Sorbonne Université

Master 2 Science et Technologie du Logiciel
2018 - 2019

Rapport de projet de fin d'études

Automatisation de Test fonctionnel

Etudiants :
ZHANG Qilin

Tuteurs :
CHRISTIANSEN Camilla,
COUDRAY Sébastien

Septembre 2019

Table des matières

Table des figures

Liste des tableaux

1 Remerciements

Je tiens tout d'abord à remercier toutes les personnes qui m'ont aidé au succès de mon alternance de cette année et de la rédaction de ce mémoire.

Je voudrais dans un premier temps remercier à l'ensemble de l'équipe Finance France de SAP France, pour son accueil et sa bienveillance tout au long de cette année. Je tiens à remercier particulièrement des personnes suivantes :

— Mes tuteurs, Monsieur Sébastien Coudray, Developer Senior, et Madame Camilla Christiansen, Senior Quality Specialist, avec qui que je travaille tout au long de l'année, pour leurs esprits professionnels, leurs patiences pour m'expliquer et m'aider pour mon travail.

— Monsieur David Dufour, Development Senior Manager IMS, pour son implication dans mon recrutement au sein de SAP France.

— Monsieur Thibault Lefaix, Development Manager, pour son explication qui m'a beaucoup aidé à comprendre l'architecture Financial Consolidation et à finir ce mémoire.

— Monsieur Thierry Gosselin, Quality Expert, Scrum Master, pour sa façon de gérer l'équipe et le projet Financial Consolidation.

Mes remerciements ne sont pas envoyés que les personnes indiquées dans la liste précédentes, y contient aussi tous les collègues d'équipe, durant cette année de travail, ils m'ont aidé à bien "On-Boarding" à SAP France, m'ont aidé à déboguer mes programmes, m'ont pris le temps pour discuter mon sujet d'alternance.

Je remercie également toute l'équipe pédagogique de Sorbonne Université et de CFA-INSTA, particulièrement monsieur Binh-Minh Bui-Xuan et monsieur Emmanuel Chailloux pour leurs conseils tout au long de l'année, madame Emilie Auger pour son travail administratif de la promotion. Je remercie aussi la promotion M2 STL 2018 auprès de laquelle j'ai passé une année formidable, avec qui j'ai passé des très belles histoires, ces dernières vont tenir pendant longtemps dans ma tête.

Enfin, je pense à ma famille, mes proches, mes meilleurs amis, qui m'ont encouragé et soutenu pour ma vie et mes études en France tout au long de ces années.

2 Introduction

2.1 Cadre

Ce rapport est écrit dans le cadre de ma formation en Master 2 Informatique, spécialité de Science et Technologie du Logiciel à Sorbonne Université, dans ce cadre, j'effectue une année d'alternance entre un parcours théorique auprès des intervenants et professeurs à Sorbonne Université et à CFA INSTA (Institut National Supérieur des Technologies Avancées), et une expérience professionnelle chez SAP France en tant que développeur de test automatisé.

2.2 Organisation du rapport

Ce rapport décrit l'essentiel de contenu de mon alternance, il est divisé globalement en trois parties :

1. La première partie fait une présentation d'organisation où j'effectue mon alternance, il parle de SAP SE, de SAP France et l'équipe dans lequel je travail.
2. La deuxième partie décrit le contexte et la problématique du projet : Automatisation des tests d'impressions de l'application *SAP Financial Consolidation* Client Web HTML5.
3. La troisième partie parle des travaux que j'ai réalisé, il décrit la solution qui a été développé pour résoudre le problème de la deuxième partie.

3 Présentation d'entreprise

3.1 Vue globale

SAP SE (***S**ysteme, **A**nwendungen und **P**rodukte in der Datenverarbeitung*, "***S**ystems, **A**pplications & **P**roducts in Data Processing*") est une entreprise qui conçoit et vend des logiciels, notamment des systèmes de gestion et de maintenance, principalement à destination des entreprises et des institutions dans le monde entier. Elle est fondée par 5 anciens employés d'IBM - Dietmar Hopp, Hans-Werner Hector,Hasso Plattner, Klaus E. Tschira, et Claus Wellenreuther en 1972. SAP est une entreprise internationale et son siège se localise à Waldorf, Allemagne.

[1] SAP est le premier éditeur de logiciels en Europe et le quatrième dans le monde. SAP a 98332 employés au tour du monde dans plus de 147 pays jusqu'à 30 Juin 2019. La stratégie de SAP est "The best-run business", le quel SAP veut aider le monde et améliorer la vie de tout le monde. SAP fournit les services pour plus de 437000 clients dans plus de 180 pays, 92% des clients sont dans la liste de Forbes Global 2000 companies. Pour l'année 2018, SAP a une revenue globale de 24,47 billons euros (non-IFRS) au tour du monde.

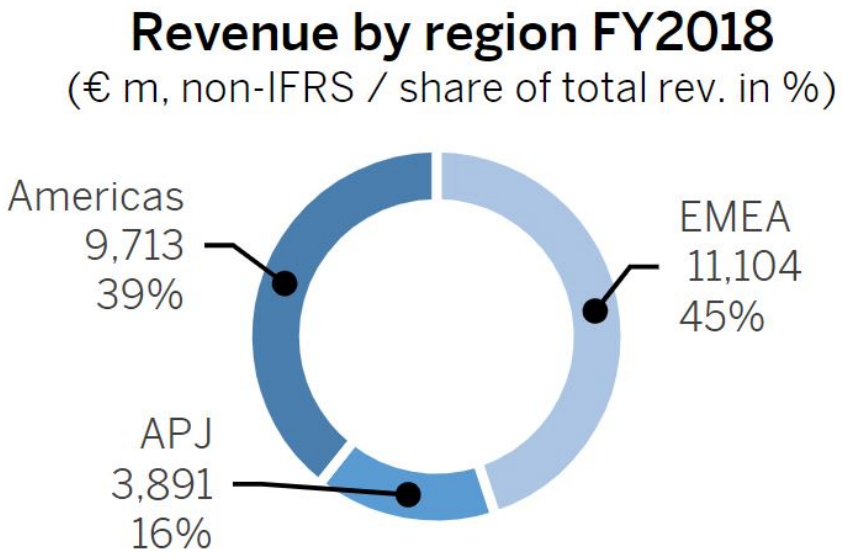


FIGURE 2 – Revenue par région FY2018

3.2 Commerce de SAP

3.2.1 Activités et marchés

[3]SAP opère dans trois zones géographiques : Europe/Moyen-Orient/Afrique (**EMEA**), **Amériques** (le siège de SAP America, qui regroupe Amérique du Nord et Amérique latine, se trouve à Newtown Square, en Pennsylvanie), et Asie/Pacifique/-Japon (**APJ**, qui regroupe Japon, Australie, Inde et plusieurs autres pays d'Asie). SAP dispose également d'un réseau de 115 filiales, et de cabinets de Recherche & Développement.

SAP se concentre sur six secteurs : procédés industriels de fabrication, d'assemblage, de distribution et de services aux consommateurs, services financiers et publics. La société propose plus de 25 produits aux grandes entreprises et plus de 550 pour les petites et moyennes entreprises.

Comme le portfolio de produit SAP ***3*** indique ci-dessous, les produits de SAP divise en 8 catégories :

— ERP et cœur numérique
— Gestion de la relation client et expérience client
— Réseau et gestion des dépenses
— Chaîne logistique digitale
— RH et implication du personnel
— Plateforme digitale
— Outils d'analyse
— Technologies intelligentes

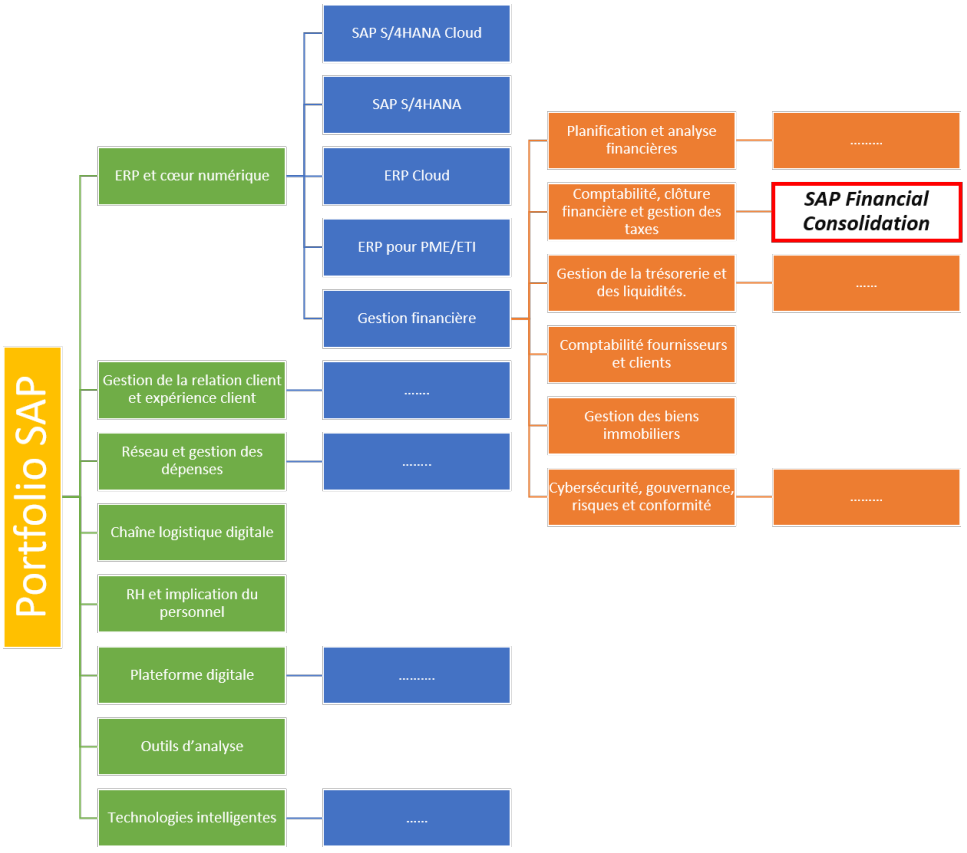


FIGURE 3 – Portfolio de SAP

Chaque catégorie contient plusieurs sous catégories et les sous catégories contient aussi plusieurs produits, le produit que je travaille au-dessus s'appelle *SAP Financial Consolidation (FC)*, il est dans la catégorie de Gestion Financière, c'est un produit conçu pour les comptabilités, clôture financières et gestion des taxes. Il y aura une explication de ce produit plus précisée dans la partie 4.1

Les différents départements opérationnels de SAP sont divisés en trois catégories : Recherche & Développement, activités de terrain et assistance aux consommateurs. SAP Labs conçoit et développe les produits, tandis que des bureaux installés dans chaque pays partenaire se chargent des opérations de terrain comme la vente, le marketing et le conseil. La stratégie et la gestion globale sont décidées et menées au siège de SAP AG, en Allemagne, tout comme le travail d'ingénierie lié à la fabrication des produits.

3.3 SAP Aujourd'hui et Demain : SAP vers Cloud

Depuis 2012, SAP a acquis plusieurs sociétés qui vendent des produits dans le cloud pour être compétent dans le marché Cloud. Par exemple : En 2014, SAP a acheté Concur Technologies, un fournisseur de logiciels de gestion des déplacements et des dépenses en nuage

SAP a aussi coopéré avec les autres entreprises comme IBM, Microsoft, Google pour mieux fournir les services cloud aux clients. SAP a également investi IOT(Internet of Things) et lancé sa propre cloud solution : SAP HANA Solution. En 2015, SAP a lancé SAP S/4 HANA, une nouvelle génération de SAP Business Suite, ce produit est écrit nativement pour le SAP HANA plateforme .

3.4 SAP France

Après l'acquisition de *BuisinessObjects* en Octobre 2007, SAP SE a fondé sa filiale en France - SAP France.

Aujourd'hui, SAP France présente plus de 1500 de salarié en temps plein. Le siège local de SAP France est à **Tour SAP** 35 rue d'Alsace, 92300 Levallois-Perret, où j'effectue mon alternance. SAF France présente aussi un centre de formation et un SAP Labs in Paris qui sont aussi se situent à Tour SAP ; 2 bureaux de vente à Lyon et à Toulouse ; 2 autres SAP Labs France à Caen et Sophia Antipolis.

3.4.1 Equipe Finance

L'équipe dans laquelle j'effectue mon alternance se situe à la Tour SAP répartie principalement sur le 7ème étage de la tour mais aussi au 9ème étage de la Tour SAP.

L'équipe nommée formellement P&I S/4HANA LoB Finance France (équipe "Line of Business Finance France" de l'équipe "Product and Innovation S/4HANA"), et communément appelée équipe Finance, fait partie du département P&I Analytics DW IMS EPM de SAP Labs. Les SAP Labs sont les principales entités de R&D de SAP, développant et améliorant constamment les solutions clés de SAP. SAP Labs Paris est fondé en 2009, il se concentre sur l'application des technologies de SAP, tels que la conversation, l'intelligence artificielle, Machine Learning automatique et l'automatisation intelligence des processus robotiques.

Organisation du groupe L'équipe composée de 43 personnes, David DUFOUR est le manager. Dans cette équipe, une sous-équipe nommée FC (Financial Consolidation) développe le produit éponyme. Les produits qui sont maintenus et développés par cette équipe sont des logiciels d'aide à consolidation financière. David DUFOUR est au management de cette équipe, dont je fais partie.

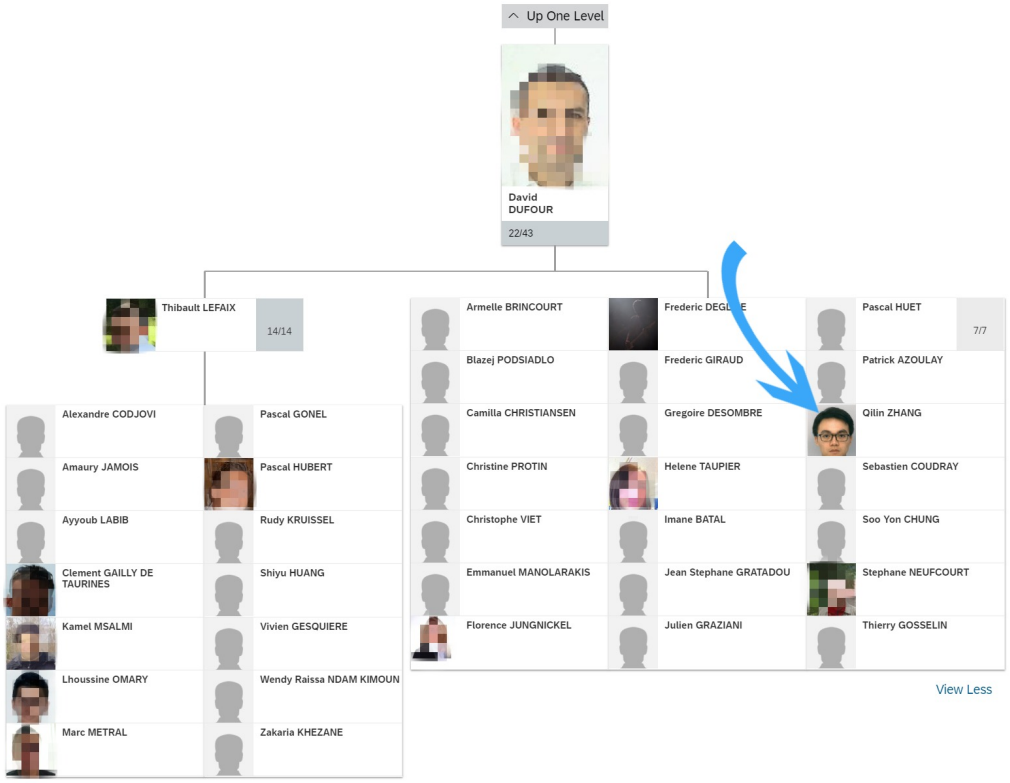


FIGURE 4 – Organisation du groupe

Mon rôle et ma mission Mon rôle durant mon alternance est développeur de test automatisé, Camilla CHRISTIANSEN et un autre Senior Quality Specialist Amaury JAMOIS, qui sont très compétents sur le produit Financial Consolidation rédigent les scénarios de test, je le mets en place et intègre sur notre projet Test-Automatisé. Mon tuteur technique COUDRAY Sébastien va m'expliquer et m'aider à déboguer si je rencontre des difficultés techniques.

4 Analyse et Compréhension de la problématique

4.1 SAP Financial Consolidation

SAP Financial Consolidation est une application de consolidation légale et de reporting de gestion développée depuis 1999. Il permet aux grandes entreprises de répondre à des exigences de consolidation complexes, de rationaliser la conformité réglementaire, d'unifier le reporting légal et de gestion et d'accélérer le processus de clôture financière global.[2]

Avec toutes les évolutions des nouvelles technologies, aujourd'hui SAP Financial Consolidation contient plusieurs versions de clients selon les besoins d'utilisateurs :

Client Windows Desktop : Client installer sur Windows

Client Legacy Web : Client version Web avant l'utilisation de HTML5

Client HTML5 Web : Client version Web avec l'utilisation de HTML5, nouvelle UX.

4.1.1 Architecture

Architecture globale Le produit SAP Financial Consolidation est conçu et développé comme une *n-tiers application* :

- **Couche de présentation :**
  - Une couche de ASP .NET web présentation permet de soutenir les HTTP clients légers : Web Client pour l'entrée de donnée et le contrôle de package ; Web Admin permet d'avoir une administration au niveau de Web ; Excel Web Schedule et Excel Web Links qui est présenté comme un plugin FC dans Microsoft Office Excel.
  - Une nouvelle couche Web de HTML5/SAPUI5 qui est introduise à partie de FC 10.1.
  - Une couche de Web Service permet d'intégrer avec autres applications : Legacy .NET SOAP Web Service sont utilisés par plusieurs clients de FC et de EPM produits ; New .NET REST Web Service pour le nouvel HTML5 Web Client.
- **Couche de traitement :** Le cœur est un **DCOM** (*Distributed Component Object Model*) **server** développé en C++ qui permet de communiquer avec la partie Financial Consolidation Server.

- **Couche d'accès aux données :** Une couche d'accès aux bases de données permet d'accéder la base de donnée (SQL Server, Oracle ou SAP Hana).

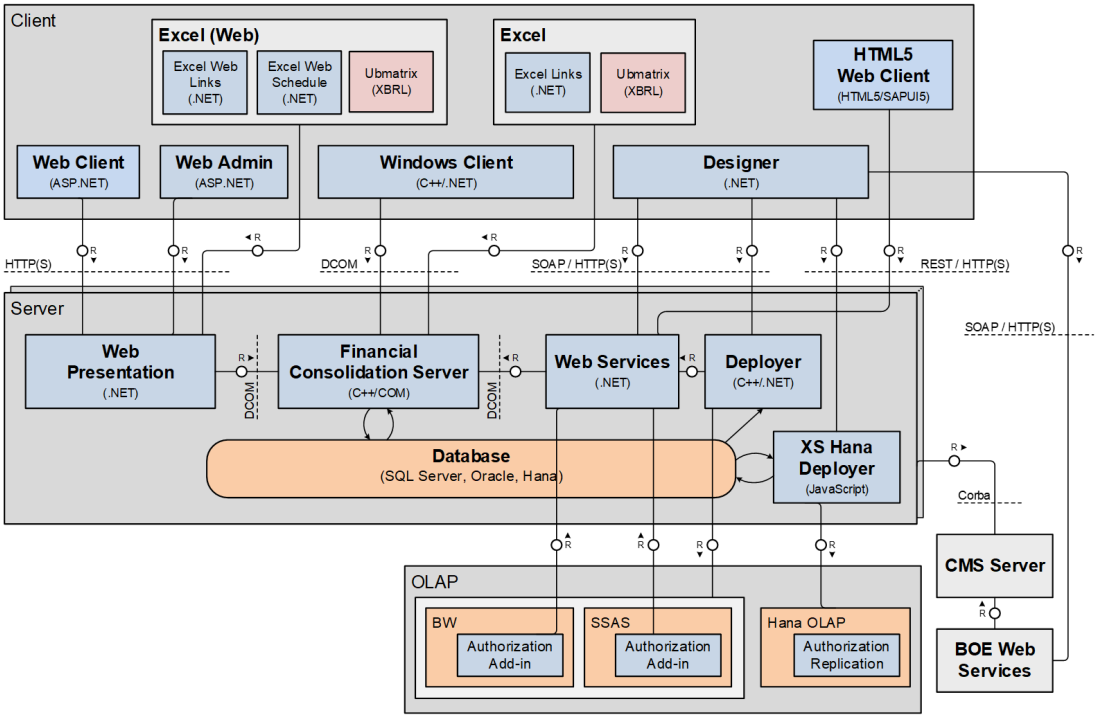


FIGURE 5 – Financial Consolidation architecture globale

Architecture du Client HTML5 Web Avec l'évolution des nouvelles technologies et des besoins de nos clients et existences de beaucoup d'inconvénients pour le client Windows par exemple les traitements sont à locale qui demande beaucoup d'espace sur la machine, en plus, il faut une personne s'occupe d'installation et de la mise à jour afin d'être cohérent entre deux machines. Avec les raisons ci-dessus, l'équipe a décidé de développer un nouveau client HTML5 Web qui a livré aux clients à partir de la version FC 10.1 pour remplacer l'ancien client Web développé avec ASP .NET. La première version de Client HTML5 Web a été sortie en 2015, l'équipe continue à développer les nouvelles fonctionnalités et à faire les maintenances pour ce client.

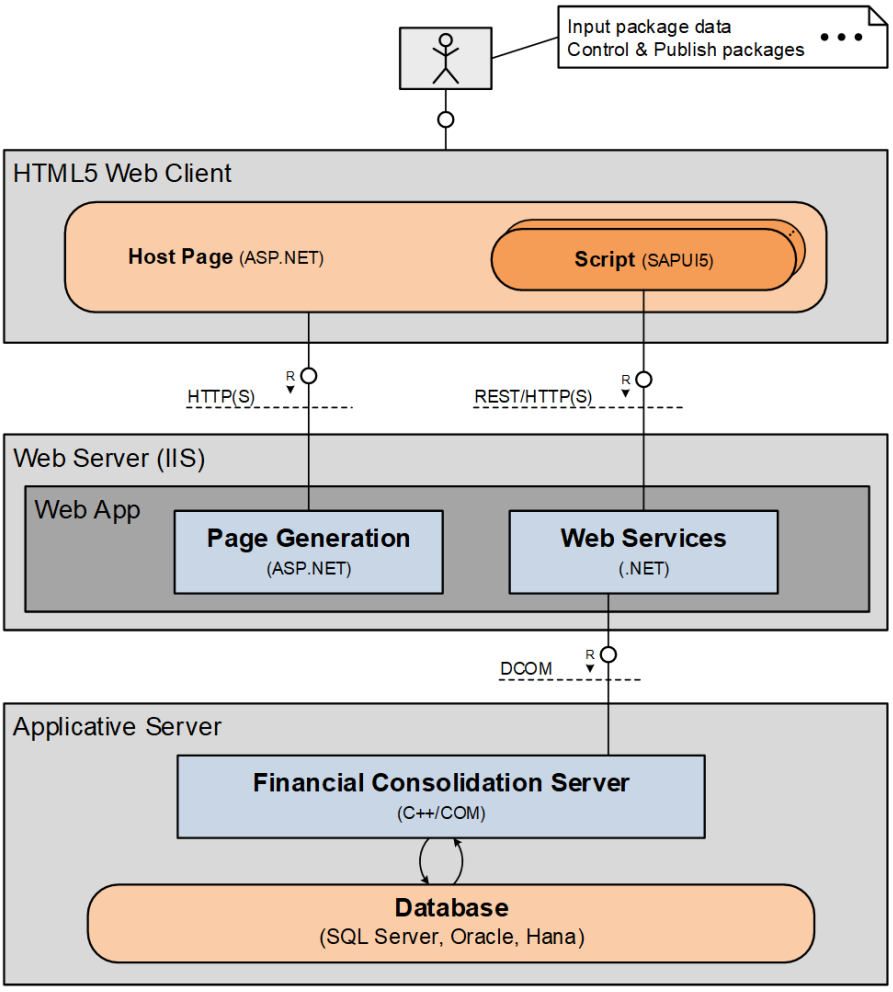


FIGURE 6 – Financial Consolidation HTML5 Web Client Architecture

Le Client HTML5 Web utilise la technologie SAPUI5 et fonctionne sur IIS(*Internet Information Services*), il communique avec le Web Serveur en appelant le REST Services écrits en C#, ce dernier communique avec application serveur par DCOM afin de finir les traitements.

4.1.2 Méthodologie du travail - Agile

Le produit FC existe depuis presque 20 ans, à l'époque, l'équipe FC avait travaillé avec les méthodes comme *Modèle en cascade*, *Cycle en V* comme la figure indique ci-dessous :

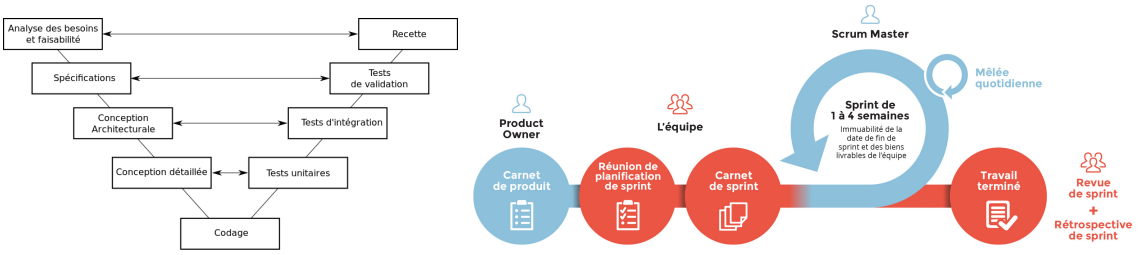


FIGURE 7 – Cycle en V vs Agile

La méthode de cycle en V sépare les testeurs et les développeurs, cette méthode propose de retravailler par exemple la conception détaillée si les tests unitaires ne se valident pas ou de retravailler les spécifications si les tests de validation ne passent pas, contrairement, cette méthode ne convient pas beaucoup au besoin, elle prend beaucoup de temps à retravailler et faire la régression. Avec ces inconvénients de cycle en V et l'évolution de méthodologie de gestion du projet, SAP a demandé à l'équipe FC de travailler avec la méthode Agile, qui s'adapte très bien aux besoins du projet FC.

Nous avons un tableau de mission collé sur mur qui indique l'avancement de chaque mission comme suivant :

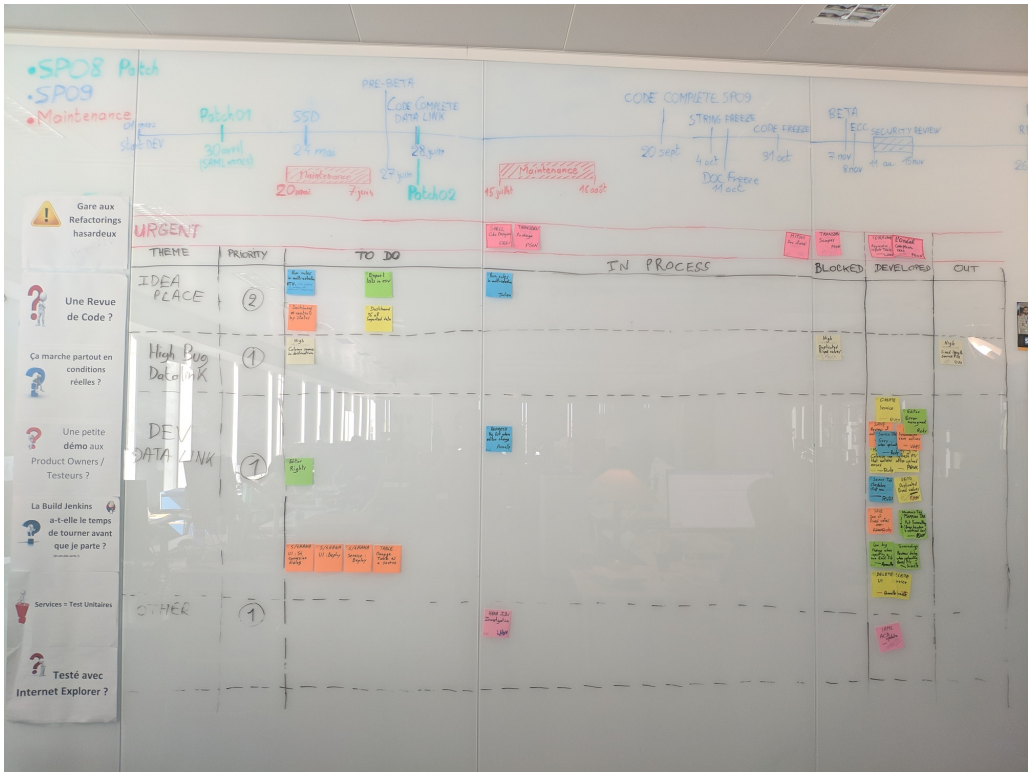


FIGURE 8 – Scrum - Table des tâches

Chaque jour entre 13h40 et 14h00, notre Scrum Master Thierry GOOSSELIN organise une "Stand-up" réunion, avec *Skype for Business* pour les collègues qui sont en télétravail, dans cette réunion, tous les membres d'équipe participent et restent debout au cours de laquelle chacun répond principalement à 3 questions :

- Qu'est-ce que j'ai terminé depuis la dernière réunion ?
- Qu'est-ce que j'aurai terminé d'ici la prochaine réunion ?
- Quels obstacles me retardent ?

Et pour chaque remonte du code importante, on a nos propres règles à respecter :

- Est-ce que j'ai fait une revue de code ?
- Ça marche partout en condition réelles ?
- Une petite démonstration aux Product Owners / Testeurs ?
- La Build Jenkins a-t-elle le temps de tourner avant que je parte ?
- Testé avec Internet Explorer ?

4.1.3 Build et Livraison, Maintenance et Update

L'équipe FC a une organisation robuste pour la maintenance et la mise à jour des produits. Pour chaque version de Financial Consolidation, il y a plusieurs SP (*Support Package*), chaque SP a plusieurs Patch, on livre un nouveau SP environ chaque 9 mois qui contient des nouvelles fonctionnalités et des Bugs fixés, on livre un nouveau patch environ chaque mois qui ne contient que des Bugx fixés.

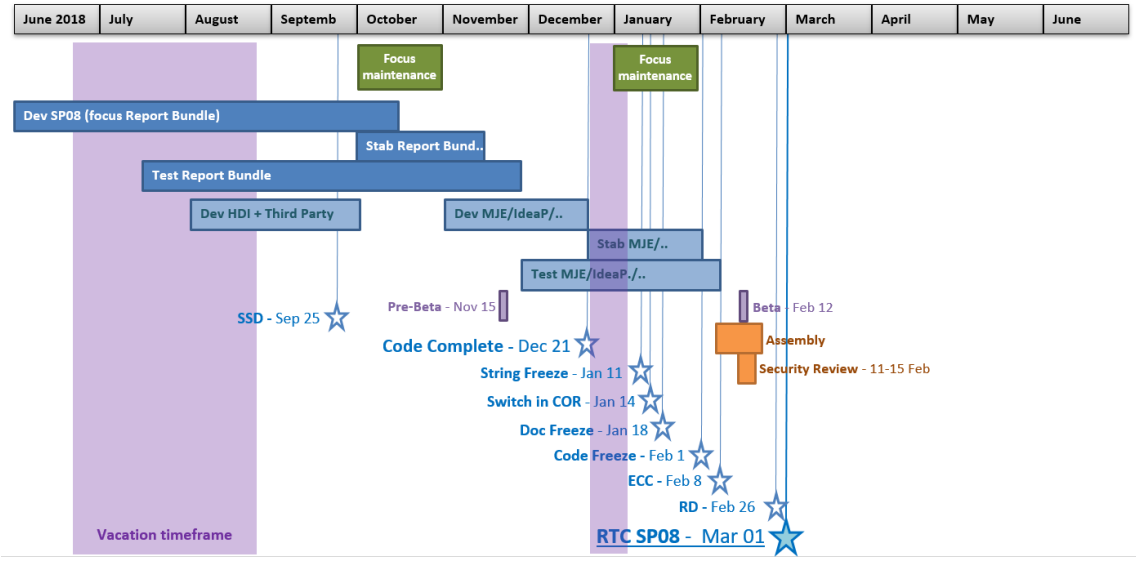


FIGURE 9 – Time Line de Support Package

Quand on reçoit un bug envoyé par client, en raison de travailler avec la méthodologie d'Agile, chaque développeur d'équipe peut prendre ce bug, ce fixe de bug va livrer au client dans le prochain patch. Dans l'image ci-dessous, on peut voir que pour le SP07, nous avons livré 6 patchs en total.

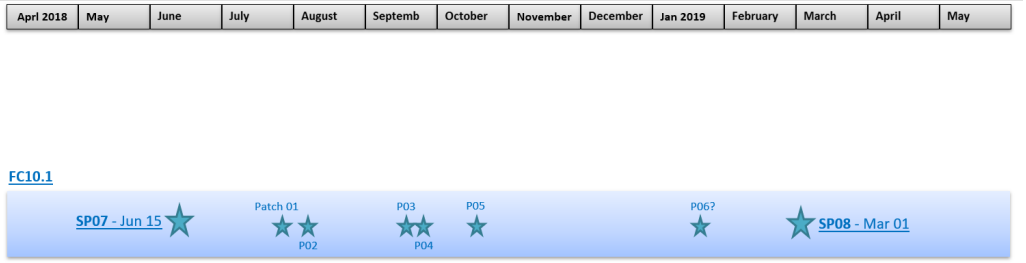


FIGURE 10 – Time Line de Patch

Cycle de Build - Test - Release Le produit FC est développé et délivré en mode pipeline.

- Afin de mieux organiser le développement du produit, l'équipe FC a décidé de soumettre les codes sur 3 branches :
  1. **Core** : La branche qui contient toutes les fonctionnalités et toutes les nouvelles fonctionnalités du produit FC.
  2. **PI** : La branche qui contient les fonctionnalités que l'équipe veut livrer au client, ***c'est sur cette branche que l'équipe Test va faire les tests***, les développeurs corrigent les bugs et soumettent leurs codes sur cette branche aussi.
  3. **Rel** : La branche qui va être livré et installé directement sur le côté client après la validation des tests.
- Le Build de FC est fait sur ces 3 branches ci-dessus, il se déclenche à chaque nuit, ce build va donner à lendemain un résultat qui est coloré en rouge ou en vert. Rouge veut dire qu'il y avait une erreur sur le build ou des bugs sur les codes, on doit attendre la correction d'erreur et rebuild du soir prochain ; vert veut dire que le build marche très bien, l'équipe Test peut récupérer et faire les tests. Chaque build prend plus de 6 heures.
- Pour les tests après le build, l'équipe Test va faire les tests manuel et test automatisé, nous faisons souvent les tests manuels sur les nouvelles fonctionnalités, les tests automatisés sur les non-régression fonctionnalités. Cette étape prend pas mal de temps et potentiellement trouver des bugs, les bugs viennent du développement sur les nouvelles fonctionnalités, ou les non-régression après le développement des nouvelles fonctionnalités, si on ne trouve pas de bug, on peut livrer le produit sur la branche *Rel*.
- Si le test trouve des bugs, nos développeurs vont les fixer et remonter les codes pour le prochain build.

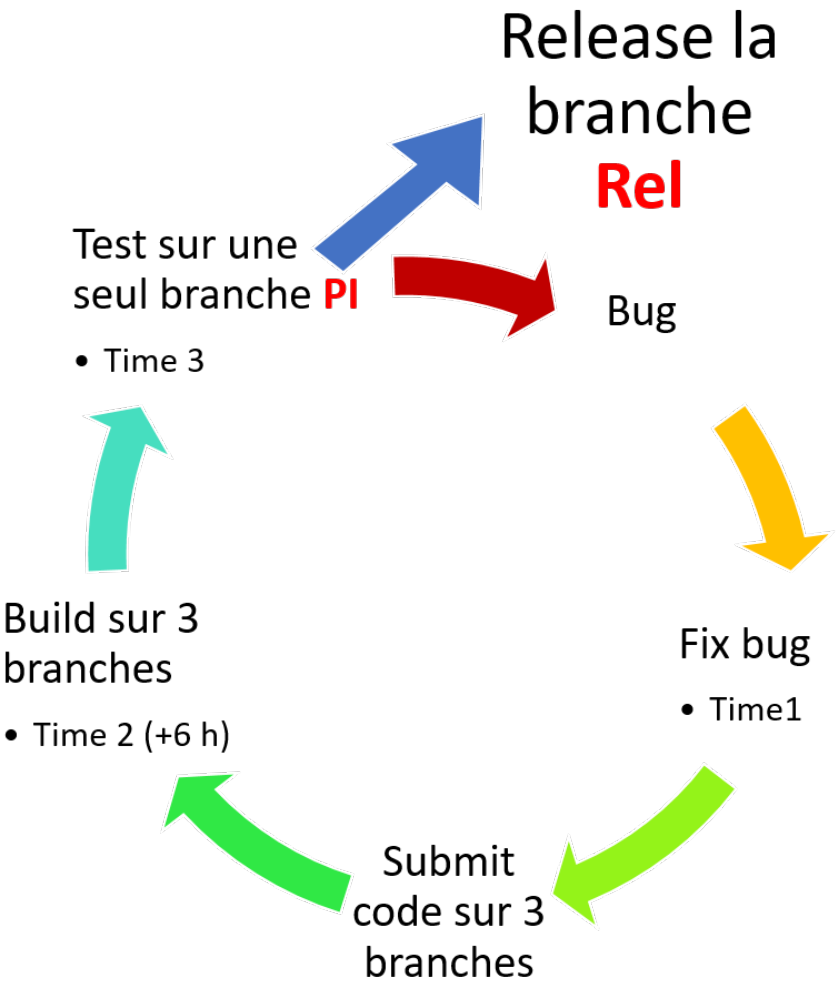


FIGURE 11 – Cycle de Maintenance : Build et Release

4.2 Client HTML5 : Test régression et Test non-régression

Le développement du produit amène deux théories de test : Test de régression et Test de non-régression. Un test de régression est un ensemble de tests d'un programme préalablement testé, après une modification, pour s'assurer que des défauts n'ont pas été introduits ou découverts dans des parties non modifiées du logiciel. Test de non-régression est le même test avec test de régression mais sans modification

pour assurer la qualité de produit.

4.2.1 Test Manuel VS Test Automatisé

Grâce aux technologies utilisés pour le développement de Client HTML5, nous pouvons automatiser certains tests fonctionnels en manipulant les XPath (*XML Path Langage*) de contrôle sur navigateur. Pour le choix de faire test manuel ou test automatisé, les comparaisons sont dans le tableau ci-dessous. Finalement, après la comparaison des avantages et désavantages de ces deux tests, l'équipe a choisi ces deux méthodes avec test automatisé couvre 50%-60% de fonctionnalité, les restes sont les tests manuels.

	Avantages	Désavantages
Test Manuel	• Permet de trouver les erreurs ne sont pas dans le scénario qui peut potentiellement gênant pour le produit. • Permet de tester le UX. • Très utile pour faire les tests régressions.	• Facile d'être fatigué faces aux scénarios *répétés*, perdre de temps.
Test Automatisé	• Scripts de test peuvent être réutilisés • Pas de fatigue, moins d'erreurs • Augmenter la couverture : plus de langue, plus de type de navigateur • Assurer la qualité du produit, détecte les régressions (une fonctionnalité ne marche pas après ajouter les nouvelles features) • Très utile pour faire les tests de non-régression sur les fonctionnalités existantes.	• Dépendance d'environnement, par fois ne marche pas même si le Test Robot marche bien sur la machine de développeur • Test Robot fait ce que les codes demandes, ignore les erreurs visuels. • Stabilité faible, une petite modification UI d'application peut bloquer le Test.

TABLE 1 – Comparaison entre Test Manuel et Test Automatisé

4.2.2 Pourquoi Test Automatisé est demandé ?

Après la discussion de la comparaison entre Test Manuel et Test Automatisé, et le cycle de Build et Release dans les parties précédentes, on peut savoir que Test Automatisé est plus pratique pour faire des tests de non-régression sur les fonctionnalités existantes parce qu'il peut répéter le scénario de test facilement.

Dans le développement du produit, les QA(Quality Assurance) assurent la qualité du produit en testant les nouvelles fonctionnalités et en faisant des tests de non-régressions sur les fonctionnalités existantes. Avec cette raison, l'équipe avait décidé d'automatiser le plus nombreux de tests non-régressions possible, c'est dans ce cas le Test Automatisé est demandé comme un point important dans

4.2.3 Langages et Technologies utilisées pour réaliser Test-Auto

Nous avons choisi d'utiliser SilkTest et Java pour développer le test-auto, les raisons de faire ce choix sont suivantes :

— SilkTest est déjà utilisé par une autre équipe de SAP France avant le commence du projet test-auto, les expériences d'utilisations d'un autre équipe nous a aidé beaucoup.

— Le choix du langage Java a été de préférence du développeur pour commencer le projet test-auto.

SilkTest VS Selenium Néamoins SilkTest est actuellement utilisé pour le test-auto de FC, l'équipe de Test a décidé d'utiliser Selenium comme outil de test-auto pour le projet à l'avenir, les raisons sont dans le tableau ci-dessous :

	Silk Test	Selenium
Prix	Acheter Licence	Open source - Gratuit
Test Système	Microsoft Windows	Cross-plateform : Windows, Linux, Mac
Applications à tester	Web, Modbile, Windows C\S	Web application
Développement langage supporté	C#, Java	Java, C#, Perl, Python, JavaScript, Ruby, PHP

TABLE 2 – SilkTest VS Selenium

4.3 Test d'impression

4.3.1 Analyse du problème et Etat de l'Art

Dans le cadre de test fonctionnel du FC, nous avons eu besoin de tester s'il la fonctionnalité d'impression de rapport dans l'application marche bien, c'est à dire pour le même rapport, l'impression donne toujours le même fichier pdf entre différentes versions du client.

Avant je suis arrivé, pour ce test, nous avons fait manuellement, c'est à dire que nous imprimons ces fichiers pdf et vérifié la différence entre ces pdf, c'est très fatigant pour les testeurs, en plus, nous en tant que humains, nous ne pouvons pas vérifier les petites différences, par exemple la différence de police des lettres, la différence entre lettre **'o'** et **'0'**.

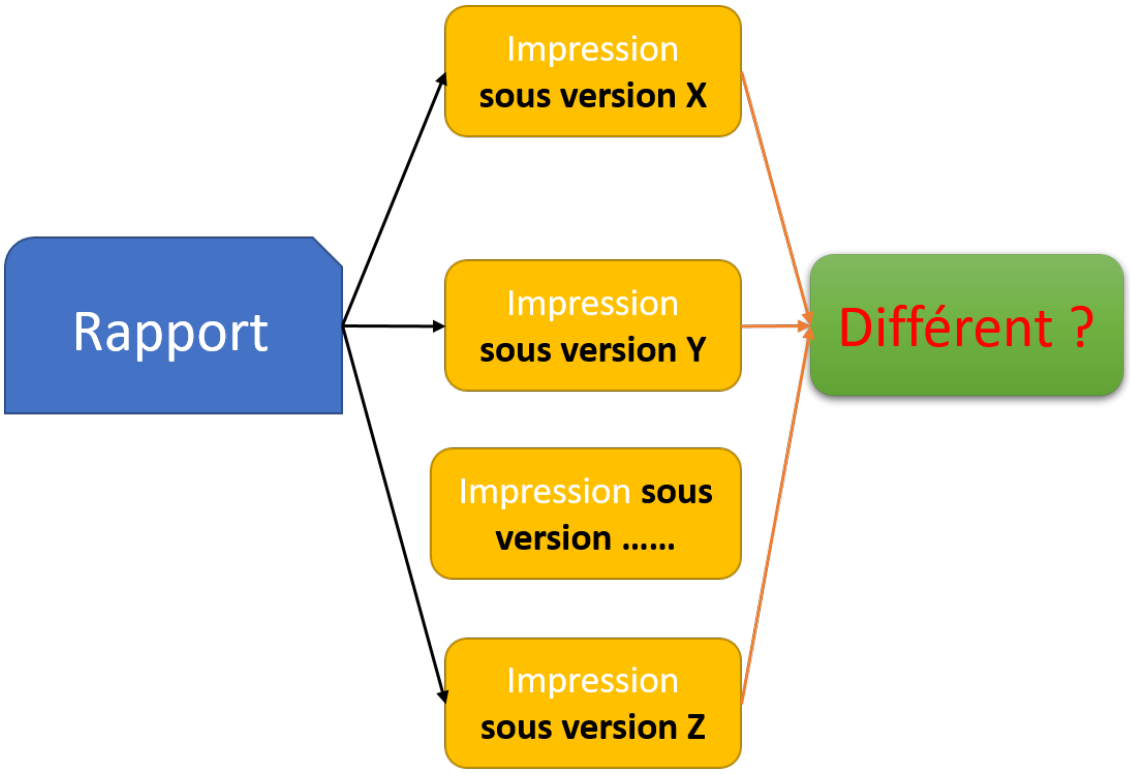


FIGURE 12 – Problématique du test d'impression

4.3.2 Comparaison les pdf avec checksum ou pixel par pixel

Dans cette condition, ça nous fait un besoin de comparer la différence entre deux fichiers pdf par machine. La façon la plus rapide est de générer une valeur de checksum (*MD5, SHA-1, SH-2*, etc.) pour chaque pdf, puis on vérifier la différence entre ces deux valeurs, s'ils sont les mêmes, ça va dire que ces deux pdf sont les mêmes, sinon, ça va dire que les deux pdf sont différents. Mais la comparaison avec checksum ne peut pas indiquer la position où il y a des différences entre 2 fichiers. Pour cette raison, j'ai décidé de faire une comparaison de pixel par pixel, qui est plus précisé et plus lisible par les humains.

	Avantages	Désavantages
Checksum comparaison	Rapide	Ne peut pas indiquer les positions de différences
Pixel par Pixel comparaison	Peut indiquer les positions de différences	Compiliquer à programmer, traitement lent

TABLE 3 – Checksum comparaison VS Pixel par Pixel comparaison

Puisque deux fichier pdf n'est jamais le même, donc nous pouvons aussi définir une cohérence de différence, on dit qu'il y a la différence entre deux pdf si le taux de différence est supérieur à la cohérence (0,3% par exemple).

Avec les conditions et raisons ci-dessus, l'équipe de test m'a donné la tâche d'automatiser le test d'impression et l'intégrer dans le processus de test-auto sur Jenkins comme la mission principale de mon alternance.

5 Travail réalisé

5.1 Schéma global

5.1.1 Architecture de Test-Auto

La mission de mon alternance est d'automatiser les scénarios de test automatisé, quand je suis arrivé, l'architecture de Test-Auto est déjà conçue et réalisée. Nous avons utilisé les machines virtuelles, Jenkins slave, et Perforce afin de lancer les tests et héberger les bases de données que ce dernier aura besoin pendant les tests. Les principes de test automatisé sont suivant :

— 3 types de machines virtuelles : machine qui contient la base de données utilisé par FC ; machine qui contient les ressources de référence de tests, cette machine va faire une copie de base de donnée celle de précédente ; machine virtuelle avec Jenkins slave, produit FC et Junit installés, c'est dans cette machine qu'on lancer les tests de FC.

— Une machine virtuelle sauvegarde les logs de tous les test-auto dans une base de donnée, et ces logs vont afficher dans un fichier Excel qui est plus lisible au lieu d'être en format .txt.

— Les test-auto codes sources de *JUnit* manipulés par Perforce, ces codes vont être exécutés dans une ou toutes les hosts machines selon la configuration.

— Un Jenkins master qui va lancer tous les tests et bien générer les résultats de test.

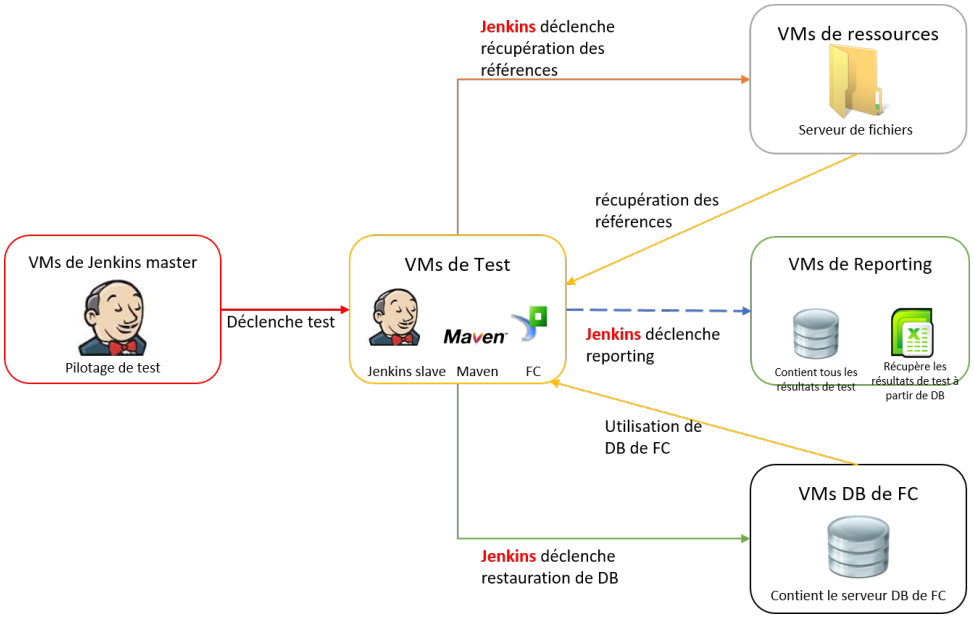


FIGURE 13 – Architecture de Test Automatisé

5.2 Développements Réalisés

5.2.1 Réalisation les scénarios des opérations sur les packages

Quand je suis arrivé, l'équipe Test m'a donné une tâche de réalisation les test-auto pour les opérations de packages de consolidation financière pour me faire familiariser avec les outils, l'environnement de travail. Il y avait 3 scénarios en totale et je l'ai fini pendant environ 10 jours, cette tâche me permet de bien comprendre l'architecture de Test-auto présentée dans la partie précédente, me permet de parcourir les codes existants, de comprendre le mécanisme de comment sauvegarder les logs de Test, de comment déclencher une application sur par exemple navigateur *IE* et *Google Chrome* sous exploitation Windows avec SilkTest, de comment manipuler les contrôles d'une application avec XPath, etc.

Après cette tâche, je suis demandé de concentré sur la tâche principale - test-auto d'impression, en réalisant les petites tâches peuvent être arriver de temps en temps.

5.2.2 Comparaison entre deux fichiers PDF

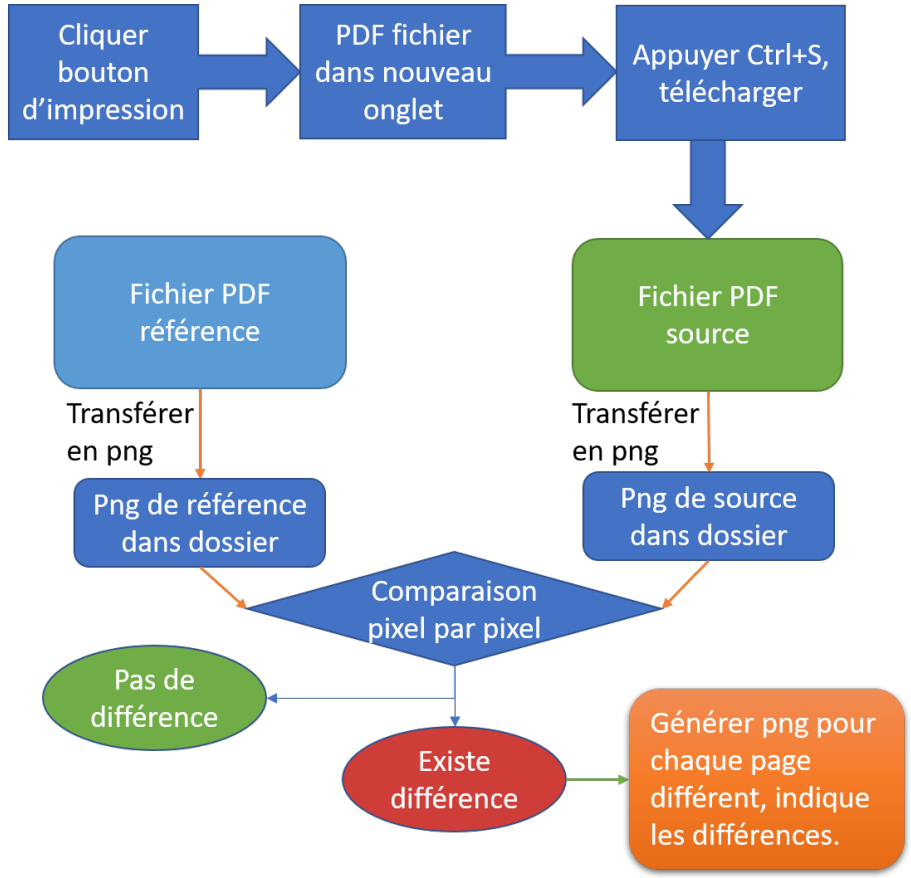


FIGURE 14 – Processus de comparaison de deux rapports

Comme la figure ci-dessus décrit, dans l'application, le bouton d'impression d'un rapport ou plusieurs rapports résulte d'un téléchargement à partir de navigateur. Pour le téléchargement de plusieurs rapports, je vais discuter dans la partie prochaine. Pour la comparaison entre deux fichiers pdf, le processus que j'ai conçu est ci-dessous :

1) Le rapport de référence est déjà fourni par le testeur, navigateur est préconfiguré pour que le téléchargement de pdf s'ouvre dans un nouvel onglet.
2) Cliquer sur le bouton d'impression dans l'application FC, le rapport pdf s'ouvre dans le nouvel onglet, mettre focus sur le nouvel onglet.

3) Appuyer **Ctrl**+**S** sur le nouvel onglet, il y a un dialogue de téléchargement apparaître, télécharger le fichier avec le nom fournis.
4) Pour chaque page de fichier pdf de référence et de source, générer une image en format png, les mettre dans un nouveau dossier créé dans le même répertoire du rapport avec le nom "*png_Nom_du_pdf*".
5) Comparer les images dans les deux dossiers, s'il n'y a pas de différence, on passe à la prochaine étape ; sinon, générer les images en noir et blanc comme ci-dessous indique les différences, ensuite rejeter une exception qui va être écrit dans log et arrêter le test de scénario problématique.

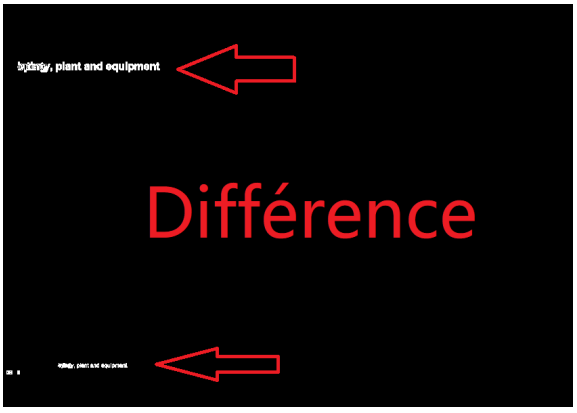


FIGURE 15 – Exemple de différence dans un pdf

Parce qu'un autre groupe de SAP France a déjà fait des traitements similaires, donc nous avons récupéré leurs codes de comparaison entre deux pdf, j'ai tout compris ce que leurs codes ont fait et je les ai adapté à notre projet. Dans leurs codes, un point très bien et je peux toujours me servir est leurs codes utilisent deux threads différentes pour traiter le pdf de référence et le pdf de ressource en parallèle, cette méthodologie a beaucoup amélioré l'efficacité du projet, la partie d'annexe **A.1** contient les codes sources de cette réalisation.

5.2.3 Télécharger et décompresser un zip

Pour certains scénarios de tests, ils nous demandent de télécharger un fichier zip, le nom de ce zip contient un GUID *(globally unique identifier)* préfix, le nombre des fichiers pdf et les noms sont déjà connus, ils ont aussi le même préfix pour leurs noms. Pour ces fichiers pdf dedans, on m'a demandé de les renommer avec les noms donnés selon l'ordre de choix de rapport dans FC.

Pour cela, J'utilise une bibliothèque standard de Java *java.util.zip.ZipEntry* et *java.util.zip.ZipInputStream* afin de décompresser le zip, ensuite, j'ai utilisé lambda expression de classer les fichiers selon leurs temps de création, puis les renommer.

5.2.4 Comparaison les pdf dans deux répertoires avec *Lambda Expression*

Après avoir fini la tâche dans la partie précédente, nous avons des nouveaux besoins : on ne sait pas le nom ou le préfix de nom des fichiers pdf dans le zip, ni le nombre des pdfs dans le zip. Autrement dit, ces besoins nous demandent de faire la comparaison entre tous les fichiers pdf ont le même nom dans deux répertoires.

Pour la comparaison, on peut appeler la méthode qu'on a déjà développé dans la partie précédente, pour comparaison de tous les fichiers pdf, on peut parcourir le répertoire et faire la comparaison dans plusieurs boucle ***for***, mais cette méthode demande trop de complexité.

Après avoir réfléchi, j'avais trouvé qu'on peut faire le parcourir de répertoire avec ***Lambda Expression***, qui est une très importante mise à jour depuis la création du langage Java, et il est beaucoup utilisé par les autres langages, c'est très "en mode" dans le monde de programmation aujourd'hui. Les codes utilisent ***Lambda Expression*** sont élégants et ne prennent pas beaucoup de place, pour la comparaison entre deux répertoires, il ne m'a pris que 10 lignes de codes. La partie d'annexe **A.2** contient les codes sources de cette réalisation.

Lambda Expression est une façon élégante de penser pour la programmation !

5.3 Outils utilisés

Pour réaliser les travaux parlés ci-dessus, nous utilisons les outils ci-dessous :

Gestion du code - PERFORCE Pour la gestion de version du code, nous utilisons Perforce.

PERFORCE

Perforce est un outil de gestion de configuration utilisé dans le processus de développement logiciel, Perforce est un système de contrôle de version centralisé,

c'est ici la différence par rapport les autres systèmes de contrôle de version comme Git, Mercurial. Nous utilisons un des ces outils Helix Visual Client (P4V), P4V est une application de bureau qui permet d'accéder aux fichiers versionnés dans Helix Core via une interface graphique. Il comprend des outils permettant de fusionner et de visualiser l'évolution du code.

Avec P4V, il est facile de personnaliser notre espace de travail afin de ne voir que les fichiers dont nous avons besoin. Nous pouvons travailler hors ligne et à distance. Et lorsque nous avons terminé, transférez facilement les modifications locales sur un serveur distant.

Nous pouvons :

— Voir les changements de time-lapse et de révision.

— Obtenez un aperçu des métadonnées du projet.

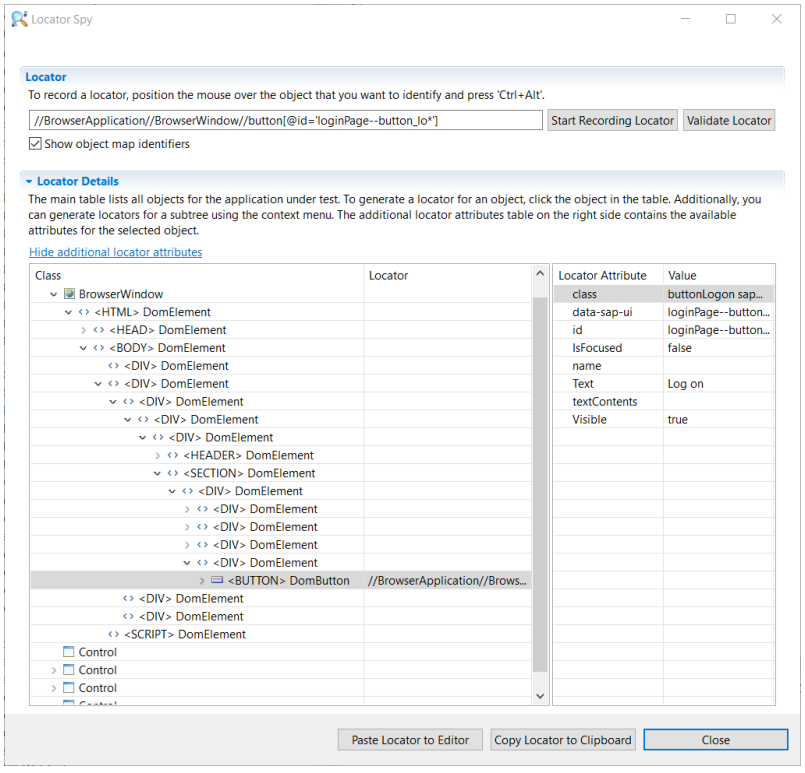
— Demander des révisions de code sur les modifications en attente.

Dans notre équipe, chaque remonte du code est revue par l'autre développeur, pour moi, c'est mon encadrant technique Sébastien Coudray relit mes codes.

Édition du code - Eclipse Avec la limite que le plugin Silk4J ne peut qu'installer sur Eclipse, l'équipe de test-auto a décidé d'utiliser Eclipse comme éditeur du code.

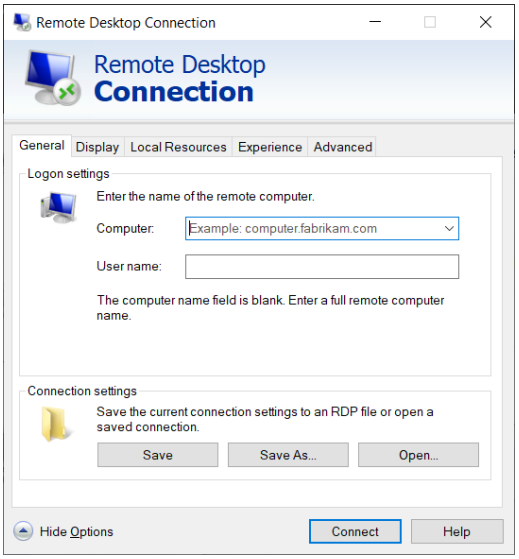
Outil de Testauto : SilkTest - Silk4J Silk Test est un outil de test de fonction automatisé et de régression des applications d'entreprise. Le plugin Silk4J installé sur Eclipse nous permettons de créer des tests fonctionnels avec la programmation en Java, avec la Java runtime librairie fournis par Silk4J, on peut lancer des JUnit test.

L'outil "Locator Spy" installé ensemble avec SilkTest nous permet de localiser les contrôles d'une application et afficher comme un **XPath**, comme la figure ci-dessous, c'est un outil que je peux l'utiliser pour reconnaître le ID d'un contrôle puis l'utiliser dans mes codes.



Intégration continue - Jenkins + machines virtuelles Afin d'automatiser le build et les tests fonctionnels, l'équipe FC a mis en service une Intégration Continu

de build et de Test-auto avec Jenkins et plusieurs machines virtuelles. Nous pouvons se connecter aux ces machines virtuelles avec le client *Remote Desktop Connection* sous Windows.



Partage de fichiers et d'informations - One Note + One Drive Pour partager les fichiers et les informations nécessaires d'équipe, nous utilisons OneNote et OneDrive, OneNote pour sauvegarder les informations des projets : Notes pour chaque réunions, IP des machines virtuelles, liens des dossiers partagés, tâches de chaque personne avant prochaine réunion de revue, etc. OneDrive pour partager des documents en commun d'équipe.

Communication interne - Skype for Business + Outlook Pour communication interne, nous utilisons Skype for Business et Outlook.

Skype for Business est un logiciel d'entreprise de messagerie instantanée développé par Microsoft, ce logiciel nous permet de se communiquer entre membre de l'équipe, il nous permet aussi d'organiser et de participer les réunions en ligne, cette fonctionnalité est très utile, et surtout beaucoup utilisé pour participer des réunions d'entreprise à l'étranger et quand l'on est en télétravail par exemple le *Scrum "Stand-up"* réunion de chaque début d'après-midi (Skype for Business permet d'ajouter jusqu'à 250 participants aux réunions en ligne).

Quand on a des informations importantes à échanger et partager, on utilise Microsoft Outlook, Outlook est un gestionnaire d'informations personnelles et un client de courrier électronique propriétaire édité par Microsoft.

Outlook

Cet outil nous permet aussi d'organiser les réunions selon la fonctionnalité de calendrier, il nous permet aussi de vois les agendas de chaque membre d'équipe par exemple les temps de disponibilité, les périodes de vacances, les jours de télétravail, les jours de trainning, etc grâce à un plugin interne s'appelle TeamCalendar.

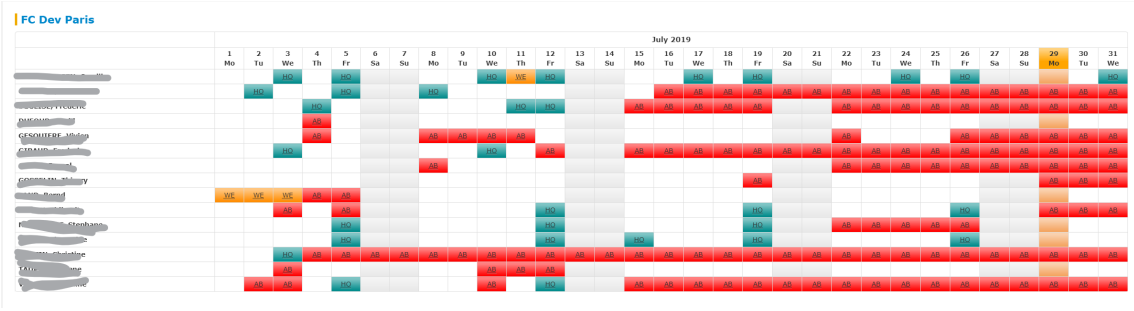


FIGURE 18 – Team Calendar

6 Conclusion

Pour conclure, l'automatisation de test d'impression, la partie que je travaille durant mon alternance, est presque fini jusque à la date de rédaction de ce mémoire, il marche bien et il est bien intégré dans le projet Test-Auto de l'équipe Test, je vais bientôt commencer des autres tâches de Test-Auto.

Après d'avoir suivi les cours théoriques à Sorbonne Université et à CFA INSTA, et d'avoir travaillé chez SAP France, je me sens beaucoup amélioré mes compétences sur le développement en Java, les connaissances sur les XPath, la compréhension théorique de Lambda Expression, des connaissances sur le domaine de finance.

Durant cette année d'alternance, j'ai appris beaucoup aussi sur la gestion de projet, sur la méthodologie de comment travailler ensemble dans une grande équipe, de comment travailler dans une petite sous-équipe, de comment être patient pour travailler, de comment travailler efficacement.

Pour finir, j'ai appris beaucoup de chose pendant cette année d'alternance et je me sens d'être à l'aise à SAP. C'est mon honneur d'effectuer l'alternance chez SAP et de suivre ces années d'études à Sorbonne Université, dans lequel j'ai appris beaucoup de chose, pour mon futur professionnel, pour ma vie.

Bibliographie

[1] SAP Global Corporate Affaires. Sap corporate fact sheet. https://www.sap.com/corporate/en/documents/2017/04/4666ecdd-b67c-0010-82c7-eda71af511fa.html, July 18, 2019.

[2] sap help portal for Financial Consolidation. A propos de financial consolidation. https://help.sap.com/viewer/9781cc5319034ed18687566eaa9ac40c/10.1.8/fr-FR/277af635f7d1406a8b7433624013b911.html, 2019.

[3] Wikipedia. Sap wikipedia page. https://fr.wikipedia.org/wiki/SAP_(entreprise), 2019.

A Annexe des codes

A.1 Multi-Thread pour la comparaison des pdf

```
// First Thread - Ref
Thread_PDF_To_PNG thread_one = null;
try
{
   thread_one = new
       Thread_PDF_To_PNG(1,bar,GeneralFuncs.normalizePath(ref_filePath));
   es.execute(thread_one);
}
catch (Exception e)
{
   throw new Exception("Cannot proceed with PDF extraction to PNG. "
       + e.getMessage());
}
es.isTerminated();
// Second Thread - Res
Thread_PDF_To_PNG thread_two=null;
try
{
   thread_two = new
       Thread_PDF_To_PNG(2,bar,GeneralFuncs.normalizePath(res_filePath));
   es.execute(thread_two);

} catch (Exception e) {
   throw new Exception("Cannot proceed with PDF extraction to PNG. "
       + e.getMessage());
}
try  {
   // Wait for the end of two thread
   System.out.println("Wait for the end of multi-thread pdf
       extraction");
   bar.await();
   bar = null;
} catch (InterruptedException ex)  {
   throw new Exception("Error Message : " + ex.getMessage());
}
```

A.2 Lambda Expression pour comparaison entre deux répertoires

```
public static void compareFolderRefAndSrc(String refFolder, String
    srcFolder) throws Exception
{
   TestLogger.getInstance().beginCheckpoint("Ref folder: " + refFolder +
       " - Src foler: " + srcFolder);

   try
   {
      Path dirRef = Paths.get(refFolder);
      Path dirSrc = Paths.get(srcFolder);

      // filter the .pdf file
      List<Path> listRefPath = Files.list(dirRef)
            .filter(f -> f.toString().endsWith(".pdf"))
            .collect(Collectors.toList());

      for (int ii = 0; ii < listRefPath.size(); ii++)
      {
         Path refPath = listRefPath.get(ii);
         Path srcPath = Files.list(dirSrc).filter(path ->
             path.getFileName().equals(refPath.getFileName())).findFirst().get();
         comparePDF(refPath.toString(), srcPath.toString());
      }
   }
   catch (Exception e)
   {
      TestLogger.getInstance().logError(e.getMessage());
   }

   TestLogger.getInstance().endCheckpoint();
}
```