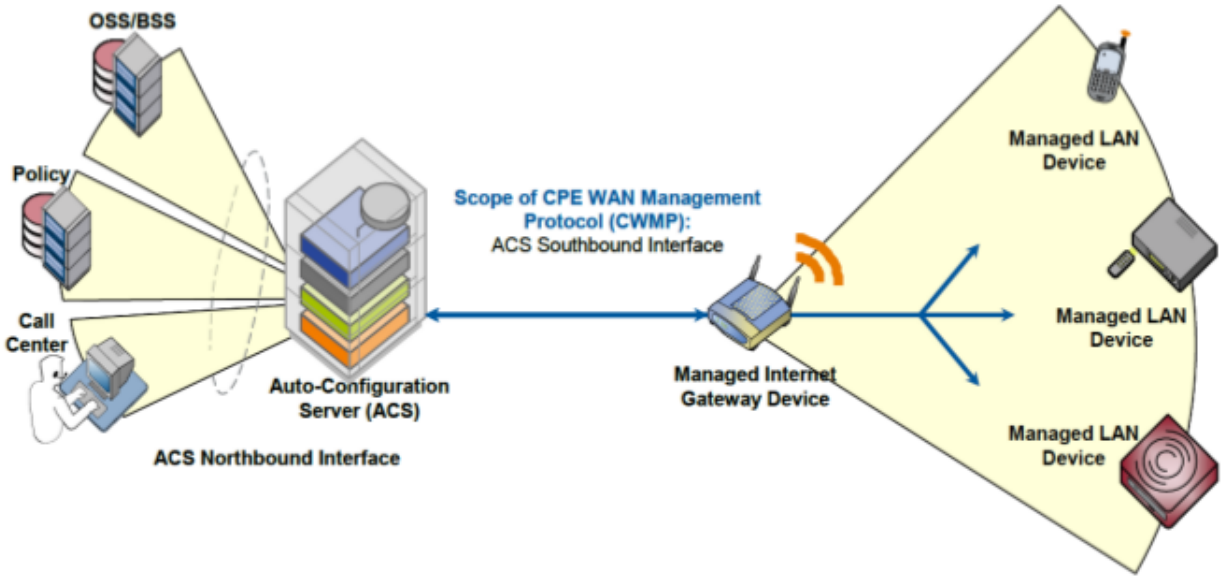

FIGURE 2 – Réseau de Device Management, côté Opérateur et côté Client

— Le coté client, où l'on retrouve le réseau privé du client, dit le Local Area Network (LAN), avec généralement divers équipements, que l'on nomme Customer Premise Equipment (CPE), tels que, une passerelle Internet, un décodeur TV, un téléphone, une caméra Internet Protocol (IP), des capteurs domotiques etc.
— Le coté serveur, se trouvant chez Orange, où l'on va retrouver les serveurs, que l'on nomme Auto Configuratione Server (ACS), qui vont permettre de faire ce que l'on nomme du Device Management.

L'un des objectifs du Device Management, pour l'équipe CARE, est d'apporter un service d'aide et de dépannage aux clients, tous en restant à distance. Dans le but de ne pas avoir à faire déplacer un technicien sur place, pour un problème qui peut être résolu à distance par l'exécution de scripts, lancement de test et analyse, ou encore correction de bug. Le rôle de l'équipe CARE, est de concevoir l'intégration de ces outils qui pourront être utilisés à distance.

La supervision et la maintenance du parc Orange sont d'autres activités dans le périmètre de l'activité du Device Management. Ce parc contient les différents produits vendus par Orange et qu'Orange s'engage à maintenir. On comprend alors l'importance des activités de supervision et de maintenance. Pour gérer ce parc, Orange a besoin, entre autres, d'identifier les différents équipements présents et d'accéder à leurs caractéristiques. Les outils de Device Management développés au sein de l'équipe CARE permettent, cette fois, de remonter aux ACS toutes les informations nécessaires pour superviser et maintenir le parc. Il permet également de mettre à jour et corriger des bugs en envoyant de nouvelles versions de Firmware aux équipements concernés.

1.2.2 Document TR-069 et protocole normalisé CWMP

Avant de continuer, il est important de préciser quelques élèments de vocabulaire propre au Device Management et décrire plus en détail comment ACS et CPE communiquent.

Comme nous l'avons vu précédemment, un ACS est un serveur qui permet de manager un CPE, et par conséquent de réaliser du Device Management sur le parc client.

Les interactions entre ACS et CPE sont standardisées par le Document TR-069. Ce document permet, entre autre, de décrire comment implémenter la norme TR-069 sous la forme du protocole CPE WAN Management Protocol (CWMP), tant sur les ACS que sur les CPE.

Ce Document TR-069 est le résultat d'un consortium de plusieurs industriels. Ce consortium, appelé le BroadBandForum (BBF), ce compose de plus d'une centaine de membres dont Orange, CISCO, Deutch Telecom, Huawei, Juniper, le gouvernement du Canada,

Intel et bien d'autre. Le BBF vise à décrire la gestion des équipements clients, dit CPE, par les serveurs de gestion d'équipements, dit ACS. C'est par le Document TR-069 que le BBF décrit un standard permettant la communication entre CPE et ACS pour une bonne gestion des équipements clients.

Ce standard décrit un modèle de donnée, que l'on nomme Data Model, comportant une partie commune pour chaque équipement implémentant le Document TR-069. Le management d'un équipement par un ACS repose en partie sur la présence de ce Data Model. Le standard décrit également différentes méthodes, que l'on nomme RPC Methode, obligatoire ou facultative, qui doivent être implémentées soit par l'ACS soit par le CPE. Ces RPC Méthodes [1] vont permettre la communication entre ACS et CPE, et ainsi rendre possible à l'ACS le managment de ses CPE.

Plus précisément, un CPE, afin de pouvoir échanger avec un ACS, doit implémenter le Document TR-069 sous la forme d'un client CWMP. Les échanges CWMP sont transportés sur du HyperText Transfer Protocol (HTTP) et encapsulés dans des messages Simple Object Access Protocol (SOAP). La création d'une session CWMP ce fait toujours par le CPE. L'ACS ne peut pas créer de session, en revanche il possède différentes techniques [2] lui permettant de demander au CPE de venir initialiser une session CWMP.

1.3 Objectifs

Au cours des trois années passées dans l'équipe CARE, il m'a été confié différents objectifs d'importance et de responsabilité croissante. Ces objectifs m'ont permis de monter en compétences tant sur l'aspect technique que l'aspect transversal du métier d'ingénieur informatique.

1.3.1 Première année

À mon arrivée le principal objectif été de me faire monter en compétence sur le domaine du device management, et plus particulièrement sur le protocole CWMP. Il a donc été

1. Les principales RPC Methodes des ACS et CPE sont visibles en annexe.
2. Ces techniques feront l'objet de parties présentées plus loin dans le document.

décidé de me faire tout d'abord monter en compétence sur le côté client, puis sur le côté serveur, par différentes missions que nous verrons par la suite [3].

1.3.2 Deuxième année

En début de deuxième année, ayant acquis les connaissances nécessaires lors de la première année, l'objectif était de me faire développer une application permettant de réaliser une série de test de client CWMP de manière automatique [4].

Un autre objectif de cette deuxième année a été d'encadrer un stagiaire de Master 2 Informatiques, Jean-Pierre ONA, qui devait travailler avec moi sur ce projet. Jean-Pierre a été présent de mars à aout 2016, son sujet de stage portait sur la conception et le développement d'une Interface Homme Machine (IHM) pour l'application de mon projet.

Enfin, l'un des objectifs de cette deuxième année était de trouver une personne pour prendre la suite du projet de test d'équipements. Pour ce faire je devais mener une activité de tutorat afin de faire monter en compétence une personne de l'équipe sur ce projet.

1.3.3 Troisième année

Pour la troisième année l'objectif était avant tous de continuer le développement et l'ajout de fonctionnalité à l'application réaliser l'année précédente, mais également de la mettre en production et d'en assurer l'aide aux utilisateurs.

Un autre des objectifs était de prévoir mon départ en réalisant un transfert de compétence sur l'outil développer. Cela devait se faire par la rédaction de différentes documentation permettant aux futurs développeurs de continuer mon travail, ainsi que la présentation de l'outil aux équipes susceptibles de récupérer l'outil.

3. La partie intitulée "Monté en compétence sur le protocole CWMP" est entièrement dédié à cet apprentissage du domaine du device management et du protocole CWMP

4. La partie intitulée "Conception et développement de TINK" est entièrement dédiée à la réalisation de cet outil.

2 Monté en compétence sur le protocole CWMP

2.1 Études d'équipements

Ce projet s'est déroulé tous au long de la première année. La personne avec qui je travaillais sur ce projet, Serge MARTIN, étant partie à la retraite en octobre 2015, et ayant dû me consacrer moi-même à d'autre missions, ce projet a été progressivement arrêté durant ma deuxième année. L'un des objectifs de ma deuxième année avait été de continuer ce projet le temps que l'on trouve une autre personne à former pour prendre la suite. Malheureusement, cet objectif n'a été que partiellement réalisé, faute de disponibilité de la part de la personne devant reprendre le projet, menant ainsi à son arrêt.

Le projet consiste à faire différents tests sur des équipements Orange que l'on peut retrouver côté clients. Ces tests peuvent être demandés par d'autres équipes ou bien des membres même de CARE. Ces demandes sont faites lorsque l'ont a besoins de savoir comment fonctionne certains équipements Orange au niveau CWMP, connaître leurs comportements réseau dans certaines situations, ou encore récupérer une partie voir l'intégralité de leur Data Model. Le domaine de ces tests est très diversifié et peut prendre quelques heures, comme plusieurs semaines.

2.1.1 Présentation du réseau isolé

Afin de reproduire au mieux les conditions réels, avoir une visibilité intégrale sur les échanges ayant lieu, et un contrôle à tous les niveaux du réseau opérateur, nous avons décidé de reproduire un réseau opérateur que l'on nomme « réseau isolé ».

Ce réseau est constitué d'un micro Digital Subscriber Line Access Multiplexer (DSLAM) faisant la jonction entre la partie opérateur Orange et les clients Orange. Il permet de gérer les « clients » en leur attribuant une adresse IP publique via un service de Dynamic Host Configuration Protocol (DHCP) et en créant des lignes Asymmetric Digital Subscriber Line (ADSL) avec login et mot de passe attribués.

Du cotés clients, on retrouve les différents types de box internet auxquels sont connectés les équipements clients, dans leur réseau LAN. Du coté opérateur, nous avons mis en

place un serveur DHCP pour attribuer des adresses IP aux machines du réseau opérateur. Un serveur Domain Name Service (DNS) est également présent pour la résolution des noms. Ce serveur DNS est aussi utilisé par les box internet des clients comme serveur DNS primaire. Nous avons aussi deux serveurs ACS. Un premier du même type que celui utilisé par Orange Labs pour les tests et recherches. Et un second, sous forme de Servlet Java [5] qui possède moins de fonctionnalités que le premier, mais qui est très utilisé pour les tests de comportement CWMP [6]. Vous pouvez voir ci-dessous le plan du réseau isolé que nous utilisons et qui est décrit ci-dessus.

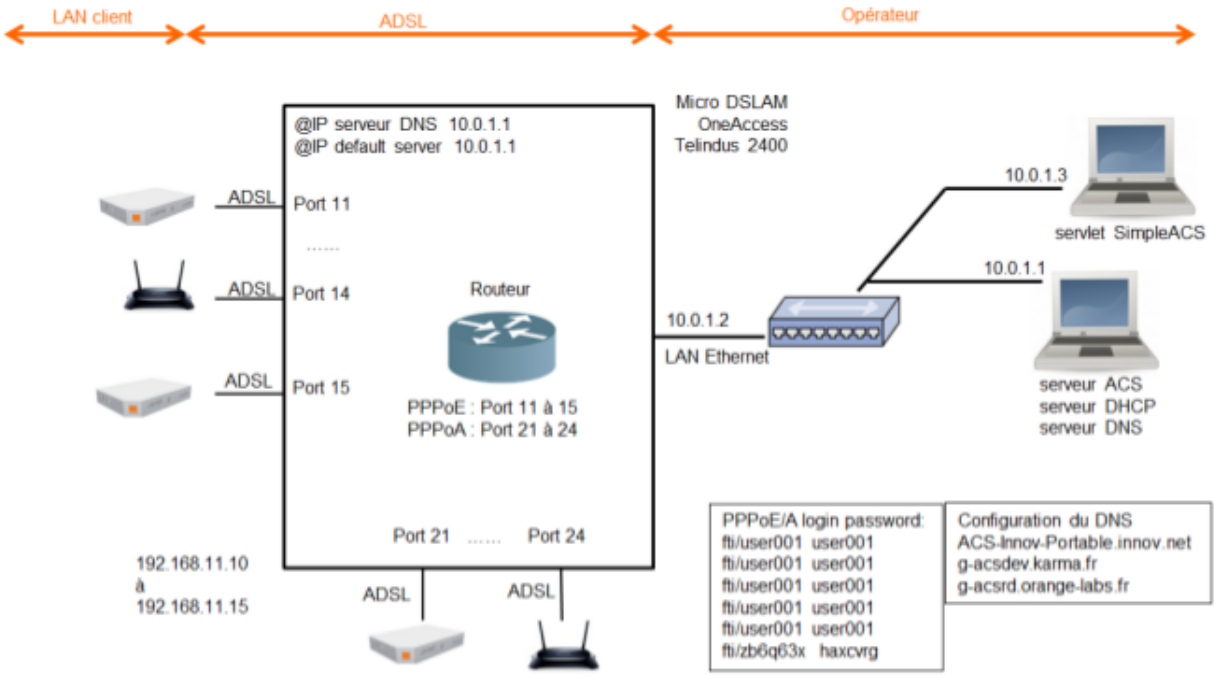

FIGURE 3 – Schéma du réseau isolé

Nous ne rentrerons pas plus en détail sur ce réseau ici. Il faut également savoir que nous avons placé un concentrateur du côté opérateur, entre les serveurs et le micro DSLAM dans le but de pouvoir analyser le trafic réseau en plaçant un sniffeur réseau tel que Wireshark.

5. La servlet Java fait l'objet d'une sous-partie intitulé "Création d'un ACS Servlet", puisque j'ai également eu à travailler dessus durant ma première année

6. Les test de comportement font l'objet de la partie intitulé "Test de comportement CWMP d'équipement"

Il arrive parfois que certains équipements dialoguent via HyperText Transfer Protocol Secure (HTTPS) avec le serveur ACS. Ce type de protocole sécurisé nous empêche de faire certains tests liés au protocole CWMP, puisque nous n'avons pas les certificats nécessaires au déchiffrage des trames HTTPS.

2.1.2 Test DNS

Au cours de la première année, une équipe a eu besoin de connaître le comportement des équipements Orange vis-à-vis de leur serveur DNS. Nous avons donc utilisé le réseau isolé afin de réaliser différents tests sur les équipements en questions.

Plus exactement, cette équipe devait migrer des serveurs ACS. Les CPE, selon leur client CWMP envoie des requêtes à leur serveur DNS respectif afin de connaître leur ACS. Les membres de cette équipe voulaient donc connaître l'impact, en terme de nombre de requêtes DNS, que cela aller avoir sur les serveurs DNS des équipements liés à ces serveurs ACS. Le but premier était de s'assurer que le nombre de requêtes faites par les équipements, pour connaître la nouvelle adresse IP de leur ACS, n'allait pas faire tomber les serveurs DNS qui sont sollicités par ces équipements.

Nous nous sommes demandés dans un premier temps, quand est ce que doivent apparaître ces requêtes DNS. Pour ce faire, nous avons regardé ce qui est préconisé par le Document TR-069, puisqu'en toute logique, les équipements qui communiquent et sont gérés par un serveur ACS, doivent respecter cette norme le plus fidèlement possible. A partir de la norme, nous avons isolé plusieurs cas d'usage où un équipement se doit de contacter son ACS, et donc avant cela, faire un appel DNS pour connaître son adresse IP.

Nous avons ensuite mis en place chaque équipement, dans notre réseau isolé, et les avons redirigés vers le serveur ACS et le serveur DNS appropriés pour procéder aux tests et à la vérification du respect des cas d'usages.

Il y avait donc deux objectifs. Le premier, était de vérifier si les équipements respectent bien le Document TR-069. Tandis que le second objectif était de contrôler et relever le nombre de requêtes faites par les équipements vers leur serveur DNS.

Après avoir testé les différents cas d'usage, sur tous les équipements, il apparaît que certains équipements ne respectent pas le Document TR-069 en termes d'appel DNS. Certains cas d'usage sont tous simplement ignorés par l'équipement, ce qui représente une grave erreur d'implémentation du Document TR-069 sur ces équipements. Afin que ces erreurs soient corrigées, nous avons remonté ces anomalies au service qui veille au respect du Document TR-069 et qui font les correctifs des bugs sur les équipements. Nous avons également pu donner des éléments de réponse à la première question par le biais d'un rapport des tests effectués et de présentations aux cours de réunions d'avancement. Au vu des données que nous leur avons remonté, ils ont pu déduire que l'impact du nombre de requête envoyées aux serveurs DNS ne générerait pas une surcharge des serveurs.

2.1.3 Test de comportement CWMP d'équipement

Les tests CWMP sont des tests qui nous sont demandés par notre équipe de manière irrégulière selon les besoins de l'équipe. Dans l'équipe nous utilisons, aux cours de projet, de nombreux équipements Orange. Le but de ces tests est de connaître les caractéristiques CWMP de ces équipements que nous utilisons. Mais aussi être capables de savoir comment ils réagissent dans différentes situations et savoir comment leur client CWMP implémente le Document TR-069. Bien entendu, nous ne testons ici que des équipements embarquant un client CWMP capable de communiquer avec son ACS.

A chaque fois que l'équipe récupère un nouveau device susceptible de pouvoir faire du Device Mangement, il doit faire l'objet de ces tests CWMP. L'ensemble des résultats de ces tests sont ensuite documentés et archivés afin de pouvoir être réutilisés si besoins. Ces demandes peuvent permettre d'anticiper la venu de nouveaux équipements dans le parc client d'Orange, elles permettent de savoir si nous auront des difficultés à manager ces nouveaux équipements avec la version actuelle de Karma.

Le Document TR-069 est extrêmement long et décrit l'ensemble des régles qu'un CPE ce doit d'implémenter, avec plus ou moins de rigueur dans certain cas. Pour tester les équipements que nous recevons, il a été décidé de regarder le respect de 4 points qui sont les plus importants pour les travaux fait dans l'équipe. Ces 4 points [7] ont été sélectionné

7. Nous ne rentrerons pas dans le détail technique de ces quatre tests

pour être testé systématiquement car ce sont ceux qui sont le plus souvent demandé par les membres de l'équipe. Un cinquième point a été rajouté récemment avec la création et l'ajout dans le réseau isolé de l'ACSServlet [8]. Ce cinquième test que j'ai pu rajouter consiste à extraire l'intégralité du datamodelg d'un équipement. L'extraction du datamodelg d'un équipement fait souvent l'objet de demande externe, et fait appel à des fonctionnalités CWMP qui sont intéressantes à être testées pour chaque équipement.

Pour procéder aux tests, il faut avant tout pouvoir placer les équipements dans notre réseau isolé, ce qui nécessite parfois son amélioration. Les améliorations et modifications peuvent être faites à plusieurs niveaux. Cela peut être une modification de la configuration du micro DSLAM, ajouter de nouveaux types de connections [9] au micro DSLAM, rajouter des plages d'adresses sur le serveur DHCP, ou bien encore créer de nouveaux domaines dans la configuration du serveur DNS.

Cependant, il arrive que l'ajout de l'équipement dans notre réseau isolé ne soit pas possible. Soit à cause de critère physique, comme on a pu le rencontrer avec des dispositifs 4G Huawei, auquel cas nous ne pouvons pas procéder à certains tests, voir à l'intégralité des tests. Soit, à cause de contraintes techniques, qui nous empêche de pouvoir faire le moindre test. Cela arrive lorsque l'équipement utilise un certificat Secure Sockets Layer (SSL), que nous ne possédons pas, dans ce cas aucuns des tests ne sont possibles.

De manière générale, les 4 principaux tests sont faits à partir de scripts JavaScript exécutés depuis le serveur ACS [10] vers lequel remonte les équipements dans le réseau isolé. Tant dit que le cinquième test est fait en redirigeant l'équipement vers l'ACSServlet. Lorsque nous rencontrons un problème avec un équipement, empêchant son installation dans le réseau isolé, nous faisons les tests sur le réseau public. Mais cela rend les tests plus longs et compliqués. Il faut parfois trouver de nouvelles façons de faire et cela ne suffit pas toujours rendant parfois des tests impossibles.

8. Voir la partie intitulé "Création d'un ACS Servlet"

9. Les équipements que nous avons peuvent communiquer soit par PPPoE soit par PPPoA, cela nécessite la configuration de port dédiés à ces deux types de connections.

10. C'est ce serveur même qui est utilisé pour la recherche et l'anticipation.

Ce projet s'est déroulé lors de ma première année. L'étude d'équipements aussi bien comportement CWMP que DNS, la mise en place du réseau isolé et la restitution sous forme oral et écrite des rapports de test, m'auront permis d'acquérir de solides bases sur le protocole CWMP, et une excellente entrée en matière dans le Device Management. Cela m'aura également permis de renforcer mes compétences en Système et Réseaux, particulièrement sur la configuration de serveur DNS et DHCP Mais aussi dans tous ce qui est protocole de communication et gestion d'OS. Enfin, cela m'aura permis de renforcer mes compétences en rédaction et présentation oral.

2.2 Étude de client CWMP

Afin de rentrer plus en profondeur sur les spécificités d'implémentation du Document TR-069 pour un CPE, j'ai étudié lors de ma première année deux clients CWMP. L'étude de ces clients CWMP avait pour objectif de détermine quel pouvait être les clients susceptibles d'être réutilisés par des équipements du parc Orange, afin de faciliter leur management par Karma.

2.2.1 Client EasyCWMP

L'objectif était de voir dans un premier temps si dans le domaine de l'open source il existe des clients CWMP qui ont été développé. Puis, après avoir listé les différents clients CWMP open source, je devais récupérer et tester ceux qui semblaient les plus prometteurs et qui implémentent le plus fidèlement le Document TR-069.

Afin de correctement comparer les différents clients open source, et vérifier qu'ils respectaient bien le Document TR-069, j'ai lu l'intégralité du Document TR-069 écrite par le BBF. Cela m'a permis de monter rapidement en compétence sur le sujet. De plus, la norme étant rédigée en anglais cela m'a également permis de monter en compétence sur l'anglais techniques.

La comparaison entre les clients open source trouvés ne sait pas faite uniquement sur le respect de l'implémentation du Document TR-069. Le logiciel devait être régulièrement

tenu à jour, et le projet open source devait être encore "vivant", c'est à dire que la communauté derrière le développement du client devait être active. Il a donc fallu s'intéresser plus en détail sur la provenance des clients, vérifier qu'une documentation était présente et à jour, ainsi que regarder la date des dernières versions des clients sélectionnés.

A la suite de cette comparaison théorique des différents clients CWMP que j'ai pu identifier, il a été décidé de n'en récupérer qu'un seul, nommer EasyCWMP et développé en C par l'entreprise PIVA Software. Seul le code source est disponible sur leur site. Les documentations d'installations et de spécifications n'étaient pas accessibles librement. Pour les récupérer il faut les commander, et donc les payer à PIVA Software, mais nous n'avons pas juger cela comme étant un frein.

Nous nous sommes intéressés à ce client CWMP open source car il est le seul à implémenter toutes les méthodes dites obligatoires du Document TR-069, contrairement aux autres clients open source rencontrés qui ne les implémentes jamais toutes. De plus, nous avons pu voir qu'il était très régulièrement mis à jour. Bien entendu cela était purement théorique à ce moment puisque nous n'avions pas encore pu tester le client en question.

Après avoir récupéré le code du logiciel, l'objectif était de l'installer sur un PC afin de le faire remonter vers l'ACS utilisé pour la recherche et l'anticipaition, que l'on nomme g-acsdev.

L'une des difficultés a été de l'installer sans avoir de documentation, d'autant plus que c'était la première fois que je devais réaliser ce type d'installation en compilant le programme et en installant toutes les libraires dépendantes entre elles. Cela m'a permis de monter en compétence en C puisque je devais comprendre comment compiler le client C avec ses différentes librairies. Puis comprendre comment, lancer le client, modifier sa configuration, utiliser les différentes méthodes CWMP qu'il implémente pour vérifier leurs bon fonctionnement, et enfin le faire communiquer avec g-acsdev. Toutes ces étapes de compilation, configuration et test, ce sont faites sur une machine linux. Une partie du code du client était en Shell, ce qui m'a permis de monter en compétence en Unix également. Mon tuteur m'a expliqué de nombreux aspects de C et de Shell afin de comprendre au mieux le code du client.

Après avoir réussi à faire communiquer parfaitement le client EasyCWMP avec g-acsdev et avoir testé les méthodes CWMP qu'il implémente, il m'a été demandé de modifier le code du client. L'objectif était de rajouter des paramètres dans son Data Model et de voir ces modifications apparaître dans l'interface web de g-acsdev lorsque l'on explorerait son Data Model. Cela m'a permis de rentrer plus en détail dans le code du client, puisque rajouter des branches au Data Model implique de modifier le code du client à différents niveaux. Tant au niveau C que Shell, j'ai dû comprendre comment le logiciel était codé, la relation entre les différents fichiers et appels de librairie. Ce qui m'a également permit de monter de nouveau en compétence en C mais également d'apprendre à comprendre un code que l'on récupère, le tout sans aucune documentation.

Cette activité m'aura donc apporté énormément dans plusieurs domaines techniques tels que le C, le Shell et renforcer ma monter en compétence sur le protocole CWMP. Elle m'aura également permis de voir comment se déroule une activité de bout en bout de manière totalement autonome. A la fin de l'activité il m'a également était demandé de créer une documentation sur l'installation des différents prérequis et du client EasyCWMP même, ce qui m'a permis d'apprendre à rédiger une documentation. De plus, le fait d'avoir fait des recherches dans le domaine de l'open source, m'a aidé à mieux comprendre l'open source avec ces différentes normes et licences, les projets collaboratifs etc. Enfin, cette activité m'aura permis de véritablement rentrer dans le device management et de mieux comprendre le côté client.

2.2.2 Comparaison EasyCWMP et tr69agent

Actuellement tous les constructeurs d'équipement n'implémentent pas systématiquement un client CWMP sur leurs équipements. Cela rend impossible le device management lorsqu'un client achète un équipement chez un de ces constructeurs. Or, Orange s'engage à pouvoir manager l'ensemble des équipements connectés à la LiveBox du client. Par conséquent, nous avons décidé de fournir ce que l'on appelle un « toolkit CWMP », qui aurait pour but d'aider les constructeurs qui le souhaitent, à implémenter sur leurs équipements un client CWMP. Ce client serait certifié par Orange comme étant capable de communiquer avec nos serveurs ACS, et donc être manager. La question qui s'est donc

posé, était de savoir ce que contiendrait le toolkit proposé par Orange.

Nous avons dégagé quatre propositions de contenu du toolkit. Ces quatre propositions de contenue du toolkit ne pourront pas être décrites ici. Afin de départager ces différentes possibilités de contenu nous devions apporter des éléments de comparaison. Par ailleurs, ce n'est pas nous qui avons rendu la décision final, mais les responsables de HomeService [11] qui à partir de cette étude ont pu rendre une décision sur la stratégie à adopter.

Ces propositions ont amené à comparer le client EasyCWMP au client nommé tr69agent. Ce client CWMP a été développé par Orange. Bien que nous possédions l'intégralité de la documentation et des droits liés à ce logiciel, nous devions vérifier s'il était plus avantageux que le client proposé par PIVA Software. Mon rôle dans cette étude a donc été de comparer ces deux clients et ainsi déterminer quels sont les points faibles et points forts de chacun de ces deux clients.

Pour ce faire je leur ai fait passer les tests de comportement CWMP et tests DNS. Les deux clients ont été installés dans des environnements identiques afin de ne pas perturber les résultats. L'analyse de ces tests n'a pas permis de départager les deux clients. En revanches, aucun des deux ne les satisfaisaient complétement, ce qui impliquait que peu import le client qui serait choisi, il faudrait le retravailler afin de le rendre parfaitement en adéquation avec le Document TR-069.

Je suis donc rentré plus en détails dans la comparaison, et j'ai regardé du côté de leurs caractéristiques. Tels que, le poids du logiciel après installation, le langage de programmation utilisé, la clarté de leur code, et les méthodes CWMP obligatoires et facultatives implémentées.

En dehors de cet aspect technique de comparaison des deux clients, j'ai pu voir comment mener une étude, présenter des résultats et préparer leurs présentations. De plus, j'ai rédigé un rapport sur l'ensemble des tests de comparaison des deux clients, ce qui m'a permis de m'exercer sur l'aspect rédactionnel. Je n'ai malheureusement pas pu assister à la présentation finale puisque cela s'est produit lors de l'une de mes périodes de cours,

11. HomeService correspond au service en-dessous d'OLS.

mais j'ai pu participer à la préparation de la présentation et du support.

2.2.3 Résultats

À la suite de la comparaison entre les client CWMP EasyCWMP et tr69agent, une des quatre propositions a été retenu. Cette solution consiste, entre autre, à fournir au sein du toolkit le client tr69agent, une documentation permettant l'intégration du client, ainsi qu'une plateforme de test [12] d'implémentation de client CWMP.

Les études de ces deux clients m'auront permis de renforcer mes compétences sur le protocole CWMP, en particulier sur le côté client. Mes compétences rédactionelles auront également été renforcé.

2.3 Création d'un ACS Servlet

Après être monté en compétence sur CWMP et avoir travaillé du côté client du device management avec EasyCWMP et tr69agent, j'ai travaillé du côté serveur. Ce projet s'inscrit encore une fois dans le courant de ma première année.

Dans le cadre des tests d'équipements, nous avions besoin d'un deuxième ACS dans le réseau isolé afin d'effectuer certain tests. C'est pourquoi, un des membres de l'équipe a conçu ce que l'on appelle une Servlet. Cette Servlet est codée en langage Java, et se comporte au sein du réseau isolé comme un serveur ACS. La Servlet s'installe sur un serveur web installé sur un PC du réseau isolé, permettant ainsi d'avoir un équivalent de serveur ACS mais extrêmement minimaliste.

L'avantage d'avoir une Servlet comme ACS est que l'on peut y rajouter les fonctionnalités dont nous avons besoin. Elle est également facile à installer, puisqu'il suffit uniquement d'un serveur web.

12. La réalisation de cette plateforme fait l'objet de la partie suivante, intitulé "Conception et développement de TINK"

On m'a donc demandé de reprendre la Servlet déjà faite et de l'améliorer afin que dans un premier temps elle puisse extraire le Data Model d'un équipement qui viendrait dialoguer avec elle. Par conséquent, elle devait aussi pouvoir communiquer avec un maximum d'équipements qui implémentent le Document TR-069. A l'origine, la Servlet qui a été développé ne pouvait communiquer qu'avec un nombre restreint de types d'équipements à cause de contraintes techniques que je devais donc résoudre.

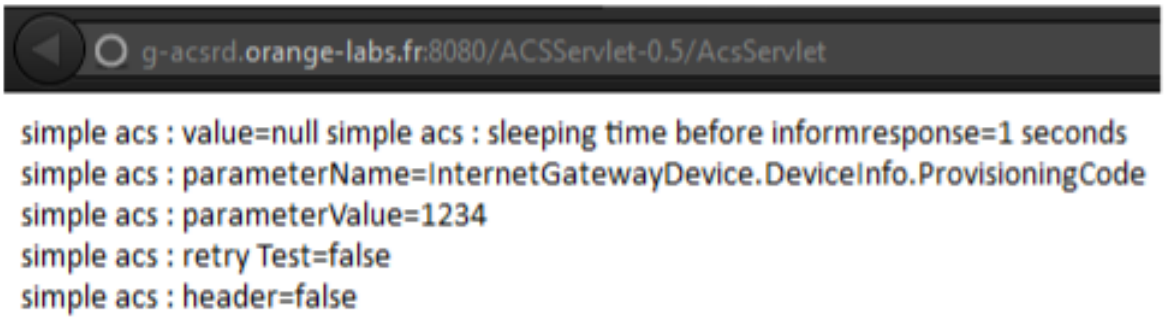

FIGURE 4 – Affichage web de l'ACSServlet-0.5, ancienne version

Comme on peut le voir ci-dessus, la dernière version de la Servlet, nommé « ACSServlet-0.5 » était extrêmement minimaliste en termes d'affichage. L'état des paramètres utilisés pour les tests d'équipements était modifié via l'url et aucune vérification n'était faite sur la cohérence des valeurs entrées. Par ailleurs, elle implémentait un nombre limité de méthodes CWMP. Je devais donc améliorer l'interface web afin de proposer une solution plus simple pour modifier les paramètres. De plus, j'ai pensé qu'un contrôle des valeurs de ces paramètres serait un plus et une sécurité pour l'utilisateur. Le tous devait bien entendu pouvoir fonctionner avec le plus grand nombre de device possible en implémentant davantage de méthodes CWMP. Enfin, elle devait pouvoir extraire systématiquement le data model de l'équipement dans un fichier au format .csv.

Comme pour les précédentes activités, j'étais libre de procéder dans l'ordre que je souhaitais pour réaliser ce travail. L'objectif étant là encore de me faire gérer de bout en bout une activité, découper les différentes tâches et étapes, le tout sans hésiter à aller voir les personnes susceptibles de m'apporter des informations et explications dans les domaines dont j'avais besoin.

J'ai tout d'abord cherché à comprendre le code de la Servlet que l'on m'avait donné. N'ayant pas vraiment de connaissance dans le langage Java, je devais également monter en compétence dans ce domaine. Cela m'a pris plusieurs jours pour voir et comprendre le code, les fonctionnalités mises en place, ainsi que les outils utilisés pour développer la Servlet.

Je me suis familiarisé avec le code en ajoutant des méthodes CWMP qu'un serveur ACS doit avoir pour communiquer avec différents équipements. J'ai pu identifier et implémenter ces différentes méthodes depuis le Document TR-069, mais également en analysant les échanges de trames sur le réseau isolé entre un équipement et son ACS. L'implémentation de ces méthodes m'a permis de me familiariser plus rapidement avec l'ensemble du code. J'ai également pu avoir plusieurs explications de la personne qui a développé les premières versions de la Servlet.

Une fois cette première étape de découverte passée, j'ai pu faire communiquer la Servlet, avec ces nouvelles méthodes, au client EasyCWMP sur lequel j'avais travaillé précédemment. Et ainsi vérifier son bon fonctionnement sur un premier type d'équipement.

J'ai part la suite réfléchie à la méthode à employer pour réaliser l'extraction du Data Model d'un équipement quelconque. Un Data Model ce présentant sous forme d'un arbre. Je me suis donc renseigné sur les différents types de parcours d'arbre possible et algorithme associé, afin de choisir le plus optimisé, mais aussi le plus en adéquation avec les contraintes techniques que j'avais. Le Data Model devait être extrait sous forme d'un fichier .csv.

Une fois l'implémentation de cet algorithme fait, j'ai cherché à le tester sur un maximum d'équipements. Il fallait donc modifier la Servlet de telle façon à ce qu'elle puisse s'adapter à l'équipement qu'on lui présente en entrée, et communiquée avec. Cela a été assez compliqué et partiellement réaliser, puisque tous les équipements ne dialogue pas forcément de la même façon, et apporte donc régulièrement de nouvelles contraintes. Ce travail d'adaptation de la Servlet à un équipement nécessiterait beaucoup plus de temps que celui disponible pour l'activité entière.

Après avoir permis à la Servlet de communiquer avec différents équipements et d'extraire leur Data Model, j'ai voulu faire des tests d'extraction de Data Model. L'objectif

était de comparer le temps d'extraction du Data Model de plusieurs équipements depuis ma Servlet, avec le temps d'extraction depuis un script JavaScript exécuté sur le serveur ACS g-acsrd. Ces mesures ont été réalisé selon différents critères tels que, le nombre de paramètres (feuille), le nombre de groupe de paramètre (branche), le poids final du Data Model dans un .csv et l'équipement dont le Data Model est extrait [13]. Cela m'a amené à optimiser mon algorithme d'extractions, et plus généralement le code entier de ma Servlet.

Une fois l'étape d'optimisation du code effectué, j'ai créé une page d'accueil de la Servlet en html et css avec un formulaire pour modifier les paramètres plus proprement. De plus, lors de l'envoie des valeurs des paramètres, j'ai rajouté une vérification afin de détecter différentes anomalies ou valeurs impossibles qui serait entré par l'utilisateur et entrainerais un blocage de l'équipement.

Enfin, après avoir testé le bon fonctionnement de la Servlet, j'ai pu l'installé sur le serveur g-acsrd et donc la rendre accessible depuis l'extérieur, comme on peut le voir ci-dessous.

13. Les capacités de RAM et cpu de l'équipement sont très influentes sur l'éxecution des requêtes permettant de parcourir le Data Model

FIGURE 5 – Affichage web de l'ACSServlet-0.6, nouvelle version

J'ai également ajouté une partie sur la page d'accueil expliquant les fonctionnalités et le rôle de cette Servlet, avec des exemples de valeurs attendues pour les paramètres et une note d'explication pour chacun d'eux.

Cette activité m'aura permis de monter en compétence en Java, plus particulièrement du côté Servlet. Mais également de m'améliorer dans la gestion d'une activité de plusieurs mois entre coupé par les périodes de cours et les autres activités. Enfin, j'ai pu de nouveau monter en compétence sur CWMP.

2.4 Impact sur mon parcours

Comme nous avons pu le voir tous au long de cette partie, les activités réalisées durant la première année de mon alternance m'ont permis de monter rapidement en compétence sur le protocole CWMP, et permis d'avoir une bonne vision du domaine du device management. La lecture du Document TR-069, puis l'activité de test d'équipements réalisé sur l'ensemble de la première année, m'ont permis de comprendre le protocole CWMP, voir différentes implémentations et comportement d'équipements. Le projet d'étude du client EasyCWMP, puis la comparaison avec le client tr69agent, m'ont permis de rentrer plus en profondeur sur les aspects clients du protocole, mais aussi de développer des compétences en développement et génie logiciel. Enfin, le projet d'ACS Servlet, m'a quant à lui permis de rentrer plus en détails sur l'implémentation de la partie serveur du protocole. J'ai par ailleurs pu travailler avec une grande partie des membres de mon équipe durant ces projets, me permettant ainsi de m'intégrer au mieux au sein de mon service et de mon entreprise.

Sur les domaines techniques, j'ai pu renforcer mes compétences en administration réseaux et système développer durant mon DUT R&T. Plus précisément, j'ai consolidé les compétences sur la configuration de serveur DNS et DHCP lors des études de comportement de client. J'ai pu également acquérir des bases de développement en C et Java dont je n'avais que très peu de notion avant mon arrivé. Ces bases en développement, comme nous le verrons par la suite, ont été extrêmement utiles et sollicitées durant la suite de mon alternance. De plus, mes compétences acquises en entreprise sur le domaine logiciel, ont pu être renforcées par les cours de développement et le projet Java de première année suivi à l'école.

Sur les domaines transversales, j'ai pu acquérir de solide base en rédaction de documentation et rappport d'étude. Ces documents ont été rédigé en français, mais cela m'a permis d'avoir connaissance du rendu attendu pour ces différents types de documents. J'ai également pu acquérir, grâce au projet de test d'équipements, des compétences sur les aspects de travail collaboratif. Enfin, le fait d'avoir eu une très grande autonomie sur

l'ensemble des projets m'a permis de renforcer mes compétences en gestion du temps, planification et organisation de projet. Là encore, ces compétences seront grandement sollicitées et renforcées par la suite.

Ainsi, ces projets n'auront pas uniquement servi à conforter mes compétences sur le protocole CWMP et le domaine du device management. Ils m'auront aussi permis d'acquérir des compétences transversales tel que la gestion de projet ou encore la connaissance d'entreprise.

3 Conception et développement de TINK

3.1 Contexte

L'ACS que nous utilisons en production pour manager le parc client d'Orange se nomme Karma. Il a été développé en Java par des équipes internes Orange. Il respecte l'ensemble des parties obligatoires du Document TR-069 et une majorité des parties optionnelles. Les trames CWMP qu'il reçoit sont modifiées pour s'adapter au fonctionnement interne de Karma. De même, lorsqu'il répond à un équipement, ses messages sont créés de manière à être compris par l'équipement en question. Le module de Karma permettant ces traductions du protocole CWMP, se nomme X69. Comme nous le verrons par la suite, c'est essentiellement sur ce module de Karma que le projet TINK s'appuie. Ce module permet ainsi de générer des trames correctement construites au niveau CWMP à destination des équipements, tout en contrôlant que celles reçues des équipements sont compréhensibles et correctement formées pour l'instance de Karma déployée en production.

Actuellement, les constructeurs d'équipements, souhaitant produire des équipements manageables, implémentent leur propre client CWMP sur leurs appareils. Lorsqu'ils veulent vérifier leur bon fonctionnement pour le management par Orange, ils prennent contact avec Orange pour leur faire passer une série de tests validant, ou non, la bonne communication entre l'équipement du constructeur et Karma. Cependant, de nombreux constructeurs contactent Orange avec des embryons de client CWMP, ne pouvant parfois pas émettre les requêtes CWMP obligatoires du Document TR-069, y compris les requêtes les plus basiques. Ceci fait perdre un temps considérable à l'équipe en charge de ces tests. Les tests sont alors refaits, après modification du client par le constructeur, jusqu'à ce que le client CWMP soit finalement jugé apte à communiquer avec Karma. Cette demande de renouvellement des tests implique également un coût financier non négligeable aux équipes d'Orange. Cela pose donc un problème, puisque certains constructeurs ne pouvant pas être autonomes pour réaliser les tests sont obligés de prendre contactent avec nos équipes régulièrement pour tester leurs équipements.

Ainsi, TINK se propose de résoudre ce problème, en permettant aux constructeurs d'être plus autonomes dans les tests de leurs équipements avant de les présenter pour les

tests plus poussés.

3.2 Présentation

Il a été décidé durant le premier semestre 2015, à la suite de l'étude comparative entre les client EaysCWMP et tr69agent, de proposer un toolkit CWMP afin d'aider les constructeurs à implémenter un client CWMP compatible Karma sur leurs équipements. Les différentes solutions de toolkit CWMP proposables ont fait l'objet d'une étude lors de ma première année. Cette étude a permis d'aboutir à la décision de proposer un toolkit CWMP comportant le client CWMP tr69agent développé par Orange et mis en Open Source, ainsi qu'une plateforme en ligne de test de client CWMP. Ainsi, le constructeur aura le choix entre utiliser notre client CWMP pouvant communiquer avec Karma, ou bien développer son propre client CWMP et le tester avec notre service de test en ligne. Bien entendu, ce service de test de client CWMP n'a pas vocation à remplacer la procédure de tests effectuée par Orange, il servira à réaliser une première vérification, sans pour autant exécuter les tests plus complexes. Par la suite, l'équipe de test d'Orange sera sûre que le client qu'on leur apporte satisfait une majeure partie de ce qui est exigé.

Ce projet de conception et réalisation d'un service de test CWMP a donc débuté en Octobre 2015, il m'a accompagné tout au long de ma deuxième, puis de ma troisième année.

D'octobre 2015 à mars 2016 nous étions trois à travailler sur le projet, puisque comme nous allons le voir plus tard, la solution technique n'était pas encore choisie, il n'y avait donc pas la nécessité d'une personne supplémentaire. Initialement, il y avait Marc DOUET, Matthieu ANNE et moi-même, puis en mars 2016, Jean-Pierre ONA, stagiaire encadré par Matthieu ANNE, nous a rejoints afin de développer un portail web pour TINK. Puis, à partir d'aout 2016 Jean-Pierre ONA a fini son stage, nous sommes donc repassé à trois jusqu'à aujourd'hui.

Nous avons opté pour une gestion du projet de type agile durant la présence de Jean-Pierre ONA. Le fait que ma présence soit discontinue a rendu le déroulement du projet très complexe à gérer. Par conséquent, l'organisation du projet a dû être revue au fur et à

mesure, ce qui convient à la gestion de projet agile. Pour Jean Pierre ONA et moi-même, c'était la première fois que nous mettions en pratique la gestion de projet en agilité, ce qui m'a permis de monter en compétence sur ce type de gestion de projet.

À la suite de la fin du stage de Jean-Pierre ONA, nous nous sommes de nouveau retrouvé à trois sur le projet. Il nous a fallu recupèrer le travail effectué par Jean-Pierre ONA sur l'IHM du portail web afin de pouvoir, le moment venu, le faire évoluer.

Dans ce projet, Jean-Pierre ONA et moi-même avions les rôles de concepteurs développeurs, puisque nous avons défini et réalisé l'architecture du logiciel ; mis en place et configuré l'environnement de déploiement ; et enfin développé l'intégralité du service. Matthieu ANNE a eu le rôle de Product Owner en ayant une vision globale du produit final non seulement par sa participation aux choix d'architecture logicielle, mais aussi en priorisant les différentes tâches. De plus, il s'est occupé de tenir à jour l'avancement du projet sur la plateforme de gestion interne à Orange, en ajoutant les différents User Stories, Sprint et Release définis lors des réunions. Marc DOUET a supervisé le déroulement du projet et participé avec moi-même au choix de la solution technique dont fait l'objet la prochaine partie.

3.3 Déploiement

Sachant que TINK se veut être une solution d'automatisation de test en ligne, il nous a fallu chercher une solution d'hébergement. Pour la suite du document, il est important d'expliquer et apporter des précisions sur cette solution de déploiement qu'est Kermit.

3.3.1 Qu'est-ce que Kermit ?

Kermit est un projet interne à Orange, basé sur l'outil OpenShift de RedHat Enterprise. Ce projet interne Orange consiste à aider les développeurs dans leur cycle agile, en fournissant une plateforme de Platform-as-a-Service (PaaS) permettant de faire rapidement de l'intégration et du déploiement continue.

Les solutions PaaS sont avant tous des solutions d'infrastructure cloud. Le fait que les applications soient hébergées dans le cloud offre la possibilité d'avoir un accès rapide et simple aux ressources physiques, et de pouvoir réduire ou augmenter les ressources de calcul à moindre coût. De plus, la solution de PaaS se basant sur les conteneurs Docker, cela permet une meilleur isolation des applications, ainsi qu'une simplification du déploiement de celles-ci. L'utilisation de Docker encourage également les développeurs à faire des mircro services. Enfin, l'ensemble des bénéfices apportés par les solutions PaaS permettent de favoriser le DevOps.

On peut donc rapidement déployer notre application de manière automatique, dans différents langages de programmation, le tous basé, sur la solution de conteneurisaiton Docker. Ces conteneurs sont facilement instanciables soit via une IHM en ligne, soit avec un outil en ligne de commande. Les conteneurs sont orchestrés et gérés par l'outil Kubernates, ce qui est transparent pour nous.

3.3.2 Environnement

L'environnement de Kermit se compose de plusieurs éléments. L'ensemble des conteneurs reposent sur une couche physique commune. Sur cette couche physique, vient se placer se que l'on appel des "nodes". Les nodes fournissent un environnement d'exécution aux conteneurs. Ils possèdent les services requis à l'exécution de Docker, Kubernates et tout autre service de proxy. Enfin, dans les nodes nous avons ce que l'on appel les "pods". C'est dans les pods que les conteneurs sont instanciés. Un pod peut contenir un ou plusieurs conteneur, il représente un hôte. On peut retrouver cet environnement sur la figure de l'architecture simplifié de Kermit ci-dessous.

Les données devant être sauvegardées, tels que les données d'une base de données, sont stockées sur des disques dits persistants, en dehors de la solution de PaaS.

Il est important de noter que chaque pod possède sa propre adresse IP publique. De même, nous pouvons créer autant d'Uniform Resource Locator (URL) que l'on souhaite pour chacune de ces adresses IP. Les URL pouvant être sécurisé par l'ajout de son propre certificats, ou bien en utilisant celui déjà existant de Kermit.

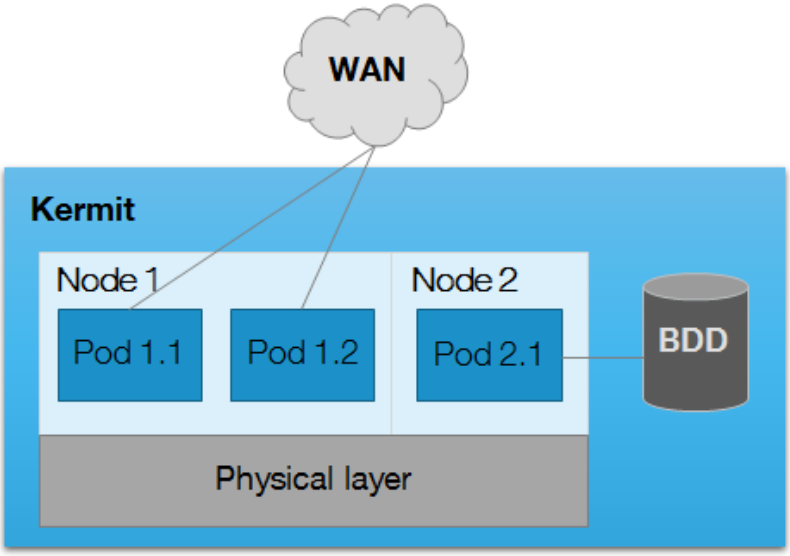

FIGURE 6 – Architecture simplifiée de Kermit

L'utlisation de Kermit pour pouvoir faire de l'intégration continue m'a permis de renforcer mes compétence en conteneurisation. J'ai été amené à développer des scripts d'installation pour leur client sur des machines Linux, puisque jusqu'à présent il n'y avait pas d'installation du client de Kermit sur Linux. Cela m'a permis de renforcer mes compétences de scripting Linux.

Bien que je connaissais déjà Docker grâce à mon auto formation faite suite au projet de 2 année de l'école, la prise en main de Kermit, m'a permis de renforcer mes compétences sur la conteneurisation et particulièrement sur Docker.

3.4 Travail de préparation

3.4.1 Recherche de solution technique

Avant la phase de réalisation il a fallu identifier les différents choix techniques possibles, puis en choisir un selon nos besoins, tout en comprenant son fonctionnement. Cette première phase du projet s'est déroulée d'octobre 2015 à mi-janvier 2016.

Le but de cette phase était de trouver une solution technique qui répond aux besoins de TINK tout en comprenant comment l'exploiter. Les besoins identifiés pour le projet TINK sont de permettre aux constructeurs d'avoir un accès libre à un service de test

reproduisant le comportement de Karma, sans pour autant donner un accès à Karma même. Ainsi, les choix techniques devaient porter intégralement sur le choix du ou des module(s) Karma à utiliser, dans quelle version et de quelle branche, puisque comme nous le verrons Karma possède différentes branches versionnées.

Le choix d'une solution technique devait permettre à TINK de remplir au mieux ses objectifs, de pouvoir cadrer l'environnement de déploiement de la solution, ainsi que donner une direction pour la réalisation du projet en lui-même. Ce choix avait une très grande importance pour la suite du projet, puisque de lui dépendait la manière donc le projet serait réalisé, les contraintes qui seraient établies, mais également les difficultés qui allaient être rencontrées. La compréhension de la solution choisie, devait permettre la mise en place d'une première version de TINK lors de la deuxième phase du projet.

Karma est divisé en deux branches logicielles, qui sont elles-mêmes divisées en plusieurs versions. Les deux branches, Karma Fr et Karma BU, offrent les mêmes fonctionnalités du protocole CWMP, et diffèrent uniquement dans leur implémentation et dans les traitements des équipements qui sont faits plus en profondeur. Ainsi, le choix ne pouvait pas être fait au niveau des branches, qui sont par ailleurs maintenues par deux équipes différentes. Karma Fr est destinée aux clients français uniquement. Elle est développée et maintenue par une équipe française, tandis que Karma BU est destinée à l'ensemble des clients en-dehors de la France, avec un code maintenu par une équipe roumaine.

Nous avons décidé d'étudier les versions les plus récentes, à savoir 9 et 10, car des fonctionnalités manquaient aux versions précédentes. Que ce soit pour Karma Fr ou Karma BU, aucune version n'offre de fonctionnalité supplémentaire. Pour Karma Fr, la différence entre les versions 9 et 10 réside dans l'architecture. En effet, la version 9 fait communiquer les différents modules Karma via des Web Services, tandis que la version 10 les réunifie et les fait communiquer via des packages java 1 . Pour Karma BU, il n'y a pas de différence d'architecture logicielle, les deux versions utilisent des Web Services pour la communication de leurs modules, la différence se trouve au niveau de l'architecture système. Dans la version 9, les modules sont déployés sur des machines différentes, tandis que pour la version 10, ces derniers se trouvent sur la même machine, offrant ainsi une refonte du code et une optimisation de celui-ci.

Ainsi, j'ai commencé par me renseigner sur les branches existantes, puis les spécificités de leurs versions auprès de différentes personnes compétentes, à la suite de quoi, j'ai sélectionné les versions 9 et 10 de Karma Fr et la version 10 de Karma BU pour une phase de test. N'ayant pas de contrainte d'architecture imposée ou définie, j'étais libre de choisir la version la plus arrangeante et la plus simple d'utilisation pour moi pour la suite. La version 9 de Karma BU, bien qu'intéressante pour sa facilité de récupération et d'utilisation de ses modules individuellement, ne bénéficie pas de toutes les optimisations de la version 10 et n'était donc pas intéressante.

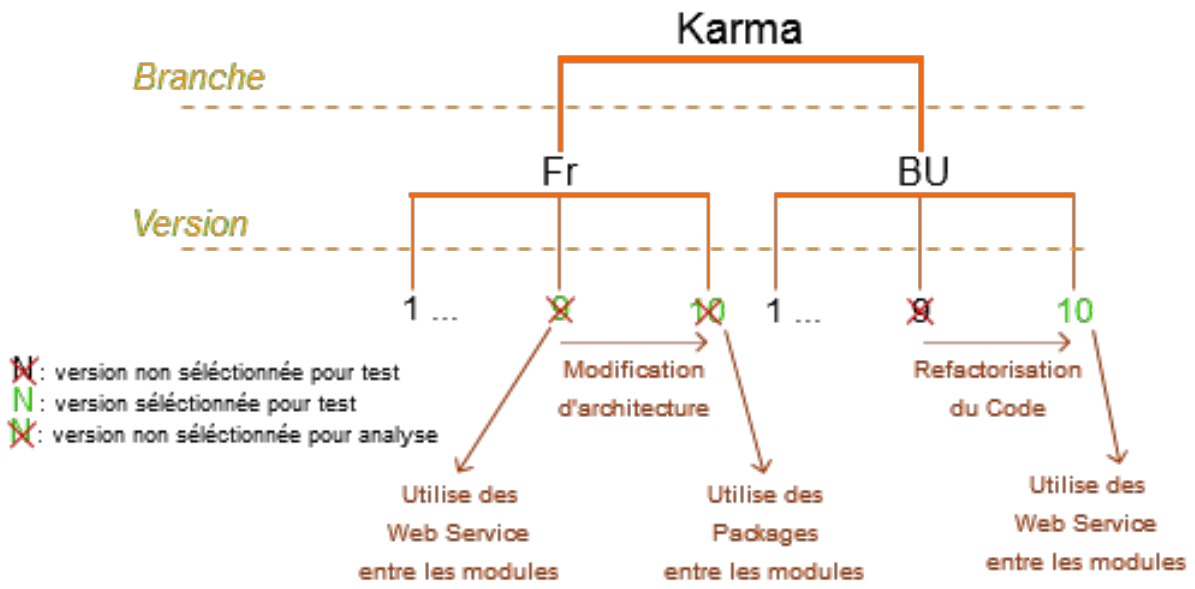

FIGURE 7 – Choix de la version de Karma à utiliser

Le but n'était pas d'étudier le fonctionnement de l'intégralité de Karma mais uniquement les modules qui pouvaient être utiles. Marc DOUET m'a indiqué que le module qui

semblait le plus approprié pour le projet était le module X69. J'ai donc commencé à compiler puis déployer chaque module X69 des versions retenues, pour ensuite comprendre leur fonctionnement. Pour cette étape, je me suis mis en relation avec les équipes de développeurs de chacune des deux branches. Ils m'ont aidé avec les spécificités de compilation et déploiement de chaque version. Cette étape a permis non seulement de connaitre les contraintes d'environnement pour chaque version, mais également servis à comprendre comment le module X69 communiquer avec un client CWMP et la manière de l'utiliser selon les versions.

Finalement, il est apparu que la version 10 Karma Fr des modules COM et X69 ne peuvent pas se passer de la base de données de Karma Fr, ce qui rend cette version inutilisable. La version 9 de Karma Fr est complexe à compiler et ne possède aucune documentation pour configurer son environnement de déploiement. De plus, le code de la branche Karma Fr n'est pas très propre et non documenté, ce qui a permis d'éliminer ces deux versions, ne laissant que la version 10 de Karma BU. Le code de cette version est correctement documenté, de même pour la configuration de l'environnement de déploiement et son installation. Ainsi, mon choix s'est porté sur le module X69 de la version 10 de Karma BU.

3.4.2 Analyse de faisabilité

Après avoir identifié la solution technique que nous devions utiliser, il fallait utiliser la version de X69 afin de créer une première version de TINK. Bien entendu, cette première version ne servirait que de test, permettant uniquement des échanges de requête avec un client CWMP, sans logique de test CWMP. Cette phase d'analyse de faisabilité s'est déroulée de mi-janvier à mi-février 2016 et comportait plusieurs objectifs. Il est important de noter que contrairement à la précédente phase, celle-ci avait une date limite et devait être terminée pour mi-février 2016, c'est-à-dire à la fin de ma période entreprise. Cette date limite se justifie par le fait que ce travail allait servir de base pour Jean-Pierre ONA, qui devait arriver courant mars 2016, durant ma période d'école.

Le principal but de cette phase était bien évidemment de commencer un premier travail d'adaptation du module X69 de Karma BU pour nos besoins. Cela a consisté à comprendre

les mécanismes d'interactions des différents packages et classes, afin de pouvoir générer des requêtes CWMP autres que celles utilisées lors de la première phase, c'est-à-dire les méthodes impliquées dans l'initiation de la session CWMP. Cela m'a demandé de me concentrer sur la manière d'utiliser ce module pour choisir les méthodes CWMP envoyées à un client CWMP lors d'une communication.

Un second objectif fut de déterminer quelles sont les méthodes CWMP que le module X69 implémente et nous permet donc d'utiliser. Cette étape était cruciale puisqu'elle a permis de déterminer quels tests nous allions pouvoir faire, ou non, et avec quelle complexité nous pouvions accéder et formater ces méthodes CWMP.

Un troisième objectif, dépendant des deux précédents, était de définir de manière plus structurée le projet dans une première version du cahier des charges. Par la suite, il m'a également fallu réaliser une première version du dossier de spécification. Le dossier de spécification devait être uniquement constitué des premiers tests susceptibles d'être fait. Ces documents devaient également aider Jean Pierre ONA à prendre connaissance du projet plus facilement.

En somme, l'objectif global de cette phase était de définir ce que nous allions faire lors de la réalisation, en ayant une première version de TINK capable d'envoyer des requêtes CWMP.

Lors de la réalisation de cette étape, j'ai dans un premier temps cherché à mieux comprendre le code de X69. Cela n'a pas été simple pour moi puisqu'il utilise plusieurs Framework Java assez complexes. J'ai donc régulièrement communiqué avec Eduard COJOCARU, qui a retravaillé le code de X69 lors du passage de la version 9 à la version 10 de Karma BU. Par conséquent, il a une très bonne connaissance de ce module. J'ai par ailleurs été amené à rencontrer d'autres développeurs seniors pour qu'ils m'expliquent les concepts et le fonctionnement de certains mécanismes de ce code. Ce travail m'a permis de trouver par quel moyen un CPE sait quelle interface interroger pour communiquer avec X69.

Quand j'ai eu une vision globale du module X69 et compris le fonctionnement de celui-ci, je me suis concentré sur la compréhension du processus de détermination d'une réponse

du module X69 au CPE. Pour cela, j'ai eu à retracer le parcours d'un message d'un CPE arrivant sur ce module, jusqu'à l'envoi de la réponse. Ce travail de Reverse Engineering m'a permis de remonter aux parties qui « parsent » les messages reçus, et plus important aux classes qui créent les messages CWMP à envoyer au CPE. Encore une fois, lors de cette étape, Eduard COJOCARU m'a indiqué et expliqué de nombreux mécanismes de X69 impliqués dans cette communication avec le CPE, ainsi que des mécanismes plus généraux de Java.

Lorsque j'ai su comment X69 traite les messages CWMP des CPE et comment moi-même en envoyer dans l'ordre de mon choix, j'ai créé une première Servlet qui utilise le module X69. Cette Servlet avait pour but de communiquer avec un CPE qui voudrait établir une session CWMP et lui envoyer les requêtes CWMP voulues. Cela m'a permis de mettre en pratique pour la première fois l'ensemble des connaissances acquises depuis le début de ce projet.

Une fois les tests de communication entre ma Servlet et le CPE effectués, j'avais une meilleure vision des tests qu'il serait possible d'implémenter parmi la liste de ceux proposés par l'équipe qui les fait habituellement. Il s'agissait donc maintenant de rédiger et spécifier cette première suite de tests afin que Jean Pierre ONA, qui arriverait durant ma prochaine période de cours, puisse avoir une aide pour comprendre le projet plus facilement.

Parallèlement à ce travail effectué sur le module X69, nous avons commencé à mettre en place la gestion de projet de manière agile. Cela s'est traduit pour ma part à lister sous forme de Trackers l'ensemble des tâches à effectuer. J'ai par conséquent dû mettre à jour ces Trackers au fur et à mesure que j'avançais dans la réalisation de la tâche qu'ils représentaient. J'ai également suivi une formation sur Kanban et Scrum, ainsi que sur l'utilisation d'un outil de gestion agile de projet interne à Orange.

3.5 Méthode de projet

Nous avons utiliser la méthodologie de projet agile durant la période de mars 2016 à août 2016. Après cette période j'étais le seul à développer l'application, il n'était donc

plus utile de poursuivre ainsi. Néanmoins, nous avons continué à nous référer aux User Stories décrites durant la période de présence de Jean-Pierre ONA.

Après que Jean-Pierre ONA ai pris connaissance de l'objectif du projet, nous avons commencé à réfléchir puis décrire en détail les User Stories desquelles ont découlé les différentes tâches qui, plus tard, ont constitué les différents Sprint.

Nous avons opté pour des Sprint de trois semaines, à la fin desquels une démonstration était attendue. Les Sprint ont été définis au fur et à mesure. De plus, le fait que je ne sois pas toujours présent en entreprise a rajouté une contrainte supplémentaire au déroulement du projet et au contenu des Sprint.

Tout au long des Sprint, des Stand-up daily meetings ont été réalisés afin que, quotidiennement, chacun indique aux autres membres de l'équipe sur quoi il allait travailler et quels étaient ses objectifs. Nous avons mis en place des points d'avancement une à deux fois par semaine afin de répondre aux problèmes qui pouvaient survenir, discuter de certaines solutions techniques, ou bien encore reprioriser certaines tâches. Le contenu des Sprint étaient également défini lors de réunions en début de Sprint, où nous étions présents tous les quatre.

Comme dit précédemment, la méthodologie de projet en agilité a permis de nombreuses modifications tout au long du projet et sur différentes parties. Bien que les User Stories n'aient pas été modifiées au cours du projet, les contenus des Sprint et de la Release d'août 2016 ont été fortement modifiés entre le début du projet et aujourd'hui. De même, l'architecture logicielle et le modèle relationnel de données ont été revus entre les différents Sprint pour s'adapter au Sprint qui suivait et à l'ajout de fonctionnalités que cela impliquait.

Le premier Sprint n'a été débuté qu'une fois les questions d'architecture logicielle réglées pour une première version, et les spécifications sur les différents tests rédigées. Lors du premier Sprint qui a débuté en mai 2016, le but était d'envoyer une suite de requêtes simples à tous les équipements qui viendraient interroger l'interface de l'outil de test. Il ne devait pas y avoir de vérification du contenu des réponses des équipements. Il y avait seulement la possibilité de créer des utilisateurs dans le portail et de leur associer des

équipements. Lorsque ces équipements viendraient contacter l'interface de l'outil de test dédiée aux équipements, différentes requêtes et réponses seraient échangées. La sauvegarde de ces données impliquait la création d'un modèle relationnel de données qu'il a également fallu établir. De plus, lors de ce Sprint, les modules constituant l'outil et la base de données devaient être déployés sur Kermit. Ce Sprint a donc été l'occasion de prendre en main Kermit en allant interroger l'équipe en charge du développement et maintien de cet outil.

Les second et troisième Sprint n'ont concerné que le portail web puisque je n'étais pas présent en entreprise. L'interface et le design ont été revus et différentes fonctionnalités d'ergonomie ont été rajoutées.

A mon retour en juillet 2016, nous avons lancé le quatrième Sprint, qui devait initialement aboutir à une première version de la plateforme, qui permettait la création d'un utilisateur ; l'ajout d'équipements ; la création d'un plan de test ; et le déroulement de celui-ci lorsque l'équipement viendrait établir une session CWMP avec notre outil. Cependant, nous avons revu l'architecture globale du logiciel, et avons, à la suite de réflexions communes, décidé de reprendre la manière dont communiquent les modules entre eux, lesquels peuvent discuter avec la base de données ; et de clarifier de nouveau les rôles de chacun entre Jean-Pierre ONA et moi-même. La fin de ce Sprint a été reportée à mi-août 2016 avec une diminution du nombre attendu de tests réalisables, et une redéfinition plus propre de l'architecture logicielle, ce qui permettrait de reprendre le code plus facilement dans le futur.

Après le départ de Jean-Pierre ONA, nous avons arrêté d'effectuer des Sprint, tous en continuant de réaliser les différentes User Stories que nous avions précédemment planifiées.

3.6 Conception

Les différents points de conception ont été réalisé par Jean-Pierre ONA et moi-même, puis présenté et soumis à Matthieu ANNE et Marc DOUET, avant de pouvoir être implémenté. Il n'a pas été exigé de nous que nous proposions dès le premier Sprint une architecture compléte répondant à l'ensemble des problèmes techniques des futurs Sprint. Au contraire, à chaque Sprint nous devions modifier, et réviser le cas échéant, ces points

de conception [14] en analysant les nouveaux objectifs. En revanche, nous devions défendre et argumenter nos choix de conception à chaque révision et modification.

A la suite du départ de Jean-Pierre ONA en aout 2016, peu de modification au niveau de la conception de l'application ont été réalisé. Le cas échéant le procéder de validation a été le même.

La plus grosse partie de la phase de conception de l'outil à eu lieu durant la période de mars 2016 à juillet 2016. D'autres évolutions ont eu lieu au cours de la troisième année pour mettre en place de nouvelles fonctionnalités, ou en renforcer d'autre. Comme mentionné précédemment, nous avons travaillé en méthode agile, l'architecture logicielle et le modèle relationnel de données de l'application ont donc été fortement amené à évoluer au fur et à mesure des Sprint, pour finalement être fixé à la fin de l'année 2016 à ce qui est présenté ci-dessous.

3.6.1 Modélisation des tests

Le but de cette plateforme étant de réaliser des tests d'implémentation CWMP sur des équipements, il nous a fallu concevoir un modèle de test. Ce modèle devait faciliter l'ajout de nouveaux tests dans le futur, l'analyse des messages CWMP reçu, et la génération des rapports de tests. Ainsi nous avons produits un modèle de test basé sur quatre couches hiérarchiques défini comme suis :

- Un `Test Plan`, est un scénario testant un ensemble de fonctionnalités CWMP sur un équipement donné, et aboutissant à la création d'un rapport de test.
- Un `Test` permet de vérifier l'implémentation d'une fonctionnalité CWMP en particulier au travers d'un ou plusieurs échange avec l'équipement, durant un temps donné. Dans un `Test Plan`, il doit y avoir un ou plusieurs `Test`. Un `Test Plan` est réussi uniquement si tous ses `Test` le sont.
- Une `Step`, correspond à un échange CWMP avec l'équipement à tester. Un `Test` est composé d'une ou plusieurs `Step`. Il comporte donc le nom de la méthode CWMP

14. Les points de conception sont ceux présentés dans cette partie : Modèle relationnel de données, Architecture logicielle et Modélisation des test

attendu ou à envoyer à l'équipement. Un `Test` n'est réussi que si toutes ses `Step` le sont aussi, et si elles sont faites dans le temps définis pour le `Test`.

— Un `Parameter`, représente un paramètre envoyé dans une méthode CWMP. Ils peuvent à la fois être donnés par l'utilisateur, mais aussi défini par défaut par Orange, selon ce qui doit être testé. Une `Step` peut avoir plusieurs paramètres comme aucun.

3.6.2 Modèle relationnel de données

Le modèle relationnel de données sur lequel nous nous sommes arrêtés courant mars 2017 est formé de huit tables :

— `user` : Table permettant de sauvegarder les données utilisateur.

— `device` : Table permettant à un utilisateur associé d'enregistrer un ou plusieurs équipement à tester.

— `test_plan` : Table permettant à un utilisateur de créer un scénario de tests à exécuter pour un équipement donné.

— `test` : Table permettant de retenir un test composant un scénario de tests.

— `message` : Table où sont stockés les messages générés par l'analyse d'un test, constituant le rapport d'un plan de test.

— `test_type` : Table stockant les différents type de test pouvant être sélectionnés par l'utilisateur lors de la création d'un plan de test. Il contient les caractéristique de chaque tests.

— `step` : Table permettant de connaître les échanges CWMP constituant un test donné.

— `parameter` : Table permettant de savoir quel(s) paramètre(s) passer à la méthode CWMP qui sera envoyée à l'équipement.

Ces tables forment quatre parties utilisées par les modules qui constituent l'architecture logicielle de TINK. La partie User, composée de la table user, la partie Device composée de la table device, la partie TestType composée de la table test_type, et la partie Test composée des tables test_plan, test, step, message et parameter.

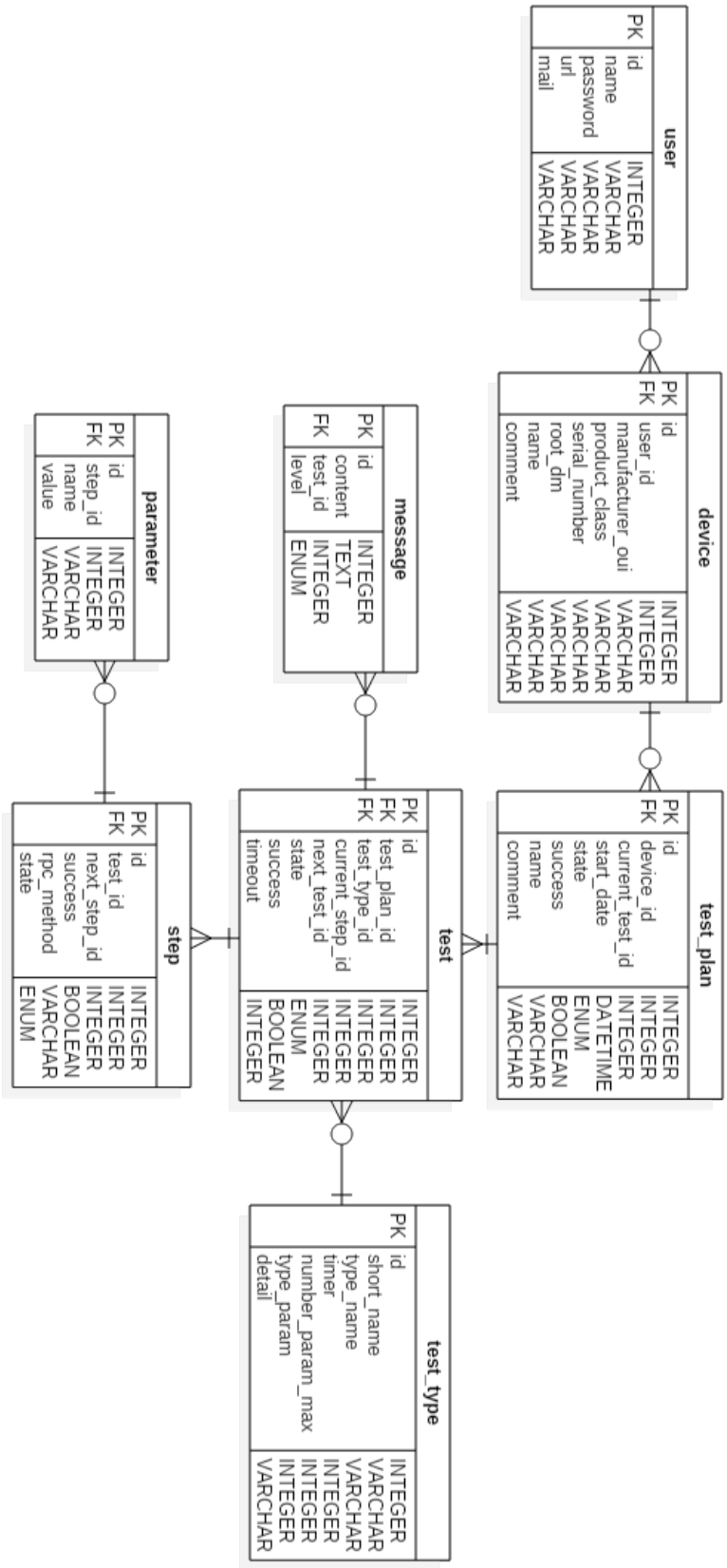

FIGURE 8 – Modèle relationnel de données

3.6.3 Architecture logicielle

Trois modules composent la plateforme TINK : GUI, TestManager et CWMP_I. Ces trois modules sont indépendants les un des autres et communiquent uniquement par les Application Programming Interface (API) qu'ils exposent. Ainsi ils ne partagent pas de dépendances et peuvent être déployés sur des serveurs d'applications différents.

— `GUI` : Permet aux utilisateurs d'intéragir avec TINK à travers une IHM. Ce module est responsable des interactions classiques tels que la création et modification d'un compte utilisateur, la création de test, la consultation et l'exportation de rapport de test, ou encore l'ajout d'un équipement à tester. Il peut communiquer avec les parties utilisateur, équipement et type de test de la base de données. Il a été décidé qu'il n'avait pas de besoins à connaître la logique de plan de test. De plus les informations sur les tests lui sont fournis par le module TestManager.

— `CWMP_I` : Permet aux équipements de communiquer avec TINK à travers le module X69 qui provient directement des dépôts de Karma BU. Le fait que ce module est importé par une dépendance, permet de faciliter sa mise à jour en cas de développement de nouvelles fonctionnalités sur ce dernier. Il n'a aucun besoin de communiquer avec la base de données car il est uniquement en charge des interactions CWMP avec les équipements se présentant à TINK. Il récupère les méthodes et paramètres à envoyer aux équipements grâce aux API du module TestManager. De la même façon, il envoi les messages CWMP reçu et pars par le module X69 au module TestManager pour leur analyse et la validation des tests.

— `TestManager` : Permet de connaître la logique de déroulement des test. Il porte également l'intelligence pour la validation des tests. Il peut communiquer les résultats des tests au module GUI à travers ses API, et envoyer de la même façon les méthodes et paramètres CWMP au module CWMP_I qui doivent être utilisés pour former les requête et réponse CWMP envoyées aux équipements. Il n'a pas connaissance des utilisateurs et ne communique donc pas avec cette partie de la base de données.

Ci-dessous nous pouvons voir un schéma de l'architecture logicielle de TINK où sont présent les trois modules précédemment cité.

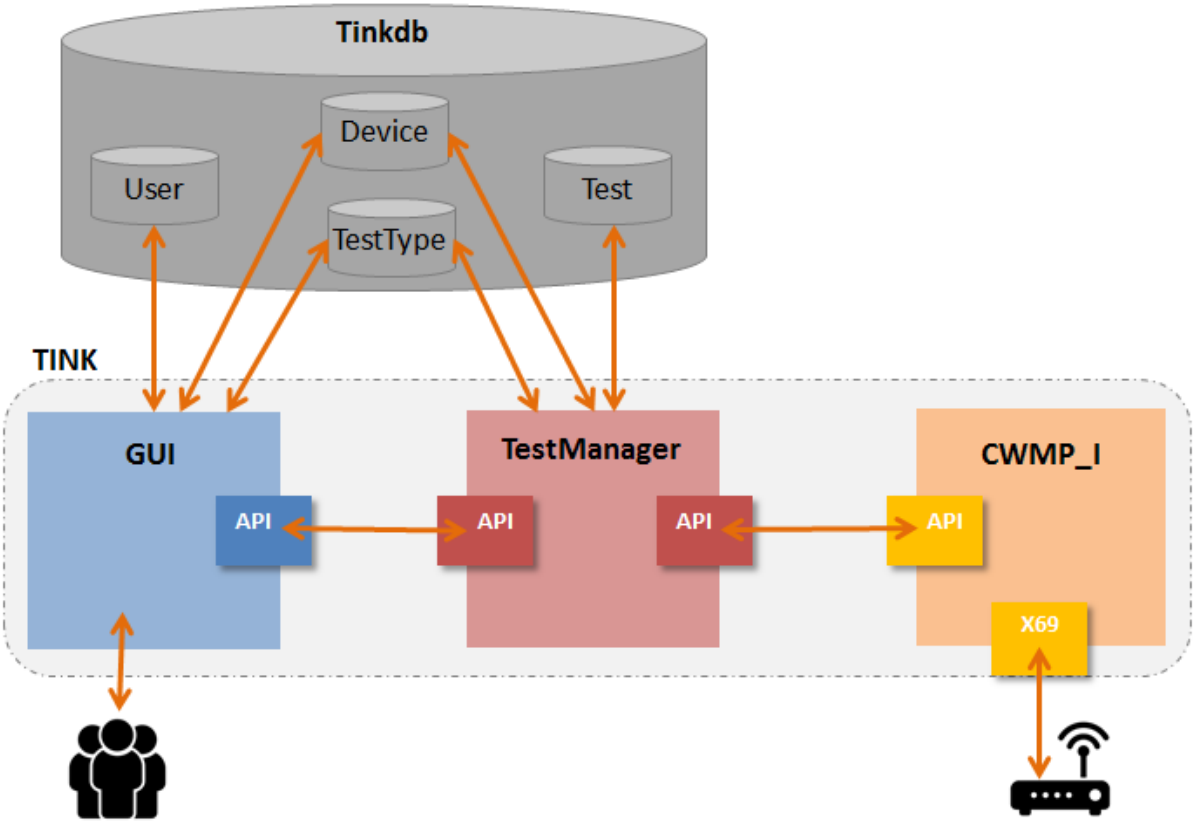

FIGURE 9 – Architecture logicielle de TINK

3.7 Réalisation

3.7.1 Travail en équipe

L'équipe de travail sur ce projet a été amené à changer plusieurs fois tous au long de la réalisation du projet. Comme mentionné précédemment, de Octobre 2015 à mars 2016 nous étions trois, et mon travail était principalement axé sur la recherche de solution technique. Ainsi vis à vis des autres membre du projet, j'avais uniquement à présenter les résultats lors de points d'avancement hebdomadaire.

Le travail en équipe à fortement évoluer et pris plus d'ampleur avec l'arrivée en mars 2016 de Jean-Pierre ONA. Il m'a été demandé d'encadrer ses activités et tâches lors de mes périodes en entreprise. Par conséquent, l'arrivée de Jean-Pierre ONA dans le projet en tant que développeur web et java, m'a demandé une adaptation dans mon organisation et les tâches que je devais accomplir ou lui donner. Nous avons choisi de travailler ensemble sur l'architecture logicielle de TINK en réalisant différentes évolutions de l'architecture qui, petit à petit, ont permis de définir le périmètre de chacun. A terme, chacun avait sa

propre partie de l'architecture à gérer, nous avons fait en sorte d'être cohérents dans nos choix.

Nos compétences techniques étant très différentes, cela nous a permis d'établir clairement le périmètre de chacun une fois l'architecture choisie. Jean-Pierre ONA est plus axé développement java et web, tandis que je suis plus orienté sur la partie algorithmique et base de données, et possède une connaissance du protocole CWMP. Ainsi, Jean-Pierre ONA s'est occupé de la partie IHM utilisateur, tandis que je me suis occupé de la partie interface équipement et réalisation/validation des tests. De plus, ayant une formation accès système, j'ai été en charge du déploiement et de la configuration système du serveur d'application et de la base de données sur Kermit.

Durant les phases de développement des Sprint, nous avons régulièrement modifié le modèle relationnel de données afin de s'adapter aux exigences des Sprint. Ces modifications ont nécessité une importante communication, dans un premier temps entre Jean-Pierre ONA et moi-même, puis avec Marc DOUET et Matthieu ANNE pour présenter les modifications et discuter ensemble les points bloquants. Ces révisions nous ont demandé une forte capacité d'adaptation, tant individuelle qu'au niveau organisationnelle. Les évolutions faites sur la base de données et sur l'architecture sont parfaitement inhérentes à la méthodologie de projet d'agilité.

À la suite du départ de Jean-Pierre ONA en août 2016, nous nous sommes de nouveau retrouvé à trois. Le travail était alors à la fois de continuer l'implémentation des fonctionnalités planifiées, mais également de récupérer et s'approprié le travail réalisé sur le module GUI par Jean-Pierre ONA. Matthieu ANNE et moi-même sommes montés en compétences sur ce module et y avons progressivement développés de nouvelles fonctionnalités.

Nous avons cessé de travailler sous forme de Sprint, mais avons continuer de développer l'outil fonctionnalité par fonctionnalité, en faisant des point plusieurs fois par semaine. Nous n'avons pas apporter d'évolution majeure à l'architecture de TINK, ni repris la modélisation des tests. Cependant, nous avons adapter le modèle relationnel de données aux nouvelles fonctionnalité développées, cela a pu se faire assez facilement grâce au travail

réalisé précédemment.

3.7.2 Développement

3.7.2.1 Développement d'un PoC

Nous avons choisi de coder TINK en Java J2EE, puisque nous souhaitions développer une plateforme en ligne. De plus le module X69 de Karma BU que nous utilisons pour communiuqer avec les équipements est lui-même codé en Java. Nous avons donc décidé, pour des raisons de simplicité, de coder TINK en Java.

Le module X69 utilise plusieurs frameworks. Pour exposer une interface CWMP en écoute pour les équipements, il utilise le framework CXF qui permet de simplifier l'utilisation et la gestion des Servlet. Afin de communiquer avec les équipements via le protocole CWMP, et donc d'implémenter le protocole côté serveur, le module X69 utilise le framework Spring. Avant même de commencer le développement de TINK, il m'a fallu prendre en main ces frameworks, comprendre leur fonctionnement et leur utilisation dans le cas présent.

La première phase de développment a été faite durant la période de février à mars 2016 [15]. Elle a permis de prendre en main les framework Spring et CXF, mais aussi de comprendre comment se greffer au code de X69 et récupérer les données provenant des messages CWMP qu'il parse. A la fin de cette phase de développement un PoC de TINK a pu être réalisé. Cela consistait dans un premier temps à faire communiquer le module X69 avec un équipement. Dans un second temps je devais réussir à demander l'envoi de certaines méthodes CWMP par X69 à l'équipement, et à en analyser les réponses.

3.7.2.2 Implémentation des bases

La deuxième phases de développement c'est déroulée de mars à août 2016, avec la présence de Jean-Pierre ONA. Elle avait pour objectifs de réaliser une première version de TINK, avec la possibilité de se créer un compte sur la plateforme, tester des équipements,

15. Cette phase a déjà été détaillé dans la partie "Analyse de faisabilité"

et avoir des rapports de test détaillés et précis. Toutes ces fonctionnalités devaient être accessibles depuis la plateforme pour n'importe quel utilisateur. De plus, une majeur partie des tests devaient être présents. Afin de communiquer entre le code Java avec la base de données, nous avons mis en place et pris en main le framework Hibernate. Aucun de nous deux ne connaissait ce framework, il nous a donc fallu un certain temps pour monter en compétence dessus, mais cela nous a par la suite aider. Cette phase a été particulièrement importante puisque nous avons développé à la fois la partis IHM, la partie de logique de test [16] et la communication avec les équipements. Cette phase m'a permis d'acquérir de solides base en développement et en Java.

3.7.2.3 Évolution de TINK

Enfin, la troisième et dernière phase de développement du projet, c'est déroulée durant l'ensemble de ma troisième année. Elle a principalement consisté à prendre en main la partie IHM développée par Jean-Pierre, mais également de rajouter de nouvelles fonctionnalités et tests plus complexes qui avaient été laissé de côté pour la première version par manque de temps. Jusqu'à présent seul des tests réalisés sur une seule session CWMP avait été fait. De plus, les scénarios de test une fois lancés n'avaient pas de temps d'expiration et nous ne pouvions pas les supprimer.

L'une des fonctionnalités était de pouvoir placer un temps maximum d'exécution pour chaque test, modifiable par l'utilisateur. Cela a nécessité de prendre en main l'IHM, mais également de modifier le modèle relationnel de données. Une autre fonctionnalité était de pouvoir récupérer l'intégralité du Data Model d'un équipement, cela n'est pas considéré comme un test mais comme une fonction supplémentaire. Nous avons voulu proposer aux utilisateurs la possibilité d'exporter un rapport de test dans le format .xls depuis l'IHM. Cette nouvelle fonctionnalité a nécessité l'apprentissage de la librairie Apache POI, mais elle rajoute un réel plus pour les utilisateurs qui souhaite nous transmettre leur résultats. Il a également été rajouté la possibilité de supprimer des scénarios de test. Enfin, nous avons voulu réduire le nombre d'informations à apporter par un utilisateur pour ajouter un équipement, ainsi nous avons simplifié le formulaire d'ajout d'équipement et déduit les

16. Par logique de test on entend : validation, déroulement et création des tests pour un équipement donné

informations manquantes durant les échanges entre TINK et l'équipement.

Une nouvelle version du module X69 a été développé par les équipes de Karma BU, avec l'implémentation de méthodes CWMP supplémentaire. Cela a été pour nous l'occasion de tester l'intégration d'une nouvelle version du module X69, et d'évaluer concrètement l'impact sur le code que cela représente. L'importation du module X69 étant uniquement réalisée par une dépendance Maven [17], cela a été transparent. Il nous a seulement fallu prendre en main les fonctionnalités proposées dans cette nouvelle version.

Au niveau des tests supplémentaires, avec l'ajout des temps d'expiration et la gestion du temps durant les tests, j'ai pu mettre en place des tests utilisant des calculs de temps. J'ai également implémenté des tests réalisés sur plusieurs sessions CWMP. En tous nous sommes passés de sept tests à douze, couvrant ainsi l'ensemble des tests que nous nous étions fixé de faire initialement.

De plus, en parallèle du développement de l'ajout de nouvelles fonctionnalités, j'ai été amené à réviser et factoriser le code qui avait été produit lors de ma deuxième année. Le premier but de cette réorganisation de code était d'améliorer les performances. Le second but était de rendre plus propre et maintenable le code pour la suite de la vie de l'application. Cela m'a permis de renforcer mes compétences en développement Java, mais également d'apprendre des bonnes pratiques de code.

Enfin, avec le déploiement de l'application en production sur Kermit en mars 2017, nous avons voulu rendre complètement automatique la génération de la base de donnée au démarrage des trois modules de TINK. Pour ce faire j'ai configuré le framework Hibernate via le code Java afin qu'il créer la base de données sur le serveur de base de données. Il crée également les tables du modèle relationnel par la suite, et les remplis avec les valeur fixées dans le code. Ainsi, l'ensemble du déploiement, des modules à la base de données ce fait automatiquement et ne nécessite aucune intervention extérieur.

17. Maven est un logiciel permettant, entre autre, de synchroniser les dépendances d'un projet Java avec les dépôts d'où elles proviennent. Il permet également de compiler le code Java selon certaines règles définies par le développeur

L'ensemble de ces nouvelles fonctionnalités ont été implémenté sur la totalité de ma troisième année, et m'ont occupé une majeure partie de l'année.

3.8 Lancement de l'application

A partir de mars 2017, nous avons créer deux URL, pour accéder via l'intranet et internet à TINK, lançant ainsi en production l'application. Cela nous à lancer dans une nouvelle phase du projet.

3.8.1 Communication et présentation

Afin de il nous a fallu faire connaître notre plateforme. Nous avons donc communiquer et présenter TINK. Pour cela nous avons tout d'abord présenter l'application à notre département, puis plus largement à différentes équipes susceptibles d'être intéressées par l'utilisation de l'outil. Les présentations se sont déroulées en anglais et en français selon le publique visés. Un texte de présentation de l'application a été rédigé en anglais par Matthieu ANNE, afin de le diffuser sur l'intranet et de le mettre sur la page d'accueil de l'application.

Dès lors que des équipes en interne souhaitaient des informations sur un de leur équipement ou bien tester le comportement CWMP, nous les redirigions vers TINK afin qu'ils puissent tester l'outil. Cela nous a permis d'avoir nos premier utilisateurs, et les premiers retour sur l'application. Certains membres de l'équipe ont également contribué à tester leurs équipements sur TINK et nous faire des retours sur les points d'améliorations et bugs rencontrés.

3.8.2 Support utilisateur

3.8.3 Livrable du projet

3.9 Bilan du projet

3.9.1 Difficultés et solutions

3.9.2 Apport personnel

3.9.3 Les suites pour TINK

4 Transfert de compétences

5 Bilan de compétences

5.1 Environnement professionnel

5.1.1 Connaissance de l'entreprise

Lors de mon arrivée au sein de l'entreprise, une présentation des différents services avec lesquels j'étais susceptible de travailler m'a été faite. Cela m'a permis dans un premier temps d'avoir une connaissance partielle des ressources humaines avec lesquelles je serais amené à travailler tout au long de mes trois années de formation. Grâce à cette connaissance de l'environnement de travail, j'ai pu savoir qui contacter directement en cas de besoins dans de nombreuses situations. Notamment, lorsque j'ai débuté certains projets. Comme je savais que certaines personnes avaient au par-avant abordées des sujets similaires ou dans des domaines équivalents, j'ai pu obtenir davantage de connaissances pour les sujets sur lesquels j'allais être amené à composer. De la même façon, lorsque j'ai rencontré des points bloquant sur la partie technique, j'ai su quelles personnes contacter afin de pouvoir monter en compétences plus rapidement dessus afin de résoudre la difficulté.

Enfin, si au cours de ma première année j'ai souvent eu à contacter différentes personnes pour leur demander des informations ou de l'aide, j'ai vu durant la deuxième année l'inversion progressive des rôles. En effet, des personnes sont venues me demander des renseignements.

5.1.2 Anglais et contexte international

Mes activités et projets de première année ne m'ont pas permis d'interagir avec des équipes internationales. Cependant, lors de ma deuxième année j'ai été amené à travailler avec un membre d'une équipe de conception et développement roumaine. Ce travail en collaboration a eu pour conséquence de me faire monter en compétence non pas uniquement sur l'aspect technique, mais aussi sur mon anglais écrit, puisque nous échangions en anglais par écrit le plus souvent.

De plus, durant les deux premières années j'ai pu traiter et rédiger de nombreux documents en anglais. Lors de mon arrivée au sein de l'équipe, il m'a été aussi demandé de

lire plusieurs documents techniques concernant le domaine dans lequel j'allais évoluer et travailler. L'ensemble de ces documents était rédigé pour la majorité en anglais, ce qui fut pour moi l'occasion d'acquérir des notions en anglais technique. Dans un second temps, j'ai eu à rédiger des documentations et participer à la rédaction de wiki de mon projet de deuxième année. Ces rédactions ont été un moyen de mettre en pratique les notions d'anglais précédemment acquises. Lors de la deuxième année il m'a été demandé de rédiger un cahier des charges ainsi qu'un dossier de spécification intégralement en anglais. Cela m'a ainsi permis de monter en compétence en anglais technique tout au long de ces deux ans. Enfin, lors de ma troisième année j'ai pu continuer de pratiquer mon anglais écrit en dialoguant avec l'équipe roumaine qui maintien la plate-forme où TINK est hébergé. De plus, j'ai eu a rédiger une documentation technique entiérement en anglais, servant de base au futurs dévéloppeurs qui seront amenés à reprendre la suite de TINK.

5.2 Management

5.2.1 Travail en équipe et communication

Les premières missions qui m'ont été confiées n'impliquaient pas de travail collaboratif. Je devais faire des rapports à mon tuteur, selon l'avancée du projet. Ce dernier supervisait alors mes missions, en me laissant toutefois déjà une importante autonomie. Lors de réunion ayant lieu toutes les deux semaines, je présentais mon avancée de manière plus synthétique au reste de l'équipe. J'ai ainsi pu acquérir des notions en communication, ainsi que dans les compétences de restitution et présentation orale.

C'est dans la seconde moitié de ma première année que j'ai pu travailler sur des missions impliquant d'autres collaborateurs. Ce fut pour moi l'opportunité d'apprendre à organiser mon travail en prenant en compte également celui de mes collègues. Pour ce qui est de l'aspect communication, cela m'a permis de renforcer mes compétences en relatant mon travail lors des points d'avancés, mais aussi en étant force de proposition lors des réunions de travail où j'exprimais mon opinion sur les solutions envisagées et proposais d'autres possibilités.

Mon attitude professionnelle et l'évolution de mes compétences en travail d'équipe et en management ont conduit mon tuteur à me confier l'encadrement partiel d'un stagiaire de Master 2 sur la seconde moitié de ma deuxième année. Il m'a été demandé de l'encadrer durant mes périodes en entreprise en répartissant entre nous les tâches composant le projet sur lequel nous travaillons. Cet exercice m'a permis de renforcer davantage ma communication et ma gestion du travail en équipe, en planifiant des réunions et des points d'avancement, ainsi qu'en assignant des activités au stagiaire. Cet exercice s'est correctement déroulé et m'a permis de renforcer ma confiance en moi. Cela m'a également appris à organiser, découper et répartir le travail d'un même projet entre plusieurs ressources.

Mes compétences en communication ont pu être renforcées lors de la dernière année, puisque j'ai eu a réalisé plusieurs présentation en duo de TINK. Ces présentations ont été faite à plusieurs type de publique, nécessitant d'adapter son discours selon les connaissances sur le sujets des personnes présentes.

5.2.2 Gestion du temps et du stress

Les missions qui m'ont été confiées tout au long des six premiers mois de ma formation n'impliquaient pas de date limite de rendu de projet. J'ai cependant eu besoin d'organiser mon travail et gérer le temps que je passais sur chacune de mes activités. Tout au long de la première année mon travail fut partagé entre deux activités principalement. Même si ces activités n'incluaient pas toujours un travail en collaboration, cela a tout de même nécessité l'organisation et la planification de mes temps de travail et actions sur chacune de ces deux activités. Cela m'a permis d'acquérir des notions dans ma gestion du temps.

Ce n'est qu'à partir de la seconde moitié de ma deuxième année que des contraintes de date limite m'ont été imposées. Cela fut notamment le cas dans le projet qui a occupé l'intégralité des deux dernières années, où différents jalons avaient été définis soit initialement, soit au cours du projet. J'ai alors mis en place des outils d'organisation afin de gérer mon temps de manière plus efficace et renforcer encore mes compétences dans ce domaine. J'ai appris à estimer la priorité, la complexité et le temps de réalisation de différentes actions, dans le but d'estimer de la façon la plus efficace possible le temps de travail nécessaire à la réalisation de ces actions.

L'apparition des dates limites et des enjeux à finir une activité dans le temps imparti m'a permis d'apprendre à gérer également mon stress.

5.3 Conduite de projet

5.3.1 Analyse des besoins

Les premières missions qui m'ont été confiées à mon arrivée au sein de l'équipe n'impliquaient pas de réelle analyse des besoins. En revanche, à partir de la seconde partie de la première année, j'ai eu progressivement des projets nécessitant que je recueille des besoins, fonctionnalités et critères essentiels pour mener à bien mes projets. Dans un premier temps cela consistait essentiellement à poser les bonnes questions à mon tuteur, qui m'avait attribué le projet. Progressivement j'ai eu à réaliser des études de faisabilités afin de démontrer que mes solutions étaient viables, ou tout simplement pour sélectionner la solution la plus adéquate au problème posé.

Mon tuteur m'a confié progressivement des missions qui nécessitaient une réflexion plus approfondie sur la recherche de solutions techniques et sur l'analyse de besoins, à partir de la seconde partie de ma première année. Afin que je puisse réaliser ce travail correctement lors de ma deuxième année. L'une des phases de mon projet de deuxième année consistait justement à définir et décrire les besoins, puis réaliser une étude de faisabilité de la manière la plus autonome possible puisque mon tuteur souhaitait que je réalise ces étapes moi-même tout en étant capable d'argumenter mes choix. Le fait d'avoir eu à examiner et à décomposer le projet en intégrant différents paramètres m'a permis d'avoir une certaine prise de recul sur ce que l'on me demandait de réaliser.

5.3.2 Planification et méthode de gestion de projet

Dès ma première année j'ai eu à planifier mes activités. Mais c'est à partir de ma deuxième année que j'ai vraiment pu utiliser des méthodes de projet, notamment l'agilité avec les méthodes Scrum et Kanban. Ces dernières m'ont permis d'organiser et préparer au mieux mon projet et mon travail collaboratif, puisque de cette façon je savais où en étaient mes collaborateurs dans leurs tâches sur le projet. Le fait de mettre en place cette

méthode de projet m'a permis d'apprendre à décomposer le travail et à pouvoir le répartir entre les différents membres de manière égale et la plus adéquate possible en fonction des compétences de chacun et des tâches à réaliser.

Monter en compétence sur la gestion de projet et sur la décomposition des tâches m'aide à préparer au mieux mon travail et avoir une meilleure visibilité sur celui-ci.

J'ai de plus été amené à rédiger un dossier de spécification pour mon projet de seconde année, constitué d'un cahier des charges ainsi que de plusieurs diagrammes UML. Cet exercice de rédaction permet d'avoir encore une fois une meilleure vue sur ce qui est attendu, de son contexte et des différentes étapes. Lors de la troisième année, j'ai réalisé plusieurs diagramme d'activité, classes et séquences permettant d'illustrer le fonctionnement de TINK, ce qui là aussi m'a permis d'avoir une meilleur vision sur le travail à réaliser sur un projet donné.

5.4 Systèmes d'Information

5.4.1 Architecture logicielle

Durant ma première année j'ai pu, dans le cadre du projet de première année, concevoir une application utilisant l'architecture logicielle Model View Controller (MVC). Cela m'a permis d'acquérir des notions dans le domaine de patron de conception. Mon tuteur m'a également encouragé à utiliser d'autre patron de conception lors de l'un de mes projets qui nécessitait que je conçoive une application depuis le dossier de spécification, nécessitant donc de concevoir son architecture.

Lors de ma deuxième année, mon tuteur m'a confié la conception, puis le développement, d'un outil plus complexe que ce que j'avais pu réaliser jusqu'à présent. Cela a nécessité de ma part de prendre connaissance de plusieurs types d'architectures logicielles existantes. À savoir l'architecture n-tiers ; comprendre chacune de ces couches qui composent cette architecture ; leurs rôles et leurs fonctionnements. De plus, j'ai eu besoin de comprendre comment faire communiquer les différents modules de cette architecture, tout en prenant en compte les contraintes et les besoins impliqués dans le projet. Cela

a nécessité de ma part de questionner des collaborateurs, développeurs seniors, ayant un niveau de compétences avancé dans l'architecture de logiciels et le développement. Le fait d'avoir pu contacter des développeurs seniors m'a permis de monter en compétence et d'avoir une importante source de connaissances à disposition. J'ai également récupéré les cours donnés aux étudiants INFRES ED de deuxième année qui abordent ce sujet, ce qui m'a permis d'acquérir davantage de connaissances.

J'ai ainsi conçu l'architecture logicielle d'une application, basée sur une architecture n-tiers, tout en prenant en compte les évolutions futures et la maintenabilité de celle-ci. Cela m'a permis de monter en compétence dans les architectures logicielles.

5.5 Logiciels

5.5.1 Conception et développement d'applications

Lors de mon arrivée dans l'entreprise, mon tuteur m'a fait travailler sur l'étude d'un client embarqué développé en C. Cela dans le but de me faire monter en compétence en développement C, puisqu'il m'a fallu comprendre le fonctionnement de ce client à l'aide de reverse-engineering. Cela avait aussi pour but de me donner des notions en conception d'applications, en devant essayer de comprendre comment ce client avait était conçu. Puis dans un second temps, le modifier afin d'étudier la complexité à rajouter des fonctionnalités. Cette seconde phase m'a permis de mettre en application mes compétences en développement C mais aussi en conception.

Dans la seconde partie de ma première année, afin d'acquérir des notions supplémentaires dans la conception d'application, mon tuteur m'a fait travailler sur la conception d'un outil doté de fonctionnalités simples. La conception de cette application avait pour second objectif de me faire rentrer davantage dans le domaine d'activité sur lequel j'allais travailler durant mes trois ans de formation, en utilisant des concepts de cet outil. Cela m'a permis de monter à la fois en compétence sur les aspects de bases de conceptions d'applications, mais également de renforcer mes compétences en développement, particulièrement sur le langage orienté objet Java.

Lors de ma deuxième année, il a été décidé de me faire concevoir et développer un outil plus complexe que l'application implémentée en première année. Ce projet a occupé ma deuxième et troisième année. De plus, le travail de conception a été réalisé en travail collaboratif avec un second collègue. J'ai également pu demander des conseils, des explications, et des aides à différents développeurs seniors pour la conception de cet outil. De plus le développement de celui-ci s'est effectué en Java et m'a permis de travailler avec de nombreux frameworks fréquemment utilisés et relativement complexes à prendre en main. Le fait d'avoir pu apprendre à utiliser ces frameworks m'a permis d'acquérir de solides bases et compétences en développement Java.

De part la conception, l'étude et le développement de ces différentes applications, dans les langages C et Java, j'ai pu monter en compétence dans la conception logicielle et le développement d'applications.

5.6 Base de données

5.6.1 Création et implémentation de modèle de données

Lors des projets de cours de première et deuxième année j'ai eu à concevoir des modèles de données. Un premier, pour le projet de création de site web effectué à Bristol. Et un second durant le projet de deuxième année qui nécessité la création d'un site web la encore. Cela m'a permis de renforcer mes compétences en conception de modèle de données, mais aussi en langage MySQL, puisque leur création c'est fait sur des serveurs MySQL.

Lors du projet d'entreprise TINK, j'ai également eu à concevoir de manière agile, un modèle de données. Ce modèle a dû être revu et amélioré tous au long du projet. Là encore cela a été réalisé en MySQL.

J'ai ainsi pu renforcer mes compétences en création de modèle de données tous au long de ma formation, autant en entreprise qu'en école.

5.6.2 Gestion et administration de SGBD

Les connaissances de gestion et administration de Système de Gestion de Base de Données (SGBD) ont été en premier acquises par les cours suivis en première année, puis renforcées en deuxième année. Cette fois-ci cela était pour des serveurs Oracle et non MySQL, permettant ainsi de voir les différences de ces deux types de SGBD.

Les projets de première année et deuxième année précédemment sités m'ont également permis de monter en compétences en administration de SGBD, mais MYSQL cette fois.

En entreprise, c'est par le projet TINK là encore, que j'ai pu renforcer mes compétences en administration de SGBD, en particulier sur MySQL. J'ai pu créer différents compte, leur attribuer des droit en conséquences, et également réaliser des procédures stockés.

5.7 Systèmes et Réseaux

5.7.1 Administration et maintenance d'OS

Depuis le début de ma formation je travaille la majorité de mon temps sur des machines Linux, ce qui favorise le développement de mes compétences d'administration OS Linux et le langage orienté OS associé. De plus, lors de ma première année j'ai eu à recréer un réseau opérateur dans le cadre de tests d'équipements. Pour cela il m'a fallu configurer différents services et serveurs, me permettant de renforcer mes compétences en administration de système. J'ai par ailleurs eu à configurer et déployer des serveurs d'applications dans le cadre de ma deuxième et troisième année, ce qui a là encore pu renforcer mes compétences dans ce domaine acquis une première fois lors de mon DUT, mais aussi lors des cours de deuxième année que nous avons pu avoir.

Durant les périodes d'école de ma deuxième année j'ai également pu monter en compétence en administration d'OS, tant Linux et Microsoft, grâce aux différents cours qui m'ont été donnés.

Tous au long des deuxième et troisième années, j'ai pu monté en compétence sur certaines technologies de conteneurisation. En premier lieu grâce à un auto apprentissage

lors du projet d'école de deuxième année, où mon équipe et moi-même sommes montés en compétence sur LXC et Docker. Puis en entreprise, où ces connaissances ont pu être renforcées et réutilisées lors de l'hébergement de l'application du projet TINK basé sur Docker.

5.7.2 Protocole de Communication

Tout au long de ma formation, mon domaine d'activité en entreprise a nécessité que j'ai une forte connaissance du protocole de communication client/serveur, nommé CWMP. Ce protocole est une importante brique de l'ensemble de mes projets. Il m'a donc été nécessaire de monter rapidement en compétence sur ce protocole dès mon arrivée dans l'entreprise. L'apprentissage de ce protocole m'a permis de monter en compétence dans mes connaissances des protocoles de communication.

Les différents cours suivis lors de ma deuxième et troisième année, en réseaux et en télécommunications, m'ont également permis de monter en compétence sur d'autre protocole de communication, tels que les protocoles lié au réseaux mobile.

5.8 Axe d'amélioration

Ainsi que l'on peut le voir en annexe avec le *Tableau d'auto-évaluation de compétences*, il y a de nombreuses compétences qui n'ont pas pu être encore développées ou approfondies. Cela s'explique par les sujets couverts jusqu'à présent lors des cours proposés à l'école, ainsi qu'aux missions effectuées en entreprise, qui ne peuvent bien évidemment pas couvrir l'ensemble des domaines du référentiel de compétence. Il est donc normal de voir des compétences plus avancées que d'autres, puisque l'on ne peut pas travailler sur toutes. Toutefois, il me semble intéressant pour mon futur professionnel de renforcer davantage mes compétences dans les domaines de *Systèmes et Réseaux*, *Conduite de projet* et *Systèmes d'Information*. Ce sont pour moi des domaines de compétences qui m'attirent et vers lesquels je souhaiterai davantage me tourner.

5.9 Conclusion

Comme nous avons pu le voir tout au long de cette partie, ma première année fut pour moi riche en matière d'acquisitions de bases et notions dans de nombreux domaines de compétences. La seconde année m'a permis de renforcer ces compétences en améliorant ma manière de travailler, d'appréhender un problème, ou encore de faire face à un problème. Durant cette deuxième année, j'ai pu acquérir de nouvelles compétences, qui ont été elles aussi renforcées durant ma dernière année. Ces différentes phases d'acquisitions des compétences de l'ingénieur tendent ainsi à me rapprocher davantage du statut d'ingénieur souhaité par l'école.

6 Conclusion

6.1 Atteintes des objectifs

6.2 Progression

6.3 Synthèse de parcours

Annexe A

RPC Method CWMP

RPC Method	Description
InformResponse	Permet à l'ACS de répondre à un Inform.
GetParameterValues	Permet de récupèrer la valeur d'un ou plusieurs paramètre du datamodelg passé en paramètre.
SetParameterValues	Permet de modifier la valeur d'un ou plusieurs paramètre du datamodelg passé en paramètre.
Reboot	Permet à l'ACS de demander au CPE de redémarrer.
FactoryReset	Permet à l'ACS de demander au CPE de se réinitilisaer à l'état d'usine.
ScheduleInform	Permet à l'ACS de demander au CPE de revenir initilisaer une session TR-069 dans t seconds.
Download	Permet à l'ACS de demander au CPE de télécharger un fichier, souvent un nouveau firmware.

TABLE A.1 – Liste des RPC méthodes devant être implémentées par l'ACS.

RPC Method	Description
Inform	Permet au CPE d'initier une session TR-069.
GetParameterValuesResponse	Permet au CPE de répondre à un GetParameterValues en indiquant la/les valeur(s) du/des paramètre(s) demandé(s) par l'ACS.
SetParameterValuesResponse	Permet au CPE de répondre à un SetParameterValues en indiquant si la demande a pu être réaliser.
RebootResponse	Permet au CPE d'acquitter l'ordre de redémarrage.
FactoryResetResponse	Permet au CPE d'acquitter l'ordre de réinitilisation à l'état d'usine.
ScheduleInformResponse	Permet au CPE d'acquitter l'ordre de ScheduleInform.
DownloadResponse	Permet au CPE d'acquitter l'ordre de Download.

TABLE A.2 – Liste des RPC méthodes devant être implémentées par le CPE.

Annexe B

Tableau d'auto-évaluation de compétences

Domaines	Familles d'activités	Niveaux de maîtrise				
		1	2	3	4	5
A. Ingénierie	A.1. Méthodologie		V			
	A.2. Analyse		V			
	A.3. Modélisation		V			
B. Environnement socio-économique	B.1. Anglais, contexte international		V			
	B.1. Connaissance de l'entreprise, économie		V			
C. Management	C.1. Communication		V			
	C.2. Travail collaboratif		V			
	C.3. Ethique de l'ingénieur		V			
	C.4. Responsabilité juridique	V				
	C.5. Gestion du temps, du stress		V			
D. Conduite de projet	D.1. Analyse des besoins		V			
	D.2. Relation et satisfaction client	V				
	D.3. Planification, organisation		V			
	D.4. Calcul des coûts	V				
	D.5. Négociations	V				
	D.6. Relations contractuelles	V				
	D.7. Livraison de produit ou service	V				
E. Systèmes d'Information	E.1. Modélisation, Conception		V			
	E.2. Architecture d'application		V			
	E.3. Urbanisation	V				
F. Logiciels	F.1. Conception d'applications		V			
	F.2. Développement		V			
	F.3. Test	V				
	F.4. Maintenance	V				
	F.5. Support		V			
G. Normes et procédures de sécurité	G.1. Sécurisation des échanges des systèmes	V				
	G.2. Sécurité des réseaux	V				
H. Bases de données	H.1. Création, optimisation de SGBD	V				
	H.2. Gestion et administration de SGBD	V				
I. Systèmes et Réseaux	I.1. Administration et maintenance d'OS		V			
	I.2. Gestion matérielle et logicielle		V			
	I.3. Protocoles de communication		V			
	I.4. Administration de services réseaux		V			
	I.5. Déploiement, optimisation		V			
	I.6. Maintenance évolutive et corrective	V				

FIGURE B.1 – Tableau d'auto-évaluation des compétences

Glossaire

ACS Type de serveur permettant d'administrer et gérer des équipements TR-069 que les clients Orange possèdent.. 11

ADSL Technique de communication, permettant de transmettre et recevoir sur le réseau téléphonique des données numériques de manière indépendante du service téléphonique conventionnel. 15

API is a particular set of rules and specifications that a software program can follow to access and make use of the services and resources provided by another particular software program that implements that API. 46

BBF Consortium à but non-lucratif se concentrant sur le développement de protocole de réseaux télécoms. 12

CARE Équipe dans laquelle j'évolue depuis 3 ans, constituée d'une quinzaine de chercheurs/ingénieurs. 10

COM Module de Karma permettant d'ordonnancer le traitement des requêtes des équipements clients selon leur contenu. 38

CPE Nom défini par le BBF pour désigner un équipement client. 11

CWMP Protocole du Document TR-069. C'est le protocole de communication employé pour que les serveurs ACS et les équipements puissent dialoguer. 12

Data Model Modèle de données sous forme d'arbre contenant l'ensemble des caractéristiques d'un équipement. Il peut aller de quelques dizaines de paramètre à plusieurs milliers selon l'équipement. En modifiant les paramètres du data model d'un équipement, on peut modifier son fonctionnement. Chaque équipement respectant le Document TR-069 doit implémenter un data model, qu'il renvoie à son ACS. 13, 15, 21, 24, 26, 50

DHCP Attribue un paramétrage IP (adresse IP, masque de sous-réseau etc) aux machines qui le contactent, afin qu'elles puissent communiquer avec d'autres réseaux. 15

DNS Traduit des noms de domaines en l'adresse IP de la machine qui porte ce nom. 15

Docker Logiciel libre automatisant le déploiement d'application dans des conteneurs. 33, 34

Document TR-069 Document édité par le BBF. Il contient l'ensemble des méthodes et spécification à respecter pour implémenter la norme CWMP. 12, 13, 17, 18, 20, 21, 23, 24, 26, 29, 31, 70

DSLAM Appareil mis en place par le fournisseur d'accès, ici Orange. Il a pour objectif de faire transité le trafic issu du client vers le réseau opérateur afin de fournir un accès internet. Il se trouve juste avant l'arrivée chez le client. 15

Firmware Ensemble d'instruction et structure de données qui sont intégrées dans un matériel informatique pour qu'il puisse fonctionner. 11, 12

Framework Ensemble de composants logiciels permettant de faciliter la construction et la maintenance d'un programme. 39

IHM Interface graphique permettant à un utilisateur final de communiquer avec l'application. 14

Kanban Méthode agile, consistant à diviser le travail en différentes cartes limitées afin de ne pas surcharger les développeurs à un instant t. 40

Karma ACS utilisé et développé par Orange en production. 4, 18, 20, 31, 32, 35, 36, 37

Karma BU Branche internationale de Karma. Instance de Karma destinée aux filiales d'Orange autres que la France. 36, 38, 39, 46, 49, 51

Karma Fr Branche française de Karma. Instance de Karma destinée aux clients français.. 36, 38

Kermit Outil de service cloud de PaaS d'Orange basé sur Open Shift. 4, 33, 34, 41, 48, 51

LAN Réseau informatique tel que l'on peut le retrouver chez soi. 11

Release Version livrable du projet après plusieurs Sprint. 33, 41

Reverse Engineering Consiste à étudier le fonctionnement apparent d'un logiciel ou application pour déterminer le fonctionnement interne. 39

Scrum Méthode agile, consistant à découper le projet en différents Sprints qui aboutissent chacun à une démonstration du produit. 40

Servlet Permet l'extension des fonctions d'un serveur web sur lequel elle est déployée. Une servlet est une application web java. 24, 25, 26, 27, 28, 40, 49

SGBD Permet à un utilisateur de manipuler une base de données. Il peut être sous la forme d'un logiciel applicatif, ou bien d'un serveur. Certain peuvent proposer leur propre langage. Parmis les plus connu on peut citer Oracle, MySQL, DB2... 61

sniffeur réseau Permet d'analyse le trafic circulant sur un réseau. Il permet de détecter et filtrer les trames que l'on souhaite observer. Il se trouve sous forme de logiciel à installer sur une machine. 16

Sprint Période de développement de quelques semaines au bout de laquelle on livre une version potentiellement livrable du produit. 33, 41, 42, 43, 48

Stand-up daily meetings Réunion quotidienne d'une quinzaine de minutes au maximum faite debout où chacun explique où il en est et ce qu'il va faire. 41

TINK Nom du projet réalisé durant ma deuxième et troisième année correspondant à l'outil de test d'équipement en ligne du Toolkit TR-069. 2

Trackers Outil de traçage, permettant de tenir à jour l'avancement et les détails d'une tâche. 40

User Stories Regroupement de fonctionnalités que le logiciel doit embarquer. Constitue les Sprint. 33, 40, 41, 42

Web Services Programme exposé sur internet ou intranet permettant la communication et les échanges de données entre application de manière synchrone ou asynchrone. Dialogue le plus souvent par HTTP. 36

X69 Premier module de Karma interrogé par les équipements souhaitant communiquer avec Karma. Il permet de rendre compréhensible les trames CWMP des équipements clients pour Karma, et inversement de modifier les trames créées par Karma pour qu'elles soient compréhensibles pour les équipements. 31, 37, 38, 39, 40