Sommaire

Chapitre 1

Software tester

1.1 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 autre d'un code. Il existe de nombreux niveaux de test : unitaire, fonctionnel, intégration, performance, ... Les tests sont très importants dans un projet logiciel, ils permettent de 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].

1.1.1 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 possible 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 aux 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 se stabilités permet 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. Dans système d'exploitation à un autre, les comportements sont-ils les mêmes ? Les fonctionnalités sont-elles identiques ?

1.2 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 où leurs codes sont hébergés sur le gestionnaire de code source Perforce.

Après chaque livraison de code effectuée, le logiciel est compilé par Jenkins, son utilisation étant principalement de vérifier l'état de la compilation après chaque livraison. Et, de plus, le logiciel est compilé quotidiennement par ASTEC dont les résultats sont automatiquement envoyés aux personnes concernées (cf. annexe B page 27 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 est extrait 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 à 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 1.1 page 4. 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 fonctionnalié. Mon intégration dans ce processus de développement m'a appris beaucoup, non seulement quant à la gestion d'une anomalie logicielle, mais surtout quant à la valeur est à l'utilisation du test qui peut-être implémenter 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.

1.3 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.

1.4 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 1er jours à SAP se sont déroulés de la manière suivante :

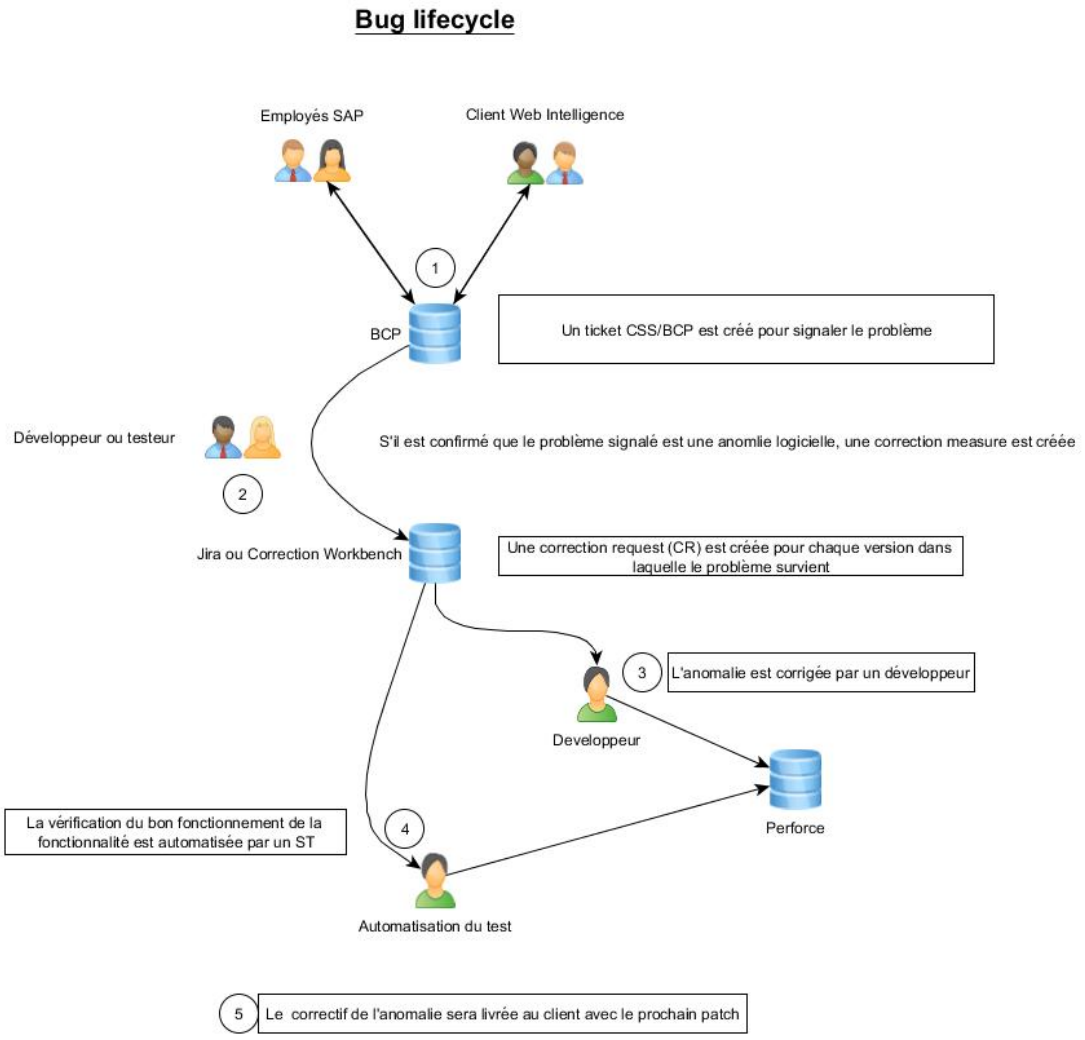


FIGURE 1.1 – Le process de test

— Présentation à l'équipe et visite des locaux
— 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 1ère 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 1er tests.

1.5 Travail réalisé

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 posés 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.

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

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

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 6).

FIGURE 1.2 – Document sur lequel j'ai basé mes première expérimentations

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 exitante. J'ai donc chercher un code dont la structure me permettait de manipuler un document Web Intelligence déjà existant. Il en existe beaucoup de types 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 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 reçevais en masse.

J'avais pris le soin de coller mon document Web Intelligence précisément dans le dos-

sier qu'il fallait mais pourtant, les messages d'erreur me disait que le document était introuvable. En investiguant 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'allais 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 compilais, 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'avais 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 :

```
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.

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 I page 47 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 utilisés 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 dossier 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 retrouvé 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 c'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.

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 considérai que toute modification applicable depuis l'interface graphique pouvait être reproduite depuis le test. J'avais tord. 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 simlple 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 paraîssaient 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 heirarchy », me permettant d'avoir le détail de tous les emplacements où cette classe est instanciée. De cette manière j'ai rapidement trouvé des exemple de codes me permettant d'en déduire les quelques informations qui m'étaient nécessaires.

J'ai comprise 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 :

```
1 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 comment ce tableau a été appelé ? 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 contenu dans le corps du rapport et d'en afficher les propriétés. Cela me permettais, non seulement, d'obtenir son nom, son identifiant et tout ses paramètres, mais aussi, d'obtenir son type. Au fur est à mesure du temps et que j'implémentais des tests j'ai découvert beaucoup de types différents, leurs imbriquations, 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 :

```
1 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 fais plusieurs essais avant de comprendre son fonctionnement. Cette Méthode prends 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 1.3.

1.6 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.

1.6.1 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 des 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 ! Voyons ceci plus en détail.

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 implémenter que quelques-uns, l'intérêt en terme de difficulté et de choses à

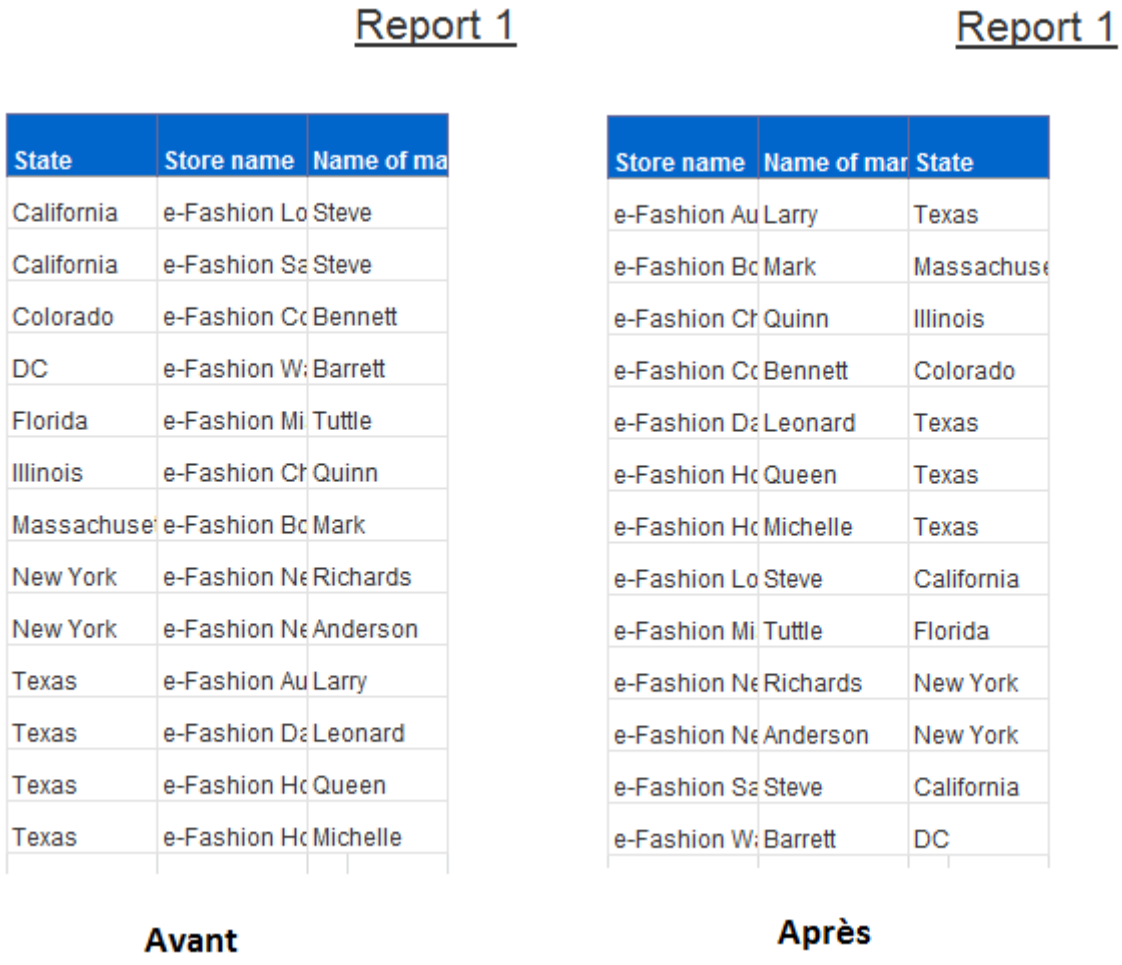


FIGURE 1.3 – Allure du tableau avant et après l'exécution du code

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 est 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 :

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

import model.filters.Mode;
import model.filters.Severity;
import model.filters.Type;
import model.filters.testplan.Suite;
import model.filters.testplan.TestPlanType;
import model.filters.testplan.features.Features_REBEAN;

import org.junit.Test;

import tests.exported.annotations.BOTest;
import tests.exported.annotations.BOTestPlan;
import extensions.toolbox.WIStaticTestPlanDefinition;

@BOTestPlan(Type = TestPlanType.FEATURE_VERIFICATION, Suite = Suite.AURORA,
    Feat = Features_REBEAN.WI)
public class CM_{cmNumber} extends WIStaticTestPlanDefinition{
  private static String _scriptID = "CM_{cmNumber}_{short
text}";
  public CM_{cmNumber} () {
    super(_scriptID);
  }

  @Test
  @BOTest(Objective = "Test CM_{cmNumber}_{short
text}' functionality", Severity = Severity.CRITICAL, Author = "", Mode = Mode.
    PROD, Type = Type.FUNCTIONAL)
  public void CM_{cmNumber}_{short text}" () {
    logNumericResults(launchTC(getTestcase("CM_{cmNumber}_{short text}")));
  }
}
```

J'ai été étonné, la première fois que j'ai compiler un test de ce type, 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 1.4 page 14), 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 copier 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.

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

```
<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 1.4 – Contenu du script.xml pour générer une référence

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 à 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 grande 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, étapes rendu 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 même 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 :

```java
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 :

```java
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
  }
}
```

1.6.2 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 effectué au niveau Software development kit (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 d'arriver à mes fins. De sorte qu'en règle générale je n'arrivai 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 aller essayer sur Web Intelligence le Workflow qui faisait survenir l'anomalie. Toutes les informations que l'on a sur celle-ci se trouvent sur JCWB (voir figure 1.5 page 16) 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 le deuxièmement pour valider que celle-ci a été correctement corrigée.

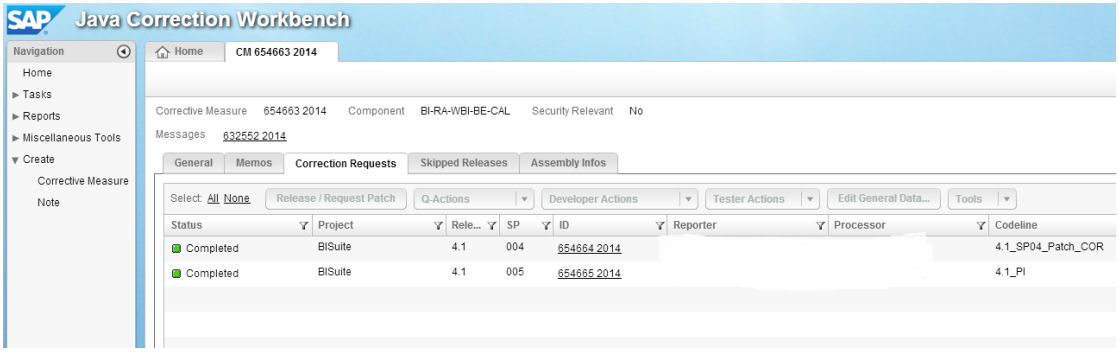


FIGURE 1.5 – Écran de JCWB propre à une CM

Cette partie préalable à l'implémentation du test automatique m'a permi 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 de puis l'interface graphique, je devais étudier les correctifs appliqués pour orienter ma logique d'implémentation du testcase. Cela a été très formateur car les différents codes que j'étudiais avaient été implémenter 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 vu de la logique de la programmation orientée objet que de la complexité des procédures utilisées. De plus, en observant les modifications faitent 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 1.6 page 17). Ce qui peut-être, entre autre, une condition manquante, un problème d'identifiant ou une mémoire saturée.

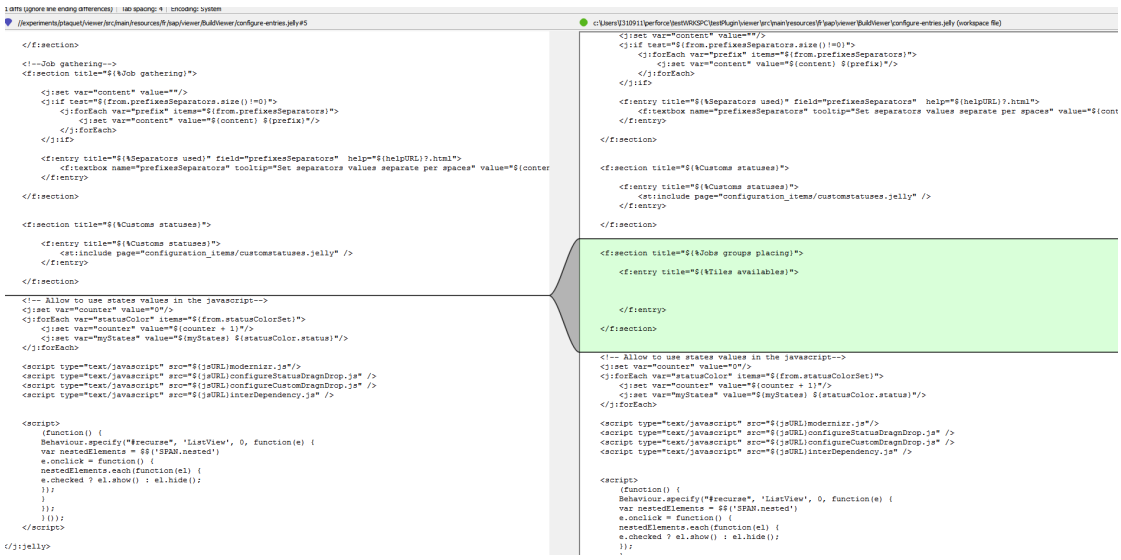


FIGURE 1.6 – Écran de comparaison des 2 versions d'un même fichier (avant et après correctif)

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 configuration 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 à été de ne pas me dirigé plus tôt vers la personne qui aurait pu me conseiller, cela m'aurait épargné plusieurs heures d'effort 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 coquile 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 1.7 page 18, chaque test automatique est intégré à un plan de test, lui-même intégré à une suite de tests.

FIGURE 1.7 – Diagramme UML des suites de tests

Les fichiers essentiels au bon fonctionnement du test sont décrit ci-dessous. Il n'était pas évident, au début, de compléter correctement tous ces documents car chacun ayant 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 coquile 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 1.8 page 20. Et, de la même manière, la figure 1.9 page 21 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 arriver plusieurs fois de générer des fichiers de références à partir de la mauvaise version, ce qui avais pour effet de faire échouer mes tests sans que je puisse, rapidement, savoir pourquoi.

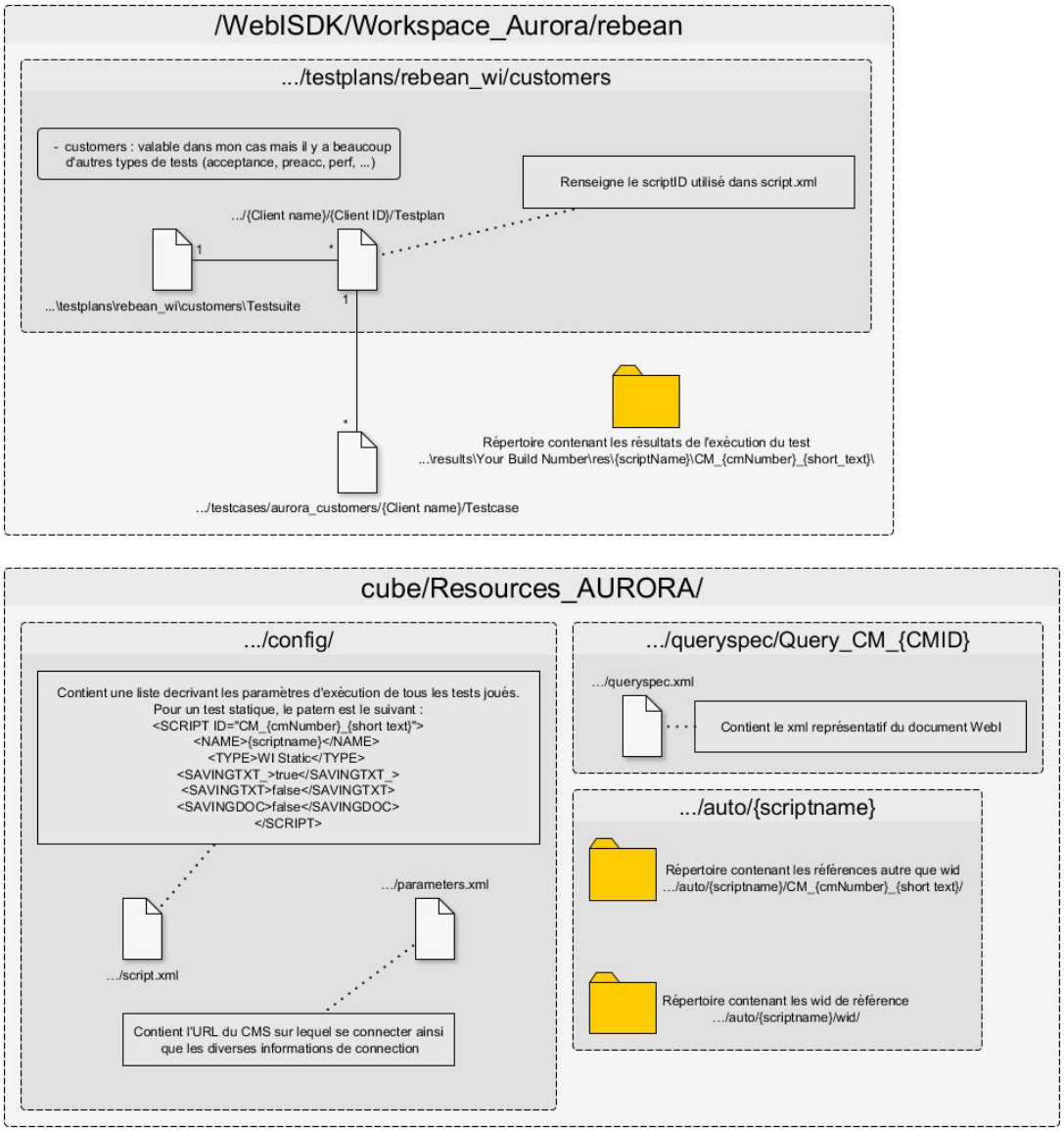


FIGURE 1.8 – Diagramme représentant les différents éléments qui compose la coquille vide d'un test dynamique

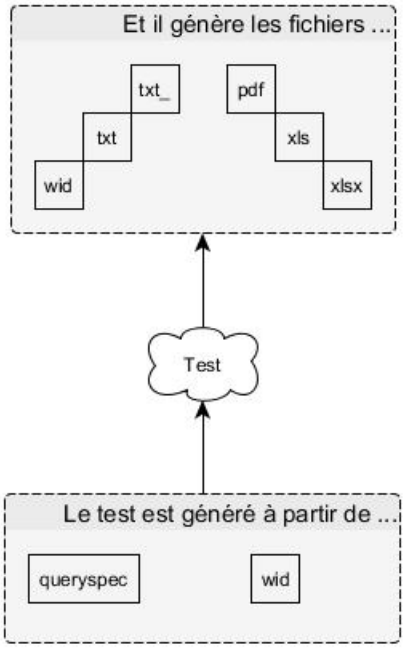


FIGURE 1.9 – Les fichiers utilisés ou générés par le test

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 propiétés du fichier, il y a son nom complet (différent du nom dans WebI) avec son arborescence à partir du dossier Input (figure 1.10 page 22). 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 fais 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.

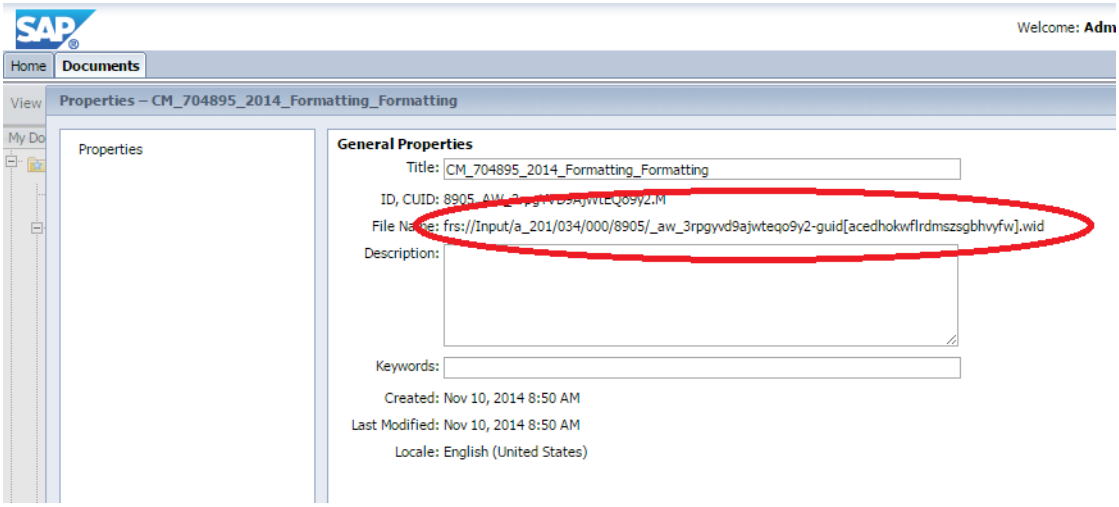


FIGURE 1.10 – Capture de l'écran de propriété d'un document WebI

1.7 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 une fonctionnalité ne fonctionnant plus du tout. 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.

1.7.1 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.

1.7.2 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 que si j'avais implémenté uniquement l'appel à tester, j'aurais pu implémenter ce test en quelques heures seulement !

2. cf. annexe C page 31 pour la liste complète des tests implémentés

1.7.3 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 testeurs, développeurs et par d'autres. J'étais donc intégré à une grande équipe, leurs travaux ayant des répercussion 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 à informé tout le monde de mon erreur de manipulation (illustrée annexe F page F) !

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.

Annexe A

Les niveaux de test

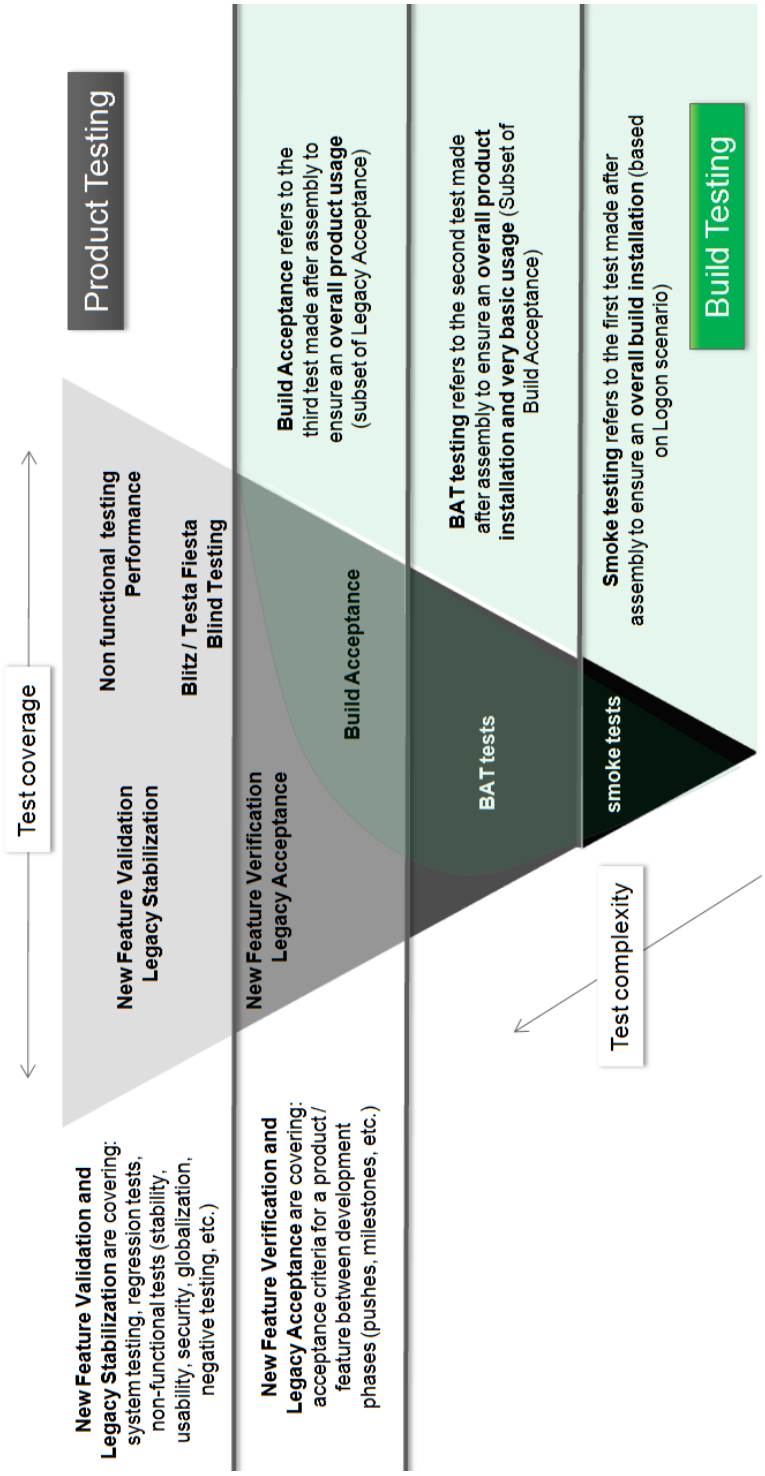


FIGURE A.1 – Les différents niveaux de test

Annexe B

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 C

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 D

Les bases pour comprendre Jenkins

cf annexe différence entre job et projet http ://jenkins-ci.361315.n4.nabble.com/Is-it-called-quot-Project-quot-or-is-it-called-quot-Job-quot-td4628610.html

cf annexe status JUnit vs status Jenkins

- https ://wiki.jenkins-ci.org/display/JENKINS/Terminology
!!!! La base de l'implémentation d'un plugin
https ://wiki.jenkins-ci.org/display/JENKINS/Extend+Jenkins
https ://wiki.jenkins-ci.org/display/JENKINS/Plugin+tutorial

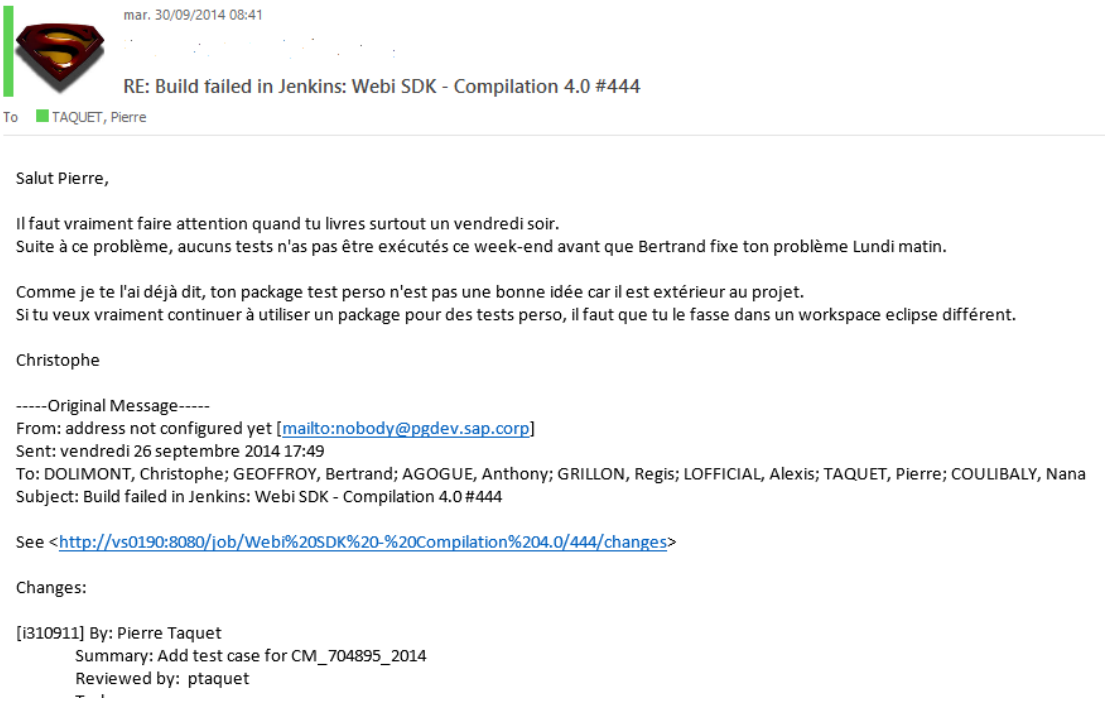
Annexe E

Maven

C'est quoi ?
Concrètement ça sert à quoi ?
Comment ça s'installe ?
Exemple d'utilisation : génération de skeleton du plugin Jenkins

Annexe F

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

Annexe G

Maven

explication des paramttreds -d

Annexe H

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

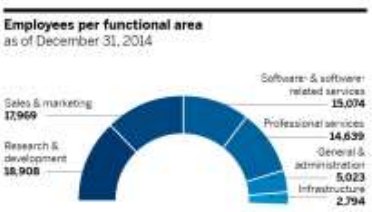
Leading Mobility Vendor

- 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

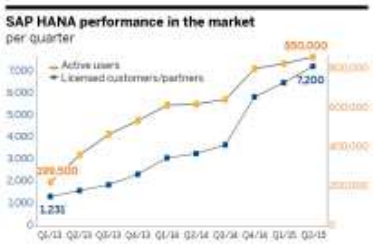
SAP's End-to-End Solutions

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 I

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

```
[0]               -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]
[0]                       -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]
[0]                   -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]
[0]               -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]
[0]                       -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]
[0]                   -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]
[0]                       -Try to find InfoObjects with name = "BEACH" and with
     type = "Universe"   :: OK
[<]                       -executeQuery [START]
[0]                           -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]
[0]                       -1 InfoObject(s) found   :: OK
```

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


%%%%%%%%%%%%%%%%%%%%%%%% Pendant la methode StopTestcaseStepWithAction()


[<]                 -applyFormat [START]
[>]                 -applyFormat [END]
[<]                 -verifyWithRef - PageMode Listing [START]
[<]                     -prepareResFiles [START]
[<]                         -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
[>]                         -createPath [END]
[<]                         -checkSupportedViews [START]
[>]                         -checkSupportedViews [END]
[<]                         -saveAsXMLaurora [START]
[<]                             -saveReportAsCharacterViewAurora [START]
[O]                                 -set LISTING for pagination mode   :: OK
[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
[>]                       -compareTXTFiles [END]
[>]                   -comparePreparedFiles [END]
[>]               -verifyWithRef - PageMode Listing [END]
[>]           -Testcase step: createdoc [END]
[<]           -closeDocument [START]
[>]           -closeDocument [END]
[<]           -closeAll [START]
[>]           -closeAll [END]
[<]           -logoff [START]
[O]               -close report engines   :: OK
[>]           -logoff [END]
[<]           -logoff [START]
[O]               -Close session   :: OK
[>]           -logoff [END]
[O]           -Total Time Spent for Testcase TestcaseCreateANewDocument =
    22.083 second(s)   :: OK
[O]           -Finished at Mon Jul 28 14:32:42 CEST 2014   :: OK
[>]       - 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'
Passed.TestcaseCreateANewDocument=
```

Glossaire

classe Une classe, en programmation orientée objet, déclare un ensemble d'attributs et de méthodes communs à un ensemble d'objets. 6–8, 10

client lourd Un client lourd est un logiciel que l'on installe sur un machnie physique. 21

CMS Le . 7–9

Eclipse Eclipse est un IDE open source de développement Java. 10

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, ... 5, 8–10, 19, 27

interface Une interface définit le comportement d'une classe, concrètement une interface définit un ensemble de méthodes que doit nécéssairement implémenter une classe. 14

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. 5–7, 9, 11

méthode Une méthode est un comportement propre à une classe. 11

plan de test Un plan de test (ou testplan) est une collection de suite de test. 12, 15, 18

queryspec C'est une spécification de requête au format XML propre à Business Objects. 18, 19, 21

SDK Software development kit. 16, 18

software tester Ingénieur écrivant, entre autre, des tests automatiques pour s'assurer, d'une part, que les fonctionnalités correspondent aux fonctionnalités, et d'autre part, permet aux régressions d'être traitées rapidement. 13

suite de tests Une test suite est l'ensemble des tests effectués lors de son exécution. 11, 18

testcase C'est un cas de test, correspond à l'implémentation d'une classe de test. 11, 15, 17

Workflow Un workflow est une succession d'action permettant de partir d'un état A à un état B du logiciel. 14, 16, 17

Table des matières