ESIGELEC
Saint-Etienne du Rouvray (76)

Pierre TAQUET

Contrat de professionnalisation

Équipe ST Automation

Date Visa du tuteur industriel	Date Visa du tuteur pédagogique	Date Visa du service formation continue

Remerciements

Je tiens à remercier, tout d'abord, l'ensemble des membres de l'équipe ST Automation pour leur accueil chaleureux.
Je voudrais aussi remercier tout particulièrement Antoine MAILLARD, Christophe DOLIMONT et Fabien COAT pour m'avoir accompagné tout au long de mes différentes missions, pour avoir répondu à mes questions et pour être restés à mon écoute.
Un grand merci également à Ziad AKL, mon manager, pour son accueil et son suivi régulier.
Enfin, je remercie Leslie SUDRE pour tout le temps qu'elle a passé à la relecture de ce présent rapport.

Sommaire

Introduction

Ce présent rapport a pour objet la présentation, d'une part, de SAP, entreprise dans laquelle j'ai fais mon contrat de professionnalisation, et d'autre part, de mes différentes activités effectuées lors de cette période.

Mon contrat était divisé en deux périodes, la première s'étendait du 15 juillet au 30 septembre 2014 et la seconde du 15 février au 30 septembre 2015. Ces longues périodes de travail ont permis une immersion complète dans le monde de l'entreprise et surtout dans mes missions.

Au commencement de mon contrat de professionnalisation j'ai participé aux tests, raison d'exister de l'équipe ST Automation dans laquelle j'ai été intégré. Ainsi, après un certain temps de formation, j'étais capable d'implémenter des tests de qualités et fiables. Aujourd'hui encore ces tests sont exécutés quotidiennement pour s'assurer de la non-régressions du logiciel Web Intelligence.
Dans la seconde partie de mon contrat de professionnalisation, je me suis occupé d'implémenter l'accès à leur logiciel de traçabilité, et ce dans le respect des normes de sécurités prescrites par SAP. Une fois cela terminé et fonctionnel, j'ai entamé le projet qui aura été le plus formateur de mon contrat : le développement d'un plugin pour Jenkins devant se substituer à un autre déjà existant. Développer un plugin pour un logiciel qui m'était inconnu, et aux fonctionnalités qu'alors je découvrais, s'est présenté à moi comme un véritable challenge. Après un certain temps de recherche et de développement, celui-ci a commencé à être utilisé dès lors que ses fonctionnalités égalaient celles du plugin précédant, aujourd'hui désintallé, le mien l'ayant remplacé.

Ce temps que j'ai passé dans l'équipe ST Automation à été très formateur et j'ai pu progresser tant en Java que dans ma méthode de travail. Ces différentes missions que je me suis vu effectuer m'ont permises d'étudier le Framework de test de Web Intelligence sous plusieurs angles. Cela m'a permis d'avoir aujourd'hui une vision globale de celui-ci, sans pour autant en comprendre tous les détails, et d'être ainsi plus à même de m'adapter à tout nouveau Framework que je serai appelé à utiliser.

Chapitre 1

Présentation de l'entreprise

1.1 SAP dans le monde

SAP, qui signifie « Systeme, Anwendungen und Produkte in der DatenverarbeitungSystems » ou « Systems Application and Products in data processing », à été formé en 1972 par cinq anciens employés d'IBM et ont inventé l'ERP [1] ou PGI en Français [2]. Son siège est à Walldorf, en Allemagne. SAP conçoit et vend des logiciels de gestion et de reporting à destination des entreprises et des professionnels.

SAP est le premier éditeur de logiciels en Europe (devant Dassault Systèmes, Sage et Software AG) [3] et le quatrième mondial (derrière Microsoft, IBM et Oracle) [4]. Ses principaux concurrents sont les autres éditeurs d'ERP et de logiciels de gestion. Le chiffre d'affaire de l'année fiscale 2014 est de 17,56 milliards €.

SAP fournit plus de 293 000 clients venant de 190 pays et regroupe plus de 74 400 employés dans plus de 130 pays [5]. Pour plus d'informations à ce sujet consulter l'annexe D page 85 [6].
Les membres du conseil d'administration de SAP sont représentés dans le tableau page 4.

1. Entreprise Ressource Planning
2. Progiciel de gestion intégré
3. Source : http://www.journaldunet.com/solutions/dsi/truffle-100-europe-2014.shtml
4. Source : http://www.journaldunet.com/solutions/ssii/classement-mondial-editeur-2010/editeur-logiciel-classement-mondial.shtml
5. Source : http://www.sap.com/corporate-en/about/our-company/index.html
6. http://www.sap.com/bin/sapcom/en_us/downloadasset.2015-07-jul-22-01.SAP-Corporate-Fact-Sheet-en-2015-07-22-pdf.html

— Bill McDermott
— CEO
— a rejoint SAP en 2002
— a rejoint le conseil d'administration en 2008
— le terme de son mandat au conseil expire en 2017
— autres sièges en conseil d'administration : ANSYS, Inc ; Under Armour, Inc

— Robert Enslin
— Président, opérations client
— a rejoint SAP en 1992
— a rejoint le conseil d'administration en 2014
— le terme de son mandat au conseil expire en 2017
— responsabilités particulières : stratégie de marché, cloud, ventes régionales, ventes de l'industrie spécialisée, écosystème, expérience client end-to-end

— Bernd Leukert
— produits et innovation
— a rejoint SAP en 1994
— a rejoint le conseil d'administration en 2014
— le terme de son mandat au conseil expire en 2017
— responsabilités particulières : organisation du développement global, analyse, application, cloud, bases de données et technologies, mobile

— Lika Mucic
— Directeur financier, Directeur des opérations
— a rejoint SAP en 1996
— a rejoint le conseil d'administration en 2014
— le terme de son mandat au conseil expire en 2017
— responsabilités particulières : finance et administration, relations investisseurs et protection des données

— Gerhard Oswald
— a rejoint SAP en 1981
— a rejoint le conseil d'administration en 1996
— le terme de son mandat au conseil expire en 2016

SAP mène une stratégie d'acquisition [7] des leaders des technologies de l'information. La croissance organique étant le moteur de cette stratégie, tout en continuant à investir dans le développement de ses propres produits et dans les technologies innovatrices. Cette stratégie permet à SAP de mieux soutenir l'innovation en la dirigeant vers les partenaires possédant les domaines d'expertise requis. SAP perdure dans ce sens pour acquérir les technologies stratégiques cibles pour couvrir un marché toujours plus grand et pour satisfaire toujours plus les besoins de ses clients. Le nouveau marché que vise SAP est celui du Big data, dans lequel nous pouvons l'imaginer leader dans quelques années.

La structure actuelle est le fruit du rachat de Business Objects par SAP France dont la date légale est le 1er janvier 2010, l'annonce ayant été faite le 7 octobre 2007. Précédemment, Business Objects ayant absorbé deux autres entités, CARTESIS et Crystal decisions. Les rachats de SAP se succèdent : Sybase en 2010, SuccessFactors en 2011, Ariba et Syclo en 2012, Hybris en 2013, FieldGlass et pour finir Concur en 2014.

1.1.1 Les produits SAP

Les solutions logicielles de SAP [8] sont principalement à destination des professionnels et offrent un grand nombre de logiciels destiné au traitement de tout type de problématique.
Historiquement, SAP est le leader (et inventeur) des progiciels, logiciels de gestion d'entreprise qui ont su, avec le temps, devenir incontournable pour toute moyenne ou grande entreprise.
En terme de logiciel de gestion, SAP propose :

- Business suite
- Développement durable
- Gestion comptable et financière
- Progiciel de gestion intégré (ERP)
- Gestion de la chaine logistique (SCM)
- Gestion de la relation client (CRM)
- Gestion du cycle de vie des produits (PLM)
- Gestion des achats
- Gestion des actifs d'entreprise
- Gestion des ressources humaines (GRH)

Outre le fait de pouvoir organiser les flux d'information de l'entreprise, SAP s'est imposé comme l'un des leaders de l'analyse de ces données.
Parmi ces outils d'analyse, nous avons :

7. Source : `http://www.sap.com/corporate-en/about/investors/newsandreports/acquisitions/index.html`
8. Source : `http://www.sap.com/france/pc/index.html`

— Analyse prédictive
— Applications analytiques
— Business intelligence
— Entrepôt de données (Data warehousing)
— Gouvernance, risques et conformité
— Pilotage de la performance

SAP se voulant une entreprise aux solutions complètes et innovantes, elle s'est ouverte aux technologies et aux données. La stratégie d'entreprise de SAP changeant, ainsi que le marché, les domaines de compétences évoluent aussi. Pour illustrer ceci, nous pouvons citer les quelques domaines dans lesquels son expertise est reconnue :

— Bases de données
— Could computing
— Entrepôt de données
— Gestion de contenu et collaboration
— Gestion de l'information
— Gestion et intégration des processus métier
— Gestion informatique
— Infrastructure / Sécurité des applications métier
— Mobilité
— Platefome de données temps réel (RTDP)
— Technologie In-memory SAP HANA

Et pour compléter son large panel de compétences, la stratégie de SAP s'est orientée ces dernières années vers le cloud et vers les technologies mobiles, très à la mode en ce moment. En terme de cloud et de mobilité, nous pouvons citer :

— Application métier
— Collaboration sociale
— Infrastructure
— Plate forme
— Commerce collaboratif
— Applications mobiles
— Gestion de flottes de terminaux mobiles
— Gestion de mobilité
— Plate forme d'applications mobiles
— Services mobiles
— Solutions de commerce mobile

De cet échantillon de solution que propose SAP, nous pouvons être certain de deux choses :

1. SAP est le leader et expert des technologies de gestion, de stockage et de traitement des données

2. Sa stratégie d'expansion et son domaine d'expertise sont orientés vers les systèmes d'information

1.1.2 SAP en France

En France, SAP a implenté des bureaux commerciaux à Lyon, Nantes, Toulouse, Strasbourg, Bordeaux, Aix, Sofia Antipolis et Caen. Leur centre de recherche (SAP Labs) était anciennement à Levallois-Perret (Paris). Paris abritait trois sites SAP différents, ils étaient présents à la défense (principalement pour les ressources humaines), à Capital 8 Paris (principalement pour le développement) et à Levallois-Perret (Développement de solutions et SAP France). Tous ces centres ont été rassemblés en un seul et la fusion s'est terminée mi-avril 2015 quand les locaux de Levallois-Perret front de Seine ont été vidés. Aujourd'hui c'est la tour SAP, à Levallois-Perret, qui accueille les employés des trois sites précédents pour un total de plus de 1600 personnes.

1.2 Contexte antérieur à mon arrivée à SAP

Après le rachat de Business Objects, SAP a sorti la version 4.0 de BOE (Business Objects Entreprise). Le rachat de Business Objects ayant précipité la sortie de BOE, ses fonctionnalités n'étaient pas au rendez-vous, ce qui a grandement déplu les clients. Le logiciel comportait beaucoup de bugs et certaines fonctionnalités n'étaient pas implémentées. Ceci s'explique d'une part par le fait que beaucoup trop de fonctionnalités ont été planifiées. Et d'autre part parce que les différentes équipes travaillaient en tunnel, que le trop grand nombre de dépendances rendait très difficile l'intégration complète du produit et que la date de sortie du logiciel ne pouvait pas être repoussée.
Mais, grâce aux initiative qualités du groupe Web Intelligence, les versions qui ont suivies ont eu beaucoup de succès. Dans la continuité des initiatives qualité, le groupe Web Intelligence a ainsi amené les huit initiatives qualité.

1.2.1 Les 8 initiatives qualité

Performance

Améliorer les performances des versions 4.x pour les ramener aux performances de la version 3.1.
Les performances critiques étant celles que le client note en premier lieu :

— temps d'ouverture d'un document

— fonctionnalités de base : création d'un document, sauvegarde, rafraîchissement

— temps de connection au CMS

Intégration continue

L'intégration continue doit être assurée de manière à pouvoir détecter le plus tôt possible les régressions (par exemple en compilant ou à chaque livraison de code) et pour pouvoir limiter le risque qu'une compilations casse et de limiter les risques de régressions fonctionnelles en lançant les tests appropriés.

Automatisation

Contrôler la qualité de l'héritage inter-versions (compatibilité ascendante) à travers des tests automatiques et assurer la qualité des fonctionnalités en cours de développement.

— import d'un document créé sur une version 3.x vers une version 4.x
— préserver les fonctionnalités et les performances

Optimisation de l'exécution des tests

Optimiser la couverture de test basée sur les livraisons.
Ne pas exécuter les tests qui ne sont pas nécessaires et exécuter ceux qui le sont.

1 bug = 1 test

Améliorer la couverture de test de régression en intégrant les anomalies trouvés par les clients dans les fonctionnalités principales. Autrement dit, dès qu'un bug est trouvé un test sera implémenté pour reproduire le problème de manière automatique pour assurer la non-régression. Ce principe permet, comme son nom l'indique, que chacun des bugs corrigés par un développeur soit testé systématiquement et automatiquement par un ST.

Augmenter la couverture de test

Trouver les problèmes nouveaux et pas encore découverts en complétant les tests existants, en se concentrant sur les zones à risque.

ADEPT

Réduire le temps de configuration de l'environnement de développement en fournissant des environnement prêts à l'emploi (comprenant IDE, outils de test, codes source, ...) pour toutes les branches BI 4.x.

Outil de validation des rapports

Faciliter la migration des clients de la BI4.1 à la BI4.2 en accélérant l'inspection des rapports Web Intelligence après une mise à jour.

Chapitre 2

Présentation de mon environnement à SAP

2.1 Présentation de l'équipe

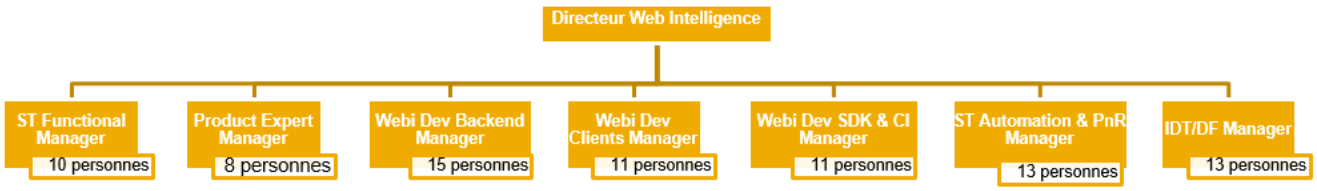


FIGURE 2.1 – Équipe Web Intelligence

2.2 Sa mission

L'équipe transversale fournit les services transversaux, faisant appel à des compétence spécifiques.
Particulièrement foculisée sur le test coverage, différentes missions ce l'équipe transverse peuvent être divisées en cinq catégories [1] :

Tests fonctionnels Assurés par 3 personnes, leurs tests portent sur le SDK de Web Intelligence.

Tests automation Assurés par 3 personnes, leur mission est d'ajouter les tests implémentés et de les mettre en production, quelque soit le test (son Framework ou le langage utilisé).

Benchmark Assurés par 3 personnes, ils s'occupent des tests de benchmark, autrement dit, les tests de performance, de stabilité et de scalabilité. Ils sont parfois appelés à implémenter quelques codes mais utilises, en règle général, des logiciels pour générer des charges, ouvrir de multiples documents, installer le produit, répéter la même action des centaines de fois, ... Ils sont aussi en charge des acceptances, autrement dit la validation de tous les tests existants afin de certifier que le produits et fonctionnel.

Mise en production Sa mission est de mettre en production les produits développées par et pour les équipes SAP. Une personnes s'occupe de cette mission.

Développement Développement d'outils internes et d'outils de reporting, construction de rapport WebI. Une personnes s'occupe de cette mission.

1. se reporter à l'organigramme page ??

Chapitre 3

Software tester

3.1 Généralités sur le test automatique

Pour répondre à la question « Qu'est-ce qu'un test automatique ? », je dirais que c'est un code vérifiant le bon fonctionnement d'un autre code. Il existe de nombreux niveaux de test : unitaire, fonctionnel, intégration, performance, . . . Les tests sont très importants dans un projet logiciel, ils permettent de s'assurer de la conformité du logiciel par rapport aux spécifications (autrement dit, le besoin client). La mise en place de ces tests automatiques est facilitée par un Framework, il en existe de nombreux mais de manière générale un Framework est un ensemble de librairies et de normes (de modélisation, d'architecture, ...) [1].

3.1.1 Description des différents types de test

Le test unitaire

Un test unitaire vérifie le bon fonctionnement d'une petite portion de code. En règle générale il s'agit de tester que la valeur retournée par une méthode est la bonne, ces tests sont donc implémentés par les développeurs. Mais il s'agit aussi de valider le fonctionnement de la méthode aux limites et de vérifier la cohérence de ses résultats. Ces tests sont censés être les plus nombreux possibles pour couvrir le maximum de cas possibles, on appelle cela le test coverage (ou couverture de test).

Le test fonctionnel

Le test fonctionnel est axé sur l'interaction des différentes méthodes, il est axé sur les fonctionnalités du logiciel. Ces tests permettent de vérifier le bon fonctionnement des spécifications du produit et de valider que les fonctionnalités correspondent aux spécifications. Ceux-ci sont assurés par les testeurs automatiques (ST).

1. Source : `https://boulaich.wordpress.com/2009/07/08/framework-generalite/`

Le test d'intégration

Le test d'intégration est au test fonctionnel ce que le test fonctionnel est au test unitaire, avec plusieurs niveaux de granularités (plus ou moins grand nombre de méthodes concernées). Le test d'intégration s'assure du bon fonctionnement du logiciel en interaction avec d'autres logiciels.

Le test de performance

Le test de performance s'assure qu'une action est effectuée dans le temps imparti. Les actions testées sont au minimum celles qui composent les workflows nominaux, tant séparément que simultanément (multi utilisateurs).

Le test de stabilité

Le test de stabilité permet de tester qu'un comportement est systématique, autrement dit qu'une même action implique toujours le même résultat.
Les tests de stabilités permettent de s'assurer que la réponse est bien là dans le temps imparti et qu'il n'y a pas d'erreurs, ceci en faisant subir une montée en charge des requêtes du logiciel sur une durée pouvant aller de 24 à 72 heures.

Le test de scalabilité

Le test de scalabilité permet de s'assurer que le logiciel peut évoluer de manière à pouvoir offrir les mêmes performances dans un contexte d'utilisation qui a évolué. Par exemple, dans le cas où la charge d'utilisation double, est-il possible de modifier l'environnement client afin que les utilisateurs profitent des mêmes performances ?

Le test de validation de plateforme

Ce type de test est consacré aux logiciels destinés à être utilisés sur plusieurs plateformes. D'un système d'exploitation à un autre, les comportements sont-ils les mêmes ? Les fonctionnalités sont-elles identiques ?

3.2 Le test dans l'équipe d'automatisation des tests

Lorsque je suis arrivé dans cette équipe, j'ai découvert des logiciels que je ne connaissais pas : Perforce, ASTEC, Jenkins, et d'autres. J'ai mis un peu de temps à me les approprier et à bien comprendre à quoi ils servaient exactement. La complexité de leur environnement d'utilisation ne m'a pas facilité la tâche.

Pour résumer ce que j'ai appris sur les généralités de cette équipe c'est qu'elle utilise plusieurs outils pour gérer le code des différents logiciels, possédant chacun plusieurs versions. Pour chacune de ces versions une suite de tests est exécutée quotidiennement, les codes en question sont hébergés sur le gestionnaire de code source Perforce.

Après chaque livraison de code effectuée, le logiciel est compilé par Jenkins. Et, de plus, le logiciel est compilé quotidiennement par ASTEC dont les résultats sont automatiquement envoyés aux personnes concernées (cf. annexe A page 73 qui présente l'un des résultats de ces mails). L'utilisation de ASTEC dans l'équipe est beaucoup plus large que cela mais je n'ai pas eu l'occasion de l'étudier plus en détail.

Le testeur automatique est focalisé sur les fonctionnalités du logiciel. Lorsque les spécifications d'une nouvelle fonctionnalité voient le jour, un plan de test est rédigé et est envoyé à l'équipe du test automatique. Du test plan sont extraits tous les tests pouvant être automatisés et ceux-ci sont implémentés.
Je n'ai pas beaucoup participé à cet aspect de l'équipe, ma principale mission relevait de l'initiative « 1 bug - 1 test ». Mon travail n'était pas de tester une nouvelle fonctionnalité mais plutôt, une fonctionnalité déjà existante sur laquelle une anomalie a été trouvée, soit par un client (à éviter) soit par un individu interne à SAP.

Le cycle de développement de l'automatisation de tests dans lequel j'ai évolué est résumé (et simplifié) figure 3.1 page 18. Ce cycle est celui de l'automatisation d'un test dans le cas d'une anomalie trouvée sur une fonctionnalité existante et pas sur une nouvelle fonctionnalité. Mon intégration dans ce processus de développement m'a appris beaucoup, non seulement quant à la gestion d'une anomalie logicielle, mais surtout quant aux cas d'utilisation du test qui peut être implémenté soit en réponse à une anomalie soit dans le processus initial de développement pour garantir qu'une fonctionnalité se comporte telle qu'elle est décrite dans les spécifications.

3.3 Présentation du produit testé : Web Intelligence

Web Intelligence est un logiciel de BI permettant d'accéder à des données stockées dans une base de données. Cet accès aux données ne se fait pas directement, tout l'intérêt de Web Intelligence repose sur l'utilisation d'une couche sémantique, « l'univers ».
Il existe trois interfaces graphiques permettant d'accéder à ces données, un client lourd et deux clients légers : l'applet et le client dhtml.

3.4 La première semaine dans l'équipe d'automatisation des tests

À mon arrivée dans l'équipe et avant de commencer à implémenter des tests automatiques, j'ai été accueilli par mes collègues et on m'a fourni le matériel nécessaire au bon déroulement de mon travail.
Mes premiers jours à SAP se sont déroulés de la manière suivante :

— Présentation à l'équipe et visite des locaux

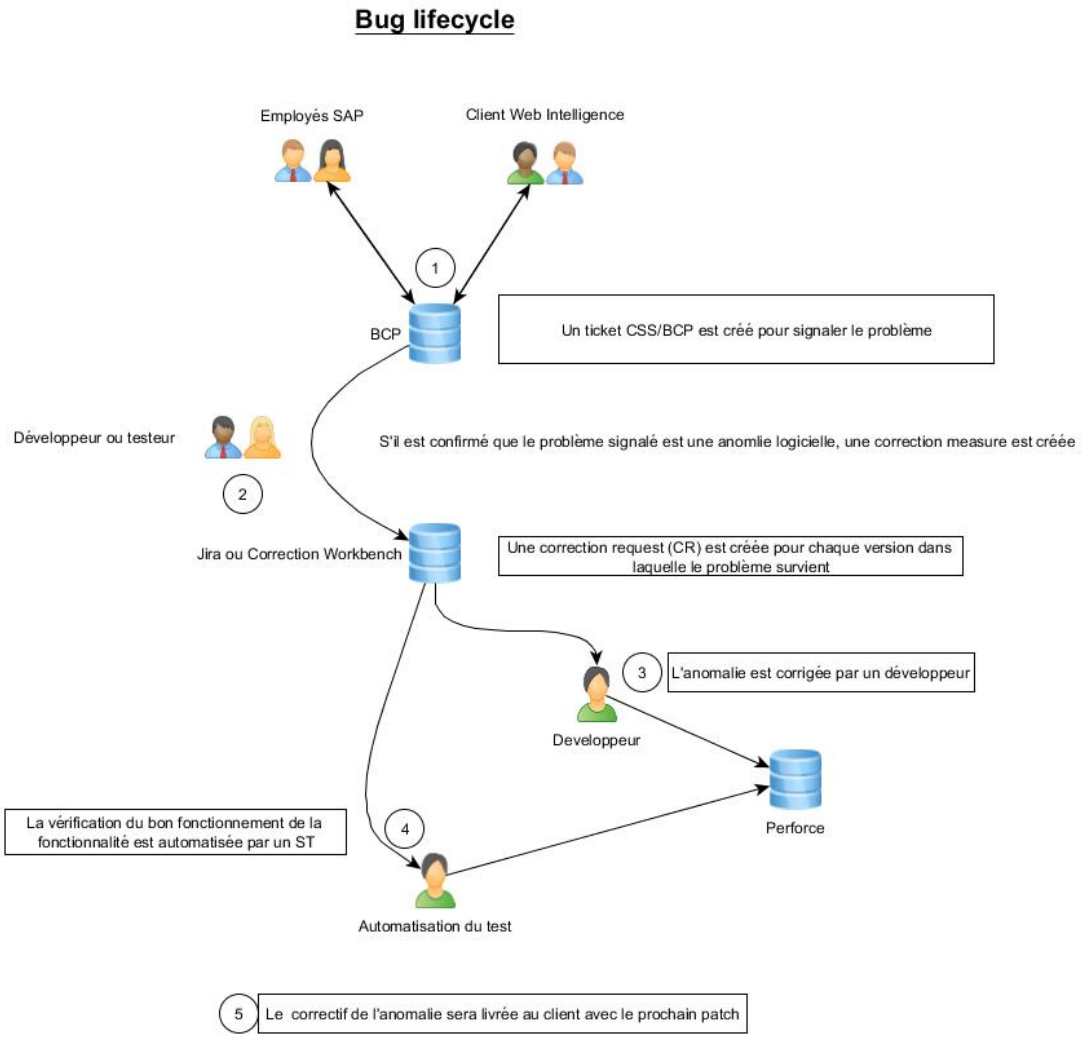


FIGURE 3.1 – Le process de test

— Réunion avec mon tuteur et mon manager pour une description de la mission
— Récupération des différents droits d'accès aux serveurs
— Familiarisation avec les outils internes (tickets HR, CSS, IT, ...)
— Installation des logiciels nécessaires au développement (IDE, SCM, éditeur de texte, ...)
— Mise en place du framework de test (création du workspace perforce)

Au terme de la première semaine, j'ai pu commencer à étudier le framework de test en me basant sur les tests déjà existants.
C'est au cours de la deuxième semaine que j'ai pu implémenter mes premiers tests.

3.5 Les recherches relatives aux tests automatiques dans le cadre du framework utilisé

Tout au long de cette mission, ou plutôt ces missions (un test achevé laisse toujours la place à un autre), j'ai travaillé à implémenter des tests en Java permettant de valider automatiquement le comportement de WebI au niveau SDK.

L'objectif principal de cette mission était d'être, à terme, capable d'implémenter seul et dans les plus brefs délais un test automatique. La grande difficulté de début de cette mission a été de comprendre, d'une part, la logique du framework de test, et d'autre part les différentes choses que celui-ci me permettait de faire.

Les concepts à assimiler étaient nombreux et j'ai passé beaucoup de temps à essayer de comprendre le framework sur lequel repose l'implémentation des tests. Les problèmes majeurs que j'ai rencontrés au début sont les différents points suivants :

— Comment intégrer un test à l'ensemble des tests exécutés ?

— Quel type de test mettre en place, statique ou dynamique ?

— Comment initialiser un test ?

— Comment gérer les sources de telle ou telle version du logiciel pour implémenter un test automatique ?

— Où trouver les fichiers de références nécessaires ?

J'ai éclairci tous ces points au fur et à mesure de mes missions et de mes expérimentations. Certains sont très simples à assimiler mais d'autres m'ont posé beaucoup de problèmes. Je détaille dans les parties suivantes comment est-ce que j'ai été confronté à ces différents problèmes et la manière dont je m'y suis pris pour les résoudre. Dans la partie qui suit je présenterai les différents problèmes que j'ai rencontré et la manière dont je m'y suis pris pour les résoudre. Dans le souci de rester clair dans mes explications je ne m'éparpillerai pas sur tous les tests que j'ai implémenté qui m'ont mis face à tel ou tel problème. Je partirai plutôt d'un cas simple de test automatique en considérant que tous ces problèmes se sont succédés dans le cadre du même test.

Ne connaissant rien ni de Web Intelligence ni du framework de test, je me suis d'abord penché sur la manière d'utiliser ce framework pour implémenter un code simple manipulant un document Web Intelligence.

Pour se faire, j'ai choisi une fonctionnalité très simple, le déplacement d'une cellule dans un tableau, me basant sur un document de test mis à ma disposition (illustration de ce document figure 20).

Partant de ce document, comment devais-je m'y prendre pour automatiser une action sur celui-ci ? Pour se faire il me fallait me baser sur une implémentation de test automatique déjà existante. J'ai donc cherché un code dont la structure me permettait de manipuler un document Web Intelligence déjà existant. Il en existe beaucoup de types

FIGURE 3.2 – Document sur lequel j'ai basé mes première expérimentations

différents mais j'ai rapidement trouvé l'exemple d'un test qui se basait sur un document Web Intelligence. J'ai donc créé une nouvelle classe pour mon test et j'y ai collé le code épuré à l'intérieur, j'avais donc une classe Java me permettant d'effectuer un test sur mon document. Mais à cette étape j'ai rencontré un certain nombre de problèmes m'empêchant de tester mon code qui ne compilait pas.
J'ai passé beaucoup de temps à chercher les solutions dans les messages d'erreurs que je recevais en masse.

J'avais pris le soin de coller mon document Web Intelligence précisément dans le dossier qu'il fallait mais pourtant, les messages d'erreur me disaient que le document était introuvable. En investiguant dans tous les documents rentrant en jeu dans l'exécution du test et en réfléchissant à la signification de chacune des variables, je me suis rendu compte que je n'avais pas modifié le « script ID » du fichier script.xml. Celui-ci permet d'identifier un plan de test, qui lui-même identifie une ou plusieurs classes de test. Une fois la modification effectuée, en prenant soin de correctement renseigner le nom de mon

test, je suis tombé une fois de plus sur une erreur similaire : le document demeure introuvable. Le plan de test étant exécuté, cette fois-ci, l'erreur ne pouvait se trouver que dans ma classe de test. J'ai rapidement identifié le problème puisque c'est dans cette même classe que le chemin d'accès à mon fichier est renseigné. Les répertoires étant nommés différemment en fonction de la suite de test à laquelle ils appartiennent, et, le test que je voulais exécuter ne faisant pas partie de la même suite, lors de l'exécution Java n'allait pas chercher mon document dans le bon répertoire. J'ai donc corrigé cette erreur et ai renseigné la bonne valeur.
En exécutant de nouveau, je constate qu'il n'y a plus de problèmes d'accès à un fichier. Cette fois-ci les messages d'erreur m'annoncent qu'il est impossible d'accéder au CMS. Je me dirige donc vers le CMS en question pour récupérer son adresse puis vers le fichier parameters.xml dans lequel l'url du CMS est renseignée, effectivement ce n'était pas la même, mais d'autre part en essayant d'accéder au CMS dont il était question je constatais que celui-ci n'existait plus, je ne pouvais pas y accéder. J'ai donc corrigé l'adresse du CMS sur lequel je voulais faire mon test.
En exécutant une fois de plus j'ai été ravi de voir que mon test compilait, celui-ci ne faisait encore rien mais je pouvais alors commencer à manipuler mon document Web Intelligence en implémentant le code de mon test automatique.

La question qui se posa alors était de savoir comment y accéder et de quelle manière m'y prendre pour le modifier.
En parcourant les tests automatiques existants et en cherchant les informations qui me manquaient, j'ai constaté avec surprise que nul part dans le code il n'y avait de commentaires pour expliquer la démarche utilisée ou la logique appliquée. Ce qui m'a troublé, puisque l'on m'avait toujours enseigné que les commentaires sont essentiels dans toute implémentation. J'ai évidemment posé la question pour en connaitre la raison. Les équipes de développement et de test de Web Intelligence travaillant en suivant la méthode agile, celles-ci sont censées écrire des codes clairs au point que le code lui-même suffit à sa compréhension. Les quelques rares commentaires servant à expliquer le fonctionnement général d'une grande portion de code. Je comprenais ce principe mais j'étais dérangé par le fait que chacun des objets utilisés dans le code m'étaient inconnus, ce qui augmentait beaucoup le temps qu'il me fallait pour comprendre une certaine implémentation.

En cherchant parmi les tests automatiques existants j'ai trouvé le code qui me permettrait de récupérer mon document :

```
1 IRSReport monRapport = docCalc.getReportAurora(0);
```

L'objet « docCalc » appartenant à la super classe de ma classe de test, contient un grand nombre de méthodes permettant de manipuler un document Web Intelligence. L'argument « 0 » que je passe à la méthode me permet de récupérer le premier rapport que contient mon document. Pour compléter ce code ci-dessus et le faire ressembler à test il faut marquer le début et la fin du test. Le code complété étant le suivant :

```
1 IRSReport reportAurora = docCalc.getReportAurora(0);
```

```
startTestcaseStep("monTest");
stopTestcaseStepWithAction();
```

En observant les logs de l'exécution de ce test, on remaque que la première méthode « startTestcaseStep("") ; » permet de marquer le début des tests, celle-ci ne fait rien d'autre que d'ajouter un log. Ensuite, « stopTestcaseStepWithAction() ; » marque non seulement la fin du test dans les logs mais aussi exécute une série d'actions qui sont décritent dans les logs. Il est intéressant de les noter ici car ces actions peuvent nous être très utiles.
L'étude des ces logs m'a été d'une grande aide car ceux-ci décrivent précisément l'exécution du test.

3.5.1 Recherche effectuées pour connaître les différentes actions de l'exécution d'un test

Les logs sont nombreux et j'ai passé beaucoup de temps à les analyser. J'ai pu en conclure les trois étapes essentielles, et évidentes, de l'exécution d'un test : l'avant test, le pendant et l'après. Un exemple d'une de ces sorties consoles que j'ai ainsi étudié est disponible annexe E page 87 dans laquelle j'ai distingué les différentes parties depuis lesquelles les logs étaient générés.
Pour faire ces obsvervations j'ai simplement ajouté des logs au moments clés de l'exécution : au début et à la fin du constructeur et de mon test (en distinguant la méthode de test et l'implémentation du test). Cela m'a permis d'observer et ainsi de comprendre les différentes étapes du test tels qu'ils étaient dans le framework que j'utilisais. Les tests que j'implémentais étaient exécutés dans un cycle propre au framework, cycle qui m'était inconnu et sans documentation. De cette manière j'ai pu, d'une part, observer dans quelle exécution s'inscrivait mon test, d'autre part, étudier l'exécution de mon test lui-même et comprendre l'utilité de certaines fonctions. Ci-dessous la liste déscriptive de ce que j'ai pu déduire de cette étude (la méthode « run() » dont je parle et la méthode contenant l'implémentation de mon test automatique) :

1. Avant la méthode « run() »
   - Avant toute chose, ce sont les informations contenues dans le fichier parameters.xml qui sont récupérées, autrement dit, le CMS sur lequel le test devra être effectué
   - Vérifie l'existence du dossier contenant les ressources utilisées dans le cadre de tout test
   - Création du fichier dans lequel seront enregistrés les logs
   - Connexion au serveur (CSM)
   - Recherche le dossier FavoritesFolder sur le CMS
   - Supprime, en local, les dossiers où sont enregistrés les résultats (\res\et \ref\)
2. Pendant la méthode « run() » et avant la méthode « stopTestcaseStepWithAction() ; »

— Recherche de l'univers mentionné dans l'implémentation du test automatique
— Création d'un nouveau document sur les CMS, celui-ci est suivi du nom que j'ai donné à l'exécution de mon test (« startTestcaseStep("monTest") ; ») de telle sorte que je vais retrouver sur le CMS le document que j'avais en local.

3. Pendant la méthode « stopTestcaseStepWithAction() ; » (cette méthode étant comprise dans la méthode « run() »)
   — Création du dossier où les résultats seront enregistrés
   — Recherche le dossier « Personnal Documents » sur le CMS
   — Recherche sur le CMS du document concerné par mon test (ce document se retrouve en deux endroits et dans deux états différents, sur le CMS et en local, avant test et après test)
   — Sauvegarde du document en local
4. Après la méthode « run() »
   — Sauvegarde du document dans le personal folder du CMS
   — Copie la référence du cube (référence que j'ai moi-même déposé) vers le dossier \ref\
   — Crée le dossier qui recevra les références
   — Compare le document créé avec la référence

Durant cette partie de mon travail, où l'investigation s'est étalée sur plusieurs semaines et portant sur plusieurs tests à implémenter, j'ai découvert énormément de choses aussi bien en Java que sur l'utilisation du framework de test. J'ai pu mettre au point des méthodes de résolution de mes problèmes et ai pu découvrir la majeure partie des objets utilisés lors de l'implémentation des tests automatiques de Web Intelligence.

3.5.2 Modification d'un document Web Intelligence grâce au framework

Ayant maintenant étudié toute l'exécution du test, je suis plus à même de situer le test automatique dans sa globalité.
La question qui se pose maintenant est de savoir quoi mettre dans le test, avant de pouvoir se poser la vraie question : « Comment tester automatiquement une fonctionnalité ? ».
Au début, je considérais que toute modification applicable depuis l'interface graphique pouvait être reproduite depuis le test. J'avais tort. Mes tests automatiques étant des tests au niveau SDK, les fonctionnalités auxquelles j'avais accès et que je pouvais tester étaient donc limitées à celui-ci.

Pour en revenir à l'implémentation de mon test de base, devant déplacer une cellule, la manière la plus simple était de trouver quelque part dans les tests existants comment cette fonctionnalité était utilisée. Je n'ai pas trouvé d'exemple me permettant d'appliquer rapidement ce changement à mon document mais j'ai trouvé des pistes qui paraissaient prometteuses. La recherche du mot « drop » dans le framework m'a retourné beaucoup

de résultats dont un qui a retenu mon attention, il existe un objet « DropHelperImpl ». On en déduit assez facilement, à partir de son nom, que celui à quelque chose à voir avec le déplacement d'une cellule. Il me fallait trouver un exemple d'utilisation de cet objet pour faciliter mon travail, fort heureusement il existe une fonctionnalité d'Eclipse que je ne connaissais pas avant (et que, depuis lors, je me suis mis à utiliser systématiquement) : « get call hierarchy », me permettant d'avoir le détail de tous les emplacements où cette classe est instanciée. De cette manière j'ai rapidement trouvé des exemples de codes me permettant d'en déduire les quelques informations qui m'étaient nécessaires.

J'ai compris que je ne pouvais pas faire ce que je voulais directement, il me fallait préparer le terrain et récupérer tous les objets qui entraient en jeu dans le déplacement de la cellule.

Il fallait d'abord récupérer, évidemment, le corps de mon document, cela se fait simplement grâce à l'une des méthodes propres au rapport :

```
reportAurora.getPageZone(PageZoneType.BODY);
```

L'étape suivante était de récupérer le tableau contenu dans le corps du document, le problème étant que pour se faire il faut appeler l'objet par son nom. Comment savoir quel était le nom de ce tableau ? Car je pouvais accéder à mon document depuis Web Intelligence mais, depuis l'interface, je n'avais aucun moyen de savoir ce qui se passait derrière, côté logiciel.
La méthode que j'ai utilisé à ce moment là et que j'ai toujours continué à utiliser était de parcourir l'ensemble des éléments contenues dans le corps du rapport et d'en afficher les propriétés. Cela me permettait, non seulement, d'obtenir son nom, son identifiant et tous ses paramètres, mais aussi, d'obtenir son type. Au fur et à mesure que j'implémentais des tests j'ai découvert beaucoup de types différents, leurs imbrications, utilités et correspondances avec les éléments graphiques du document Web Intelligence.
Grâce à cette méthode, j'ai pu déterminer le nom de ce que je cherchais.
Ensuite, de cette table il me fallait récupérer les colonnes, pour pouvoir les manipuler.
Dans les différents tests que j'avais pu étudier auparavant, j'avais remarqué l'utilisation d'une Méthode générique permettant de récupérer une cellule d'un tableau. J'ai utilisé cette Méthode de la même manière que celle utilisée dans les tests existants.
Alors j'ai pu implémenter le code suivant qui me permettait de déplacer une colonne :

```
IRSPageZone pageZone = reportAurora.getPageZone(PageZoneType.BODY);
IRSReportElement findReportElement = docCalc.findReportElement(pageZone.
    getChildren(), "Block 1");
IRSDynamicBlock table = (IRSDynamicBlock) findReportElement;
IRSCell cell1 = docCalc.getCellFromTable("State",(IRSTable) table);
IRSCell cell2 = docCalc.getCellFromTable("Store name ",(IRSTable) table);
IRSCell cell3 = docCalc.getCellFromTable("Name of manager ",(IRSTable) table);

DropHelperImpl dropHelp = new DropHelperImpl();
List<IRSCell> cells = new ArrayList<IRSCell>();
```

```
cells.add(cell1);
cells.add(cell2);
dropHelp.dropCells(cells, cell3, CellZone.RIGHT);
docCalc.applyFormat();
```

Au premier abord, l'utilisation de la méthode « dropCells() » n'est pas instinctive, et j'ai fait plusieurs essais avant de comprendre son fonctionnement. Cette méthode reçoit trois paramètres : un tableau de cellules, une cellule et une position. Le changement appliqué par cette Méthode est que le contenu du tableau de cellule est déplacé comme désiré, ici, à droite de la cellule spécifiée.
La modification appliquée par ce code est illustrée figure 3.3.

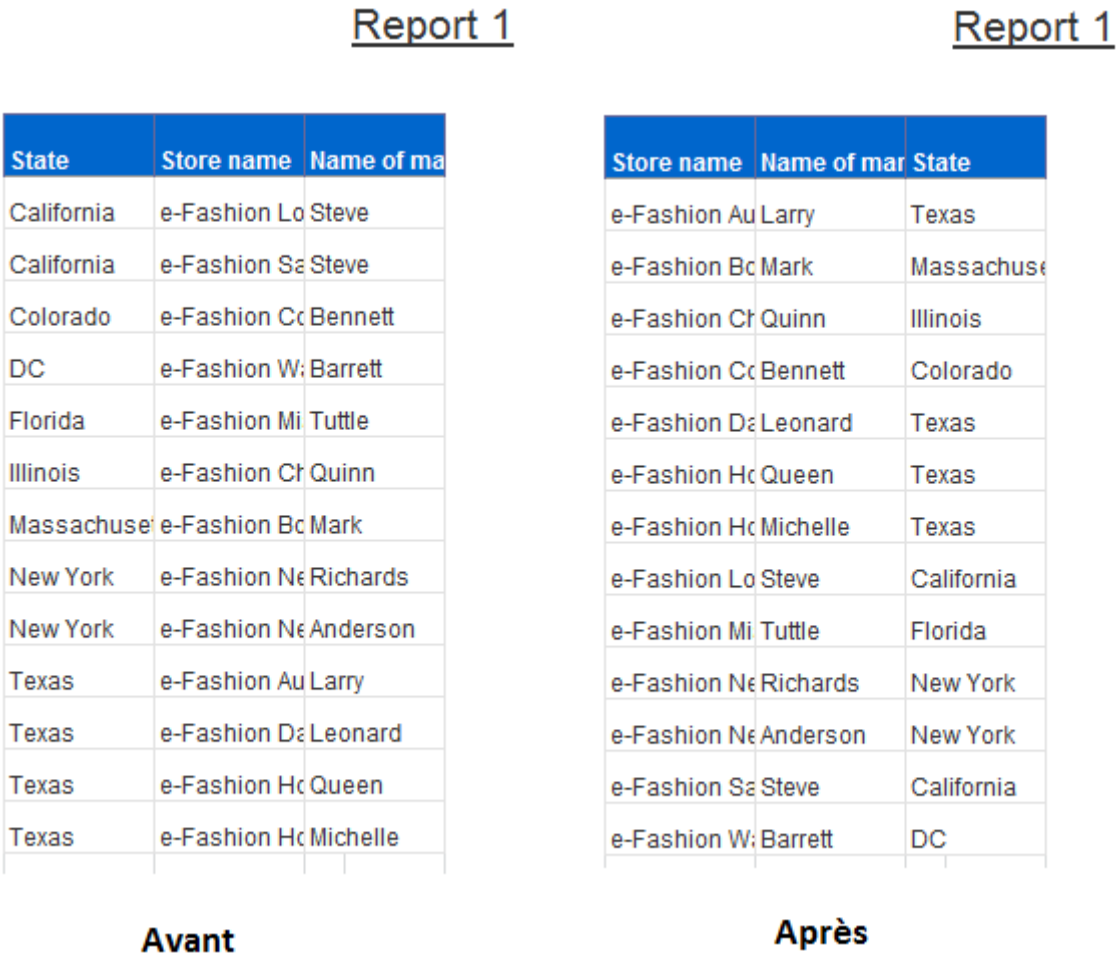


FIGURE 3.3 – Allure du tableau avant et après l'exécution du code

3.6 Synthèse des connaissances

Dans cette partie, je détaille l'ensemble des connaissances que j'ai accumulé sur l'implémentation d'un test automatique d'une fonctionnalité de Web Intelligence.

3.6.1 Tests statiques ou dynamiques

Le test, qu'il soit statique ou dynamique, sera exécuté avec tous les autres dès lors que le nom du test plan est inscrit dans la suite de tests.
La différence fonctionnelle entre ces deux types de tests est que l'un ne fait que comparer deux documents Web Intelligence alors que l'autre permet d'appliquer un certain nombre de modifications sans pour autant les comparer systématiquement à la fin.
La différence en terme de consommation de temps est très importante, puisque pour un test statique il n'est pas nécessaire d'écrire le code du testcase.

Tests statiques

Globalement, le test statique :

— compare un document par rapport à une référence

— demande peu de connaissances techniques, que ce soit en Java ou sur Web Intelligence

La question à se poser est : « Quand implémenter un test statique ? ».
Je n'en ai implémenté que quelques-uns, l'intérêt en terme de difficulté et de choses à apprendre, étant limité. Mais je pense pouvoir dire qu'un test de ce type n'est utilisé que pour valider que l'ouverture d'un document se fait correctement. Autrement dit, ce test permet de s'assurer des compatibilités ascendantes de Web Intelligence.

Principe du test statique

Lors de l'exécution d'un test statique, un fichier est généré à partir d'un fichier puis, celui-ci est comparé avec un fichier de référence.
Si les deux fichiers sont identiques, le test est correct, sinon il échoue.
Au fur et à mesure de mes tests j'ai pu déduire ce que devait contenir nécessairement les différents fichiers à implémenter et les relations qu'ils entretenaient entre eux.
Pour l'exécution d'un test statique, il faut implémenter un plan de test qui respecte la structure suivante :

```
1 package rebean_wi.customers.{customerName}.{customerID};

3 import model.filters.Mode;
  import model.filters.Severity;
5 import model.filters.Type;
  import model.filters.testplan.Suite;
7 import model.filters.testplan.TestPlanType;
  import model.filters.testplan.features.Features_REBEAN;
```

```
9
import org.junit.Test;
11
import tests.exported.annotations.BOTest;
13 import tests.exported.annotations.BOTestPlan;
import extensions.toolbox.WIStaticTestPlanDefinition;
15
@BOTestPlan(Type = TestPlanType.FEATURE_VERIFICATION, Suite = Suite.AURORA,
    Feat = Features_REBEAN.WI)
17 public class CM_{cmNumber} extends WIStaticTestPlanDefinition{
  private static String _scriptID = "CM_{cmNumber}_{short
19 text}";
  public CM_{cmNumber} () {
21   super(_scriptID);
  }
23
  @Test
25  @BOTest(Objective = "Test CM_{cmNumber}_{short
text}' functionality", Severity = Severity.CRITICAL, Author = "", Mode = Mode.
    PROD, Type = Type.FUNCTIONAL)
27  public void CM_{cmNumber}_{short text}" () {
    logNumericResults(launchTC(getTestcase("CM_{cmNumber}_{short text}")));
29  }
}
```

La première fois que j'ai compilé un test de ce type, j'ai été étonné de le voir échouer. Ce que j'ai rapidement compris car son rôle est de comparer le fichier résultant de ce test avec une référence, référence qui n'existe pas. Le problème auquel j'ai été confronté était de savoir comment générer cette référence. J'avais déjà remarqué que, grâce à une configuration particulière du fichier script.xml (voir figure 3.4 page 27), des fichiers étaient générés lors de l'exécution de mes tests. Je me suis alors rendu dans le répertoire où ces fichiers se trouvaient et ai copié celui-ci dans le dossier des références. Ceci fait le test n'échouait plus, mais comment savoir si ce fichier de référence est fiable ? Le rôle du software tester est de s'assurer que ce fichier peut-être utlisé comme référence.

```
<SCRIPT ID="CM_835743_2014_CalcIssue">
    <NAME>webi_customers_issues</NAME>
    <TYPE>WI Static</TYPE>
    <SAVINGCHART>true</SAVINGCHART>
    <SAVINGTXT_>false</SAVINGTXT_>
    <SAVINGTXT>false</SAVINGTXT>
    <SAVINGDOC>false</SAVINGDOC>
</SCRIPT>
```

FIGURE 3.4 – Contenu du script.xml pour générer une référence

Tests dynamiques

À la différence du test statique, le test dynamique va permettre de modifier le document après ouverture. Ceci permettant de s'assurer que la mécanique interne de Web Intelligence produit l'effet escompté sur le document.
C'est principalement ce type de test que j'ai implémenté. La grande difficulté est que, pour chaque test, la logique du code qui doit être implémenté est complètement différente. Et surtout il y a toujours plusieurs manières de résoudre le problème.
J'ai rencontré beaucoup de problèmes à implémenter ces tests automatiques, pour la simple raison que chacune des fonctionnalités que je devais tester était nouvelle pour moi. Mais en plus de connaître la fonctionnalité, ceci se faisant en l'essayant sur une version du produit où celle-ci a été implémentée, il faut pouvoir en reproduire le fonctionnement au niveau SDK.
À cette fin, le plus pratique était d'aller voir directment dans le code du produit ce qu'il se passait. Mais ce n'était pas toujours facile et j'ai passé beaucoup de temps à investiguer pour parfois ne rien trouver.
La méthode que j'utilisais était de me connecter au client DHTML de Web Intelligence, de cette manière je pouvais utiliser le débugger du navigateur, et ceci me permettait deux choses :

1. observer les données qui transitaient et les quelques variables et méthodes en jeu. De là, je pouvais connaître les classes concernées par la fonctionnalité à tester. Cette étape est particulièrement pénible et porte rarement ses fruits. J'ai d'abord eu de grandes difficultés à déceler les bonnes données dans le flot continu d'information. Les quelques fois où j'arrivais à trouver une information (une méthode dont le nom laisse entendre qu'elle a quelque chose à voir avec ce que je cherche), il me fallait retrouver l'implémentation concernée dans le code du produit, étape rendue très difficile par la très grande quantité de classes et d'interfaces différentes.
2. exécuter le Workflow pour arriver jusqu'au bug pour déceler la portion du code où l'anomalie survient. Les difficultés rencontrées sont les mêmes que pour le point précédent.

Outre le fait de trouver le moyen de tester la fonctionnalité, j'ai rencontré un certain nombre de problèmes à implémenter le test automatique, car il en existe de toute sorte et son implémentation de base n'était pas la même en fonction de ce que je voulais faire avec. Au fur et à mesure de mon expérience, je me suis construit une implémentation générique qui permettait de me faire gagner un temps précieux. Les codes que je me suis ainsi construit (plan de test et testcase) et que je copiais/collais sont les suivants :

```
package rebean_wi.customers.{customerName}.{customerID};
//imports
@BOTestPlan(Type = TestPlanType.FEATURE_VERIFICATION, Suite = Suite.AURORA41,
    Feat = Features_REBEAN.WI)
public class CM_{CMNumber} extends WIDynamicTestPlanDefinition {
  private static String _scriptID = " CM_{CMNumber}_{text}";
  public CM_{CMNumber}() {
```

```
    super(_scriptID);
  }
  @Test
  @BOTest(Objective = "Test 'CM_801959_2014_ValuesMissing' functionality", Severity = Severity.CRITICAL, Author = "ptaquet", Mode = Mode.PROD, Type = Type.FUNCTIONAL)
  public void CM_{CMNumber}_{text}() {
  logNumericResults(launchTC(getTestcase("aurora_customers. .{customerName}.CM_{CMNumber}_{text}");
  }
}
```

Et son testcase respecte l'implémentation suivante :

```
package aurora_customers.{ customerName};
public class CM_{CMNumber}_{text} extends MonoDocTestcase {
  MonoDocTestcaseConfigInfo _tccInfo;
  //private final static String _docPath = "";
  MSGStep _step = null;

  Public CM_{CMNumber}_{text} (MonoDocTestcaseConfigInfo tccInfo,String docName) {
    super(tccInfo, docName, "corporate", _docPath, true);
    _tccInfo = tccInfo;
  }
  @Override
  protected void run() throws Exception {
    startTestcaseStep("");
  //%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    //
    _step = _msgHelper.startStep("");
    //Here, the implementation of the code
    _msgHelper.stopStep(_step);


    stopTestcaseStepWithoutActionWI();//Stop without compare
  }
}
```

3.6.2 De l'étude à l'intégration

D'abord, les demandes de tests automatiques ne sont pas toujours réalisables au niveau SDK. Il convient donc de vérifier, en premier lieu, la faisabilité du test. Cette étape nécessite de particulièrement bien connaître le produit pour savoir si telle ou telle action est effectuée au niveau SDK. La méthode que j'utilisais était de tester la fonctionnalité sur le client DHTML afin de savoir quelles méthodes étaient appelées. Ensuite, je me

rendais dans le code de Web Intelligence afin de rechercher le code en question. Si les appels en question se faisaient du SDK, je pouvais implémenter mon test, sinon, si les appels à tester se faisaient plus bas dans l'architecture, je ne pouvais pas implémenter le test.
Cette manière de procéder est longue et sa complexité ne me garantissait pas d'arriver à mes fins. De sorte qu'en règle générale je n'arrivais pas à trouver la réponse. J'essayais toujours mais je finissais souvent par implémenter le test sans savoir si cela serait possible d'en arriver au terme. Cela m'a valu de perdre beaucoup de temps à implémenter des codes qui, au final, n'ont jamais servi.

Pour analyser le test automatique à implémenter je devais d'abord essayer le Workflow qui faisait survenir l'anomalie sur Web Intelligence. Toutes les informations que l'on a sur celle-ci se trouvent sur JCWB (voir figure 3.5 page 30) ou Jira, je pouvais alors connaître :

— L'identifiant du defect (utilisé par les conventions de nommage)
— La description détaillée du Workflow pour faire l'expérience de l'anomalie
— La/Les branche(s) sur laquelle/lesquelles elle a été décelée, de manière à en faire l'expérience deux fois. La première pour vérifier l'existence de l'anomalie et la seconde pour valider que celle-ci a été correctement corrigée.

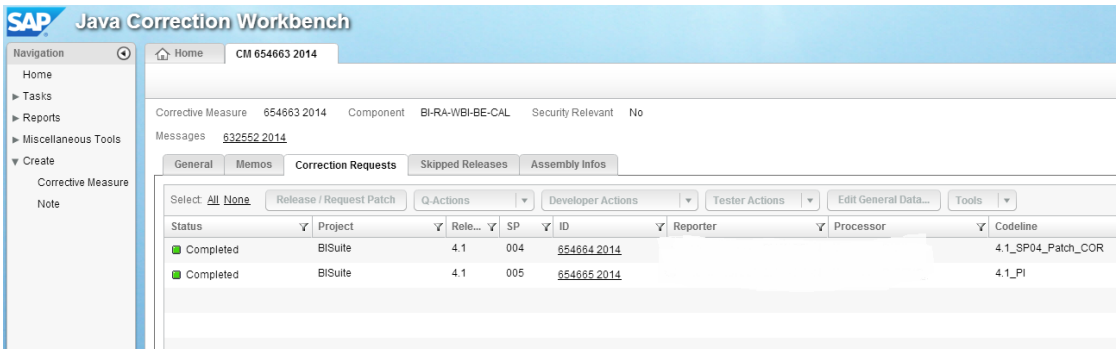


FIGURE 3.5 – Écran de JCWB propre à une CM

Cette partie préalable à l'implémentation du test automatique m'a permis de manipuler le produit Web Intelligence et plus particulièrement les fonctionnalités que je devais tester. J'ai appris beaucoup de choses sur son utilisation et ses fonctionnalités.
Pour cela j'ai du apprendre à l'utiliser, créer des rapports et effectuer des changements dessus. Web Intelligence est un produit incroyablement riche en fonctionnalités dans lequel il est possible de faire beaucoup de choses. Cette expérience est d'autant plus riche que toutes les connaissances et savoir-faire que j'ai acquis pourront me servir dans le futur. Je suis très loin de maîtriser Web Intelligence mais j'ai pu en comprendre l'utilité et la logique d'utilisation. Je serai donc capable, à l'avenir, de faire valoir cette compétence.

Lors de mes investigations sur les fonctionnalités et les anomalies, en plus de devoir reproduire le Workflow à la main depuis l'interface graphique, je devais étudier les cor-

rectifs appliqués pour orienter ma logique d'implémentation du testcase. Cela a été très formateur car les différents codes que j'étudiais avaient été implémentés par des développeurs confirmés. Il est connu que c'est en lisant le code d'autrui que l'on progresse, et j'ai appris énormément de choses, tant du point de vue de la logique de la programmation orientée objet que de la complexité des procédures utilisées. De plus, en observant les modifications faites pour corriger l'anomalie j'ai pu comprendre plus en détail les différentes raisons qui font qu'une telle anomalie puisse exister (voir l'interface graphique qui me permettait de faire ces observations figure 3.6 page 31). Ce qui peut être, entre autre, une condition manquante, un problème d'identifiant ou une mémoire saturée.

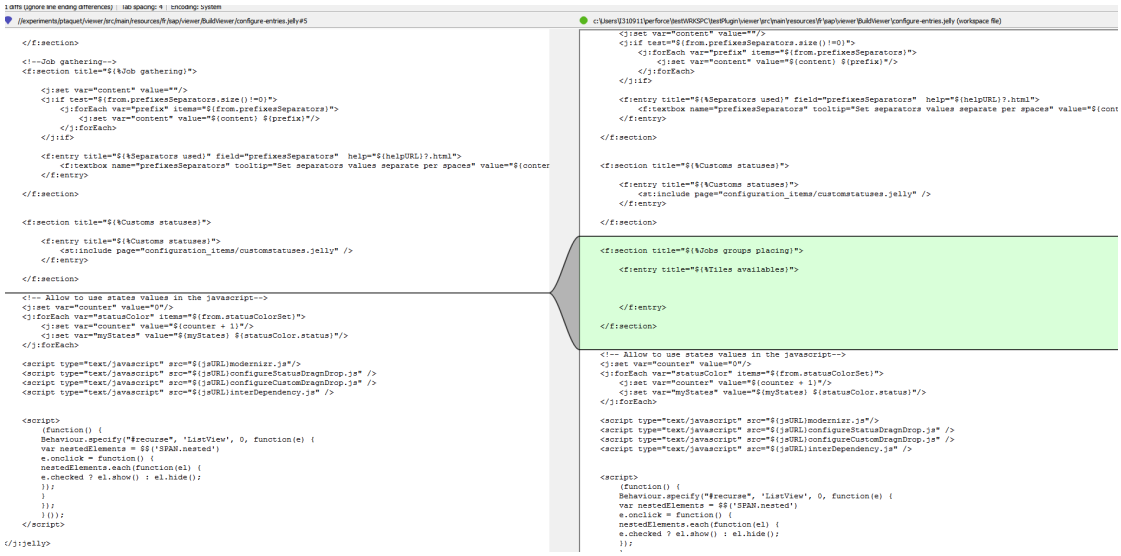


FIGURE 3.6 – Écran de comparaison des 2 versions d'un même fichier (avant et après correctif)

Implémentation de la coquille vide

Après avoir fait l'étude de l'anomalie, il faut commencer à implémenter le test automatique. La première difficulté que j'ai rencontré étant le choix du point de départ, il en existe plusieurs. Mon tuteur m'a guidé dans le choix de ceux-ci et m'en a expliqué les différents intérêts. Il était difficile de comprendre comment et pourquoi utiliser l'un ou l'autre car cela dépendait directement de ce qui devait être fait dans le test automatique. Si le test portait sur des éléments graphiques ou des configurations il était plus simple de baser son test sur un document Web Intelligence, car celui-ci me permettait d'appliquer la majeure partie des actions accessibles depuis l'interface graphique. Si je voulais plutôt vérifier un contenu quelconque, il était plus simple de se baser sur une queryspec car toutes les données y sont directement accessibles.

J'ai majoritairement utilisé les documents Web Intelligence pour démarrer mes tests. Mais, une fois, je me suis retrouvé confronté à un problème que je ne pouvais pas résoudre car je n'avais pas choisi la queryspec. Je voulais accéder à une information du document qui n'était pas disponible depuis le SDK, en utilisant le document Web Intelligence, mais qui l'était depuis la queryspec. J'ai cherché pendant de longues heures, en étant au plus proche de ce que je cherchais mais malheureusement aux limites de ce que le SDK me permettait de faire. Je n'ai jamais réussi à résoudre le problème seul, je suis donc allé voir mon tuteur pour exposer mon problème, celui-ci m'a expliqué le pourquoi de l'obstacle que je rencontrais et m'a conseillé d'utiliser plutôt la queryspec. Après être retourné dans mon bureau, j'ai pu résoudre mon problème et finir d'implémenter mon test automatique avant la fin de la journée.

Plus que d'avoir heurté les limites du champ d'action du SDK, ma principale erreur a été de ne pas me diriger plus tôt vers la personne qui aurait pu me conseiller, cela m'aurait épargné plusieurs heures d'efforts inutiles.

Le choix du point de départ pour implémenter un test automatique est donc très important. Mais quelque soit ce choix, il faut aussi implémenter la coquille vide de ce test, qui est, dans tous les cas, quasiment similaire. La coquille vide permet d'avoir un code qui compile mais qui ne fait encore rien. Ce qui garantit que tous les éléments sont bien reliés entre eux. Car, comme schématisé figure 3.7 page 32, chaque test automatique est intégré à un plan de test, lui-même intégré à une suite de tests.

FIGURE 3.7 – Diagramme UML des suites de tests

Les fichiers essentiels au bon fonctionnement du test sont décrits ci-dessous. Il n'était pas évident, au début, de compléter correctement tous ces documents car chacun a une utilité très particulière :

— Test plan
— Test case
— Test suite
— script.xml
— ressources (queryspec ou document Web Intelligence)
— parameters.xml

La principale difficulté que j'ai rencontré lors de l'implémentation de la coquille a été la convention de nommage. En effet, la moindre erreur ne permet pas au test de

compiler.
Pour illustrer la coquille vide du test automatique et pour permettre de les visualiser dans la structure du framework, je les ai représenté dans la figure 3.8 page 34. Et, de la même manière, la figure 3.9 page 35 présente le test avec les différents documents avec lesquels celui-ci interagit.
Les ressources sont très importantes dans le contexte du test, et j'ai eu quelques difficultés au début pour arriver à les générer rapidement.
Il m'est arrivé plusieurs fois de générer des fichiers de références à partir de la mauvaise version, ce qui avait pour effet de faire échouer mes tests sans que je puisse, rapidement, savoir pourquoi.

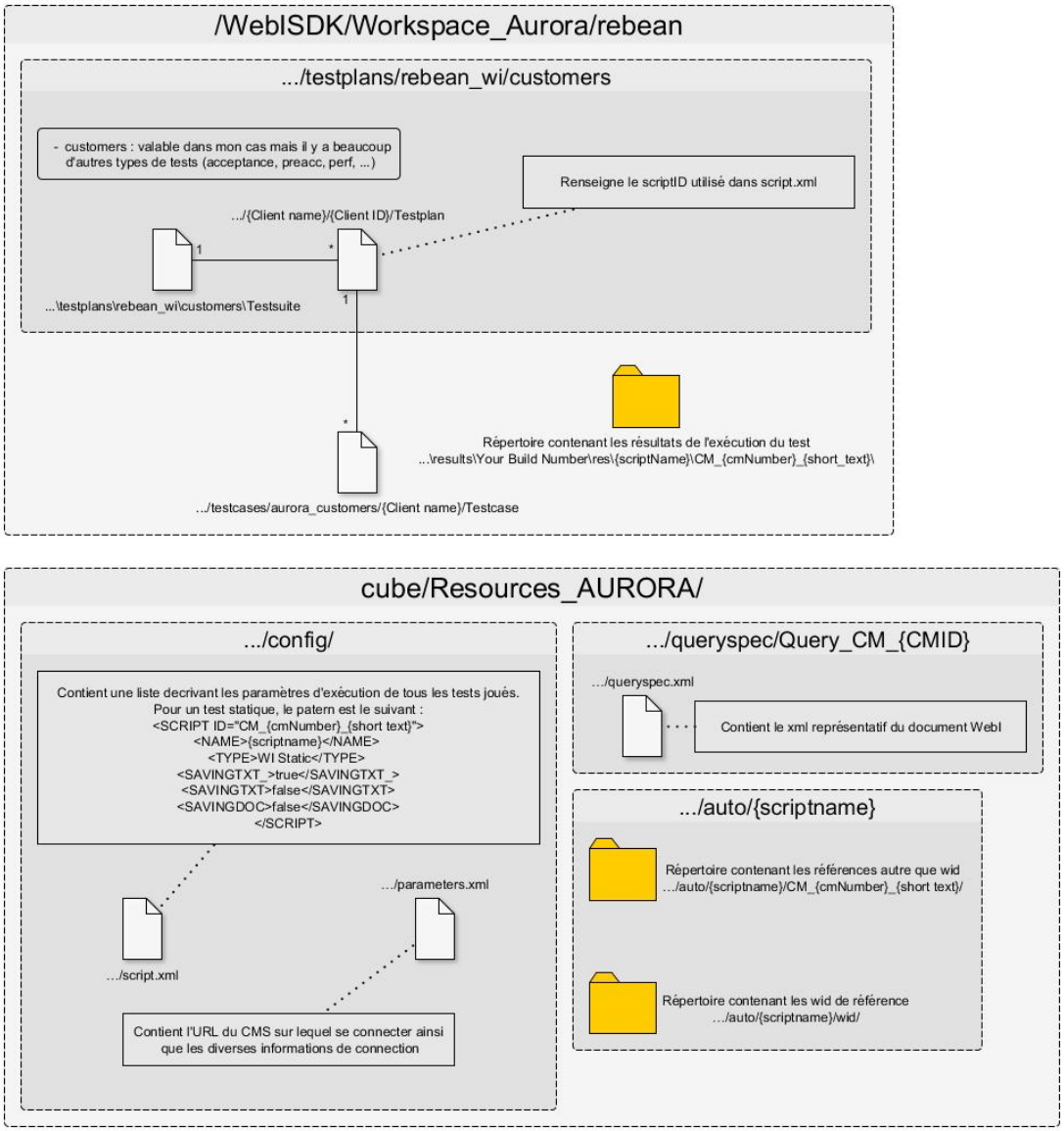


FIGURE 3.8 – Diagramme représentant les différents éléments qui compose la coquille vide d'un test dynamique

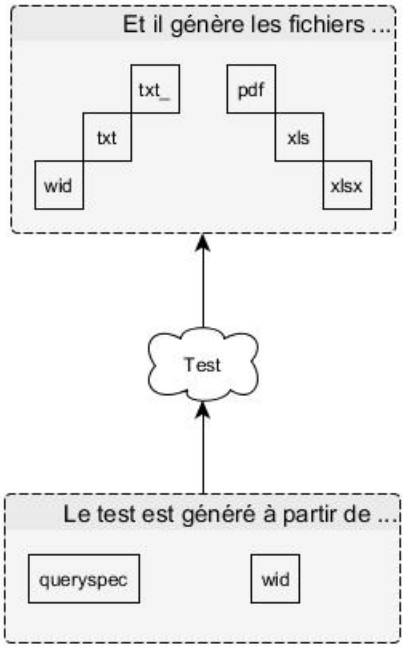


FIGURE 3.9 – Les fichiers utilisés ou générés par le test

Les ressources

Comme j'en ai déjà parlé plus haut, il existe plusieurs types de ressources. Les deux principales étant le document Web Intelligence et la queryspec. Il y a deux manières d'obtenir un document Web Intelligence, les deux sont similaires.

Obtenir un fichier wid

Via le CMS Il suffit de parcourir l'arborescence du CMS pour arriver à l'emplacement du .wid. Dans les propriétés du fichier, il y a son nom complet (différent du nom dans WebI) avec son arborescence à partir du dossier Input (figure 3.10 page 36). Ensuite, dans le système de fichiers du serveur (par exemple : « \\dewdftv01634.dhcp.pgdev.sap.corp\c$ ») aller dans « \Program Files (x86)\SAP BusinessObjects\SAP BusinessObjects Enterprise XI 4.0\FileStore\Input » et copier/coller le chemin d'accès au fichier. Le document Web Intelligence se trouve dans le répertoire en question.

Via le Rich Client Avec ce procédé il faut faire attention à la version du rich client qui doit être la même que celle du CMS, une erreur se traduit par un fichier de référence erroné, donc un test qui échoue. J'ai fait cette erreur plusieurs fois, au début, car j'avais toujours plusieurs versions de Web Intelligence qui étaient ouvertes. L'utilisation du client lourd est la plus instinctive car il suffit d'ouvrir le document et de l'enregistrer en local.

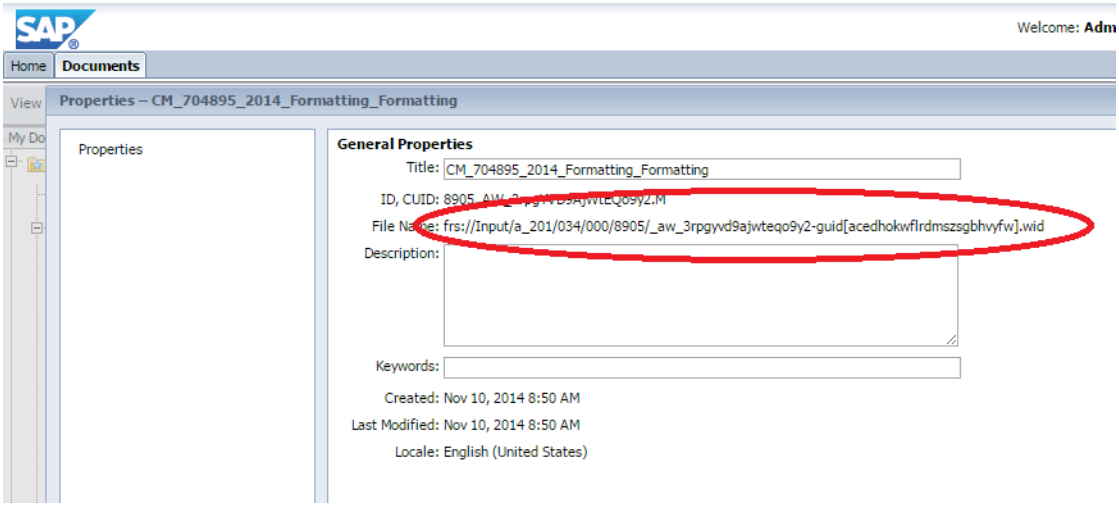


FIGURE 3.10 – Capture de l'écran de propriété d'un document WebI

3.7 Bilan de la 1ère période

Tout au long des mois de juillet, août et septembre j'ai implémenté de nombreux tests [2] pour des bugs divers, que ce soit une faute de frappe ou un dysfonctionnement. J'ai beaucoup appris de mes erreurs, surtout lorsque plusieurs jours de travail s'avéraient inutiles parce qu'une meilleure solution, évidente pour qui sait, existait.
J'ai aussi pu parfaire mes connaissances en Java ainsi que ma méthodologie lorsque je dois me former à une nouvelle technologie.
Au début de cette mission, je n'avais jamais implémenté de tests à l'exception de tests unitaires. Aujourd'hui, je suis ravi de comprendre la problématique du test et quels types de tests existent. Aujourd'hui je sais pourquoi et comment les implémenter et suis capable de faire des tests qui soient facilement maintenables.

3.7.1 Connaissance du framework

La principale perte de temps était dûe à une méconnaissance du Framework de test. Celui-ci possédant un grand nombre de méthodes aux intérêts divers et variés, d'emblée, il est très difficile de pouvoir déterminer laquelle utiliser. D'autant plus qu'au moment où le besoin de telle ou telle méthode se faisait sentir, je n'avais pas connaissance de son existence. En conséquence, je me suis vu trop souvent implémenter des méthodes déjà existantes, me faisant perdre un temps précieux.
L'expérience accumulée durant cette période m'a permis de ne plus perdre de temps à ré-inventer la roue et de me concentrer sur le code ad hoc nécessaire à un test précis et fiable.

3.7.2 Méthodologie

L'autre grande perte de temps venait surtout de ma méconnaissance du produit testé. Certaines fonctionnalités étant très complexes il m'arrivait de ne pas identifier exactement le problème pour finalement perdre du temps à implémenter du code inefficient.
Le meilleur exemple que je puisse citer s'est produit quand j'ai commencé à tester des bugs intégrés à des workflows complexes. Un certain nombre de manipulations et/ou de configurations était alors nécessaire pour que le bug survienne. Et c'est dans ces conditions que j'ai implémenté la quasi totalité du workflow au niveau SDK pour finalement arriver sur la zone à tester. Le test fonctionnait bien, il échouait sur les versions buggées et passait sur les versions fixées. Le problème ne vient pas du test que j'ai implémenté, car celui-ci faisait ce qu'il fallait, mais de la manière. J'ai implémenté ce code en approximativement 3 jours. Alors que si j'avais utilisé le CMS pour générer un document à une étape seulement du bug, et si j'avais implémenté uniquement l'appel à tester, j'aurais pu implémenter ce test en quelques heures seulement !

2. cf. annexe B page 77 pour la liste complète des tests implémentés

3.7.3 Travail en équipe

Le framework de test est entretenu quotidiennement par des dizaines de testeurs implémentant des tests et modifiant le framework lui-même. Le résultat de l'exécution des suites de tests est visible par les testeurs, développeurs et par d'autres. J'étais donc intégré à une grande équipe, leurs travaux ayant des répercussions sur les miens, et les miens sur les leurs. Il faut donc faire attention au code que l'on « push » sur Perforce ! L'anecdote me venant à l'esprit s'est passée un vendredi soir, alors que je venais de terminer un test et que celui-ci se comportait bien. J'ai livré mon test sur Perforce. L'ennui était que j'utilisais un package dans lequel je mettais des méthodes de test pour ne pas polluer mon test « final ». Je n'ai malheureusement pas pensé à revoir l'import de mes sources. Mon code, compilant correctement en local, ne compilait plus sur le serveur et a donc fait crashé toute la suite de test. Le mail automatique envoyé a informé tout le monde de mon erreur de manipulation (illustrée annexe C page 83) !

J'ai eu aussi la chance de travailler dans une équipe maitrisant très bien le framework et le produit. J'ai appris énormément pendant les entretiens que j'ai eu avec ces différentes personnes.

Chapitre 4

Migration Jira/Java Correction WorkBench

4.1 Qu'est-ce que Jira et Java Correction WorkBench (JCWB)

Jira est un logiciel de traçabilité des problèmes développé par Atlassian dont l'usage est gratuit pour les organisations à but non lucratif, de charité ou les projets open-source. SAP désire migrer de système de traçabilité pour utiliser maintenant Java Correction WorkBench (JCWB ou CWB).

JCWB est un logiciel interne à SAP dont l'utilisation à été imposée par la hiérarchie, son domaine d'utilisation est la gestion des corrections. Historiquement, SAP utilise Jira pour gérer le projet, les problèmes (defects) et les corrections.
La stratégie de gestion des bugs changeant, SAP a pris la décision d'utiliser un seul et même outil de gestion des problèmes en interne comme en externe (à l'usage des clients) : BCP. Cet outil permet de renseigner un problème, quel qu'il soit, et s'il s'avère que ce problème est un bug logiciel il est transféré vers JCWB, qui suivra ce bug toute la durée de la correction.

4.2 Présentation du contexte

À SAP, tous les codes des différents projets sont hébergés sur le gestionnaire de version Perforce (SCM : Source Code Management) et ceux-ci sont compilés plusieurs fois par jour grâce à Jenkins (CIS : Continuous Integration Software).
Chaque projet est constitué de deux choses distinctes, d'une part, son code source et, d'autre part, les tests joués pour garantir [1] le bon fonctionnement du produit. Lorsqu'il y a un problème sur la build, que ce soit une erreur de compilation ou un test qui échoue, le statut de la build change pour être représentatif du problème. Dès lors que quelqu'un

1. La valeur de la garantie dépend directement du test coverage

s'en aperçoit, il inscrit un defect dans Jira [2] ou dans JCWB, en fonction de la nature du problème

4.3 Plugin de reporting

Pour leur permettre d'avoir un rapport visuel sur l'état des builds, ils utilisent le plugin Radiator. Ce plugin permet l'affichage, sur un seul écran divisé, des groupes de jobs [3]. Un groupe de jobs est représenté par un carré coloré dont la couleur représente l'état des jobs qui le composent, voir la capture d'écran figure 4.1 page 40.
Les statuts pris en compte sont les statuts Jenkins [4] ainsi qu'un statut supplémentaire issu du plugin Claim. Ce plugin permet à un développeur de « claimer » une erreur, c'est-à-dire, ajouter au projet une information supplémentaire : le nom de l'utilisateur qui a « claimé » et le message que celui-ci a laissé aux autres utilisateurs.

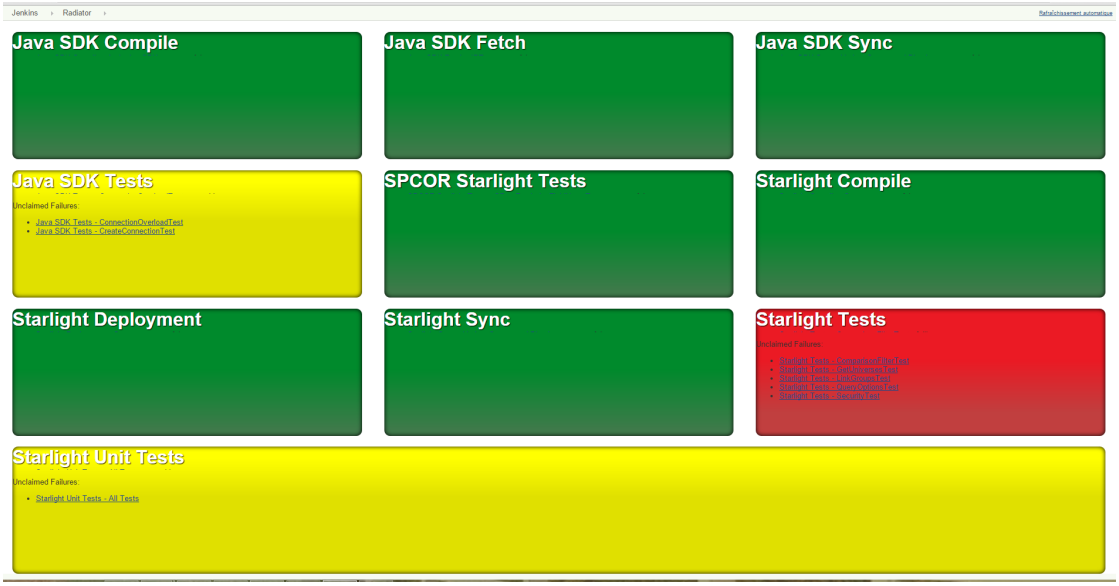


FIGURE 4.1 – Plugin Radiator utilisé actuellement

En résumé Radiator permet de :

— rassembler les jobs en un seul groupe de jobs, où chacun des jobs a un statut [5]

— afficher une couleur représentative du statut jugé le plus important

— prendre en compte les informations supplémentaires du plugin claim

2. L'utilité de Jira est bien plus large que la simple déclaration d'un test échoué, il sert à décrire n'importe quel problème quelle que soit la version, la branche ou le produit
3. Typiquement, un job correspond à un projet logiciel
4. Error, Failure, Unstable, Aborted, Not built, Success
5. statut propre à Jenkins

Prennons l'exemple où nous avons deux groupes composés chacun de trois jobs, tous différents. Supposons que, sur les 6 jobs, l'un soit en échec les autres en succès. Nous aurons donc une vue colorée en vert (parce que tous les projets sont en succès) et la deuxième vue colorée en rouge (l'un des jobs ayant échoué) ou bien en orange si l'échec est « claimé » (échec claim par un développeur).
La figure 4.2 page 41 illustre bien les sources d'informations du plugin. Il puise ses résultats depuis Jenkins et le plugin claim, les résultats de Jenkins provenant, d'une part, des résultats de tests JUnit, et d'autre part, des compilations, synchronisations vers Perforce, ...
L'inconvénient de ce procédé est que lorsque qu'un test a échoué, tout le groupe de test passe en rouge. Dans cette situation, il devient impossible de détecter les nouveaux tests qui échouent.

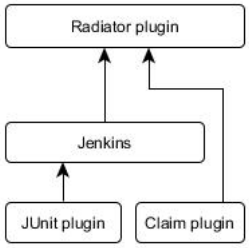


FIGURE 4.2 – Contexte d'utilisation du plugin Radiator utilisé actuellement

La solution simple, mais permettant de contourner le problème et de ne plus polluer l'affichage avec du rouge, consiste à faire passer les jobs en échec avec defect (ie. reconnus et enregistrés comme échecs dans Jira ou CWB) en vert.
De cette manière, puisque seuls les nouveaux échecs apparaissent en rouge, toute régression est visible immédiatement.
Cette transformation est mise en œuvre grâce à une annotation, celle-ci est présentée figure 4.3 page 41.

```
@Test
@Defect(url="https://sapjira.wdf.sap.corp/browse/BITBIWEBISL2-1542", message="expected:<WARNING> but was:<OK>")
public void joinDependency_missingTable() throws Exception {
	MonoSourceDataFoundation dataFoundation = create_tables_join(newDataFoundation);

	// Delete one of the derived tables
	DerivedTable table = getTable(dataFoundation, "Customer Derived", DerivedTable.class);
	dataFoundation.getTables().remove(table);

	// Save and check the IStatus returned
	IStatus status = localResourceService.save(dataFoundation, new URI(dataFoundation.getResourcePath()).getPath(), true);
	AssertHelpers.dumpStatus(status);
	TestCase.assertEquals(Severity.WARNING, status.getSeverity());
}
```

FIGURE 4.3 – Annotation utilisée pour vérifier le statut du defect Jira

De cette manière le test est relié au defect. **Et si** :

1. le test a échoué mais le statut est « Active » ou « Open »
2. le message d'erreur est de la forme attendue

Alors, le test sera considéré en succès par Jenkins. Ensuite, quand un développeur livrera la correction, le defect passera à « Validate », ce qui court-circuitera l'annotation [6]. Et en fonction de la validité de la correction, le test redevient vert ou reste rouge.
De cette manière, seuls les tests non investigués sont rouges, ce qui permet de réagir beaucoup plus rapidement et réduire le temps d'investigation.

Pour résumer, la relation qu'entretien BCP et JCWB est illustrée figure 4.4.

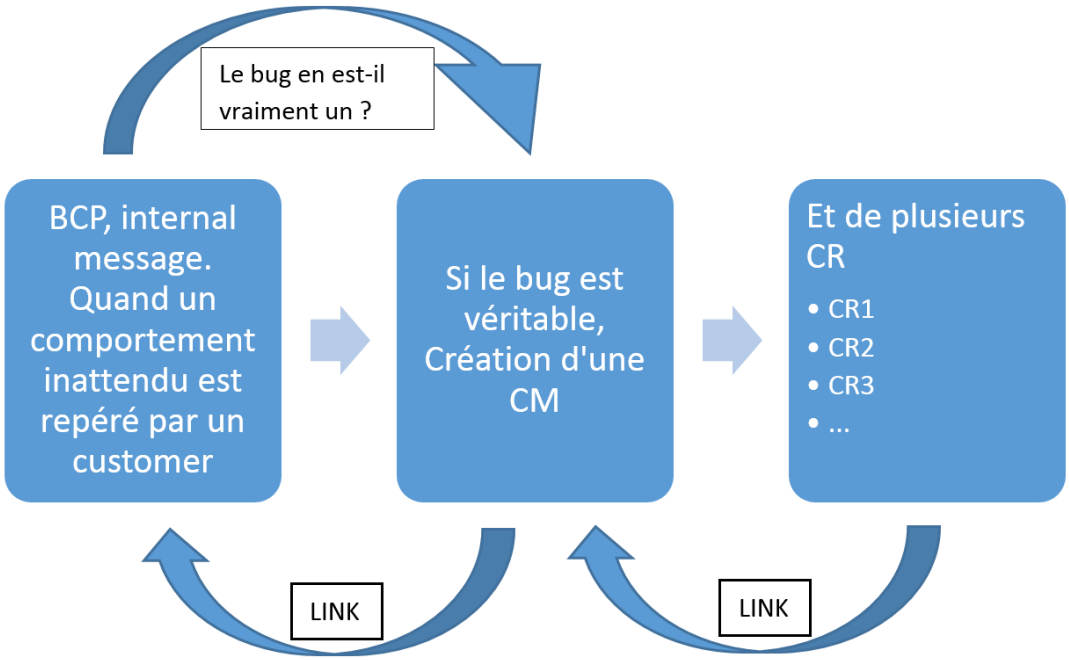


FIGURE 4.4 – Process de gestion de bug utilisé

Problématique

Actuellement, seuls les defects rentrés sur Jira sont pris en compte, la mécanique en place ne permet pas de récupérer ces mêmes informations de JCWB.
Puisque nous ne pouvons pas aller chercher le résultat de JCWB, il faut mettre à jour le process actuel et permettre aux statuts d'être ajustés quelque soit l'origine du defect (Jira ou CWB).
La problèmatique supplémentaire relève de l'architecture différente de Jira et de JCWB [7]

6. même comportement pour le statut « Closed » ou n'importe quel autre qui n'est pas « Active » ni « Open »
7. Contient la « Correction Measure » (CM) et les « Correction Requests »

(CR), ce qui nous oblige à aller receuillir des informations à deux endroits différents. Le premier objectif étant de récupérer l'identifiant de la « Correction Request » à laquel le defect est associé et l'identifiant de la « Correction Measure » à laquelle la CR est associée.

4.4 Travail effectué

La première partie du travail s'est cantonnée d'abord à des échanges de mails parallèlement à des recherches.
La difficulté de cette partie a été de comprendre le problème dans sa globalité. D'abord, je ne connaissais pas le process qui nécessitait cette modification. Dans la partie précédente de mon contrat j'ai pu utiliser Jira et JCWB ainsi que les defects qui y étaient enregistrés, mais c'était tout. Les différentes choses essentielles à mon projet sont les suivantes :

- il existe un web service permettant de récupérer les informations d'une CR : CWB HTTP API
  - l'utilisation de ce web service se fait via une authentification normalisée par SAP : PROD-PASS-ACCESS
- le fichier à implémenter pour compléter le comportement actuel est JiraIssueWatcher.java

4.4.1 Fonctionnement et utilisation de prod-pass-access

La première chose sur laquelle je me suis penché a été de comprendre le fonctionnement de prod-pass-access, API développée par et pour les services de SAP pour sécuriser ses tranferts d'informations.
Je suis donc allé sur le portail de l'entreprise pour étudier cette librairie, la première chose que j'ai apprise est que prod-pass-access signifie : « Production password access » et que son rôle est de lire et écrire des credentials [8].
Dans les différentes explications que j'ai pu trouver il était indiqué que pour utiliser cette API il fallait avoir été, au préalable, enregistré dans un dossier particulier sur un serveur. Je ne comprenais pas grand chose à comment je pouvais utiliser tout ceci, j'ai donc téléchargé l'API en question et ai fait mes expériences en lignes de commande.
Je ne connaissais pas cet outil et il me fallait découvrir comment celui-ci fonctionnait car il me permettrait ensuite de communiquer avec le serveur pour pouvoir utiliser l'API CWB HTTP API.

La documentation de cette API était plutôt claire (figure 4.5), et des renseignements ainsi obtenus j'ai pu créer un utilisateur.

J'ai suivi les informations me permettant de générer un utilisateur, et au fur et à mesure le fonctionnement de cette API s'est éclairci.

8. Les credentials sont les noms d'utilisateurs ou mots de passe

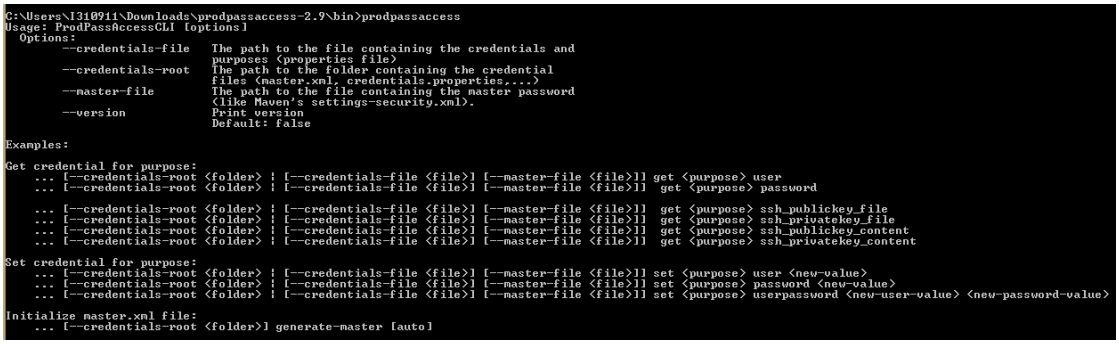


FIGURE 4.5 – Documentation fournie par l'API

Cette API sert à identifier l'utilisateur grâce aux fichiers enregistrés d'une part sur le serveur et d'autre part sur la machine faisant la requête. Les fichiers en local contiennent non seulement le nom d'utilisateur mais aussi le mot de passe en crypté. La clé pouvant décrypter le mot de passe est hébergée sur le serveur et lui seul peut décrypter le mot de passe crypté envoyé par l'utilisateur.

J'ai donc créé mon utilisateur grâce aux commandes suivantes :

```
prodpassaccess generate-master
Prodpassaccess set myPurpose user pierre
Prodpassaccess set MyPurpose password password012
prodpassaccess --credentials-file credentials.properties --master-file master.
    xml set myPurpose user pierre
```

J'ai fait plusieurs tests avant de pouvoir exécuter ces lignes de commandes mais le principe d'utilisation est assez simple.
La première ligne va permettre à l'utilisateur de fournir un mot de passe. Après l'exécution, je suis allé voir (comme précisé dans la documentation sur le portail de SAP) dans le dossier prodpassaccess contenu à la racine de mon dossier User, dans celui-ci il y a le fichier master.xml contenant une clé de cryptage.
Les deux lignes suivantes permettent de créer un utilisateur et un mot de passe pour un « purpose » en particulier.
La dernière ligne, elle, permet d'enregistrer les informations cryptées dans un fichier « credentials.properties »..
J'avais donc réussi à créer un utilisateur mais je ne savais toujours pas comment l'utiliser dans mon code java. Je me suis donc arrêté là, car je ne pouvais pas tester ce que j'avais réussi à faire.

4.4.2 Accès au web service

Maintenant il fallait que je trouve le moyen d'envoyer une requête au web service. Mais avant cela, comme précisé dans la documentation, ce n'était pas une connexion HTTP classique mais SSL nécessitant l'utilisation d'un certificat. Je ne savais pas ce que cela était. J'ai donc discuté avec un collègue ayant déjà utilisé un service similaire et il m'a envoyé un code Java qui pourrait m'inspirer. En fouillant dans son code j'ai trouvé une méthode pemettant de passer le contrôle des certificats. Après cela, je pouvais manipuler une connection HTTPS depuis mon code Java, mais encore une fois, il ne m'était pas possible de tester le code que j'avais.
À ce moment, je pouvais donc :

— utiliser l'API de cryptage des informations : prod-pass-access

— me connecter, grâce à une connexion sécurisée, à une url

Il me manquait le point essentiel, l'url du web service que je voulais utiliser. Je suis donc aller chercher l'information parmi les membres de mon équipe, ils ne savaient pas non plus, mais m'ont donné le nom de celui qui s'était occupé de développer cette API, un collègue travaillant sur le site de Walldorf, en Allemagne. Je lui ai donc envoyé un mail pour connaître le moyen d'utiliser son API et lui ai demandé sa documentation. Nous avons discuté un peu par mail car celui-ci voulait savoir pourquoi j'en avais besoin et si cela était vraiment nécessaire. Après quelques échanges de mail, celui-ci m'a transmis la documentation relative à l'API dont j'avais besoin.
J'ai donc étudié la documentation pour savoir quelle requête je devais envoyer.
J'ai trouvé exactement ce qu'il me fallait, il y avait, parmi les différentes méthodes accessibles via le web service, celle qui me retournait le statut de la Correction Request dont l'identifiant est passé en paramètre dans la requête. La requête que je devais envoyé était de la forme :

```
https://css.wdf.sap.corp/sap/bc/bsp/spn/jcwb_api_extern/get_correction_requests
    ?pointer=<id de la correction request>
```

J'ai immédiatement essayé cette url que j'ai collé dans la barre d'adresse de mon navigateur avec l'identifiant d'une correction request valide. L'accès à cette url m'affichait uniquement un pop-up me demandant de renseigner un nom d'utilisateur et un mot de passe. J'ai essayé avec mes identifiants personnels que j'utilise tous les jours pour me connecter aux services sécurisés se SAP. La connexion est refusée.
C'était donc ici que je devais trouver le moyen d'utiliser les credentials créés grâce à prod-pass-access, et de les intégrer à ma requête HTTPS.
J'ai complété mon code Java avec l'url du web service. J'ai trouvé dans la documentation de prod-pass-access la marche à suivre pour accéder aux credentials directement depuis mon code Java. J'ai donc importé le JAR de prod-pass-access dans mon projet.
D'autre part, il fallait transmettre les credentials d'une autre façon que par l'url, En étudiant les documentations des requêtes http j'ai trouvé les paramètres qui correspondaient exactement à ce que je cherchais.
Le code dont il est question est disponible dans l'annexe F page 95, la solution pour se

connecter via HTTPS est implémentée dans la méthode « sendRequest() » et l'utilisation de l'API ainsi que l'url de l'API « HTTP CWB API » est implémentée dans la méthode « findStatus() ».

J'ai exécuté mon code contenant tous les éléments pour que cela fonctionne, j'avais tout considéré dans mon code pour me connecter au web service :

— j'avais réglé le problème de la connexion sécurisé (protocole SSL)

— je pouvais accéder à mes credentials depuis mon code Java

— j'avais renseigné la bonne url et effectué la bonne requête pour accéder au statut de la Correction Request

— j'avais renseigné mes credentials de la manière préconisée par les spécifications du protocole HTTP

En réponse, le serveur m'a renvoyé ceci « ERROR : Bitte übergeben Sie Benutzer und Password »

4.4.3 Création du nouvel utilisateur

À ce moment je me suis souvenu de la documentation de prod-pass-access, particulièrement d'un élément que je n'avais pas compris quand je l'avais lu pour la première fois. Il s'agissait dans dossier particulier, hébergé sur un serveur à Walldorf. Celui-ci contient tous les credentials des utilisateurs autorisés à utiliser ce service. Je me suis donc repenché sur la création d'un utilisateur et tout se qu'il me restait à faire était de stocker mes informations sur ce serveur en question, et pas en local comme je l'avais fait lors de mes tests.
Je suis allé chercher dans la documentation l'adresse de ce fameux dossier pour y enregistrer mes credentials dont l'adresse est

```
1 \\production.wdf.sap.corp\info\make\
```

Ce dossier contient un grand nombre d'autres dossiers, d'après la documentation, il s'agit de tous les utilisateurs utilisant ce service. J'ai choisi un user générique « pblack » utilisé par beaucoup d'ingénieurs de chez SAP pour effectuer leurs tests avec ce service. Ce dossier correspond à mon dossier User que j'ai sur ma machine en local. J'ai donc ré-exécuter la commande permettant de stocker les credentials, mais cette fois-ci en précisant le nouvel emplacement.

```
1 prodpassaccess
  --credentials-file \\production.wdf.sap.corp\info\make\pblack\.prodpassaccess\
      credentials.properties
3 --master-file \\production.wdf.sap.corp\info\make\pblack\.prodpassaccess\master
      .xml
  set myPurpose pierre
```

Ceci fait, j'ai ré-exécuter mon code Java pour me connecter à l'API. La réponse que j'ai reçu était « Open », ce qui voulait dire que le statut du defect dont j'avais renseigné

l'identifiant dans la requête était toujours en cours de traitement. Mais surtout, cela voulais dire que toute mon implémentation fonctionnait.
Je pouvais maintenant revoir mon code, le clarifier et finalement le transmettre pour validation.

4.5 Conclusion de la mission

La mission qui m'a été attribué devait compléter le fonctionnement de quelque chose qui existait déjà. Je devais donc, au préalable, étudier le contexte dans le lequel celle-ci s'intégrait. D'autre part, en plus de m'intégrer dans le code déjà existant, je devais intégrer à mon code une API que l'utilisation d'une autre API nécessitait. J'ai appris énormément de choses en Java, même si en terme de lignes de code je n'en ai pas écrit beaucoup, il m'a fallu beaucoup de recherches pour effectuer ce que je voulais, que ce soit au niveau de la sécurité du transfert de données ou des différentes manières de gérer une connexion HTTP en Java. J'ai de plus implémenté une classe abstraite pour la première fois, je n'en avais jamais eu l'utilité et j'ai pu découvrir, par la nécessité, son intérêt et sa puissance. J'ai réussi à obtenir une implémentation fonctionnelle en moins de trois semaines et j'ai été particulièrement ravi d'obtenir ce résultat.

4.6 Bilan de cette mission

J'ai pris beaucoup de plaisir à développer cette solution, bien que ce soit une goutte d'eau parmi l'immensité du Framework de test de Web Intelligence, entrant en jeu dans l'exécution quotidienne des tests.
J'ai appris beaucoup de choses en très peu de temps et ai eu le plaisir de travailler avec des inconnus qui étaient mes collègues. Ceux-ci se trouvant en Allemagne, nous avons donc communiqué en anglais, ce petit aspect internationnal de la mission m'a beaucoup plu.
Cette mission, bien que très courte, à été très valorisante et enrichissante.

Cette mission étant achevée, le fonctionnement du Framework de test s'est maintenant adapté à la migration de logiciel de traçabilité des problèmes.

Chapitre 5

Plugin Jenkins

5.1 Contexte initial

Jenkins est un logiciel d'intégration continue, il est utilisé à SAP dans le contexte des tests . Son rôle est d'intégrer les différents projets, tels que configurés par l'utilisateur, et de mettre à disposition les résultats obtenus. Il permet d'avoir un retour régulier sur l'état des builds et du résultat de leurs tests.

Un plugin permettant une vision globale sur l'état des builds est déjà en place, Radiator. La figure 5.1 page 50 présente son environnement d'utilisation. Nous pouvons y voir les développeurs et les ST qui alimentent le code hébergé sur Perforce. Jenkins utilise ce code dans ses diverses intégrations. Au centre, le plugin permet un retour visuel rapide des livraisons Perforce.
Ce plugin est aussi conçu pour fonctionner avec un autre plugin, le plugin Claim. Il permet à quelqu'un visitant la page d'un job en échec de le « claimer », inscrivant son identifiant ainsi qu'un message à destination de quiconque visiterait cette même page. Ceci permet de signaler que l'investigation est en cours.

Pour résumer ses caractéristiques principales, c'est un plugin qui permet :

— d'afficher un écran où chaque couleur correspond à un statut de build

— de rassembler les jobs par leurs préfixes communs

— de prendre en compte les claims

L'inconvénient est que le plugin Radiator ne permet pas de rajouter supplémentaire. Voici la liste des différents états gérés par ce plugin :

SUCCESS Le projet compile correctement et tous les tests sont passés

ABORTED La compilation a été arrêtée en cours d'exécution

FAILURE Le code ne compile pas

ERROR Erreur de l'exécution dans Jenkins

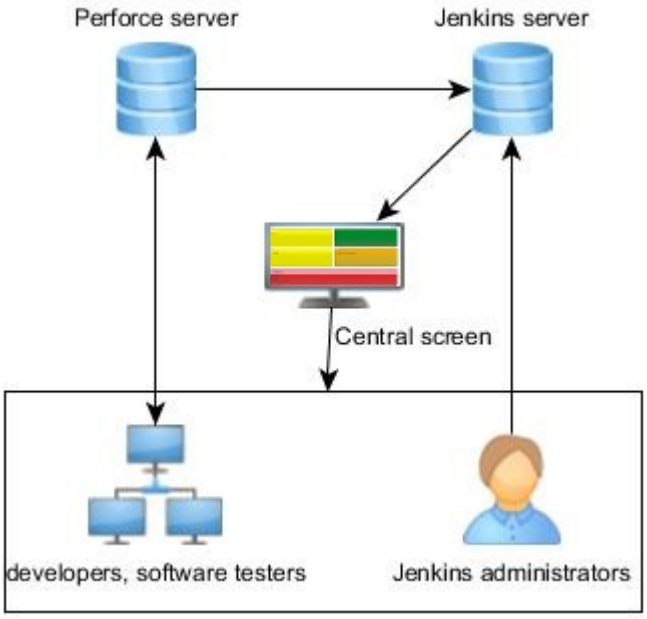


FIGURE 5.1 – L'environnement d'utilisation du plugin

NOT BUILT Le projet n'a pas été compilé

UNSTABLE Des tests JUnit ne sont pas passés mais le code compile

CLAIM Le job en échec a été claim par quelqu'un

Tel qu'il se présente le plugin ne permet pas de donner des informations autres que celles issues de Jenkins ou du plugin Claim. Il ne permet donc pas de donner de renseignements sur le defect associé à ce job. Tenir compte du defect permettrait de faire ressortir les régressions et autres erreurs, séparant ainsi les échecs pris en charge de ceux qui ne le sont pas.

Problématique

L'affichage n'est plus représentatif de la réalité, car le statut vert correspond en réalité deux status :

— la réussite d'un job

— l'échec avec defect d'un job

L'objet de la mission est donc de pouvoir définir un nouveau statut dans l'affichage, pouvoir choisir quand ce statut sera utilisé et pouvoir ajuster l'ordre de leurs priorités. La dernière chose importante est de faire en sorte que l'utilisation du plugin soit générique et pas bornée à notre contexte d'utilisation.

La solution proposée

Suite à la mission précédente, nous avons maintenant accès aux informations relatives

aux defects (celles-ci sont enregistrées dans les archives des builds sur le serveur Jenkins), dans un fichier XML que nous avons nommé test-defects (exemple page 51).
Dans un 1er temps, et en tant qu'exercice, il faudra réaliser un plugin offrant les mêmes possibilités d'affichage des statut des builds que Radiator. C'est-à-dire rassembler les jobs par préfixe et afficher le résultat général de tous ces groupes ainsi créés sur un unique écran. Ajouté aux caractéristiques, ce plugin doit pouvoir prendre en compte les informations du plugin Claim.

Dans un second temps, il faudra pouvoir ajouter un ou plusieurs statut(s) supplémentaire(s) et ré-ajuster leur ordre de priorité. L'ajout de ce nouveau statut doit être générique de sorte qu'un statut quelconque puisse être ajouté.
Le plugin est donc capable de receuillir des informations dans des fichiers sans savoir au préalable, comment seront structurées ces données. Rappelons que, lors de l'exécution des tests, un fichier test-defect est généré contenant la liste[1] des jobs avec defect (exemple page 51).

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<testcases>
  <testcase classname="test.sap.sl.sdk.authoring.IssueWatcherTest" name="
    failingTestNoDefect" status="failed"/>
  <testcase classname="test.sap.sl.sdk.authoring.IssueWatcherTest" name="
    successTestNoDefect" status="passed"/>
  <testcase actualStatus="failed" classname="test.sap.sl.sdk.authoring.
    IssueWatcherTest" message="Error" name="failingTest3_ClassDefect" status="
    passed">
    <defect expectedMessage="Error" link="https://sapjira.wdf.sap.corp/browse/
    BITBIWEBISL2-1542" status="Open" type="jira"/>
  </testcase>
  <testcase actualStatus="failed" classname="test.sap.sl.sdk.authoring.
    IssueWatcherTest" message="Error" name="failingTest2_ClassDefect" status="
    failed">
    <defect expectedMessage="Failing test" link="https://css.wdf.sap.corp/sap/
    bc/bsp/spn/jcwb?crPointer=012006153200001159182015" status="In Process"
    type="cwb"/>
  </testcase>
  <testcase actualStatus="failed" classname="test.sap.sl.sdk.authoring.
    IssueWatcherTest" message="Failing test" name="
    failingTest_CWB_MatchingMessage" status="passed">
    <defect expectedMessage="Failing test" link="https://css.wdf.sap.corp/sap/
    bc/bsp/spn/jcwb?crPointer=012006153200001159182015" status="In Process"
    type="cwb"/>
  </testcase>
  <testcase actualStatus="failed" classname="test.sap.sl.sdk.authoring.
    IssueWatcherTest" message="Failing test" name="failingTest1_ClassDefect"
    status="passed">
    <defect expectedMessage="Failing test" link="https://css.wdf.sap.corp/sap/
    bc/bsp/spn/jcwb?crPointer=012006153200001159182015" status="In Process"
```

1. Au format xml

```
    type="cwb"/>
  </testcase>
  <testcase actualStatus="failed" classname="test.sap.sl.sdk.authoring.
    IssueWatcherTest" message="Failing test" name="
    failingTest_Jira_MatchingMessage" status="passed">
    <defect expectedMessage="Failing test" link="https://sapjira.wdf.sap.corp/
    browse/BITBIWEBISL2-1542" status="Open" type="jira"/>
  </testcase>
  <testcase actualStatus="failed" classname="test.sap.sl.sdk.authoring.
    IssueWatcherTest" message="Error" name="failingTest_CWB_NoMatchingMessage"
    status="failed">
    <defect expectedMessage="Failing test" link="https://css.wdf.sap.corp/sap/
    bc/bsp/spn/jcwb?crPointer=0120061532000011591820 15" status="In Process"
    type="cwb"/>
  </testcase>
</testcases>
```

5.2 Organisation du travail

Tout au long de cette mission mon emploi du temps s'est articulé autour des meeting hebdomadaires, en fin de semaine, avec manager pour faire le point sur l'évolution du projet, présenter le produit lorsqu'il y avait une nouvelle fonctionnalité importante qui avait été implémentée et fixer les objectifs de la semaine qui allait suivre.
D'autre part, au minimum une heure par semaine, je faisait un « code review » avec Fabien, memebre de équipe et aussi commanditaire de ce projet.

5.3 Étude préalable

Les 1er impératifs ont été de maitriser le langage et les outils qui me permettraient de mener ce projet à terme. Je n'avais jamais entendu parler de Jenkins ou d'un quelconque logiciel d'intégration continue, et a forciori sur la manière de l'étendre.

Avant toute étude portant sur ce que j'allais développer, je me suis renseigné sur Jenkins. Je me suis rendu sur sa documentation officiel pour en apprendre plus à ce sujet.

Ensuite, j'ai installé Jenkins et quelques plugins, tels qu'ils sont utilisés à SAP, de manière à me familiariser avec ceux-ci. J'ai examiner le comportement de Jenkins en l'utilisant de la manière la plus simple possible.

Étant un logiciel d'intégration continu, j'ai créé des projets vides (avec implémentation de tests JUnit rudimentaires) avec différents comportements lors de la compilation. Cela m'a permis d'observer le comprtement de Jenkins.

J'ai d'abord commencé par avoir quelques problèmes à mettre en place mon Jenkins mais ja m'en suis sorti rapidement, les problèmes n'étant que des configurations de bases

sur les chemin d'accès utilisés par Jenkins. En l'occurrence le JDK et le répertoire Maven (par souci de rapidité j'ai utilisé maven pour créer mes projets, Jenkins devais donc connaître son emplacement pour avoir accès à mes configurations).

La première chose que j'ai remarqué, et qui a été ma principale incompréhension au début, était que je ne comprenais pas la différence entre les statuts Jenkins et les statuts JUnit. Ce qui portait à confusion était leur vocubulaire, qui était en contradiction. Par exemple le statut failure de Jenkins correspond au statut Unstable de Jenkins, pour complexifier un peu la chose il existe aussi un statut failure dans Jenkins !
J'ai donc fait un comparatif, simplifié, de ces différents

Description du projet	**Status JUnit**	**Statut Jenkins**
projet vide et sans test	NA	Success
projet vide et avec test réussi	Passed	Success
projet vide avec test qui échoue	Failure	Unstable
projet avec erreur de compilation	NA	Error

Passée cette étape d'installations et de configurations, je me suis penché sur la marche à suivre pour implémenter un plugin Jenkins.
La principale difficulté que j'ai eu à développer ce plugin a été de trouver l'information, il est connu sur internet que la documentation de Jenkins, lorsqu'il s'agit de l'étendre, n'est pas claire. Mais la documentation officielle [2] offre un bel éventail de ce qu'il est possible de faire, décrit les différentes technologies constituant Jenkins et offre un tutoriel trivial mais suffisant pour comprendre les bases.

5.4 Le 1er plugin

Je me suis d'abord rendu sur la page du tutoriel officiel de Jenkins [3] afin d'obtenir un plugin fonctionnel avec lequel je pouvais faire mes expériences (il existe un plugin « Hello world » de base, celui à partir duquel j'ai fais une grande partie de mes expérimentations). Ensuite j'ai suivi un autre tutoriel [4] non-officiel mais considérablement plus riche dans lequel j'ai appris une grande partie de ce que je sais aujourd'hui sur le développement d'un plugin Jenkins.

5.4.1 Configuration

Pour faire mon premier plugin je me suis basé sur le archétype proposé par Maven, le fameux « Hello world ». Maven a besoin de configurations supplémentaires dans son

2. https://wiki.jenkins-ci.org/display/JENKINS/Extend+Jenkins
3. https://wiki.jenkins-ci.org/display/JENKINS/Plugin+tutorial
4. https://cleantestcode.wordpress.com/2013/11/03/how-to-write-a-jenkins-plugin-part-1/

settings.xml[5] pour pouvoir récupérer les sources relatives à Jenkins. C'est-à-dire télécharger toutes les dépendances nécessaires à la compilation.
Dans « pluginGroups » il faut ajouter la ligne suivante :

```
<pluginGroup>org.jenkins-ci.tools</pluginGroup>
```

Et dans « profiles » il faut ajouter le profil suivant :

```
<profile>
<id>jenkins</id>
<activation>
<activeByDefault>false</activeByDefault>
</activation>
<repositories>
<repository>
<id>repo.jenkins-ci.org</id>
<url>http://repo.jenkins-ci.org/public/</url>
</repository>
</repositories>
<pluginRepositories>
<pluginRepository>
<id>repo.jenkins-ci.org</id>
<url>http://repo.jenkins-ci.org/public/</url>
</pluginRepository>
</pluginRepositories>
</profile>
```

L'attribut de la balise « activeByDefault » est, ici, à false, parce que c'est ce qui est conseillé sur beaucoup de sites. Mais j'ai eu des problèmes, que j'ai mis un certain temps à résoudre, à cause de cet attribut.
Grâce à Maven il suffit de créer un répertoire et de se placer dedans (en ligne de commande), et d'exécuter la commande ci-dessous :

```
mvn hpi:create
```

À partir de là, le projet est créé et j'ai pu commencer à travailler dessus.

5.4.2 Génération du squelette

À l'exécution de cette commande, il est demandé de renseigner le groupId et l'artifactId, le groupId peut être org.jenkins-ci.plugins et l'artifactId est le nom du plugin. Cette commande crée l'arborescence du projet ainsi que les fichiers de base.

À la génération du squelette du plugin nous obtenons un plugin qui compile, dont l'architecture est présentée figure 5.2 page 55.

Plusieurs fichiers sont générés et il est très important de connaître leur utilité car il est impossible de les déplacer ou de les renommer, c'est différents fichiers sont imposés

5. Situé dans le répertoire ~/.m2

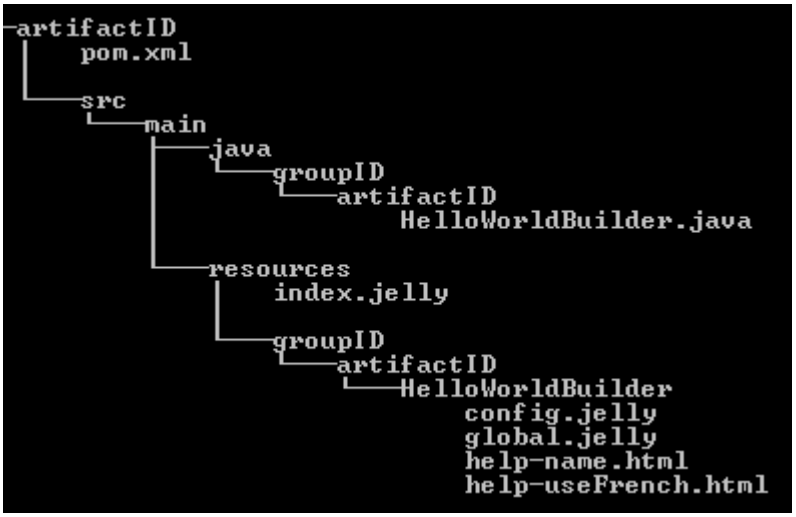


FIGURE 5.2 – Architecture d'un plugin Jenkins généré par maven

par Jenkins. J'ai lis du temps à le comprendre et à chercher pourquoi mon code ne fonctionnait pas, et s'il existe un moyen de passer outre cette norme, je ne le connaîs pas encore :

Nom du fichier	Description
index.jelly	Description utilisée à la création d'une nouvelle instance du plugin
config.jelly	Correspond à la page d'une instance du plugin
global.jelly	Correspond à la page de configuration générale du plugin
HelloWorldBuilder.java	Implémentation d'exemple du plugin
help-name.html	Fichier d'aide à la configuration lorsque clic sur le symbole d'aide
help-useFrench.html	Fichier d'aide à la configuration lorsque clic sur le symbole d'aide

5.4.3 Déploiement du squelette du plugin

J'ai eu quelque difficulté à comprendre la logique d'intégration du plugin à Jenkins. Mais J'ai trouvé deux méthodes qui fonctionnent plutôt bien. Plutôt car j'ai rencontré certains problèmes avec ces deux procédés. À certains moments, alors je n'avais pas modifier le code, je ne pouvais plus appliquer l'une ou l'autre de ces méthodes car la compilation ne fonctionnait pas.
Ces deux méthodes sont les suivantes, mais je ne les aies pas appliquées très longtemps, car j'ai trouvé une bien meilleure solution pour exécuter mon plugin.

Installation « à la main »

En ligne de commande et depuis la racine du projet[6], il faut exécuter la commande suivante :

```
mvn clean install -Pjenkins
```

Celle-ci va télécharger les sources nécessaires, les compiler, exécuter les tests et, si tout se passe bien, générer le fichier .hpi dans le dossier target.
Ceci fait, il faut se rendre dans Jenkins (par exemple http ://localhost :8080/), s'identifier et aller dans les paramètres avancés de gestion de plugins. Ici, il faut renseigner le chemin d'accès vers le .hpi (extension reconnue par Jenkins comme étant un plugin) et terminer.

Commande du plugin Jenkins de maven
En ligne de commande et depuis la racine du projet, il faut exécuter la commande suivante :

```
mvn hpi:run -Djetty.port=8090 -Pjenkins
```

Cette commande va déployer une instances de Jenkins propre à maven, ce qui nous permet d'exécuter le plugin et le debugger. On peut très bien omettre de préciser le port utilisé si celui par défaut[7] est disponible, mais il vaut mieux éviter. Une fois que le déploiement est terminé, maven signale que « Jenkins is fully up and running » nous pouvons aller voir Jenkins à l'url *http ://localhost :8080/jenkins* qui devrait ressembler à la figure 5.3 page 57.

5.5 Implémentation de la base du nouveau plugin

Jenkins possède beaucoup de points d'extensions et, à cette étape, il faut déterminer lequel, ainsi que la classe à étendre. Ce choix m'a posé beaucoup de problème au début et je n'ai jamais su comment répondre à cette question. Je me suis plutôt inspiré du plugin Radiator car celui-ci fonctionnait déjà !

Mais plus tard, quand j'ai commencé à être plus à l'aise avec ce concept de point d'extension j'ai pu déterminer une stratégie pour savoir lequel utiliser. Mais pour cela il faut déjà connaître Jenkins ainsi que son code. Par exemple, dans mon cas où je souhaite afficher la liste des jobs, en parcourant la liste des points d'extensions disponibles, nous pouvons trouver *View*, dont l'une de ses implémentations est *ListView* ; sa description étant « Displays Jobs in a flat list view ». En observant la javadoc, on peut trouver la méthode nous donnant accès aux projetx associés à la vue (figure **??** page **??**).

6. à l'emplacement du POM.xml
7. port 8080

FIGURE 5.3 – Jenkins à l'exécution de la commande maven qui l'encapsule

5.5.1 Import du code dans l'IDE

Au début de mon travail sur l'implémentation du plugin, j'ai rencontré beaucoup de problèmes pour compiler et tester mon plugin. Il m'était d'ailleurs complètement impossible de le debugger, je perdais beaucoup de temps avec toutes les manipulations que j'étais obligé de faire juste pour tester une ligne de code.
Mon premier IDE pour développer ce plugin était Eclipse. À force de perdre autant de temps et de me retrouver confronter à des problèmes n'ayant rien à voir avec l'implémentation du plugin, j'ai fini par chercher une solution avec un membre de mon équipe. Alors nous sommes allé voir un membre d'une autre équipe qui avais déjà implémenter un plugin Jenkins pour en savoir plus sur la méthode qu'il avait adoptée. Au sortir de ce meeting j'ai aussitôt installé Netbeans, lui aussi avais rencontré les mêmes problèmes que moi et avais aussi mis un certain temps à trouver la solution.

L'utilisation de Netbeans (et de son plugin Jenkins) facilite l'implémentation du code et, surtout, m'a permis de me focaliser sur les problèmes liés au plugin plutôt qu'à ceux liés aux outils de développement.

5.5.2 Implémentation de la base du plugin

Avant de pouvoir se concentrer sur l'implementation des fonctionnalités du plugin, il faut encore savoir où écrire le code et comment nommer ses fichier. Car comme j'ne ai déjà parlé précédemment, Jenkins respecte une architecture stricte pour aller chercher les différents codes que composent ses plugin. À l'heure actuelle, je n'ai pas encore compris toutes les subtilités de cette architecture.
Lors de mes première expériences, je me basais sur des plugin déjà existant pour voir comment les développeurs avaient nommés leurs différents fichiers. Mais plus tard, à force de fouiller dans le code de Jenkins, j'ai découvert une méthode bien plus efficace pour répondre à cette question.
Pour savoir comment doit se nommer le fichier de configuration de Listview, par exemple, je suis aller chercher l'emplacement de son fichier de configuration. L'arborescence de ce fichier, dans le code de Jenkins, est illustrée figure ?? page ??. Il convient maintenant de copier ce fichier et de le coller dans le répertoire *ressources* de notre projet. Et voilà, j'ai dans mon plugin un fichier de configuration qui sera reconnu par Jenkins et contenant le code par défaut.

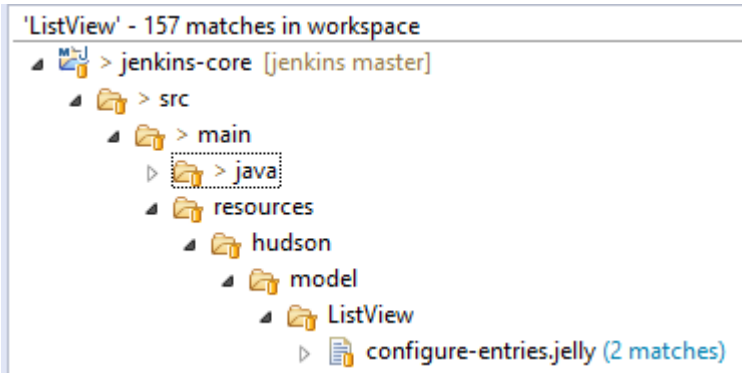


FIGURE 5.4 – Chemin d'accès au fichier de configuration par défaut

5.6 Travail réalisé

À partir du moment où j'ai compris la logique d'implémentation d'un plugin Jenkins, j'ai enfin pu me concentrer sur ma mission.
Je me suis fixé comme premier objectif d'afficher une nouvelle vue que j'aurais implémentée moi-même : typiquement un écran blanc n'affichant que chaine de caractère issue de mon implémentation Java, cet objectif simple permet de faire le lien entre l'interface graphique, codée en jelly, et le code Java.
J'ai trouvé dans la documentation de Jenkins les quelques éléments qui allais me permettre de réaliser ceci : les deux objets « it » et « from », ces deux objets me permettent d'accéder à la classe principale de mon plugin. Il permettent d'accéder aux méthodes et

propriétés de l'instance du plugin.
Les appels ne se font pas de la même manière mais se basent toujours sur ces deux objets.

L'accès aux attributs se fait en l'appelant simplement de la manière suivante :

```
<j:set var="maVariable" value="${from.maPropiete}">
<j:set var="maVariable2" value="${from.maPropiete2}">
```

Cela permet de d'initialiser une variable jelly et de lui donner la valeur de la proprité souhaitée. L'accès aux méthode se fait un peu différemment :

```
%<j:invoke var="maVariable" on="${from}" method="nomDUneMehode">
```

Et de la même manière, cette exemple permet d'appeler une méthode depuis l'objet from dont le retour est placé dans la variable jelly.

La logique d'implémentation du jelly est assez simple en soit et je me la suit rapidement appropriée. J'ai trouvé très agréable d'utiliser aussi facilement mon Java depuis mon jelly, car celui-ci est implémenté côte-à-côte avec le code html.

Grâce à ce code très simple, j'ai pu afficher ce que je voulais dans mon jelly en utilisant la notation jelly : ${maVariableJelly}

5.6.1 Construction d'une architecture en corrélation avec l'affichage souhaité

Pour accéder aux différents jobs de Jenkins, il suffit de faire appel à l'une des méthodes de la classe mère : getItems(), retournant une liste de TopLevelItem. Toutes ces informations sont disponibles dans la javadoc du produit que j'ai longuement parcourue pour comprendre son fonctionnement. Mais malheureusement cet objet ne correspond pas vraiment à l'utilisation que je voulais en faire. J'ai donc créé une nouvelle classe : ProjectWrapper, comprenant cet objet ainsi que d'autres, plus spécifiques à mon plugin.

La méthode Java que j'ai donc implémenté est la suivante :

```
public List<ProjectWrapper> getSelectedJobs() {
        List<ProjectWrapper> jobs = new ArrayList<ProjectWrapper>();
        ProjectWrapper project;
        for ( TopLevelItem item : super.getItems() ) {
            if (item instanceof AbstractProject) {
              project = new ProjectWrapper(this, (AbstractProject) item);
              if (!project.getAbstractProject().isDisabled()) {
                 jobs.add(project);
            }
        }
```

```
11      }
        return jobs;
13  }
```

Grâce à cette liste ainsi obtenue je pouvais avoir accès à la liste de mes jobs depuis mon interface graphique.
La plus grande difficulté que j'ai rencontré pour pouvoir afficher les jobs était de les afficher tels que je le voulais, autrement regroupés par préfixes qu'ils avaient en commun.

J'ai donc passé beaucoup de temps à construire une architecture qui me permttrait de réaliser cela, et je suis donc arrivé à créer un certain nombre de classes me permettant ceci.
Je développe en Java, et pour respecter au mieux la logique de la programmation orientée objet, j'ai fais en sorte que mes objets Java représentent vraiment les différents éléments de mon inerface graphique.
L'objet central de la logique que j'ai implémenté est le dashboard, celui-ci contenant la liste des différentes vues qui le compose, je l'ai nommé Dashboard !
Ensuite, chacune des vues intégrée à mon dashboard sont caractérisées par un préfixe (point commun à tous les projets qu'elle contient), un état dominant (état qui fera que la vue s'affiche de telle ou telle couleur) et d'une liste de projets. J'ai nommé cet objet ViewEntry.
Et pou finir, chacun des projets contenus dans une vue est caractérisé par un préfixe et un résultat personnalisé (personnalisé car un projet ne peut avoir qu'un seul état mais j'ai trois sources différentes pouvant influencer cet état : Jenkins, le plugin Claim ou un état choisi par l'utilisateur). Je l'ai nommé ProjectWrapper.
Chacun de ces objets possède un ensemble de comportement qui leur est propre. Par exemple, dans le cas du Dashboard, trier une liste de projets en une liste de vues (où tous les projets d'un vue on le même préfixe). Dans le cas de la ViewEntry, pouvoir ajouter un projet a sa liste de projets. Et dans le cas du ProjectWrapper, pouvoir fournir son préfixe s'il en a un, ou son statut.

Le diagramme de classe représentant cette architecture que j'ai ainsi construite est illustré page 61, et j'ai représenté l'algorithme me permettant d'obtenir mon Dashboard à partir d'une liste de ProjectWrapper page 62

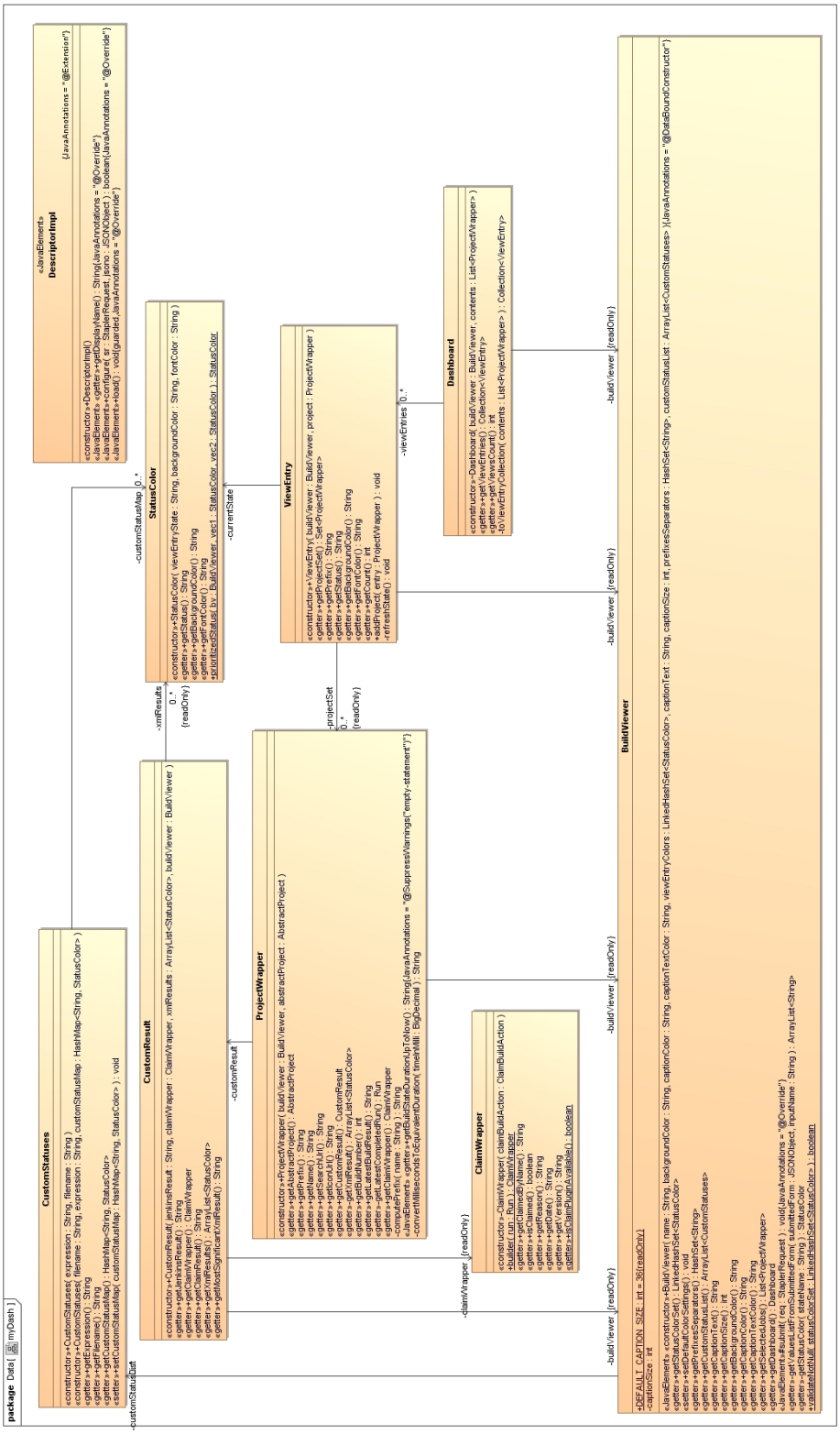


FIGURE 5.5 – Diagramme de classe du plugin réalisé

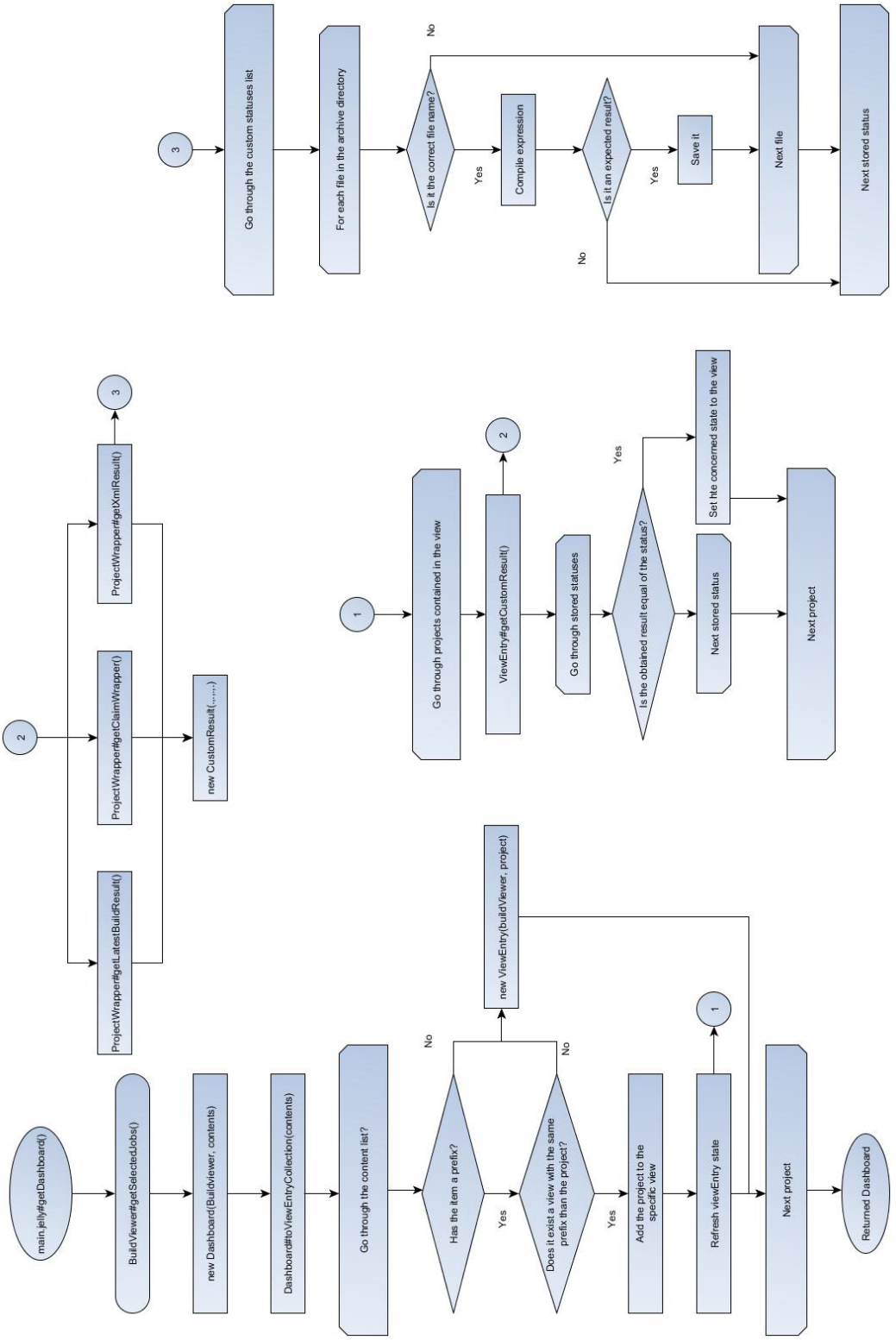


FIGURE 5.6 – Algorithme de création du Dashboard

5.6.2 Affichage des jobs dans l'interface graphique

Les objets Java représentatifs de mes vues étant implémentés, l'objectif était maintenant de pouvoir les afficher correctement. Il me fallait donc dimensionner mes vues en fonction, d'une part du nombre de vues, et d'autre part, de la taille de l'écran.
J'ai d'abord voulu essayer de dimensionner les vues directement à l'affichage, dans l'interface graphique. Cela fonctionnait très bien, puisqu'à partir du Dashboard je pouvais récupérer les nombre de vues générées et ainsi les dimensionner en fonctions de la taille de l'écran. L'inconvénient de cette méthode était que la taille de l'écran n'étant pas accessible depuis le jelly j'étais obligé d'attribuer moi-même la valeur de la taille de l'écran. Cela ne m'a pas posé de problème au début mais lorsque j'ai essayé le plugin sur d'autre écrans, plus petits ou plus grands, le dimensionnement ne pouvait plus fonctionné.
J'ai donc opté pour la solution javascript. Je ne connaissais que très peu ce langage avant cet épisode et j'ai dû passer un peu de temps à l'apprendre. Ce qui a été plutôt rapide car je ne me contentais que de mettre en forme mes éléments html en modifiant mes feuilles de styles.

La solution la plus simple pour déterminer la largeur et la hauteur d'une vue a été, d'abord, de considérer que celles-ci seraient toutes de la même taille, et ensuite de baser ce calcul sur les dimensions du dashboard (celui-ci étant configuré de manière à occuper 100% de l'espace disponible).
Concrètement, mon algorithme de dimensionnement est le suivant :

- Je calcule le carré minimum pouvant acceuillir toutes mes vues (4 vues impliquent un carré de 2, 6 vues impliquent un carré de 3, ...)
- J'en déduis mon nombre de lignes en divisant le total des vues par le carré trouvé (6 \3 = 2 implique 2 lignes)
- Si le résultat de la division n'est pas entier je rajoute une ligne (pour 7 vues j'aurais 2 lignes de 3 vues et 1 autre de 1)
- De ces valeurs j'en déduis les dimensions de mes vues

J'ai appliqué quelques coefficients tenant compte du fait que les marges entre les vues sont constantes.
Le code javascript ainsi développé est ci-dessous :

```
%breadcrumbarheight = $('div#breadcrumbBar').css("height").split("px")[0];
%h=h-breadcrumbarheight-15;
%var countOfViews = document.getElementById("hiddenCountOfViews").value;
%var captionSize = document.getElementById("hiddenCaptionSize");
        %
%if(captionSize==null){
  %captionSize=0;
%}else{
  %captionSize=captionSize.value;
%}
%xdash = 0;
```

```
%while (countOfViews &gt; xdash * xdash) {
  %xdash++;
%}
%var countOfRows = Math.floor(countOfViews / xdash);
        %
%if((countOfViews % xdash)!=0){
  %countOfRows = countOfRows +1;
%}
%var viewheight = ((h - captionSize) / countOfRows) - 4*countOfRows;
%var viewwidth = (w / xdash) - 5*xdash;
```

Ce code fonctionnait très bien et j'ai donc voulu essayer mon plugin dans un contexte d'utilisation réelle, c'est-à-dire en intégrant mon plugin à Jenkins en utilisant le fichier d'installation .hpi.
J'ai eu la surprise de constater que, malgré que le dimensionnement se passait bien, tout le contenu de mon dashboard était décalé vers la droite d'une certaine distance qui restait la même quelle que soit la dimension de l'écran que j'utilisais.
J'ai passé beaucoup de temps à chercher ce qu'il se passait, sans comprendre d'où cela pouvait venir. J'ai cherché dans le plugin Radiator s'il prenait en compte autre chose que j'avais ignoré, mais sans résultat.
Je n'ai pas investigué longtemps dans le code car je ne pouvais rien trouver de cette manière. J'ai donc lancé les deux versions de Jenkins que j'avais, l'une en exécutant le plugin depuis Jenkins (où tout se comportait comme prévu) et l'autre (où le problème avait lieu).
De ces deux versions différentes j'ai analysé les deux codes sources des pages et ai écrit un petit programme java me permettant de faire ressortir les différences. Après un certain temps d'analyse j'ai remarqué qu'il y avait une différence entre les deux versions. L'un des éléments html, commun aux deux versions, ne possédait pas les mêmes classes. Je suis donc allé dans le code de Jenkins pour voir a quoi correspondait ce code, il s'agissait en fait d'une portion de l'écran (précisément le « side-panel ») qui était réservé au menu de Jenkins, et si le plugin Radiator ne rencontrait pas ce problème c'est que celui-ci avait été développé sur une version antérieure dans laquelle le positionnement ne se faisait pas de la même manière [8].
Étant au courant de la raison exacte du problème que j'ai rencontré j'ai pu le corriger en quelques lignes de JQuery ci-dessous :

```
$("side-panel").removeClass("col-sm-9");
$("side-panel").removeClass("col-md-7");
$("side-panel").removeClass("col-lg-6");
$("side-panel").removeClass("col-xlg-4");
```

J'ai de nouveau testé mon plugin tel qu'il le serait réellement, tout fonctionnait comme je le désirais.

8. voir `https://issues.jenkins-ci.org/browse/JENKINS-29659`

Ce problème a perduré longtemps parce que, ne sachant pas comment le résoudre, je l'ai laissé de côté en attendant d'avoir une meilleure idée. Avec le temps, accumulant de l'expérience tant avec les outils que j'utilisais qu'avec les langages que j'utilisais, j'ai eu cette idée qui m'a permis de résoudre ce problème en moins de deux heures.

5.6.3 Personnalisation des statuts

J'avais donc à cette étape là un plugin me permettant d'afficher tous les jobs regroupés par préfixes dans des vues de la couleur représentative du statut jugé prioritaire par l'utilisateur. À ce niveau d'avancement de mon projet, le plugin possède déjà toutes les fonctionnalités de l'ancien plugin Radiator. À la seule différence que mon plugin permet de configurer l'ordre des priorités.
J'en arrivais maintenant à la fonctionnalité principale de mon plugin, raison d'être de mon projet : pouvoir configurer ses propres statuts se basant sur des fichiers xml.
La première question était de savoir comment récupérer ces fichiers xml.
La deuxième question, comment en extraire les résultats.
La troisième question, comment faire correspondre un résultat à un statut.

Lors de l'exécution des tests tous les fichiers générés sont enregistrés dans un répertoire d'archive (y compris les fichiers des résultats JUnit) je devais donc savoir comment y accéder. Pour répondre à cette question j'ai simplement récupérer toutes les informations disponibles dans un projet Jenkins jusqu'à tomber sur le répertoire du projet, duquel je suis parti pour me trouver dans le répertoire « archive », le code me permettant d'avoir un objet File représentatif de ce dossier est le suivant :

```
File files = new File(abstractProject.getBuildDir().getAbsolutePath() + "\\" +
    run.getId() + "\\archive");
```

Je peux maintenant tester l'existence d'un fichier (dont le nom est renseigné par l'utilisateur). Je devais pouvoir extraire une information de fichier, la méthode la plus simple, en considérant que j'ai affaire à un fichier xml, est d'utiliser un XPath pour obtenir mon information.
Je n'avais jamais utilisé cette technologie, bien que j'en connaissais son existence. Après quelques expérimentations j'ai obtenu un code fonctionnant bien me permettant, pour l'instant, de récupérer la valeur d'un attribut, que je pouvais retourner.

La vraie difficulté de cette partie à été d'implémenter une table dont les éléments sont « drag'n'droppable ». J'ai essayé un certain nombre de technologies, certaines correspondant tout à fait et étaient très faciles à utiliser mais dont le simple import de la libraire rendait le plugin inutilisable.
J'ai donc choisi de ne pas utiliser de librairie pour implémenter mon « drag'n'drop » et ai tout fait moi-même. Le « drag'n'drop », uniquement des lignes d'un tableau étant plutôt facile à implémenter, j'ai pris le temps de le faire pour en retirer un résultat très

satisfaisant.
Une fois la table implémentée il fallait, d'une part, pouvoir ajouter et supprimer des lignes, et d'autre part, que la modification de la table des statuts ait une influence sur la table des statuts personnalisés. Ceci à été assez difficile car l'ajout d'une ligne dans la table est exécuté par un code javascript contenant le code à ajouter. Ce qui nécessitait de modifier le code à deux endroits différents à la moindre modification de la table. Ensuite, les éléments à ajouter dans la ligne à ajouter, comme son identifiant, sont difficilement récupérables du simple fait que la génération ne se passe pas au même moment. Quand une table est générée, le jelly s'occupe d'incrémenter un variable pour identifier les lignes, mais quand je veux ajouter une ligne, non seulement, cette variable n'existe plus, mais aussi, je me trouve alors dans un code javascript.
J'ai donc passé beaucoup de temps à mettre en relation tous ces éléments, de manière à ce que l'interface graphique que l'utilisateur a sous les yeux soit complètement dynamique.

5.7 Les échos de mon projet

Lorsque je suis arrivé à un résultat satisfaisant il devenait nécessaire de présenter mon travail aux autres membres de mon équipe. J'ai donc préparer un PowerPoint pour étayer mes propos.
Ma première présentation, devant l'équipe ST Automation, s'est déroulée le jeudi 13 août 2015.
Mon auditoire n'a pas vraiment compris les détails de ce que je disais car je suis allé trop vite au début de la présentation et je n'ai pas pris le temps de clairement définir le domaine d'utilisation de mon plugin. De plus, une majorité des personnes présentes n'utilisait pas le vocabulaire que j'utilisais. L'autre point important, favorisant une certaine incompréhension de mon auditoire, était que je n'avais pas détaillé le contexte actuel, c'est-à-dire d'une part, l'utilisation du plugin Radiator, et d'autre part, la mécanique que j'ai mise en place précédemment pour obtenir un statut JCWB me permettant de le réutiliser. Pour terminer cette présentation, je n'ai pas fait de simulation de mon produit pour montrer à quoi il ressemble, cela aurait permis à mon auditoire de visualiser mon travail.
J'ai fait un seconde présentation, devant l'équipe SDK cette fois-ci. J'avais pris le temps de faire de nouveau slides me permettant de m'attarder un peu plus sur le contexte antérieur à mon projet et à la problématique, rencontrée. J'ai aussi fait la démonstration du plugin précédent et de mon plugin pour marquer les différences. Mon auditoire à beaucoup mieux compris le contexte de mon plugin. Le point que je devais encore améliorer était de bien séparer les différentes parties : avant, quel problème ; aujourd'hui, quelle solution ; dans le futur, toutes les fonctionnalités désirées.
Une autre présentation est prévue mais celle-ci n'est pas encore planifier à l'heure où j'écris ce rapport.
Je ferai beaucoup plus attention à présenter le contexte et je présenterai la problématique avec beaucoup plus soin de manière à ce que mon auditoire puisse comprendre tout le contexte dans lequel s'inscrit l'utilisation de mon plugin, et ainsi, son intérêt.

Bilan

Les différentes missions qu'il m'a été donné d'effectuer m'ont d'abord permis de participer activement à l'activité de l'entreprise et de me retrouver dans la peau d'un ingénieur qualité, me posant des questions nécessitant une vrai compréhension du produit et du recul par rapport à mes objectifs.

Lorsque mes collègues n'avaient pas le temps de s'occuper de ce qu'ils voulaient, je me suis vu affecter leurs tâches. Me pencher sur le genre de travaux que l'on donne aux ingénieurs a été pour moi une marque de reconnaissance. Mais paradoxalement cela a aussi été à l'origine de remises en question « En serai-je capable ? », « Ai-je les connaissances requises ? », « Vais-je réussir dans le temps imparti ? ».

Le développement d'un plugin Jenkins a été une vraie étapes dans ma carrière de d'ingénieur, un pied de posé dans la réalité du développement logiciel mais surtout du logiciel libre. Signifiant que j'ai participé activement à l'évolution de ce logiciel open-source et communautaire. J'ai communiqué avec le concepteur de l'ancien plugin, que le mien remplace, et c'est avec plaisir qu'il m'a apporté les réponses à mes questions. Mon plugin n'est, pour l'heure, pas encore sur le repository officiel mais cela est en discussion. Cette éventualité permettrai à n'importe quel utilisateur de Jenkins d'essayer mon plugin et de l'adopter, ou de le rejeter. Bien évidemment, le rendre disponible publiquement le mettrais à l'épreuve d'utilisation pour lesquelles il n'avait pas été prévu initialement. Il est donc certain que des bugs seront trouvés, autrement dit, le livrer sur le repository officiel nécessite de le maintenir et de rester disponible pour la maintenance, l'évolution et pour répondre aux questions des utilisateurs.

Techniquement, j'ai eu la chance de pouvoir me pencher sur plusieurs sujets différents, tenant de domaines techniques différents et à des fins différentes. En développant les tests j'ai implémenté du code Java, non pour apprendre une technologie, mais pour réussir à faire quelque chose, me focalisant ainsi sur l'objectif et les performances plutôt que sur la manière.

De tout ce que j'ai pu faire lors de ce contrat, seule une petite partie de ce que j'ai appris à l'école m'a été nécessaire, bien qu'essentielle. Je me suis vu aborder des problématiques et des technologies dont j'avais, tout au plus, vaguement entendu parlé. J'ai donc adopté une vrai méthode de travail où ce qui compte n'est pas la maitrise de telle ou telle technologie, comme en école d'ingénieur, mais de proposer une solution fiable et performante, quelle que soit le moyen utilisé. Ce faisant je me suis posé les vraies questions qui concernent véritablement un ingénieur : « Quels sont les différents moyens de résoudre ce problème ? », « Quels sont les différents avantages et inconvénients ? », « Quelles sont les normes, les nécessités, les limitions juridiques ? », « Quelles sont les personnes compétentes qui pourrons me donner des informations primordiales pour mon projet ? ».

D'un point de vue personnel, ce qui a été un frein au début de mon contrat de professionnalisation fût, d'une part, mon acharnement à vouloir tout comprendre dans les détails sans que cela soit nécessaire, et d'autre part, à ne pas me tourner vers les personnes compétentes qui auraient pu me conseiller, réduisant ainsi des périodes d'études de plusieurs jours à quelques heures. Je regrette de ne pas avoir été suffisamment demandeur d'informations, désirant faire le mieux possible, le plus vite et sans soutien ce que l'on attendait de moi, ainsi, je n'ai pas appris tout ce que j'aurai pu.

Si c'était à refaire je ne me jetterai pas tout suite dans l'élaboration de ma stratégie pour arriver au bout de ma mission, mais passerai beaucoup plus de temps à me diriger vers les autres pour discuter des contours de mon projets, des problèmes potentiels et des solutions déjà existentes ; chose que je faisait beauxoup plus facilement plus tard, mon contrat arrivant à son terme.

Le temps que j'ai passé dans cette entreprise est une immense fierté dont je n'hésiterai pas à vendre les mérites. Parce que c'est dans un contexte internationnal que j'ai participé au développement du framework de test, et parce que c'est pour le leader de la BI que j'ai développé un outil open-source, utilisé quotidiennement, de reporting des builds. Ce contrat de professionnalisation à été une expérience exceptionnellement riche, tant par ce que j'ai appris à faire ou à être, que par les nombreuses connaissances techniques que j'ai pu accumuler.

Conclusion

Je suis fier d'avoir pu laisser ma trace dans l'entreprise. Premièrement parceque les tests que j'ai développé continueront à être exécutés tant que le produit existera, autrement dit pour encore environ un dizaine d'années. Ensuite parceque le plugin que j'ai eu la chance de développer contribue, d'une part, au rayonnement de SAP sur le monde libre et communautaire du logiciel open source, et d'autre part, au fonctionnement interne de l'équipe de test SDK, de tel sorte que quotidiennement il sera quelque part sur un des écrans pour pouvoir visualiser l'état des builds en cours.

J'ai été au cours de mon contrat particulièrement motivé, investi et concerné par les missions qu'il m'a été donné de faire. Et quand on manager a eu échos d'une opportunité de CDI dans une autre équipe et m'en a tout de suite fait part, malheureusement, cette place devant être occupée avant le fin de l'année et mon séjour à l'étranger n'étant pas été validée j'ai été obligé, à mon plus grand regret, de décliner l'offre. Plus récemment et sur demande de ma part, mon manager a envoyé une requête à l'un de ses collègue de Bangalore. Il me rapellera probablement, s'il est intéressé, dans les prochains jours.

Mon contrat professionnel à SAP fût une expérience remarquable tant par le cadre dans lequel j'ai été intégré que par les missions sur lesquelles j'ai été affecté. Cela m'a ouvert à l'environnement complexe d'une multinationnale et à l'ampleur de projets logiciels tels que WebI. De plus, Web Intelligence est incorporée à une suite de logiciel très répandu, ayant été amené à tester les fonctionnalités du produit j'ai donc été amené à l'essayer et le comprendre. Aujourd'hui je suis capable de créer des rapports simples et à les modifier, ce qui est une compétence très demandé dans le monde de la Business Inteligence.

Annexe A

Résultats quotidiens des tests

From: JUnit@pgdev.sap.corp
Sent: lundi 17 août 2015 18:20
To: DOLIMONT, Christophe; TAQUET, Pierre; COULIBALY, Nana
Subject: JUnit 99.29% Build='aurora_dev_webi_dsl_331 Test='rebean_wi.customers.TestSuite_Customers Computer= 10.208.43.203

TEST = rebean_wi.customers.TestSuite_Customers
SUITE = Aurora 4.2
SERVERNAME = localhost:6400
OS name = Windows
OS archi = amd64
PHASE = D2P win64_x64
BUILDNAME = 14.2.0.331.businessobjects64_aurora_dev_webi_dsl
VAPP NAME = ASTEC_AURORA42_DEV_WIN_f0c68b77-27f4-475a-843e-35896aadfec4

Unit Test Results

Designed for use with JUnit and Ant.

Summary

Tests	Failures	Errors	Success rate	Time
140	1	0	99.29%	1 h 31 min 50 s

Note: *failures* are anticipated and checked for with assertions while *errors* are unanticipated.

Packages

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

Name	Tests	Errors	Failures	Time	Time Stamp	Host
rebean_wi.customers	140	0	1	1 h 31 min 50 s	2015-08-17T14:47:39	server41

Package rebean_wi.customers

Name	Tests	Errors	Failures	Time	Time Stamp	Host
TestSuite_Customers	140	0	1	1 h 31 min 50 s	2015-08-17T14:47:39	server41

TestCase TestSuite_Customers

Name	Status	Type	Time
rebean_wi.reporting_rendering.validation.Init_TC_itbi init_PrepaForITBI	Success		17 s
rebean_wi.customers.ADP_GSI_France.ID_225480.ID_225480_TestCase_1 CM_1261231_2014_CellHeightUpdate	Success		30 s
rebean_wi.customers.Aerospatiale_Matra_Airbus.ID_187739.CM_627746_2014 CM_627746_2014_Computation	Success		3 s
rebean_wi.customers.Aerospatiale_Matra_Airbus.ID_187739.CM_731851_2014 CM_731851_2014_QuerySummary	Success		2 s
rebean_wi.customers.Amdocs_Development_LTD.ID_871420.CM_654663_2014 CM_654663_2014_Unavailable	Success		2 s
rebean_wi.customers.American_Management_Systems.ID_977175.ID_977175_TestCase_1 CSSID_0020079747_0000035772_2014	Success		3 s
rebean_wi.customers.American_Management_Systems.ID_977175.ID_977175_TestCase_2 CM_1330059_2014_PDF_Truncated	Success		3 min 50 s
rebean_wi.customers.AnnTaylor_Inc.ID_570535.ADAPT01716536_Break_FormatText ADAPT01716536_Excel_Break_FormatText	Success		3 s
rebean_wi.customers.AOK_Baden_Wurttemberg.ID_101715.CM_254828_2015 CM_254828_2015_SplitColumsPageMode	Success		
rebean_wi.customers.Apple.ID_30934.ADAPT01717053_Multivalue_CrossTable ADAPT01717053	Success		3 s
rebean_wi.customers.Apple.ID_30934.CM_661468_2014 CM_661468_2014_Subtotals	Success		3 s
rebean_wi.customers.Apple.ID_30934.CM_48299_2015 CM_48299_2015_RelativePositionMismatch	Success		5 s
rebean_wi.customers.Aquitanis_Opac_De_La_Communaute.ID_944713.CM_1240269_2014_TP CM_1240269_2014_FormatNumber	Success		11 s
rebean_wi.customers.ARZ_Allgemeines_Rechenzentrum.ID_28259.ID_28259_TestCase_1 CM_1314984_2014_Export_BIG_Excel	Success		10 min 17 s
rebean_wi.customers.Asahi_Glass_Co.ID_145381.CM_1452299_2014_BreakError CM_1452299_2014_ScanBreakError	Success		2 s
rebean_wi.customers.Atlas_Copco_Business_Services.ID_642864.ADAPT01718277_Input_Controls_Lost ADAPT01718277_Input_Control_Lost	Success		26 s
rebean_wi.customers.Audi_AG.ID_12520.ADAPT01699626_PDF_Truncated ADAPT01699626_PDF_Truncated_Test	Success		6 s
rebean_wi.customers.Audi_AG.ID_12520.ADAPT01706987_ChartLegacyDefaultSettings ADAPT01706987	Success		4 s
rebean_wi.customers.Audi_AG.ID_12520.AUDI_1386567_2014_InputControlsUpgrade CM_1386567_2014_InputControlsUpgrade	Success		3 s
rebean_wi.customers.Australian_Health_And_Nutrition.ID_336882.CM_1562604_2014 CM_1562604_2014_SortIssue	Success		2 s
rebean_wi.customers.Axa_Group_Solutions.ID_890374.CM_3413_2015_TP CM_3413_2015_CustomSortIssue	Success		4 s
rebean_wi.customers.Bank_for_International_Settlements.ID_25735.CM_1511454_2014_TP CM_1511454_2014_CellAlignment	Success		3 s
rebean_wi.customers.Bank_Julius_Bar.ID_133577.CM_1542477_2014 CM_1542477_2014_AutoFitHTML	Success		2 s
rebean_wi.customers.Bank_of_America.ID_1166346.CM_704895_2014 CM_704895_2014_Formatting	Success		3 s
rebean_wi.customers.Cenovus_Energy_Inc.ID_1145846.CM_883660_2014_Config CM_883660_2014_ConfigVisuFile	Success		
rebean_wi.customers.Cenovus_Energy_Inc.ID_1145846.CM_883660_2014_Tomcat CM_883660_2014_TomcatRestart	Success		5 s
rebean_wi.customers.Cenovus_Energy_Inc.ID_1145846.CM_883660_2014_Config CM_883660_2014_ConfigVisuFile	Success		
rebean_wi.customers.Cenovus_Energy_Inc.ID_1145846.CM_883660_2014_Tomcat CM_883660_2014_TomcatRestart	Success		5 s
rebean_wi.customers.ConocoPhillips_Company.ID_161451.CM_110784_2015 CM_110784_2015_TP	Success		2 s
rebean_wi.customers.Biogen_IDEC_Inc.ID_302254.CM_677911_2014_TP CM_677911_2014_ERR_WIJ_20003	Success		3 s
rebean_wi.customers.Bouygues_Construction.ID_518883.CM_748730_2014_TP CM_748730_2014_CantCreateKeyFigure	Success		2 s
rebean_wi.customers.Bouygues_Construction.ID_518883.CM_124508_2015 CM_124508_2015_WrongBreakFooterValues	Success		2 s
rebean_wi.customers.British_American_Tobacco.ID_143839.CM_1432086_2014 CM_1432086_2014_MissingValues	Success		1 min 36 s
rebean_wi.customers.Cactus_SA.ID_185193.CM_24874_2015 CM_24874_2015_NavigationError	Success		4 s
rebean_wi.customers.Cedar_Sinai_Medical_Center.ID_977698.CM_237748_2015	Success		3 s

CM_237748_2015		
rebean_wi.customers.Cedar_Sinai_Medical_Center.ID_977698.CM_237748_2015_SumDetail CM_237748_2015_SumDetail	Success	2 s
rebean_wi.customers.Cheil_Industries_Inc.ID_270236.ADAPT01713694_Start_On_New_Page ADAPT01713694	Success	2 s
rebean_wi.customers.China_Minmetals_Corporation.ID_741504.CM_1465244_2014_RowMerge CM_1465244_2014_RowMergeExcel	Success	5 s
rebean_wi.customers.China_Shenhua_Energy.ID_918444.CM_688973_2014 CM_688973_2014_EmptyValue	Success	2 s
rebean_wi.customers.Clal_Insurance_Enterprise.ID_655943.CM_135703_2015_TP CM_135703_2015_GroupingIssue	Success	3 s
rebean_wi.customers.Coca_Cola_Refreshments.ID_1190161.ADAPT01699318_Unable_To_Drill ADAPT01699318_Unable_To_Drill_Test	Success	3 s
rebean_wi.customers.COMPAREX_Sweden_AB.ID_941001.CM_ActionCannotPerformed CM_1451542_2014_actionCannotPerformedTP	Success	1 min 2 s
rebean_wi.customers.ConAgra_Foods.ID_35284.CM_1408506_2014_RefreshError CM_1408506_2014_RefreshErrors	Success	16 s
rebean_wi.customers.Consumers_Energy_Company.ID_281086.CM_1408686_2014 CM_1408686_2014_ErrorAtOpening	Success	2 s
rebean_wi.customers.Consumers_Energy_Company.ID_281086.CM_1416174_2014 CM_1416174_2014_IsNotNullFilterNotWorking	Success	7 s
rebean_wi.customers.Dell_USA.ID_34093.CM_806638_2014 CM_806638_2014_ExcelIssue	Success	22 s
rebean_wi.customers.Dell_USA.ID_34093.CM_1543759_2014_TP CM_1543759_2014_SimpleFilterIssue	Success	2 s
rebean_wi.customers.EDF_Energy_Plc.ID_918198.CM_1562701_2014 CM_1562701_2014_MissingData	Success	2 s
rebean_wi.customers.Elia_System_Operator_SA.ID_658125.CM_760518_2014_TP CM_760518_2014_HideWhenTrue	Success	2 s
rebean_wi.customers.Entreposte_Servicios.ID_977918.CM_170068_2015 CM_170068_2015_XL_Issue	Failure	' Content is not allowed in prolog. :: ERROR_USER Content is not allowed in prolog. :: ERROR_USER Content is not allowed in prolog. :: ERROR_USER' **junit.framework.AssertionFailedError: ' Content is not allowed in prolog. :: ERROR_USER Content is not allowed in prolog. :: ERROR_USER Content is not allowed in prolog. :: ERROR_USER' Content is not allowed in prolog. :: ERROR_USER Content is not allowed in prolog. :: ERROR_USER Content is not allowed in prolog. :: ERROR_USER' at extensions.toolbox.RebeanTestPlanDefinition.launchTC (RebeanTestPlanDefinition.java:486) at rebean_wi.customers.Entreposte_Servicios.ID_977918.CM_170068_2015.CM_170068_2015_XL_Issue (CM_170068_2015.java:33)**	7 s
rebean_wi.customers.Ericsson_AB.ID_964290.CM_826799_2014 CM_826799_2014_ChartIssue	Success	14 s
rebean_wi.customers.Ericsson_AB.ID_964290.CM_1516506_2014 CM_1516506_2014_DivZeroChart	Success	4 s
rebean_wi.customers.Ethias_SA.ID_12761082014.Ethias_12761082014_mergeBreakHeader CM_1276108_2014_MergeBreakHeader	Success	7 s
rebean_wi.customers.EV_Technologies.ID_1107567.CM_841951_2014_TP CM_841951_2014_UNXTimeStamp	Success	6 s
rebean_wi.customers.F_Hoffmann_La_Roche_AG.ID_33950.CM_650796_2014_TP CM_650796_2014_SectionIssue	Success	2 s
rebean_wi.customers.F_Hoffmann_La_Roche_AG.ID_33950.CM_755589_2014_TP CM_755589_2014_DeleteSectionIssue	Success	3 s
rebean_wi.customers.Florida_Hospital.ID_950313.CM_6119_2015 CM_6119_2015_ForEachIssue	Success	3 s
rebean_wi.customers.Geopost.ID_15745047.CM_948096_2014_TP CM_948096_2014_RankingSort	Success	2 s
rebean_wi.customers.Guggenhein_Services.ID_992099.CM_171876_2015 CM_171876_2015_HideEmptyIssue	Success	8 s
rebean_wi.customers.Harley_Davidson.ID_350766.CM_217305_2015 CM_217305_2015_MigrationIssue_1	Success	5 min 24 s
rebean_wi.customers.Harley_Davidson.ID_350766.CM_217305_2015 CM_217305_2015_MigrationIssue_2	Success	5 min 23 s
rebean_wi.customers.Harley_Davidson.ID_350766.CM_217305_2015 CM_217305_2015_MigrationIssue_3	Success	3 min 33 s
rebean_wi.customers.HealthShare_NSW.ID_1003817.CM_840064_2014 CM_840061_2014_formulasChange	Success	6 s
rebean_wi.customers.Heineken_Espana_SA.ID_963339.CM_55492_2015 CM_55492_2015_TP	Success	4 min 45 s
rebean_wi.customers.Hermes_Europe.ID_783993.CM_17582_2015 CM_17582_2015_PageIssue	Success	4 s
rebean_wi.customers.Hewlett_Packard_Company.ID_30698.CM_749697_2014_TP CM_749697_2014_ChartIssue	Success	2 s
rebean_wi.customers.HSBC_Holdings_PLC.ID_991368.CM_1463644_2014 CM_1463644_2014_DoubleBorderPDF	Success	4 s
rebean_wi.customers.Huawei_Technologies_Co.ID_278504.CM_793805_2014_TP CM_793805_2014_ColorGraph	Success	4 s
rebean_wi.customers.iCreate_Software_India.ID_1349999.CM_102064_2015_TP CM_102064_2015_PDF_Border	Success	3 s
rebean_wi.customers.Itelligence_Business.ID_401022.CM_922767_2014 CM_922767_2014_Multivalue	Success	2 s
rebean_wi.customers.JR_East_Retail_Net_Co.ID_1065878.CM_645657_2014 CM_645657_2014_Runningsum_Sort_incorrect_result	Success	3 s
rebean_wi.customers.Kimco_Realty_Corporation.ID_977252.CM_1502025_2014 CM_1502025_2014_PerfIssue	Success	31 s
rebean_wi.customers.Kinki_Nippon_Railway.ID_182953.CM_828407_2014 CM_828407_2014_WebiCrash	Success	4 s
rebean_wi.customers.Kubota.ID_182457.ADAPT01720106_Extra_Column_DragnDrop_VT ADAPT01720106	Success	4 s
rebean_wi.customers.Legg_Mason_And_Co.ID_955089.CM_830966_2014 CM_830966_2014_RankBy	Success	2 s
rebean_wi.customers.Loro_Piana.ID_399446.CM_9107_2015 CM_9107_2015_FUF_Issue	Success	4 s
rebean_wi.customers.M_video_Management.ID_513617.CM_1511108_2014 CM_1511108_2014_DelegatedMesure	Success	2 s
rebean_wi.customers.Medtronic_Inc.ID_900088.CM_659376_2014 CM_659376_2014_BetweenMisspelled	Success	4 s
rebean_wi.customers.Meiji_Seika_Pharma.ID_128967.CM_117778_2015 CM_117778_2015_IncorrectCalcul	Success	3 s
rebean_wi.customers.Mitsubishi_Chemical_Corporation.unknown.ADAPT01719759_Format_Blank_Column ADAPT01719759	Success	3 s
rebean_wi.customers.Mitsubishi_Corp.ID_36135.ADAPT01699287_Tab_Charactor ADAPT01699287	Success	4 s
rebean_wi.customers.Mitsubishi_Corp.ID_36135.ADAPT01709014_spacing ADAPT01709014_Spacing	Success	4 s
rebean_wi.customers.Mitsubishi_Corp.ID_36135.CM_1357747_2014 CM_1357747_2014_Sum_If_Issue	Success	3 s
rebean_wi.customers.Nestle_Globe_BTC.ID_501821.CM_156620_2015 CM_156620_2015_Schedule_Issue_TDC_UNX	Success	20 s
rebean_wi.customers.Nippon_Access_Inc.ID_362267.ID_362267_TestCase_1 CM_1384704_2014_BadSynchro	Success	5 s
rebean_wi.customers.Nomura_Research_Institute.unknown.ADAPT01711625_Text_Garbled	Success	3 s

ADAPT01711625		
rebean_wi.customers.Novartis_Vaccines_And_Diagnostics.ID_184701.CM_787792_2014_TP CM_787792_2014_XLSX_Date	Success	2 s
rebean_wi.customers.Novartis_Vaccines_And_Diagnostics.ID_184701.CM_793170_2014_TP CM_793170_2014_Hyperlinks	Success	5 min 41 s
rebean_wi.customers.Ntt_Docomo.ID_997488.CM_138459_2015_TP_Init CM_138459_2015_Init_1	Success	
rebean_wi.customers.Ntt_Docomo.ID_997488.CM_138459_2015_TP CM_138459_2015_Drill_Snapshot	Success	2 s
rebean_wi.customers.Ntt_Docomo.ID_997488.CM_138459_2015_Tp_Reset CM_138459_2015_3_Reset	Success	
rebean_wi.customers.Odyssey_America_Reinsurance_Co.ID_945278.ID_945278_TestCase_1 CM_1446291_2014_UnableToHide	Success	2 s
rebean_wi.customers.PepsiCo_Inc.ID_692630.ID_692630_TestCase_1 CM_1326726_2014_MissingColumns_Doc1	Success	1 min 16 s
rebean_wi.customers.PepsiCo_Inc.ID_692630.ID_692630_TestCase_1 CM_1326726_2014_MissingColumns_Doc2	Success	5 s
rebean_wi.customers.PepsiCo_Inc.ID_692630.ID_692630_TestCase_1 CM_1326726_2014_MissingColumns_Doc3	Success	1 min 21 s
rebean_wi.customers.PepsiCo_Inc.ID_692630.ID_692630_TestCase_1 CM_1326726_2014_MissingColumns_Doc4	Success	5 s
rebean_wi.customers.PepsiCo_Inc.ID_809237.ID_809237_TestCase_1 CM_1431229_2014_MissingValues	Success	5 s
rebean_wi.customers.Pfizer_Inc.ID_933432.CM_644030_2014 CM_644030_2014_RunningSum	Success	5 s
rebean_wi.customers.Pfizer_Inc.ID_933432.CM_835743_2014 CM_835743_2014_CalcIssue	Success	6 s
rebean_wi.customers.Pfizer_Inc.ID_933432.CM_1504791_2014 CM_1504791_2014_XLSX_Header	Success	7 s
rebean_wi.customers.Promega_Corporation.ID_824970.ID_824970_TestCase_1 CSSID_0020079747_0001161381_2013_DrillUpdateSet	Success	5 s
rebean_wi.customers.Qatar_Airways.ID_964262.CM_1445386_2014_TP CM_1445386_2014_noConsistentData	Success	3 s
rebean_wi.customers.Ransa_Comercial_SA.ID_881393.CM_216746_2015 CM_216746_2015	Success	10 s
rebean_wi.customers.Ransa_Comercial_SA.ID_881393.CM_216746_2015_HierarchicalNavigation CM_216746_2015_HierarchicalNavigation	Success	2 s
rebean_wi.customers.Robert_Bosch_GmbH.ID_10010.CM_710039_2014 CM_710039_2014_ExcelExportCrash	Success	6 s
rebean_wi.customers.Roberto_Cavalli.ID_969639.CM_1539874_2014 CM_1539874_2014_NoReport	Success	3 s
rebean_wi.customers.Ruokakesko_Oy.ID_512535.ID_512535_TestCase_1 CSSID_0120025231_0000476351_2014_ExportCSV	Success	2 s
rebean_wi.customers.Shell.ID_22183.ADAPT01714504 ADAPT01714504_checkQueryStrippingProperty	Success	2 s
rebean_wi.customers.Shell.ID_22136.ADAPT01676242_Dupplicated_Rows_Merged_Dim ADAPT01676242	Success	3 s
rebean_wi.customers.Shell.ID_949782.CM_836429_2014_TP CM_836429_2014_AggregationIssue	Success	2 s
rebean_wi.customers.Shell.ID_22183.CM_922967_2014 CM_922967_2014_ToRefresh	Success	3 s
rebean_wi.customers.Smith_Nephew_North_America.ID_192290.CM_204685_2015 CM_204685_2015	Success	2 s
rebean_wi.customers.Socpresse.ID_458545.CM_693446_2014_TP CM_693446_2014_IncorrectSum	Success	3 s
rebean_wi.customers.Sony_Pictures_Entertainment.ID_185212.CM_858676_2014 CM_858676_2014_XLSX_SectionIssue	Success	5 s
rebean_wi.customers.Sony_Pictures_Entertainment.ID_185212.CM_1503492_2014 CM_1503492_2014_RankIssue	Success	5 s
rebean_wi.customers.Sony_Pictures_Entertainment.ID_185212.CM_79465_2015 CM_79465_2015_Reccord_Mismatch	Success	18 min 19 s
rebean_wi.customers.Southern_Wine_Spirits_Of_America.ID_492420.ID_492420_TestCase_1 CSSID_0120025231_0001134551_2013	Success	4 min 29 s
rebean_wi.customers.SUISAG_AG_fur_Dienstleistungen.ID_1009623.CM_747939_2014_TP CM_747939_2014_ExportFormat	Success	2 s
rebean_wi.customers.Tata_Consultancy_Services_Ltd.ID_31117.CM_1509927_2014 CM_1509927_2014_OutlineIssue	Success	4 s
rebean_wi.customers.Technotrans_AG.ID_834183.ID_834183_TestCase_1 CM_1368118_2014_ErrorColumns	Success	3 s
rebean_wi.customers.Technotrans_AG.ID_834183.Cm_761697_2014_TP CM_761697_2014_HeaderIssue	Success	2 s
rebean_wi.customers.Texas_Instruments_Incorporated.ID_37915.CM_707176_2014 CM_707176_2014_RsError_TC1	Success	2 s
rebean_wi.customers.Texas_Instruments_Incorporated.ID_37915.CM_707176_2014 CM_707176_2014_RsError_TC2	Success	3 s
rebean_wi.customers.Texas_Instruments_Incorporated.ID_37915.CM_799629_2014 CM_799629_2014_ExcelImageName	Success	6 s
rebean_wi.customers.The_Bank_of_Tokyo.ID_347072.CM_1156790_2014_InitTC CM_1156790_2014_Init	Success	14 s
rebean_wi.customers.The_Bank_of_Tokyo.ID_347072.ID_347072_TestCase_1 CM_1156790_2014_MaxCharacterStreamSize	Success	26 s
rebean_wi.reporting_rendering.validation.Init_TC_itbi init_PrepaForITBI	Success	13 s
rebean_wi.customers.The_Hartford_Financial_Services.ID_353355.CM_45776_2015 CM_45776_2015_ReportFilterIssue	Success	18 s
rebean_wi.customers.The_Lubrizol_Corporation.ID_197246.ID_197246_TestCase_1 CM_1383807_2014_FormattedValuesErrors	Success	3 s
rebean_wi.customers.The_Procter_And_Gamble_Company.ID_27345.CM_187536_2015 CM_187536_2015_MultiValue	Success	52 s
rebean_wi.customers.The_Procter_And_Gamble_Company.ID_27345.CM_127265_2015 CM_127265_2015_XCELMissingParts	Success	5 s
rebean_wi.customers.The_Standard_Bank_of_South_Africa_Ltd.ID_126497.CM_1573726_2014_TP CM_1573726_2014_CollapseHierarchy	Success	2 s
rebean_wi.customers.TIS_Inc.ID_300998.CM_67357_2015_TP CM_67357_2015_Missing_Reccord_CDP	Success	8 s
rebean_wi.customers.Union_Gas_Limited.ID_195098.CM_1288179_2014 CM_1288179_2014_CustomSqlNotEnable	Success	2 s
rebean_wi.customers.University_of_Queensland.ID_966000.CM_76245_2015_TP CM_76245_2015_ExcelExport	Success	6 s
rebean_wi.customers.US_Dept_of_Health_And_Human_Svcs.ID_11111.CM_672308_2014 CM_672308_2014_MultivalueIssue	Success	6 s
rebean_wi.customers.Warner_Bros.ID_132729.CM_97749_2015 CM_97749_2015_MissingData	Success	13 s
rebean_wi.customers.Zurich.ID_960077.CM_495925_2014_TP CM_495925_2014_ExportTXT	Success	51 s
rebean_wi.customers.Zurich.ID_960077.CM_721110_2014_TP CM_721110_2014_PreviousIssue	Success	4 s
rebean_wi.customers.Zurich.ID_960077.CM_932326_2014_TP CM_932326_2014	Success	2 s

Properties »

Annexe B

Tous les tests implémentés

DATE	ID	SUMMARY	Testcase name	LINK
04/08/2014	6665695	add a testplan		08\04082014071708.xps
04/08/2014	6665796	Add the testplan in the testsuite. Opening and outline tests.	- BIT BIWEBISL2_1464_OpenADoc - BIT BIWEBISL3_1605_outline	08\04082014084249.xps
05/08/2014	6669508	Add wid files for webi_perf_wrkflw_amandine		08\05082014031026.xps
11/08/2014	6696723	Add testcase for break header	CM_1276108_2014_MergeBreakHeaderTC	08\11082014084911.xps
13/08/2014	6706082	Add testcase for input controls upgrade	CM_1386568_2014_InputControlsUpgrade	08\13082014093545.xps
13/08/2014	6706165	Modify the test file		08\13082014104552.xps
14/08/2014	6709692	Modify test case about upgrade		08\14082014020039.xps
14/08/2014	6710130	Add test case architecture for ADAPT01709014_spacing	ADAPT01709014	08\14082014040011.xps

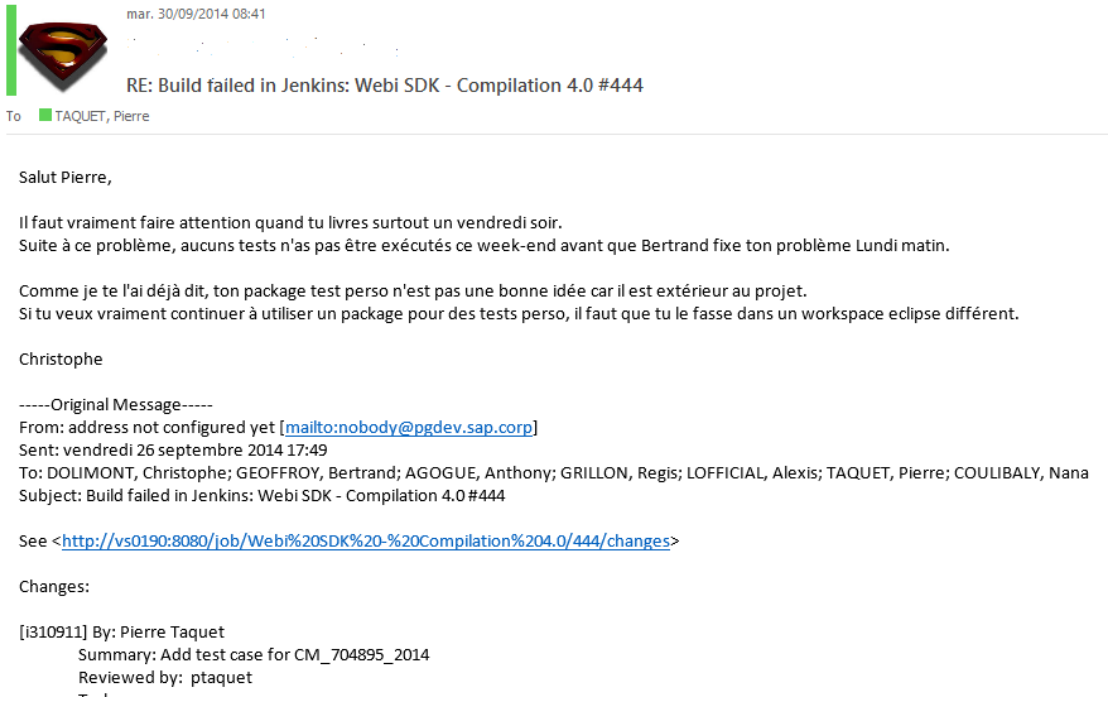
20/08/2014	6735429	Add testcase for check spacing. Add an html output for IE9	ADAPT01709014	08\20082014082612.xps
20/08/2014	6735554	Add reference for ADAPT01709014		08\20082014093427.xps
21/08/2014	6739792	Add testcase for CM_1445386_2014	CM_1445386_2014_noConsistentData	08\21082014032600.xps
21/08/2014	6740001	Add static testcase for CM_1445386_2014	CM_1445386_2014_noConsistentData	08\21082014064431.xps
22/08/2014	6744534	Add a txt_ reference for CM_1445386_2014		08\22082014030624.xps
25/08/2014	6756622	Add a txt_ ref for CM_1445386_2014		08\25082014031203.xps
05/09/2014	6806998	Add a new helper file specific to aurora		09\05092014062655.xps
09/09/2014	6820688	Improve AuroraHalper and complete		09\09092014021645.xps

		the search universe method		
17/09/2014	6856535	Add testcase CM_1451542_2014. Reopen doc after restart server	CM_1451542_2014_actionCannotPerformed	09\17092014064619.xps
19/09/2014	6866152	Add query spec for CM 1451542 2014		09\19092014024806.xps
19/09/2014	6866582	Add testcase for CM 659376_2014	CM_659376_2014_Misspelled	09\19092014081811.xps
19/09/2014	6866619	Add document save for CM 659376 2014	CM_659376_2014_Misspelled	09\19092014085603.xps
22/09/2014	6878587	Improve testcase CM_659376_2014	CM_659376_2014_Misspelled	09\22092014031008.xps
22/09/2014	6879117	Modify the save of documents in personal documents		09\22092014093842.xps
23/09/2014	6883507	complete auroraHelper	CM_659376_2014_Misspelled	09\23092014040249.xps
24/09/2014	6888819	save in folder when savingDoc is true	CM_1451542_2014_actionCannotPerformed	09\24092014032401.xps

25/09/2014	6894669	Add setFormula method in auroraHelper		09\25092014102220.xps
25/09/2014	6894688	Modify test which use auroraHelper. The docInst setter of docCalc	CM_659376_2014_Misspelled	09\25092014102742.xps
26/09/2014	6899893	Add test case for CM_704895_2014	CM_704895_2014_Formatting	09\26092014094053.xps
31/10/2014	**7056735**	Add a test case for change source feature		10\changesource.xps
13/11/2014	**7113504**	Add CM_840064_2014_formulasChange	CM_840064_2014_formulasChange	11\840064.xps

Annexe C

Mail reçu d'un mauvais push



mar. 30/09/2014 08:41

RE: Build failed in Jenkins: Webi SDK - Compilation 4.0 #444

To TAQUET, Pierre

Salut Pierre,

Il faut vraiment faire attention quand tu livres surtout un vendredi soir.
Suite à ce problème, aucuns tests n'as pas être exécutés ce week-end avant que Bertrand fixe ton problème Lundi matin.

Comme je te l'ai déjà dit, ton package test perso n'est pas une bonne idée car il est extérieur au projet.
Si tu veux vraiment continuer à utiliser un package pour des tests perso, il faut que tu le fasse dans un workspace eclipse différent.

Christophe

-----Original Message-----
From: address not configured yet [mailto:nobody@pgdev.sap.corp]
Sent: vendredi 26 septembre 2014 17:49
To: DOLIMONT, Christophe; GEOFFROY, Bertrand; AGOGUE, Anthony; GRILLON, Regis; LOFFICIAL, Alexis; TAQUET, Pierre; COULIBALY, Nana
Subject: Build failed in Jenkins: Webi SDK - Compilation 4.0 #444

See <http://vs0190:8080/job/Webi%20SDK%20-%20Compilation%204.0/444/changes>

Changes:

[i310911] By: Pierre Taquet
Summary: Add test case for CM_704895_2014
Reviewed by: ptaquet

FIGURE C.1 – Capture d'écran du résultat de ma mauvaise manipulation

Annexe D

Fiche d'information de SAP

SAP Global Corporate Affairs
July 22, 2015

SAP: Run Simple - The World's Largest Provider of Enterprise Application Software

As market leader in enterprise application software, SAP (NYSE: SAP) helps companies of all sizes and industries innovate through simplification. From back office to boardroom, warehouse to storefront, on premise to cloud, desktop to mobile device – SAP empowers people and organizations to work together more efficiently and use business insight more effectively to stay ahead of the competition. SAP applications and services enable customers to operate profitably, adapt continuously, and grow sustainably.

Customers

- SAP serves > 293,000 customers in 190 countries
- > 80% of SAP customers are SMEs
- SAP customers include:
  - 87% of the Forbes Global 2000 companies
  - 98% of the 100 most valued brands
  - 100% of the Dow Jones top scoring sustainability companies
- Our customers produce
  - 78% of the world's food
  - 82% of the world's medical devices
- 74% of the world's transaction revenue touches an SAP system[1]

How We Run Simple

- Our vision is to help the world run better and improve people's lives
- Our mission is to help our customers run at their best
- To fulfill our mission, we help our customers master complexity with simple and easy to use solutions focused on innovation in the Cloud and on SAP HANA, e.g. S/4 HANA, Simple Finance
- Our strategy is to become THE cloud company powered by SAP HANA with focus on
  - Public Cloud: Standard and Suite solutions
  - Business Network: Ariba, Concur, Fieldglass
  - Private Cloud on HANA Enterprise Cloud

Financials

- Revenue – FY 2014 (non-IFRS@const. curr.)
  - Cloud subscr. & support € 1.1 bn (+45%)
  - SW&Support[2] € 13.8 bn (+5%)
  - Total € 17.6 bn (+5%)
- Revenue – Q2 2015 (non-IFRS@const. curr.)
  - Cloud subscr. & support € 555 mn (+92%)
  - SW&Support[2] € 3.51 bn (+3%)
  - Business Network € 333 mn (+145%)
  - Total €4.5 bn (+8%)
- Outlook 2015 (non-IFRS @ const. curr.)
  - Cloud subscr.&support rev.: €1.95 - €2.05 bn
  - Cloud &SW[3] rev up 8%-10% (2014: €14.33 bn)
  - Operating profit in a range of €5.6 - €5.9 bn
- Ambition 2017 (non-IFRS@cc)
  - Cloud subscr.&support rev.: €3.5 - €3.6 bn
  - Total revenue: €21 - €22 bn
  - Operating profit in a range of €6.3 - €7.0 bn
- Ambition 2020 (non-IFRS@cc)
  - Cloud subscr.&support rev.: €7.5 - €8.0 bn
  - Total revenue: €26 - €28 bn
  - Operating profit in a range of €8 - €9 bn

1 Source: McKinsey/SAP analysis update 4/2013
2 SW&Support: Software and Support
3 Cloud&SW: Cloud Subscriptions and Software

Market Position

Enterprise Application Software

- SAP is market leader in
  - applications
  - analytics
  - mobility solutions
- Fastest growing database vendor
- Broadest portfolio of modular and suite solutions available on premise, in the cloud and hybrid: customers have full choice of consumption model

Top Cloud Vendor

- Fastest growing company at scale in the cloud
- Cloud user base: ~82 mn subscribers
- Largest cloud portfolio: >30 solutions for all lines-of-business (LoB) as well as Business Suite
- Market leader in Human Capital Mgmt. solutions

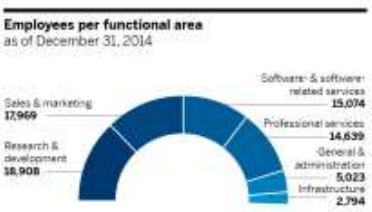
Leading Mobility Vendor

- Market leader for mobile business applications: >130 mn mobile users, >500 mobile apps
- SAP mobile solutions reach 99.9% of mobile subscribers via text messaging
- 1.8 bn text messages per day processed and delivered by SAP Mobile Platform

Innovation

- 14 Development centers (SAP Labs) worldwide
- 100 Development locations worldwide
- 13 Co-Innovation and Living Labs worldwide
- 21 Research locations worldwide
- Innovation Center in Potsdam, Germany
- Partner network with >13,000 SAP partner companies around the world
- Sapphire Ventures: Invested in >150 IT startups globally since 1996
  - US$1.4 bn capital under management
  - Operates independently from SAP
  - Gives SAP early visibility and access to markets, trends & innovation

Employees and Basic Facts



- Headquarters: Walldorf, Germany
- Founded: April 1, 1972
- Listing: Frankfurt, New York
- 74,497 employees worldwide (06/30/2015)
  - EMEA: 33,467
  - Americas: 21,574
  - APJ: 19,456
- >120 nationalities worldwide
- nearly 80 nationalities at headquarters

Useful Links

Executives – Supervisory Board – Products – Industries and Solutions – Events – Financials – Photos and Films – SAP Profile

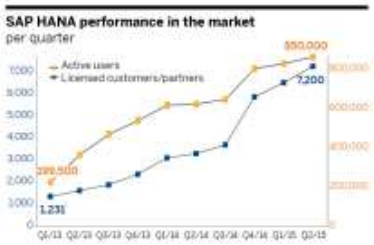
SAP's End-to-End Solutions

Simple user experience designed with a mobile first mindset

1 – Applications

- Packaged solutions for 25 industries and 11 lines-of-business. On premise, cloud, hybrid
- SAP Business Suite optimizes all business-critical processes
- Market leader in products for business analysis and a technology leader for real-time analysis: Business intelligence, Predictive Analytics etc.
- S/4HANA: the most significant business software innovation since SAP R/3
  - 10x smaller data footprint than conv. systems
  - 7x higher throughput
  - 1800x faster analytics and reporting
  - All data: text, social data, geo data, graph processing
  - Instantaneous simulations, predictions, recommendations
  - Delivery on premise, cloud and hybrid
  - Connected to Internet of Things

2 – SAP HANA Platform



- SAP HANA is the market-leading platform for real-time computing:
  - Open platform
  - Basis for all major SAP solutions, will become underlying technology for all SAP applications
  - SAP HANA Cloud Platform enables customers to extend existing Cloud applications or quickly develop entirely new ones
  - HANA Enterprise Cloud: access to the full potential of HANA via managed Cloud
- Customers:
  - >7,200 HANA customers
  - >850,000 active users
  - Suite on HANA: 1,850 customers
  - S/4HANA (launch Feb. 2015): 900 customers
  - >2,100 startups developing on HANA platform

3 – Business Network

- SAP's Business Network companies provide the leading solutions in the areas of
  - goods and services (Ariba)
  - travel and expense (Concur) and
  - contingent labor (Fieldglass)
- > 35 million users in the Networks worldwide
- The SAP Business Networks connect industry ecosystems of more than 1.9 million businesses, their partners, 3rd party developers and system integrators
- Trade volume > US$0.8 trillion p.a.

Annexe E

Exemple d'une sortie console

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Avant la methode run

Check parameters first in: C:\p4\service.td.webi_LVLD60232305A\depot\
    SoftwareTesting\WebISDK\Workspace_Aurora\rebean\parameters.xml
Resources parameters: C:\p4\service.td.webi_LVLD60232305A\depot\SoftwareTesting
    \cube\Resources_AURORA\
IsClientServerDeployment value changed (replaced by parameters): false
Log file.C:\p4\service.td.webi_LVLD60232305A\depot\SoftwareTesting\WebISDK\
    Workspace_Aurora\rebean\logs\jeu_de_test_pierre\TestcaseCreateANewDocument.
    txt=
[<]       -Testcase.java(startMSGStep) : Testcase.java(startTestcase) :
    Testcase TestcaseCreateANewDocument [START]
[O]           -Testcase.java(okMSG) : Testcase.java(startTestcase) : Started at
     Mon Jul 28 14:32:20 CEST 2014   :: OK
[<]           -initSession [START]
[O]               -Session Manager inited   :: OK
[<]               -getClusterName [START]
[>]               -getClusterName [END]
[<]               -getWebiServerName [START]
[O]                   -AdaptiveProcessingServer to be monitored is MySIA.
    AdaptiveProcessingServer   :: OK
[>]               -getWebiServerName [END]
[O]               -Login with user "Administrator" on DEWDFTV01232.DHCP.PGDEV.
    SAP.CORP:6400   :: OK
[>]           -initSession [END]
[<]           -initRepoProxy [START]
[O]               -Repository Proxy inited   :: OK
[>]           -initRepoProxy [END]
[<]           -initInfoStore [START]
[>]           -initInfoStore [END]
[<]           -initDefaults [START]
```

```
[<]               -findInfoObjects [START]
[O]                   -Try to find InfoObjects with name = "Administrator" and
    with type = "FavoritesFolder"   :: OK
[<]                   -executeQuery [START]
[O]                       -Try to execute query 'SELECT * FROM CI_INFOOBJECTS
    WHERE SI_KIND='FavoritesFolder' AND SI_NAME='Administrator''   :: OK
[>]                   -executeQuery [END]
[O]                   -1 InfoObject(s) found   :: OK
[>]               -findInfoObjects [END]
[<]               -executeQuery [START]
[O]                   -Try to execute query 'SELECT * FROM CI_INFOOBJECTS WHERE
     SI_KIND='Folder' AND SI_PARENTID=0'   :: OK
[>]               -executeQuery [END]
[<]               -executeQuery [START]
[O]                   -Try to execute query 'SELECT * FROM CI_INFOOBJECTS WHERE
     SI_ID='18''   :: OK
[>]               -executeQuery [END]
[>]           -initDefaults [END]
[<]           -waitForWebiAlive [START]
[>]           -waitForWebiAlive [END]
[<]           -initWIReportEngine [START]
[O]               -In case of Error, check that sdk.core.config is in classpath
    .   :: OK
[<]               -getGlobalContext [START]
[>]               -getGlobalContext [END]
[O]               -report engine is ready   :: OK
[>]           -initWIReportEngine [END]
[<]           -getConnections [START]
[<]               -executeQuery [START]
[O]                   -Try to execute query 'SELECT SI_ID, SI_NAME FROM
    CI_APPOBJECTS WHERE SI_KIND='CCIS.DataConnection''   :: OK
[>]               -executeQuery [END]
[O]               -34 connection(s) found   :: OK
[>]           -getConnections [END]
[<]           -getUniverses [START]
[<]               -executeQuery [START]
[O]                   -Try to execute query 'SELECT SI_ID, SI_NAME FROM
    CI_APPOBJECTS WHERE SI_KIND='Universe''   :: OK
[>]               -executeQuery [END]
[O]               -26 universe(s) found   :: OK
[>]           -getUniverses [END]
[<]           -getWIReportServerPID [START]
[O]               -Requesting MySIA.WebIntelligenceProcessingServer PID.
    Current time is: Mon Jul 28 14:32:29 CEST 2014   :: OK
[O]               -MySIA.WebIntelligenceProcessingServer PID is: 3132. Current
    time is: Mon Jul 28 14:32:29 CEST 2014   :: OK
[>]           -getWIReportServerPID [END]
[<]           -cleanResultStorage [START]
[O]               -Clear result folder: C:\p4\service.td.webi_LVLD60232305A\
```

```
    depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build
    Number\res\jeu_de_test_pierre\TestcaseCreateANewDocument\   :: OK
[<]               -emptyFolder [START]
[<]                   -emptyFolder [START]
[O]                       -clean Folder: C:\p4\service.td.webi_LVLD60232305A\
    depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build
    Number\res\jeu_de_test_pierre\TestcaseCreateANewDocument\createdoc   :: OK
[>]                   -emptyFolder [END]
[O]                   -clean Folder: C:\p4\service.td.webi_LVLD60232305A\depot\
    SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build Number\
    res\jeu_de_test_pierre\TestcaseCreateANewDocument\   :: OK
[>]               -emptyFolder [END]
[O]               -Clear result folder: C:\p4\service.td.webi_LVLD60232305A\
    depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build
    Number\ref\jeu_de_test_pierre\TestcaseCreateANewDocument\   :: OK
[<]               -emptyFolder [START]
[<]                   -emptyFolder [START]
[O]                       -clean Folder: C:\p4\service.td.webi_LVLD60232305A\
    depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build
    Number\ref\jeu_de_test_pierre\TestcaseCreateANewDocument\createdoc   :: OK
[>]                   -emptyFolder [END]
[O]                   -clean Folder: C:\p4\service.td.webi_LVLD60232305A\depot\
    SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build Number\
    ref\jeu_de_test_pierre\TestcaseCreateANewDocument\   :: OK
[>]               -emptyFolder [END]
[>]           -cleanResultStorage [END]
[<]           -newAuroraDocument [START]
[>]           -newAuroraDocument [END]


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Pendant la methode run


[<]           -updateStructureAurora [START]
[>]           -updateStructureAurora [END]
[<]           -Testcase step: createdoc [START]
[<]               -newDocument based on unv BEACH [START]
[<]                   -findUniverseInRoot [START]
[O]                       -Try to find InfoObjects with name = "BEACH" and with
     type = "Universe"   :: OK
[<]                       -executeQuery [START]
[O]                           -Try to execute query 'SELECT SI_ID FROM
    CI_APPOBJECTS WHERE SI_NAME='Universes' AND SI_KIND='Folder' AND
    SI_PARENTID=95'   :: OK
[>]                       -executeQuery [END]
[O]                       -1 InfoObject(s) found   :: OK
[<]                       -executeQuery [START]
```

```
99 [O]                          -Try to execute query 'SELECT * FROM
   CI_APPOBJECTS WHERE SI_KIND='Universe' AND SI_NAME='BEACH' AND SI_PARENTID
   =532'   :: OK
   [>]                      -executeQuery [END]
101 [O]                      -1 InfoObject(s) found   :: OK
   [>]                   -findUniverseInRoot [END]
103 [<]                   -createUniverseDataSourceInfo unv cuid Af7KxGg_CDpFp.Uq.
   gt3D58 [START]
   [O]                       -- [DSL] create DataSourceInfo : OK   :: OK
105 [>]                   -createUniverseDataSourceInfo unv cuid Af7KxGg_CDpFp.Uq.
   gt3D58 [END]
   [<]                   -DSLaddDataProvider [START]
107 [O]                      -- [DSL] add new data provider : OK   :: OK
   [>]                   -DSLaddDataProvider [END]
109 [O]                   -add a simple flat query   :: OK
   [>]               -newDocument based on unv BEACH [END]
111 [<]               -createAuroraReport My Report [START]
   [O]                   -report.setName My Report   :: OK
113 [O]                   -_repStruct.getChildren() - size: 0   :: OK
   [>]               -createAuroraReport My Report [END]
115 [<]               -createReportBody(Aurora) [START]
   [>]               -createReportBody(Aurora) [END]
117 [<]               -applyFormat [START]
   [>]               -applyFormat [END]
119


121
%%%%%%%%%%%%%%%%%%%%%%%% Pendant la methode StopTestcaseStepWithAction()
123


125 [<]               -applyFormat [START]
   [>]               -applyFormat [END]
127 [<]               -verifyWithRef - PageMode Listing [START]
   [<]                   -prepareResFiles [START]
129 [<]                       -createPath [START]
   [O]                           -Directory created: C:\p4\service.td.
   webi_LVLD60232305A\depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\
   results\Your Build Number\res\jeu_de_test_pierre\TestcaseCreateANewDocument
   \createdoc   :: OK
131 [>]                       -createPath [END]
   [<]                       -checkSupportedViews [START]
133 [>]                       -checkSupportedViews [END]
   [<]                       -saveAsXMLaurora [START]
135 [<]                           -saveReportAsCharacterViewAurora [START]
   [O]                               -set LISTING for pagination mode   :: OK
137 [O]                               -navigate to report '1'   :: OK
   [O]                               -get CharacterView view for report: My Report
       :: OK
139 [>]                           -saveReportAsCharacterViewAurora [END]
```

```
[0]                          -save as xml: C:\p4\service.td.webi_LVLD60232305A
    \depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build
    Number\res\jeu_de_test_pierre\TestcaseCreateANewDocument\createdoc   :: OK
141 [>]                       -saveAsXMLaurora [END]
    [<]                       -XML2TXT [START]
143 [>]                       -XML2TXT [END]
    [<]                       -saveDocInPersonal [START]
145 [<]                           -search (Folder) [START]
    [<]                               -findInfoObjects [START]
147 [0]                               -Try to find InfoObjects with name = "
    Personal Documents" and with type = "Folder"   :: OK
    [<]                               -executeQuery [START]
149 [0]                                   -Try to execute query 'SELECT * FROM
    CI_INFOOBJECTS WHERE SI_KIND='Folder' AND SI_NAME='Personal Documents''
    :: OK
    [>]                               -executeQuery [END]
151 [0]                               -1 InfoObject(s) found   :: OK
    [>]                           -findInfoObjects [END]
153 [>]                       -search (Folder) [END]
    [<]                       -saveDocument_aurora [START]
155 [<]                           -search (Folder) [START]
    [<]                               -findInfoObjects [START]
157 [0]                               -Try to find InfoObjects with name =
    "Personal Documents" and with type = "Folder"   :: OK
    [<]                               -executeQuery [START]
159 [0]                                   -Try to execute query 'SELECT *
    FROM CI_INFOOBJECTS WHERE SI_KIND='Folder' AND SI_NAME='Personal Documents'
    '   :: OK
    [>]                               -executeQuery [END]
161 [0]                               -1 InfoObject(s) found   :: OK
    [>]                           -findInfoObjects [END]
163 [>]                       -search (Folder) [END]
    [<]                       -search (Document) [START]
165 [<]                           -search (Folder) [START]
    [<]                               -findInfoObjects [START]
167 [0]                               -Try to find InfoObjects with
    name = "Personal Documents" and with type = "Folder"   :: OK
    [<]                               -executeQuery [START]
169 [0]                                   -Try to execute query 'SELECT
     * FROM CI_INFOOBJECTS WHERE SI_KIND='Folder' AND SI_NAME='Personal
    Documents''   :: OK
    [>]                               -executeQuery [END]
171 [0]                               -1 InfoObject(s) found   :: OK
    [>]                           -findInfoObjects [END]
173 [>]                       -search (Folder) [END]
    [<]                       -findInfoObjects [START]
175 [0]                           -Try to find InfoObjects with name =
    "TestcaseCreateANewDocument_createdoc" and with type = "Webi"   :: OK
    [<]                           -executeQuery [START]
```

```
177 [O]                                          -Try to execute query 'SELECT *
    FROM CI_INFOOBJECTS WHERE SI_KIND='Webi' AND SI_NAME='
    TestcaseCreateANewDocument_createdoc''   :: OK
    [>]                                     -executeQuery [END]
179 [O]                                     -1 InfoObject(s) found   :: OK
    [>]                                 -findInfoObjects [END]
181 [>]                             -search (Document) [END]
    [O]                             -Document
    TestcaseCreateANewDocument_createdoc saved in Personal Documents   :: OK
183 [>]                         -saveDocument_aurora [END]

185


187


189 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Apres la methode run

191


193 [O]                         -document saved in default personal folder:
    TestcaseCreateANewDocument_createdoc   :: OK
    [>]                     -saveDocInPersonal [END]
195 [>]                 -prepareResFiles [END]
    [<]                 -prepareRefFiles [START]
197 [<]                     -getFileNamesWithExtension [START]
    [O]                         -Get 1 files with extension "txt"   :: OK
199 [>]                     -getFileNamesWithExtension [END]
    [<]                     -copyFile [START]
201 [O]                         -copy file from C:\p4\service.td.
    webi_LVLD60232305A\depot\SoftwareTesting\cube\Resources_AURORA\storage\auto
    \jeu_de_test_pierre\TestcaseCreateANewDocument\txt\
    TestcaseCreateANewDocument_createdoc1.txt to C:\p4\service.td.
    webi_LVLD60232305A\depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\
    results\Your Build Number\ref\jeu_de_test_pierre\TestcaseCreateANewDocument
    \createdoc\TestcaseCreateANewDocument_createdoc1.txt in 0s   :: OK
    [>]                     -copyFile [END]
203 [>]                 -prepareRefFiles [END]
    [<]                 -createPath [START]
205 [O]                     -Directory already exists: C:\p4\service.td.
    webi_LVLD60232305A\depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\
    results\Your Build Number\ref\jeu_de_test_pierre\TestcaseCreateANewDocument
    \createdoc   :: OK
    [>]                 -createPath [END]
207 [<]                 -getFileNamesWithExtension [START]
    [O]                     -Get 1 files with extension "txt"   :: OK
209 [>]                 -getFileNamesWithExtension [END]
    [<]                 -comparePreparedFiles [START]
211 [<]                     -compareTXTFiles [START]
    [O]                         -C:\p4\service.td.webi_LVLD60232305A\depot\
```

```
    SoftwareTesting\WebISDK\Workspace_Aurora\rebean\results\Your Build Number\
    res\jeu_de_test_pierre\TestcaseCreateANewDocument\createdoc\
    TestcaseCreateANewDocument_createdoc1.txt and C:\p4\service.td.
    webi_LVLD60232305A\depot\SoftwareTesting\WebISDK\Workspace_Aurora\rebean\
    results\Your Build Number\ref\jeu_de_test_pierre\TestcaseCreateANewDocument
    \createdoc\TestcaseCreateANewDocument_createdoc1.txt are SAME   :: OK
213 [>]                       -compareTXTFiles [END]
    [>]                   -comparePreparedFiles [END]
215 [>]               -verifyWithRef - PageMode Listing [END]
    [>]           -Testcase step: createdoc [END]
217 [<]           -closeDocument [START]
    [>]           -closeDocument [END]
219 [<]           -closeAll [START]
    [>]           -closeAll [END]
221 [<]           -logoff [START]
    [0]               -close report engines   :: OK
223 [>]           -logoff [END]
    [<]           -logoff [START]
225 [0]               -Close session   :: OK
    [>]           -logoff [END]
227 [0]           -Total Time Spent for Testcase TestcaseCreateANewDocument =
    22.083 second(s)   :: OK
    [0]           -Finished at Mon Jul 28 14:32:42 CEST 2014   :: OK
229 [>]       - Testcase TestcaseCreateANewDocument [END]
    [TestcaseCreateANewDocument]    --> PASSED : 'RebeanTestPlanDefinition.java(
    StressPF) : Min=1.4065508E12|Max=1.4065508E12|Average=1.4065508E12|Count=1|
    CPUTime=0|PeakVM=0|LeakVM=0'
231 Passed.TestcaseCreateANewDocument=
```

Annexe F

Code Java permettant de se connecter à l'API CWB

```
package com.sap.authoring;

import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.Reader;
import java.io.StringReader;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.xpath.XPath;
```

```
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathFactory;

import org.apache.commons.codec.binary.Base64;
import org.junit.rules.TestWatchman;
import org.junit.runners.model.FrameworkMethod;
import org.junit.runners.model.Statement;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.xml.sax.EntityResolver;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

import com.sap.prd.access.credentials.api.Credential;
import com.sap.prd.access.credentials.api.ProdPassAccess;

//TestWatchman is used for compatibility purpose with older versions of JUnit
@SuppressWarnings("deprecation")
public class IssueWatcher extends TestWatchman {

  public static void main(String[] args) {
    try {
      Issue issue = new CWBIssue("012006153200001159182015");

      issue = new JiraIssue("https://sapjira.wdf.sap.corp/browse/BITBIWEBISL2-1542");
    } catch (Exception e) {
      // TODO Auto-generated catch block
      e.printStackTrace();
    }
  }

  @Override
  public Statement apply(final Statement base, final FrameworkMethod method, Object target) {
    return new Statement() {
      @Override
      public void evaluate() throws Throwable {
        String issueCheck = System.getProperty("issue.check");
        Throwable testError = null;
        Defect defect = (issueCheck == null) ? null : method.getAnnotation(Defect.class);

        Issue issue = null;
        if (defect != null) {
          if (defect.type().equals("jira")) {
            issue = new JiraIssue(defect.url());
          } else if (defect.type().equals("cwb")) {
            issue = new CWBIssue(defect.url());
```

```
        }
78      }

80      // Run the test
        starting(method);
82      try {
          base.evaluate();
84      } catch (Throwable t) {
          testError = t;
86      }

88      boolean success = false;
        if (issue != null) {
90        success = testError == null;
          // if the test failed
92        if (!success) {
            boolean errorMessageMatches = matches(testError.getMessage(),
    defect.message());
94          if (errorMessageMatches && "showDefects".equals(issueCheck)) {
              testError = new Exception("The defect '" + issue.getBrowseLink()
    + "' is associated to this test. Its status is '"
96                + issue.getStatus() + "'");
            } else {
98            success = errorMessageMatches && issue.isOpen();
            }
100       }

102       // Log actual result
          System.out.println("\n*** issueWatcher - test '" + method.getName() +
     "' ***");
104       System.out.println(" - Defect " + issue.getBrowseLink() + ": " +
    issue.getStatus());
          System.out.println(" - Actual test status: " + ((testError == null) ?
     "passed" : "failed"));
106       if (testError != null) {
            System.out.println(" - Expected error message: " + defect.message()
    );
108         System.out.println(" - Actual error message: " + testError.
    getMessage());
          }
110       System.out.println(" - Final test status: " + (success ? "passed" : "
    failed"));
          System.out.println("*** JiraIssueWatcher ***");
112     }
        try {
114       if (success) {
            succeeded(method);
116       } else {
            failed(testError, method);
```

```
118             throw testError;
            }
120         } finally {
            finished(method);
122         }
        }
124   };
  }
126
  /**
128  * @param url
  *            Browse URL of the defect (e.g. <i>https://jira.atlassian.com/
     browse/TST-51499</i><br>
130  *            or<br>
  *            the Correction Request id if the type is cwb).
132  * @param message
  *            Exact text or regular expression matching the message of the
     exception raised by the test. The default value matches all messages.
134  *
  */
136 @Target(ElementType.METHOD)
  @Retention(RetentionPolicy.RUNTIME)
138 public static @interface Defect {
    String url();
140
    String message() default ".*";
142
    String type() default "jira";
144 }

146 private boolean matches(String toMatch, String pattern) {
    boolean matches = toMatch.equals(pattern);
148   if (!matches) {
      Pattern p = Pattern.compile(pattern, Pattern.MULTILINE);
150   matches = p.matcher(toMatch).find();
    }
152   return matches;
  }
154
  /*
156  * #########################################################################
  *
158  * Specifics inner classes getting status from Jira or Java Correction
     Workbench
  *
160  * #########################################################################
  */
162
  private static class JiraIssue extends IssueAbs implements Issue {
```

```
164     private String jiraIssueRestUrl, jiraIssueBrowseUrl;
        private String status;
166
        public JiraIssue(String issueUrl) throws Exception {
168       this.jiraIssueBrowseUrl = issueUrl;

170       URL url = new URL(issueUrl);
          String baseUrl = url.getProtocol() + "://" + url.getHost() + ((url.
        getPort() == -1) ? "" : (":" + url.getPort()));
172       String issueId = issueUrl.substring(issueUrl.lastIndexOf('/') + 1);
          this.jiraIssueRestUrl = baseUrl + "/rest/api/latest/issue/" + issueId;
174
          this.status = getIssueStatus(jiraIssueRestUrl);
176       System.out.println("Jira status : "+status);
        }
178
        @Override
180     public String getStatus() {
          return this.status;
182     }

184     @Override
        public String getBrowseLink() {
186       return this.jiraIssueBrowseUrl;
        }
188
        @Override
190     public boolean isOpen() {
          return "Open".equals(status) || "Active".equals(status);
192     }

194     private String getIssueStatus(String issueRestUrl) throws Exception {
          String issue = execute(issueRestUrl);
196
          String regex = ".*\"status\":\\{.*?\"name\":\"(.*?)\".*?\\}.*";
198       Pattern pattern = Pattern.compile(regex);
          Matcher matcher = pattern.matcher(issue);
200
          boolean found = matcher.find();
202       if (!found) {
            throw new Exception("Issue status not found for " + issueRestUrl);
204       }

206       String status = matcher.group(1);

208       return status;
        }
210
        private String execute(String url) throws Exception {
```

```
212       String strUrl = url;
          StringBuilder buffer = new StringBuilder();
214
          try {
216         HttpsURLConnection connection = sendRequest(strUrl, null, null);

218         BufferedReader br = new BufferedReader(new InputStreamReader(connection
      .getInputStream()));
            String input;
220         while ((input = br.readLine()) != null) {
              buffer.append(input);
222         }
            br.close();
224       } catch (Exception e) {
            throw new Exception("Cannot execute URL '" + url + "': " + e.getMessage
      (), e);
226       }

228       return buffer.toString();
      }
230
  }
232
  private static class CWBIssue extends IssueAbs implements Issue {
234
      private String status, crid;
236
      public CWBIssue(String CRID) throws Exception {
238     this.crid = CRID;
        this.status = findStatus(this.crid);
240     System.out.println("Status : " + status);
      }
242
      @Override
244   public String getStatus() {
        return this.status;
246   }

248   @Override
      public String getBrowseLink() {
250     return this.crid;
      }
252
      @Override
254   public boolean isOpen() {
        return "Open".equals(status) || "Active".equals(status);
256   }

258   private String findStatus(String crid2) throws Exception {
```

```
//      System.getProperty("user.home")+"//<foldername>//"
      File localFile = new File("C://test//");
      ProdPassAccess access = new ProdPassAccess(localFile);
      Credential credential = access.getCredential("CWB");
      String username = credential.getProperties().getProperty("user");
      String password = credential.getProperties().getProperty("password");

      String strUrl = "https://css.wdf.sap.corp/sap/bc/bsp/spn/jcwb_api_extern/get_correction_requests?pointer=" + crid2;
      HttpsURLConnection con = sendRequest(strUrl, username, password);
      int status = con.getResponseCode();

      if (status == 200) {
        DocumentBuilderFactory builderFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = builderFactory.newDocumentBuilder();
        builder.setEntityResolver(new EntityResolver() {

          @Override
          public InputSource resolveEntity(String publicId, String systemId) throws SAXException, IOException {
            return new InputSource(new StringReader(""));
          }
        });

        InputStream in = con.getInputStream();
        Document xmlDocument = builder.parse(in);
        in.close();
        XPathFactory xpf = XPathFactory.newInstance();
        XPath xpath = xpf.newXPath();
        NodeList nodeList = (NodeList) xpath.compile("//status").evaluate(xmlDocument, XPathConstants.NODESET);

        return nodeList.item(0).getFirstChild().getNodeValue();

      } else {
        Reader reader = new InputStreamReader(status < 400 ? con.getInputStream() : ((HttpURLConnection) con).getErrorStream());

        System.err.print("ERROR: ");
        while (true) {
          int ch = reader.read();
          if (ch == -1) {
            System.err.println();
            break;
          }
          System.err.print((char) ch);
        }
        return null;
```

```
      }
304
    }
306
  }
308
  private interface Issue {
310   public String getStatus();

312   public String getBrowseLink();

314   public boolean isOpen();

316 }
}
318
abstract class IssueAbs {
320
  private void disableCertificateValidation() throws NoSuchAlgorithmException,
    KeyManagementException {
322   // Create a trust manager that does not validate certificate chains
    TrustManager[] trustAllCerts = new TrustManager[] { new X509TrustManager()
    {
324     public X509Certificate[] getAcceptedIssuers() {
        return null;
326     }

328     public void checkClientTrusted(X509Certificate[] certs, String authType)
    {
      }
330
      public void checkServerTrusted(X509Certificate[] certs, String authType)
    {
332   }
    } };
334
    // Install the all-trusting trust manager
336 SSLContext sc = SSLContext.getInstance("TLS");
    sc.init(null, trustAllCerts, new SecureRandom());
338 HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
  }
340
  public HttpsURLConnection sendRequest(String strUrl, String username, String
    password) throws KeyManagementException, NoSuchAlgorithmException,
342   MalformedURLException, IOException {

344 disableCertificateValidation();

346 URL url = new URL(strUrl);
```

```
    HttpsURLConnection httpsConnexion = (HttpsURLConnection) url.openConnection
    ();

    if (username != null && password != null) {
      String authStr = username + ":" + password;
      String authEncoded = Base64.encodeBase64String(authStr.getBytes());
      httpsConnexion.setRequestProperty("Authorization", "Basic " + authEncoded
    );
    }

    httpsConnexion.setRequestMethod("GET");

    return httpsConnexion;
  }
}
```

Index

Glossaire

API Signifie Application Programmable Interface. 43, 46

classe Une classe, en programmation orientée objet, déclare un ensemble d'attributs et de méthodes communs à un ensemble d'objets. 20, 21, 24

client lourd Un client lourd est un logiciel que l'on installe sur un machnie physique. 35

CMS Le . 21–23

Eclipse Eclipse est un IDE open source de développement Java. 24, 57

framework Un Framework est un environnement de développement comprenant une suite d'outils nécessaires au développement comme par exemple un IDE, des codes sources, des conventions de nommages, ... 19–25, 33, 111

interface Une interface définit le comportement d'une classe, concrètement une interface définit un ensemble de méthodes que doit nécéssairement implémenter une classe. 28

JAR Acronyme de Java ARchive. 45

Java Java est un langage de programmation orientée objet (sans être un langage purement objet, Java utilise les types primitifs int, char, float, etc.) développé par Oracle. 19–21, 23, 26, 45, 46, 58–60, 63, 105

JDK Acronyme de Java Development Kit. 53

méthode Une méthode est un comportement propre à une classe. 24, 25

Maven Outil d'automatisation du processus de production logiciel, en l'occurrence la gestion des sources. 53, 54

Netbeans Netbeans est un environnement de développement intégré. 57

plan de test Un plan de test (ou testplan) est une collection de suite de test. 26, 28, 32

queryspec C'est une spécification de requête au format XML propre à Business Objects. 31, 32, 35

SDK Acronyme de Software development kit. 29, 32

software tester Ingénieur écrivant, entre autre, des tests automatiques pour s'assurer, d'une part, que les fonctionnalités correspondent bien aux spécifications, et d'autre part, permettre aux régressions d'être traitées rapidement. 27

ST Acronyme de Software tester. 1

suite de tests Une test suite est l'ensemble des tests effectués lors de son exécution. 26, 32

testcase C'est un cas de test, correspond à l'implémentation d'une classe de test. 26, 28, 31

web service Un web service est une méthode que l'on appel grâce à une URL. 45, 46

Workflow Un workflow est une succession d'action permettant de partir d'un état A à un état B du logiciel. 28, 30

Table des figures

Table des matières

Abstract

This report focuses on my work for the duration of my internship. This content is divided into four areas. The first part provides information on the firm in which I worked and the following three parts describe each of my assignments. The common point of all is the Java development but for different purposes.
The first assignment was Java test development based on an internal Framework. It was a very interesting part because of the wealth of expertise and experience found in the project which I integrated. During this period I accumulated a large amount of knowledge, concerning the Framework itself or the Java.
The second assignment was complement an existing Java class to adapt its behavior to a new used tool. It was a hard part due to the fact there are many studies to be done for relatively little Java code. I had to integrate my Java implementation into an already existing code and I was very formative.
The third assignment, and also the most interesting and formative, was the Jenkins plug-in development. I studied Jenkins and the way to develop a plug-in for a long time. Next, I had to gradually had new features for the purpose of obtain the expected plugin. The other thing was the human side and the teamwork. Indeed I reported weekly the project progresses and work in pair.
My employer was SAP France, firstly I worked in Levallois-Perret, next to Paris, from July to mid-February. Next, SAP moved towards to a larger building, still in the same town. I worked there until end of September.
It was with a great pleasure to spent my time in there. I learned many things during this internship period, I was able to use new technologies and unknown tools.

Keywords

Software tester
JavaEE
Agile
Jenkins
Plugin