

**UNIVERSITE MOHAMMED V--AGDAL
ECOLE MOHAMMADIA D'INGENIEURS**



Filière: Génie Informatique

Option: IQL-SI

Mémoire de Projet de Fin d'Etudes

N° 2012/05

**REALISATION BRIQUES FONCTIONNELLES
ACTIONS FINANCE CONSEILS**

Réalisé par:

M. Abdeljalil AITIKKENE

M. Adil JIRARI

Dirigé par :

M. Mohammed KHALIDI IDRISSE

M. Vincent GROUX

Année 2011/2012

UNIVERSITE MOHAMMED V--AGDAL
ECOLE MOHAMMADIA D'INGENIEURS
DEPARTEMENT GENIE INFORMATIQUE
RABAT-MAROC
EMI/INF/2012



PROJET DE FIN D'ETUDES

Présenté Par

M. JIRARI Adil & M. AITIKKENE Abdeljalil

REALISATION BRIQUES FONCTIONNELLES ACTIONS FINANCE CONSEILS

Soutenu le : 04/06/2012

Membres de Jury

Mme.F.Z.BELOUADHA	Présidente
M.D.EL GHANAMI	Rapporteur
M.M.KHALIDI IDRISI	Encadrant EMI
M. V. GROUX	Encadrant Norsys

Année Universitaire: 2011-2012

Dédicaces

A nos chers parents pour leur amour, leurs sacrifices et leur affection;

A nos chères sœurs et à nos chers frères pour leur encouragement continu;

A toute notre grande famille;

A tous nos amis pour leur soutien;

A tous ceux qui nous sont chers;

Qu'ils trouvent ici l'expression de notre amour et gratitude

Remerciements :

Nous souhaitons remercier dans un premier temps, notre encadrant interne M. Mohammed KHALIDI IDRISSE, professeur à l'Ecole Mohammadia d'Ingénieurs, pour avoir assuré la partie théorique de notre formation, pour son aide, ses conseils et son encadrement tout au long de notre stage PFE.

Nous tenons à remercier tout particulièrement et à témoigner toute notre reconnaissance aux personnes suivantes, pour l'expérience enrichissante et pleine d'intérêt qu'elles nous ont fait vivre au sein de la société Norsys :

M. Vincent GROUX, directeur de notre projet, responsable de la formation TechnoMaker chez Norsys, et notre encadrant externe, pour nous avoir fait partager toute son expérience et ses compétences ; pour le temps qu'il nous a consacré tout au long de cette période de stage, sachant répondre à toutes nos interrogations ; sans oublier sa participation à la réalisation de ce mémoire.

M. Karim TAMMAR, chef d'équipe chez Norsys Afrique, et notre responsable de projet pour sa coopération professionnelle et pour nous avoir guidés et conseillés au cours de nos différentes missions.

Nos vifs remerciements s'adressent à Mme. F.Z.BELOUADHA et M. D.EL GHANAMI qui ont accepté de juger notre travail.

Enfin nous tenons à remercier aussi nos chers professeurs et camarades du département Génie Informatique à l'EMI pour ces trois années inoubliables.

Résumé :

Dans le cadre de l'informatisation des cabinets comptables, le groupe NORSYS a pris en charge la réalisation des briques fonctionnelles du SI de la société Actions-Finance-Conseils.

Cette application a pour objectif d'assurer une gestion flexible et souple des clients du cabinet AFC, en automatisant certaines tâches répétitives, et en centralisant l'ensemble d'informations du cabinet dans une seule base de données. Ce projet comporte trois modules fondamentaux à savoir : la gestion des clients, la gestion des ressources humaines et le suivi des documents comptables.

Pour répondre à ces besoins, nous avons appliqué une méthode Agile Scrum, qui est basée sur la collaboration entre le ScrumMaster, l'équipe de développement et le Product Owner.

Nous avons mené la conception et la réalisation de l'application AFC, en faisant une modélisation UML, en adoptant la plate-forme Java Enterprise Edition (JEE), et en utilisant une architecture trois-tiers tout en travaillant avec une plateforme d'intégration continue.

Abstract

As part of the computerization of accounting firms, NORSYS group has supported the realization of functional bricks of the company Actions-Finance-Conseils.

This application is designed to provide a flexible and responsive management for cabinet AFC customers, by automating repetitive tasks, and centralizing the information in a single database. This project involves three basic modules: the customer management, human resources management and accounting documents tracking.

To meet these needs, we practice the Scrum agile method; a method based on the collaboration between the development team the Scrum Master and the Product Owner.

We conducted the design and implementation of the AFC application, by realizing a UML modeling, adopting the Platform Java Enterprise Edition (JEE), and using three-tier architecture with a continuous integration platform.

ملخص

في إطار حوسبة شركات المحاسبة، أخذت مجموعة NORSY المبادرة لإنشاء نظام معلوماتي للشركة Actions Finance Conseils.

هذا التطبيق هو مصمم لتوفير إدارة مرنة ومتجاوبة مع عملاء الشركة AFC، وذلك بتنفيذ المهام المتكررة بطريقة أوتوماتيكية، وتركيز معلومات هذه الإدارة في قاعدة بيانات وحيدة. ويشمل هذا المشروع ثلاث وحدات أساسية هي: إدارة العملاء وإدارة الموارد البشرية ومراقبة وثائق المحاسبة.

لتلبية هذه الاحتياجات اعتمدنا نهج إدارة المشاريع Agile(Scrum) وهو نهج يعتمد على المشاركة البناءة و التعاون داخل فريق المبرمجين و أيضا بين ScrumMaster و ProductOwner أي الزبون AFC.

ومن اجل تصميم وتنفيذ مشروع AFC، قمنا باعتماد نموذج UML واستعمال تقنية (JEE) Java Enterprise Edition، واستخدام تصميم الطبقات الثلاث (Trois-tiers) بالارتكاز على طريقة التكامل المتواصل (Intégration continue).

Table des matières

Introduction	1
Chapitre 1 :	2
Présentation générale du projet	2
1. Présentation de l'organisme d'accueil :	3
2. Présentation du projet AFC :	3
2.1. Présentation du client Actions Finance Conseils :	3
2.2. Présentation du projet AFC :	4
3. Conclusion :	4
Chapitre 2 :	5
Etude fonctionnelle du projet	5
1. Exigences fonctionnelles :	6
1.1 Module Ressources humaines :	6
1.2 Module Client :	6
1.3 Module suivi des documents :	7
1.3.1 Suivi des déclarations :	7
1.3.2 Suivi des opérations comptables :	8
1.3.3 Suivi des dossiers juridiques :	8
2. Exigences non fonctionnelles :	9
2.1 IHM :	9
2.2 Sécurité :	9
2.3 Fiabilité :	9
3. Conclusion :	9
Chapitre 3 :	10
Analyse et Conception.....	10
1. Méthodologie de travail :	11
1.1 Méthodes Agiles :	11
1.2 Méthode Scrum :	11
1.2.1 Présentation :	11
1.2.2 Organisation :	12
1.2.3 Les avantages de cette méthode :	12
1.3 Cycle de vie.....	13
1.3.1 Description	13

1.4	Planification.....	14
1.5	Outil de gestion de projet(IceScrum):	14
1.5.1	Présentation :	14
1.5.2	Utilisation :	15
1.5.3	Backlog de produit initial :.....	16
1.5.4	Création de sprints dans le plan de release :	16
1.5.5	Plan de Sprint :	17
2.	Analyse et conception :	17
2.1	Présentation :	17
2.2	Modélisation :.....	18
2.2.1	Les acteurs du système :	18
2.2.2	Cas d'utilisation par acteur :.....	18
2.2.3	Analyse du cas d'utilisation : Gestion des clients	19
2.2.4	Analyse du cas d'utilisation : Gestion utilisateur	23
2.2.5	Analyse du cas d'utilisation : Suivi des dossiers	24
2.2.6	Diagramme de cas d'utilisation global :	27
2.2.7	Diagramme de classes :	28
3.	Conclusion :	29
Chapitre 4 :		30
Etude technique du projet		30
1.	Problématique :	31
2.	Démarche technique :	32
2.1	Implémentation:.....	32
2.1.1	Problématique.....	32
2.1.2	Solution proposée : TDD (Test Driven Development) :	32
2.2	Collaboration:	33
2.2.1	Problématique :.....	33
2.2.2	Solution proposée : GIT :	33
2.3	Intégration continue :.....	34
2.3.1	Problématique :.....	34
2.3.2	Solution proposée : Jenkins.....	34
2.4	Outils de Qualification :	35
2.4.1	CheckStyle :	35
2.4.2	Sonar :	36

2.5	Outils de développement :	37
2.5.1	Springsource tool suite (STS) :	37
2.5.2	MySQL :	37
3.	Conclusion :	38
Chapitre 5 :		39
Réalisation :		39
1.	Architecture logicielle :	40
1.1	Présentation :	40
1.2	Justification du choix de l'architecture :	40
1.2.1	Architecture JEE :	40
1.2.2	Architecture Trois-tiers :	40
1.3	Démarche de développement :	41
1.1.1	Couche DAO :	41
1.1.2	Couche Service :	43
1.1.3	Couche Présentation :	44
2.	Réalisation :	46
2.1	Présentation :	46
2.2	Authentification :	47
2.3	Gestion client :	47
2.4	Gestion des exercices :	48
3.	Conclusion :	49
CONCLUSION GENERALE ET PERSPECTIVES :		50

Table des figures

FIGURE 1 ARCHITECTURE FONCTIONNELLE DU MODULE SUIVI DE DOCUMENT	7
FIGURE 2 CYCLE DE VIE SCRUM	13
FIGURE 3: BAC A SABLE.....	16
FIGURE 4: BACKLOG DES USER STORIES	16
FIGURE 5 : PLAN DE REALEASE R1	16
FIGURE 6 : PLAN DE SPRINT	17
FIGURE 7 : DIAGRAMME CAS D'UTILISATION GESTION CLIENT	20
FIGURE 8 : DIAGRAMME D'ACTIVITE CREATION AUTOMATIQUE DU PREMIER EXERCICE	22
FIGURE 9 : DIAGRAMME D'ACTIVITE CREATION AUTOMATIQUE DES PROCHAINS EXERCICES.....	22
FIGURE 10 : DIAGRAMME DE CAS D'UTILISATION GESTION UTILISATEUR.....	23
FIGURE 11 : DIAGRAMME DE SEQUENCE GESTION AUTHENTIFICATION.....	24
FIGURE 12 : DIAGRAMME DE CAS D'UTILISATION SUIVI DES DOSSIERS.....	24
FIGURE 13 : DIAGRAMME D'ETAT D'UNE DECLARATION	26
FIGURE 14 : DIAGRAMME D'HISTORISATION DES EXERCICES.....	27
FIGURE 15 : DIAGRAMME DE CAS D'UTILISATION GLOBAL.....	27
FIGURE 16 : DIAGRAMME DE CLASSES	28
FIGURE 17 : ARCHITECTURE TECHNIQUE DU PROJET	31
FIGURE 18 : ETAPES DE LA DEMARCHE TDD	32
FIGURE 19 : MODE DE GESTION GIT CHEZ NORSYS.....	33
FIGURE 20 : JENKINS AFC	34
FIGURE 21 : SONAR.....	36
FIGURE 22 : ARCHITECTURE LOGICIELLE.....	41
FIGURE 23 : ABSTRACT DAO	42
FIGURE 24 : TEST DAO	42
FIGURE 25: HSQL DATA.....	42
FIGURE 26: DOZER BEAN MAPP.....	43
FIGURE 27: TEST SERVICE	43
FIGURE 28 : PATRON MVC	44
FIGURE 29: LAYER SPRING SECURITY	44
FIGURE 30: LA STRUCTURE DE L'APPLICATION AFC PAR TILES.....	45
FIGURE 31: TILES DEFINITION	45
FIGURE 32: FICHER D'INTERNATIONALISATION.....	45
FIGURE 33 : PAGE D'ACCUEIL.....	46
FIGURE 34 : PAGE D'AUTHENTIFICATION	47
FIGURE 35 : LISTE DES CLIENTS ET SON FORMULAIRE D'AJOUT	47
FIGURE 36 : VALIDATOR CLIENT.....	48
FIGURE 37 : LISTE DES DECLARATIONS DE L'EXERCICE COURANT DU CLIENT NORSYS	48
FIGURE 38 : MODIFICATION DES DECLARATIONS DE L'EXERCICE COURANT DU CLIENT NORSYS.....	49
FIGURE 39 : STATISTIQUE SONAR DU PROJET AFC.....	59

TABLEAU 1: PLANNING DU PROJET	14
TABLEAU 2 : ACTEURS DU SYSTEME	18
TABLEAU 3 : CAS D'UTILISATION PAR ACTEUR	18
TABLEAU 4 : FICHE DE CAS D'UTILISATION SUIVI DECLARATION	21
TABLEAU 5: FICHE DE CAS D'UTILISATION GESTION AUTHENTIFICATION	23
TABLEAU 6: FICHE DE CAS D'UTILISATION SUIVI DECLARATION	25

Table des abréviations

Abréviation	Désignation
AFC	Actions Finance Conseils
JEE	Java Entreprise Edition
SSII	Société de Service en Ingénierie Informatique
MVC	Model View Controller
UC	Use Case
TDD	Test Driven Developement
DAO	Data Access Object
XML	Extensible Markup Language
AJAX	<i>Asynchronous Javascript and XML</i>
STS	SpringSource Tool Suite

Introduction

La conjoncture internationale et nationale actuelle oblige les entreprises à améliorer leurs prestations et services afin de satisfaire les exigences du client. De ce fait, le client Actions Finance Conseils « AFC » a décidé d'informatiser ses processus métiers afin de minimiser le temps de traitement des dossiers de ses clients, il avait besoin donc de confier cette tâche à une SSII informatique pour bien mener ce travail.

Faisant partie de ces entreprises, le groupe Norsys représenté par sa filiale Norsys Afrique à Marrakech a décidé de prendre en charge la réalisation de ce projet surtout que le cabinet AFC se trouve lui aussi à Marrakech, ce qui va favoriser l'utilisation d'une méthodologie Agile afin de satisfaire les besoins du client AFC.

La mise en œuvre de cette application va apporter au client Actions Finance Conseils :

- Une automatisation de la création des différents exercices d'un client ;
- Une meilleure gestion du suivi des déclarations dans un exercice donné ;
- Une meilleure définition des responsabilités des acteurs afin de faire prendre conscience des droits et devoirs de chacun ;
- Un gain de temps remarquable dans la gestion des différentes tâches d'un client ;
- Une centralisation de ses informations ;
- Une sécurité de ses données comptables.

Dans le premier chapitre de ce rapport, nous allons faire une présentation générale de notre projet, puis dans le deuxième chapitre, nous allons faire une étude fonctionnelle du projet. Pour mener une étude conceptuelle de l'application dans le troisième chapitre. L'étude technique du projet fera l'objet du quatrième chapitre. Finalement, le cinquième chapitre sera consacré à la phase de réalisation du projet.

Chapitre 1 :

Présentation générale du projet

Dans ce chapitre, nous présentons le contexte général du projet. Nous commençons tout d'abord par présenter l'organisme d'accueil. Nous présentons ensuite le client Actions Finance Conseils pour enfin présenter l'objectif de notre projet.

1. Présentation de l'organisme d'accueil :

Créée en 1994, le groupe Norsys est une société de Conseil et d'Ingénierie informatique, dirigée par Sylvain Breuzard (président du CJD - Centre des Jeunes Dirigeants d'entreprise - de 2002 à 2004). Le groupe accompagne ses clients à la fois dans des missions de conseil auprès des différentes directions d'entreprise mais aussi dans la réalisation de projets informatiques utilisant les technologies émergentes.

Le groupe Norsys emploie aujourd'hui 200 salariés sur quatre sites (Lille, Paris, Lyon et au Maroc). Avec un chiffre d'affaires de 22 M€ (2012), en hausse de 10% par rapport à 2010, il a misé sur trois composantes essentielles pour construire sa réussite :



- Ses travaux de recherche et son expertise métier à travers son université d'entreprise,
- La compétence et des valeurs de responsabilité, de loyauté et de respect incarnées de ses collaborateurs,
- Et une politique en responsabilité sociale très engagée, tant en interne qu'à l'extérieur de la société (expérimentation du CV anonyme, partenariats associatifs pour l'aide à l'emploi et la formation de personnes d'origine étrangère).

Depuis 2005, chaque année, 70% des salariés du groupe suivent une formation et 40 nouveaux salariés sont recrutés.

2. Présentation du projet AFC :

2.1. Présentation du client Actions Finance Conseils :

Le client Actions Finance Conseils est un cabinet d'expertise comptable qui se tient aux côtés de ses clients pour les aider dans la gestion des différentes opérations fiduciaires, fiscaux, conseils juridiques, comptabilité, conseils d'entreprises ..., quelle que soit sa forme ou sa taille. Le cabinet adapte l'organigramme général ci-dessous pour organiser les rôles de chaque employé d'AFC :

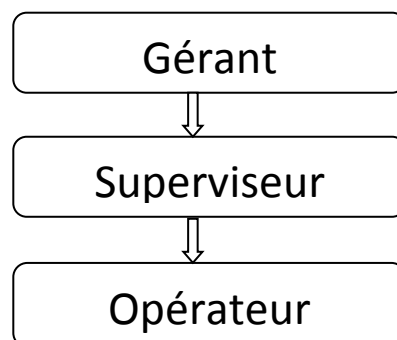


Figure 1: ORGANIGRAMME GENERAL DU CABINET AFC

Chapitre 1 : Présentation générale du projet

Le gérant est le dirigeant du cabinet, il peut y avoir plusieurs dirigeants dans le cabinet, le superviseur est un opérateur qui peut prendre plusieurs dossiers et superviser un équipe d'opérateurs.

L'aspect Workflow entre ses différents intervenants - gérant, superviseur ou opérateur – est donc primordial afin de bien mener les activités d'un client de cette société.

2.2. Présentation du projet AFC :

Pour gérer les différentes opérations fiduciaires de ses clients, le cabinet AFC fait actuellement cette gestion sur des supports papiers, ou des fiches Excel, ce qui rend :

- L'ajout d'un client ou sa modification très pénible ;
- La recherche trop lente ;
- L'accès à toutes les informations d'un client une perte de temps ;
- Le suivi de toutes les opérations comptables mises en place par les collaborateurs très complexes et suscitant un temps supplémentaire ;
- La recherche de l'historique des clients une perte de temps.

Le nombre de clients de ce cabinet ne cessant d'augmenter jour après jour, AFC s'est trouvé dans l'obligation de mettre en œuvre une application qui va prendre en charge la réalisation de ses besoins fonctionnels avec une automatisation des tâches répétitives.

Le présent projet nommé «Réalisation briques fonctionnelles Actions Finance Conseils » est une application web qui vise à développer une application JEE contenant toutes les briques fonctionnelles du cabinet AFC.

Cette application facilitera au cabinet AFC la gestion de ses clients et de ses employés, elle va permettre aussi de savoir l'état d'avancement de chaque dossier comptable et permettra un suivi des différents documents à savoir : les déclarations, les journaux, les livres légaux...etc.

Dans cette perspective, nous avons découpé ces briques fonctionnelles en quatre parties :

- Module gestion clientèles.
- Module gestion des ressources humaines.
- Module suivi de documents.

3. Conclusion :

Nous avons défini le contexte général de notre projet ainsi que ses objectifs. Dans le chapitre suivant, nous mènerons une étude fonctionnelle du projet, nous présenterons ensuite la méthodologie adoptée pour concevoir notre application.

Chapitre 2 :

Etude fonctionnelle du projet

Dans ce chapitre, nous faisons une étude fonctionnelle du projet. Nous commençons tout d'abord par décrire les exigences fonctionnelles du projet en indiquant les différents modules de l'application. Ensuite nous indiquons les exigences non fonctionnelles à mettre en place dans le développement de l'application.

1. Exigences fonctionnelles :

1.1 Module Ressources humaines :

L'objectif de ce module est de gérer les données concernant chaque collaborateur à savoir ses informations personnelles.

Un collaborateur est un utilisateur de l'application qui peut modifier son compte et faire les tâches qui lui sont attribuées, par exemple un opérateur est associé à un client (dossier comptable) dont il doit faire le suivi de tous ses documents : déclarations, journaux, ...etc.

C'est le manager qui a la possibilité d'ajouter un utilisateur, ou de le supprimer ; et de lister les utilisateurs selon leurs noms, prénoms, situations familiales, N° CIN, N° CNSS, etc.

- *Gestion des privilèges :*

L'application AFC va comporter plusieurs parties, chacune de ces parties va être accessible par des personnes précises qui auront chacune un rôle :

- Rôle opérateur : c'est le rôle le plus bas de l'application, il a comme tâches principales de faire le traitement des dossiers comptables et de modifier ses informations personnelles.
- Rôle superviseur : il possède les droits du rôle opérateur et il peut contrôler plusieurs dossiers ainsi que faire des observations pour chaque dossier.
- Rôle secrétaire : il intervient lors du suivi des déclarations des clients, il réalise le changement du statut des déclarations de « signé » à « classé et archivé ».
- Rôle gérant : c'est le rôle principal de l'application, il a le droit de faire toutes les tâches possibles dans l'application.

1.2 Module Client :

L'objectif de ce module est de gérer tous les clients de la société Actions Finance Conseils (AFC) depuis l'ajout, la modification, ou l'affichage d'un client jusqu'au suivi de son dossier comptable, de ses exercices et de ses déclarations.

En effet un client correspond à une entreprise qui effectue plusieurs activités dans un ou plusieurs secteurs d'activités. AFC se charge de gérer ses informations fiduciaires en collaborant avec lui à travers des contacts (responsables) pour maintenir à jour l'état de son dossier comptable et de ses déclarations auprès des différents organismes.

Ce module va permettre à l'utilisateur AFC de gérer ses clients, et qui dit client dit activités, exercices, déclarations, responsables, etc. Par conséquent ce module est lui-même réparti en plusieurs sous modules, à titre d'exemples :

- Gestion des activités : permet l'attribution au client des activités qu'il pratique en permettant au gérant d'ajouter de nouvelles activités et de les associer à leurs déclarations.
- Gestion des secteurs d'activités : avant d'ajouter une activité il faut mentionner son secteur, celui-ci peut avoir plusieurs activités.
- Gestion des responsables : pour collaborer avec un client il faut contacter ses responsables qui sont ajoutés comme informations du client dans l'application.
- Gestion des exercices : chaque client doit disposer d'un exercice courant par année fiscale, cet exercice est composé de plusieurs déclarations qui sont identifiées à partir des activités de ce client.

1.3 Module suivi des documents :

L'objectif de ce module est d'avoir un aperçu sur l'état d'avancement de chaque document comptable. Il comprend les fonctionnalités suivantes :

- Le suivi des déclarations.
- Le suivi des opérations comptables.
- Le suivi des dossiers juridiques.

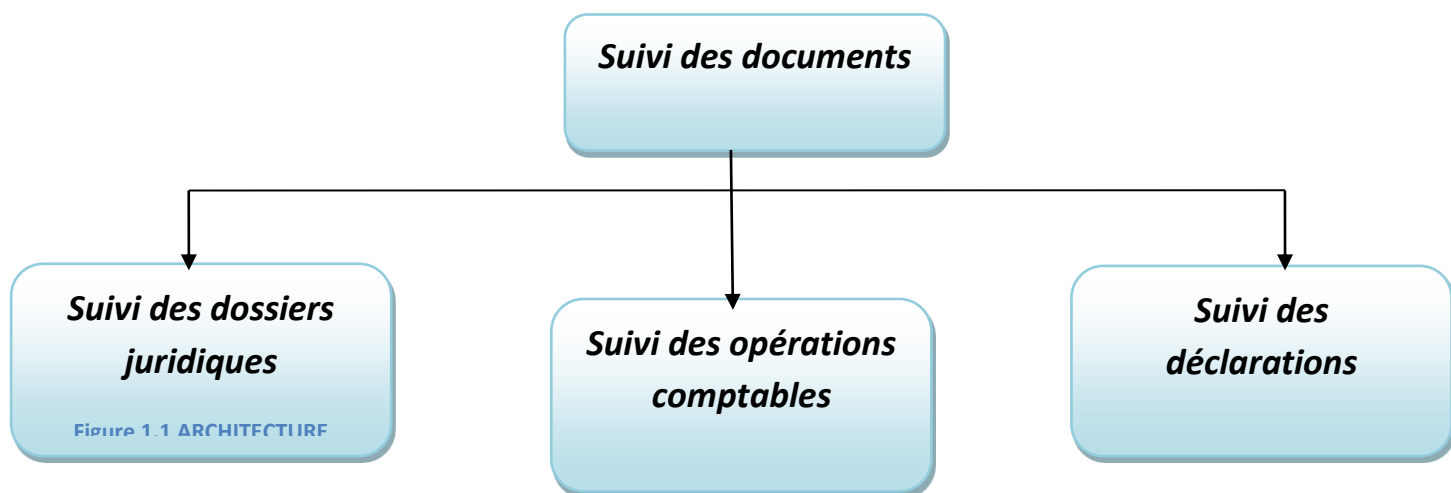


Figure2 : ARCHITECTURE FONCTIONNELLE DU MODULE SUIVI DE DOCUMENT

1.3.1 Suivi des déclarations :

L'objectif est de faire le suivi des déclarations : TVA, CNSS, IR... (Voir le tableau des déclarations en Annexe1 « Liste déclarations»). En effet une déclaration est un état qui est déclenché au moment où le client fournit ses pièces au cabinet pour transmettre ses documents. Les étapes de commencement d'une déclaration sont les suivantes :

- (1) Le client n'a pas fourni les pièces : dans ce cas la déclaration est à l'état « non commencée » et l'application doit alerter le gérant concernant cette situation pour contacter le responsable du client.
- (2) Le client a fourni les pièces partielles : dans cette étape il y a deux possibilités, soit les pièces manquantes peuvent bloquer le processus de déclenchement de la déclaration et dans ce cas retour à l'étape (1) sinon il s'agit de la situation non bloquante et l'opérateur passe la déclaration à l'état « en cours ».
- (3) Le client a fourni toutes les pièces : c'est le processus normal d'une déclaration, dans ce cas la déclaration est à l'état « en cours collaborateur ».

Après avoir reçu les pièces non bloquantes. Une déclaration peut maintenant terminer son processus standard suivant :

- Déclaration « en instance de signature » : pour être signée par le client.
- Déclaration est « signée ».
- Déclaration « en instance de dépôt » : chez la secrétaire.
- Déclaration est « envoyée pour le dépôt » : avec une date de dépôt.
- Déclaration « classée et archivée ».

1.3.2 Suivi des opérations comptables :

Le but de cette fonctionnalité est d'avoir un aperçu sur l'état d'avancement de chaque dossier comptable. En effet les dossiers comptables à éditer et à mettre à jour sont les suivants : **Grand Livre, balance, Journaux auxiliaires, Journaux général.**

Les documents ci-dessus sont classés par dossier client.

Un dossier client se compose d'un ensemble d'exercices. Ces derniers comportent plusieurs journaux. En général l'état (« Fait » ou « non Fait ») de ces documents est modifié par les opérateurs. Le gérant et le superviseur ont la possibilité de consulter et modifier cet état.

Dès qu'un journal est saisi, sa date d'établissement doit être mentionnée. Les journaux correspondant à chaque dossier sont : Achats, ventes, banques, caisse, opérations divers(OD).

1.3.3 Suivi des dossiers juridiques :

Ce suivi a pour objectif de consulter l'état de chaque document juridique :

- Les informations sur l'Assemblée Générale (AG) (« fait » ou « non fait » avec date d'établissement).
- Établissement des Procès-Verbaux (PV) (« fait » ou « non fait », par qui, date d'établissement, date de dépôt).
- Livres légaux (« fait » ou « non fait », date d'établissement).
- Tirage concernant un client pendant un exercice (« fait » ou « non fait », date d'établissement).

2. Exigences non fonctionnelles :

2.1 IHM :

L'application doit satisfaire les besoins de l'utilisateur dans sa vision ergonomique en utilisant des interfaces simples et faciles à manipuler (voir la partie Réalisation).

2.2 Sécurité :

L'architecture utilisée permet une navigation souple et sécurisée au long de la manipulation des différentes fonctionnalités.

2.3 Fiabilité :

L'application doit être fiable en termes de disponibilité totale au cours des traitements des tâches car il y a des opérations automatiques qui se déclenchent chaque jour qui sont primordiales pour le fonctionnement de l'application.

3. Conclusion :

En conclusion, nous avons défini les besoins fonctionnels de notre projet ainsi que les besoins non fonctionnels. Dans le chapitre suivant, nous présenterons la méthodologie adoptée pour concevoir notre application, nous mènerons ensuite une étude conceptuelle du projet.

Chapitre 3 :

Analyse et Conception

Dans ce chapitre, nous faisons une étude conceptuelle du projet. Nous allons présenter l'analyse et la conception réalisés au cours des quatre Sprints effectués, vu qu'on a adopté la méthode Scrum comme démarche de travail.

1. Méthodologie de travail :

1.1 Méthodes Agiles :

Afin de mener à bien notre projet de fin d'études, nous étions censés trouver une méthode de travail qui répondra à nos besoins fonctionnels et techniques. Notre client AFC n'ayant pas d'expérience dans la maîtrise d'ouvrage et la rédaction de cahier de charge, le meilleur choix était d'adopter une méthodologie agile car elle va nous permettre de :

- Concevoir notre application en impliquant au maximum le demandeur (client), ce qui permet une grande réactivité à ses demandes.
- Satisfaire réellement le besoin du client, et non d'un contrat établi préalablement : Les méthodes agiles se veulent plus pragmatiques que les méthodes traditionnelles.

Parmi les différentes méthodes agiles connues nous pouvons mentionner :

- Adaptive software development (ASD)
- Crystal clear
- Dynamic systems development method (DSDM)
- Extreme programming (XP)
- Feature driven development
- Processus Urbanisant les Méthodes Agiles (PUMA)
- Rapid Application Development (RAD)
- Scrum

1.2 Méthode Scrum :

1.2.1 Présentation :

Nous avons choisi la méthode agile Scrum pour concevoir et réaliser ce projet car une telle méthode va nous permettre de développer un logiciel de manière incrémentale en maintenant une liste totalement transparente des demandes d'évolutions ou de corrections à implémenter (backlog). Avec des livraisons très fréquentes, toutes les 4 semaines en général, le client reçoit un logiciel à chaque fois, un logiciel possédant toujours plus de fonctionnalités et en parfait état de fonctionnement. Pour cela, la méthode s'appuie sur des développements itératifs à un rythme constant d'une durée de 2 à 4 semaines. Les évolutions peuvent donc être plus facilement intégrées.

Concrètement, cette méthode nécessite 4 types de réunions :

- **Les réunions quotidiennes** : chaque jour, toute l'équipe se réunit, généralement debout, pendant 15 minutes environ pour répondre aux trois questions suivantes :
 - Qu'ai-je fait hier ?
 - Que vais-je faire aujourd'hui ?

Chapitre 3 : Analyse et conception

- Y a-t-il un obstacle gênant aujourd'hui ?
- **Les réunions de planification** : toute l'équipe se réunit pour décider des fonctionnalités qui vont composer le sprint suivant et mettre à jour la liste générale.
- **Les réunions de revue de travail** : lors de cette réunion, chacun présente ce qu'il a fait pendant la durée du sprint. Une démonstration des nouvelles fonctionnalités ou de présentation de l'architecture est organisée. Il s'agit d'une réunion informelle de 2 heures environ à laquelle participe toute l'équipe.
- **Les réunions de rétrospectives** : à chaque fin de sprint, l'équipe fait le point sur ce qui a bien fonctionné et sur ce qui a moins bien fonctionné. Lors de cette réunion d'une durée de 15 à 30 minutes où chacun est invité et parle en son nom, un vote de confiance est organisé pour décider des améliorations à apporter.

1.2.2 Organisation :

La méthodologie SCRUM fait intervenir 3 rôles principaux qui sont :

- **Product owner** : Dans la majorité des projets, le responsable produit (product owner) est le responsable de l'équipe projet client. C'est lui qui va définir et prioriser la liste des fonctionnalités du produit et choisir la date et le contenu de chaque sprint sur la base des valeurs (charges) qui lui sont communiquées par l'équipe.
- **Scrum Master** : Véritable facilitateur sur le projet, il veille à ce que chacun puisse travailler au maximum de ses capacités en éliminant les obstacles et en protégeant l'équipe des perturbations extérieures. Il porte également une attention particulière au respect des différentes phases de SCRUM.
- **Equipe** : d'une taille allant de 4 à 10 personnes en général, l'équipe regroupe tous les rôles habituellement nécessaires à un projet, à savoir l'architecte, le concepteur, le développeur, le testeur, etc. L'équipe s'organise elle-même et elle reste inchangée pendant toute la durée d'un sprint.

1.2.3 Les avantages de cette méthode :

Scrum se différencie des autres méthodes de développement par ses avantages qui font de ce procédé une réponse pragmatique aux contraintes actuelles des chefs de produits, parmi ses avantages autres que ceux des méthodes agiles on trouve :

- **Adaptabilité maximale pour du développement de produits et d'applications** : la composition séquentielle du contenu des sprints permet d'ajouter une modification ou une fonctionnalité qui n'était pas prévue au départ. C'est principalement cela qui rend cette méthode "agile".
- **Méthode participative** : chaque membre de l'équipe est invité à s'exprimer et il peut participer à toutes les décisions prises sur le projet. Il est donc plus impliqué et plus motivé.

- **Augmentation de la communication** : en travaillant dans la même salle de développement, ou en étant connecté avec différents moyens de communication, l'équipe peut communiquer facilement et échanger sur les obstacles afin de les supprimer au plus tôt.
- **Maximisation de la coopération** : les échanges quotidiens entre le client et l'équipe permettent un rapprochement et une entraide se met logiquement en place.
- **Augmentation de la productivité** : en supprimant certaines "contraintes" des méthodes classiques comme la documentation ou la formalisation exagérée, SCRUM permet d'augmenter la productivité de l'équipe. En ajoutant à cela la qualification de chaque module permettant d'en déterminer un chiffre, chacun peut se positionner par rapport à la productivité moyenne de l'équipe.

1.3 Cycle de vie

1.3.1 Description

La vie d'un produit développé en appliquant Scrum est faite d'une séquence de releases. En général, un release dure quelques mois (entre 2 et 8 mois). Quand une release est terminée, nous passons à la suivante, jusqu'à la fin de vie du produit. Normalement, les releases se suivent, mais ne se chevauchent pas.

Un release est constitué d'une séquence de sprints. Quand un sprint est fini, le suivant commence. En principe, il n'y a pas de trou entre 2 sprints et les sprints ne se chevauchent pas.

Après le début de la release et avant le début du premier sprint de cette release, il y a une période de temps de durée variable, parfois appelée improprement sprint zéro.

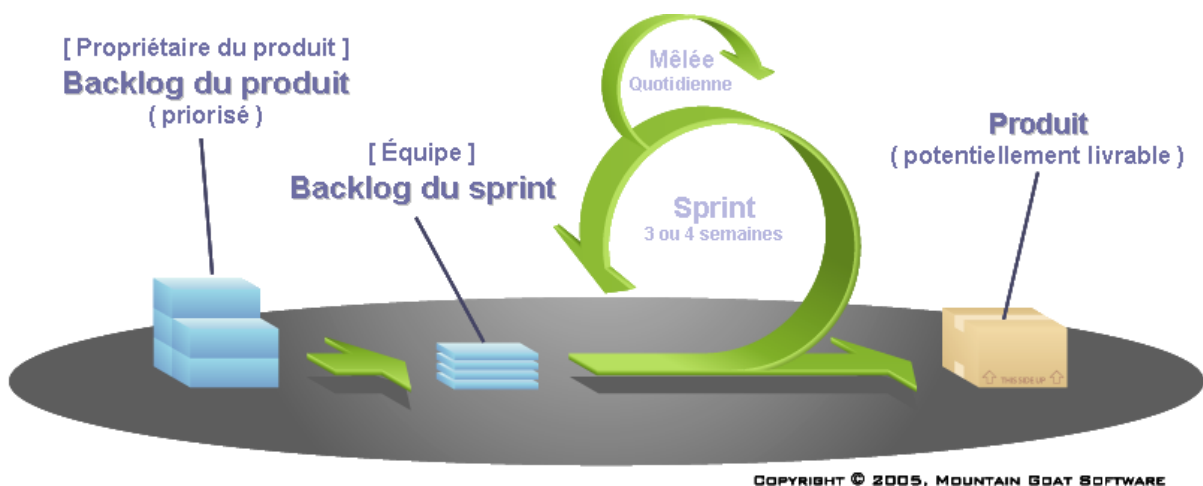


Figure 2 CYCLE DE VIE SCRUM

Chapitre 3 : Analyse et conception

1.4 Planification

Dans le contexte d'une meilleure conduite de ce projet, le planning ci-dessous a été élaboré afin de guider le travail, et avoir une visibilité sur l'état d'avancement :

Tableau 1: PLANNING DU PROJET

Date début	Date fin	Tache	Description
18/03/2012	30/03/2012	Préparation du cahier de charge pour AFC	Elaboration du cahier de charge AFC
30/03/2012	16/04/2012	Sprint Zéro	Installation des environnements de travail (voir chapitre étude technique du projet)
17/04/2012	01/05/2012	Sprint 1	Réalisation du premier Sprint : Module client
02/04/2012	14/05/2012	Sprint 2	Réalisation du deuxième Sprint : Module ressources humaines
15/05/2012	30/05/2012	Sprint 3	Réalisation du troisième Sprint : Suivi déclaration

1.5 Outil de gestion de projet(IceScrum):

La méthode Scrum se base essentiellement sur la communication d'une part entre l'équipe, et d'autre entre l'équipe et le client. Pour garantir cette communication, nous avons pratiqué la méthode classique qui consiste à échanger entre nous toutes les nouvelles informations (User Story finie, User Story en cours par qui ?,...etc.). Nous nous sommes rendu compte après que tout cela est une perte de temps et la gestion du projet n'est pas efficace car nous ne disposons pas d'un historique qui garde les traces de nos tâches réalisées.

Afin de bien gérer notre projet en appliquant la méthode SCRUM, on a décidé donc de travailler avec l'outil de gestion de projet IceScrum.

1.5.1 Présentation :

IceScrum est un outil d'aide à la gestion de projet Scrum. Il est particulièrement apprécié pour son interface aboutie et originale. Il existe plusieurs solutions sur le marché permettant d'aider les chefs de projet dans leurs gestions agiles. IceScrum a su se positionner grâce à la finition de son interface et à sa gratuité.

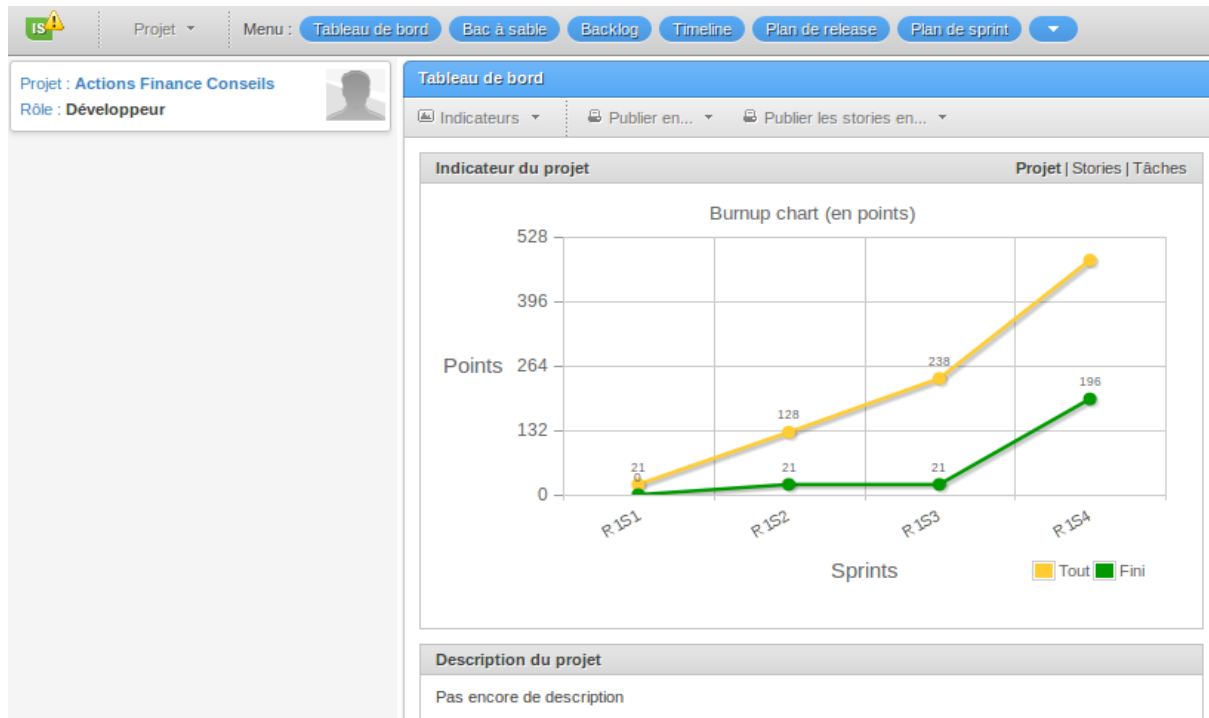


Figure 3.2 : INTERFACE GLOBALE DE ICESRUM 1

1.5.2 Utilisation :

Dans IceScrum comme dans Scrum, la notion d'équipe est primordiale : l'usage de l'outil n'est pas réservé à une seule personne qui en devient le spécialiste. Dans cette optique, la spécificité des rôles est limitée à l'essentiel :

- Un équipier peut créer des *stories* dans le *backlog* de produit, créer des tâches dans le plan de *sprint*, créer des tests d'acceptation, noter le résultat des tests et enregistrer un obstacle.
- Le ScrumMaster a la responsabilité supplémentaire de gérer l'élimination des obstacles et d'indiquer qu'un nouveau *build* est utilisable.
- Le Product Owner est le seul habilité à créer un *release*, à définir les *features*, à changer les priorités dans le *backlog* et à déclarer une *story* finie.

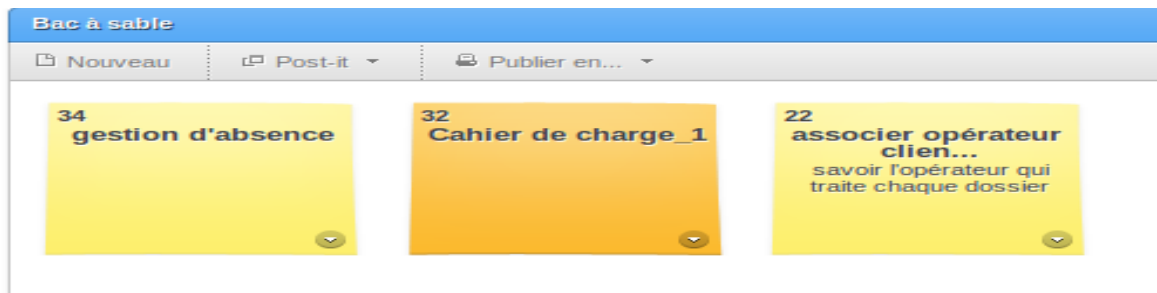


Figure 3: BAC A SABLE

1.5.3 Backlog de produit initial :

Le *backlog* de produit IceScrum montre à l'ouverture les *stories* qui sont à prioriser. L'outil permet de filtrer les éléments du *backlog* selon leur état. La vue « à prioriser » regroupe les *stories* qui ne sont pas encore planifiées, c'est-à-dire celles qui ne sont pas associées à des *sprints*.

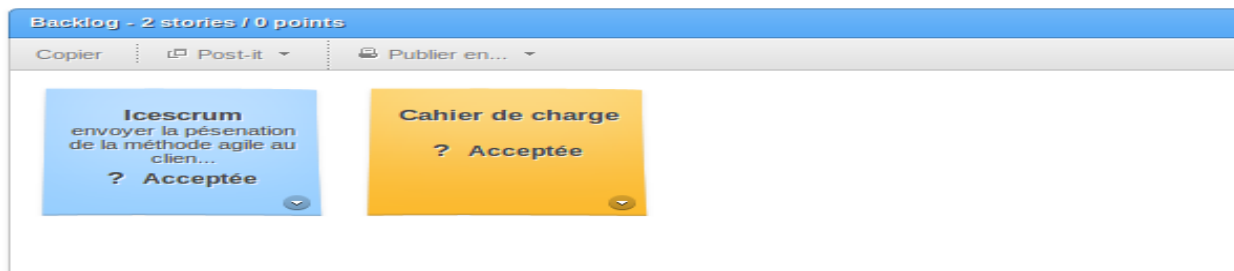


Figure 4: BACKLOG DES USER STORIES

1.5.4 Création de sprints dans le plan de release :

Le plan de release montre la vie du produit à moyen terme. On y trouve les sprints qui composent la release. Le plan repose sur l'association des stories du backlog aux différents sprints. La planification de release consiste à associer des stories du backlog à des sprints de la release. Pour cela, il faut donc que le backlog contienne des stories estimées : en effet pour planifier, il faut avoir estimé et seules les stories estimées seront candidates à être

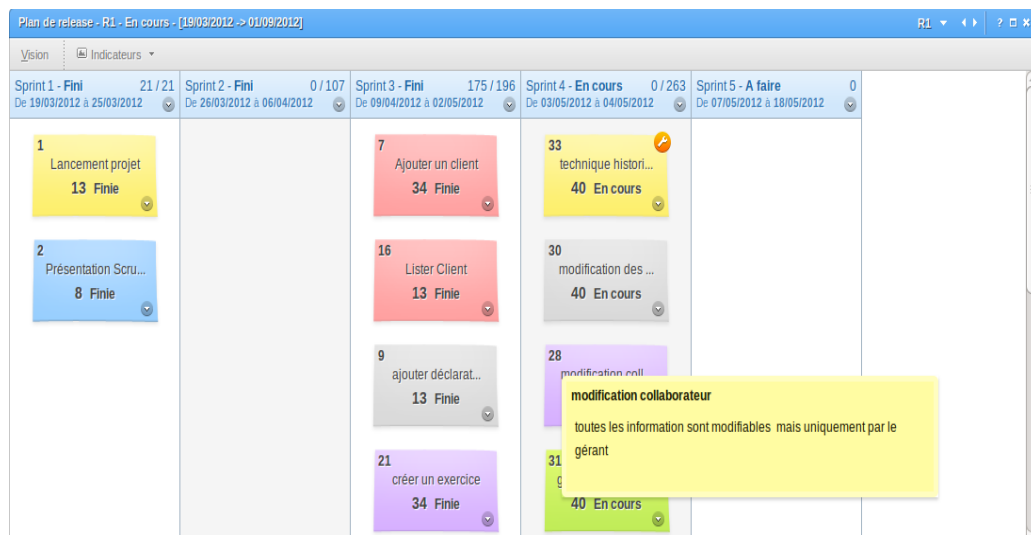


Figure 5 : PLAN DE REALFASF R1

Chapitre 3 : Analyse et conception

planes. Il faut aussi avoir créé des sprints. Il sera possible plus tard de changer d'avis, en dissociant toutes les stories, ou uniquement celles d'un sprint ou des stories individuelles ou en déplaçant une story d'un sprint à un autre

1.5.5 Plan de Sprint :

IceScrum fournit une assistance facilitant le lancement du sprint, divisant chaque User Story en tâches prises par chaque membre de l'équipe.

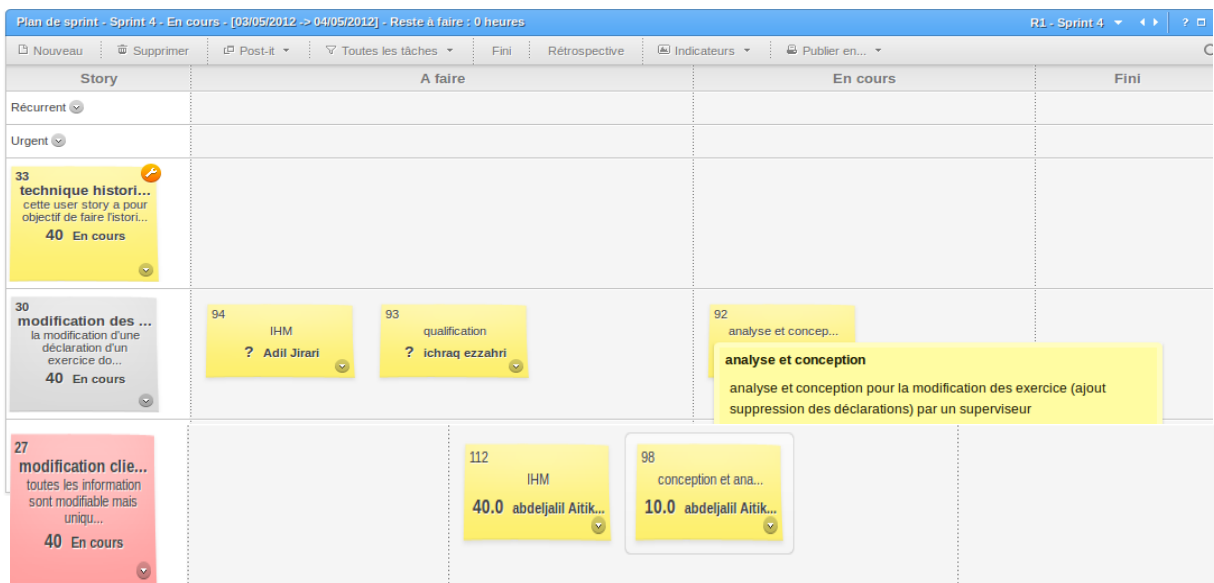


Figure 6 : PLAN DE SPRINT

2. Analyse et conception :

2.1 Présentation :

Nous allons entamer dans cette partie du rapport l'analyse et la conception des différents Sprint que nous avons réalisés à savoir :

- Sprint 0 : Installation des environnements de travail (voir chapitre étude technique du projet) ;
- Sprint 1 : Réalisation du premier Sprint : Module client ;
- Sprint 2 : Réalisation du deuxième Sprint : Module ressources humaines ;
- Sprint 3 : Réalisation du troisième Sprint : Suivi déclaration.

A chaque sprint, nous avons réalisé l'analyse et la conception des User Stories qui le composent. Pour ne pas répéter les mêmes diagrammes dans chaque Story nous avons décidé de présenter une conception générale de notre projet en détaillant certaines fonctionnalités importantes.

2.2 Modélisation :

2.2.1 Les acteurs du système :

Tableau 2 : ACTEURS DU SYSTEME

Profil	Description
Opérateur	Utilisateur qui fait le suivi des dossiers comptables, en mentionnant l'état de chaque client. Un opérateur peut être un simple opérateur ou un superviseur ou une secrétaire.
Gérant	C'est le rôle principal de l'application, il a le droit de faire toutes les tâches réalisées dans l'application, de la création des comptes utilisateurs en spécifiant leur rôle jusqu'au suivi des dossiers comptables.

2.2.2 Cas d'utilisation par acteur :

Tableau 3 : CAS D'UTILISATION PAR ACTEUR

Nom UC	Nom sous UC	Fonctionnalité	Acteur
Suivi des dossiers	Suivi des déclarations	-Changer l'état d'une déclaration : <ul style="list-style-type: none"> • En cours client. • En cours Collaborateur. • Terminé... 	Opérateur, gérant
	Suivi des journaux	-Changer l'état d'un journal : <ul style="list-style-type: none"> • Fait • Non fait 	Opérateur, gérant
	Suivi des livres légaux	-Déterminer l'avancement de chaque livre légal : <ul style="list-style-type: none"> • Terminé • Non terminé 	Opérateur, gérant
	Gestion de l'historique	-Consulter l'historique des transactions	Gérant
Gestion des clients	Gestion des déclarations	-Ajouter une déclaration -Modifier une déclaration -Supprimer une déclaration	Gérant
	Gestion des exercices	- Modifier un exercice	Gérant
	Gestion des responsables	-ajouter un responsable -modifier un responsable -Archiver un responsable	Gérant
	Gestion des activités	-Ajouter une activité -Modifier une activité -Supprimer une activité	Gérant
		-Ajouter un journal	

	Gestion des journaux	-Modifier un journal -Supprimer un journal	Gérant
	Gestion des livres légaux	-Archiver le livre légal	Gérant
	Affectation opérateurs	-Associé à chaque client un opérateur de traitement et un superviseur	Gérant
Gestion des utilisateurs	Gestion d'authentification	-Création des comptes de chaque utilisateur -Attribuer à chaque utilisateur un profil déterminé	Gérant

2.2.3 Analyse du cas d'utilisation : Gestion des clients

Un client est une entreprise qui est définie par : un code client, raison sociale, numéro de la taxe professionnelle. Selon la description du métier de notre client, l'application pourra éventuellement proposer les fonctionnalités suivantes:

- **Créer un nouveau client:** l'application fournit à l'administrateur un moyen de remplir les informations du client nouvellement ajouté.
- **Modifier un client:** les fiches clients AFC sont éditables, l'administrateur peut modifier les informations sur un client donné.
- **Archiver un client:** conserver les données du client.
- **Afficher les clients:** le gérant peut consulter la liste des clients par des critères de sélection, mode de tri...
- **Rechercher un client:** cela est possible par la recherche sur la raison sociale.

- **Diagramme de cas d'utilisation :**

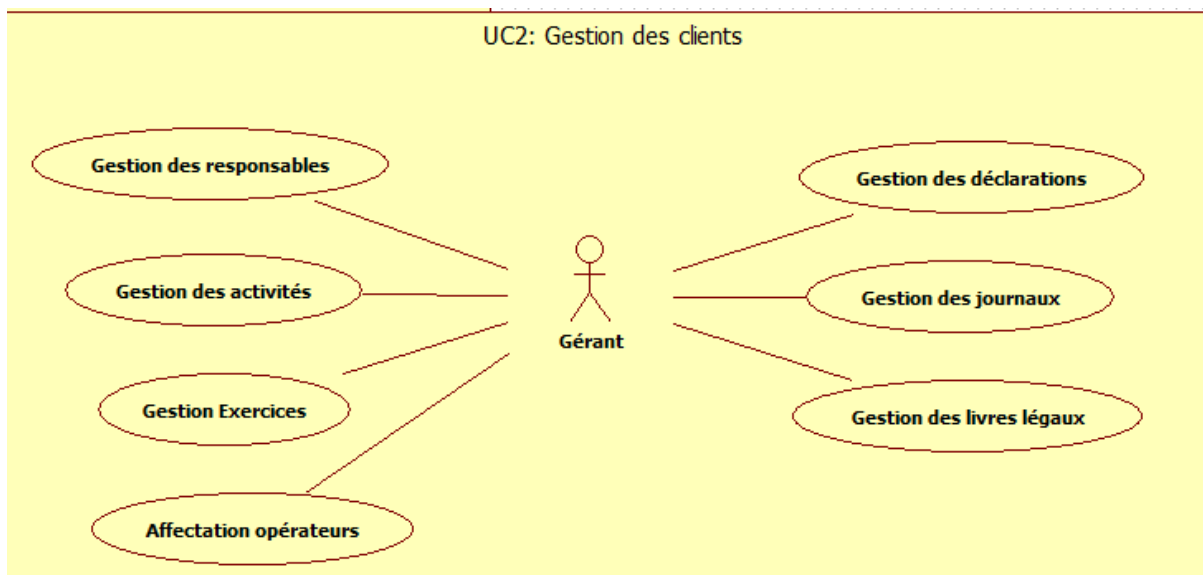


Figure 7 : DIAGRAMME CAS D'UTILISATION GESTION CLIENT

Après avoir ajouté un client une tâche automatique se déclenche pour l'ajout d'un exercice que nous allons détailler dans la fiche ci-dessous :

- **Tableau de fiches de cas d'utilisation : Gestion des exercices**

L'exercice fiscal, année fiscale ou encore exercice comptable est une période de temps délimitée au cours de laquelle une entreprise enregistre tous les faits économiques qui concourent à l'élaboration de sa comptabilité.

Dans cette perspective, AFC a besoin d'une gestion des exercices de ses clients en abordant deux aspects fondamentaux :

- Création automatique des exercices ;
- Association des déclarations à chaque exercice.

- **✚ Création automatique des exercices :**

Lors de l'ajout d'un client, l'utilisateur doit préciser la date de clôture de l'exercice. La création des exercices suivants se fait automatiquement à date anniversaire. L'exercice créé prend le statut en cours de réalisation, l'exercice précédent est lui-même passé à l'état « clôturé ».

Chapitre 3 : Analyse et conception

Association déclarations et exercice :

Lors de l'ajout d'un client, il est obligé d'indiquer les différentes activités de l'entreprise pour générer les différentes déclarations correspondantes et les ajouter automatiquement dans chaque exercice créé pour avoir une liste des déclarations de l'année fiscale en cours afin de faciliter leur suivi.

A la fin de ce processus automatique nous pouvons lister les exercices d'un client précis ainsi que les déclarations de chaque année fiscale en cliquant sur le champ « Exercices » situé dans le tableau d'affichage d'un client. Nous pouvons aussi faire une recherche de tous les exercices clôturés d'un client ou bien uniquement de l'exercice en cours.

Tableau 4 : FICHE DE CAS D'UTILISATION SUIVI DECLARATION

Titre	Cas d'utilisation gestion des exercices
Version	Version 2.0 du 15/04/2012
Scénario nominal	<ul style="list-style-type: none">• Le gérant crée un client.• l'exercice associé se crée automatiquement.• Un pop-up s'affiche pour montrer que l'exercice est créé avec succès.
Scénario alternatif	<ul style="list-style-type: none">• Le gérant crée un client en précisant une date de clôture de l'exercice inférieure à la date système.• Le valideur retourne une erreur de création de client et de l'exercice parce que la date doit être supérieure à la date système.
Acteurs principaux	<ul style="list-style-type: none">• Gérant
Pré condition	Le gérant est authentifié
Post condition	Création de l'exercice de la nouvelle année automatiquement
Fréquence	Ajout d'un dossier (client)

- **Diagramme d'activité: création d'un exercice**

La création automatique des exercices est divisée en deux parties par deux diagrammes d'activité :

- Une création de l'exercice 1 automatiquement après avoir ajouté un nouveau client :

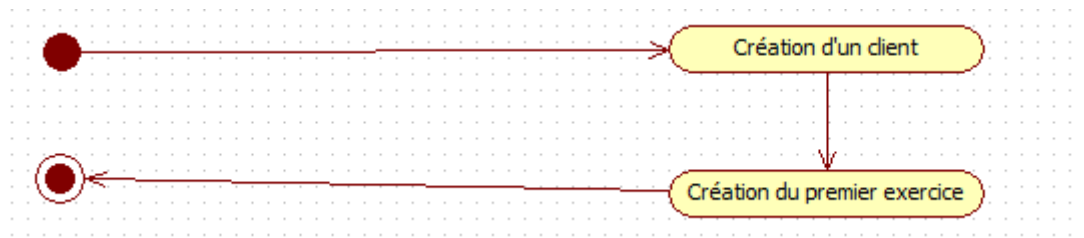


Figure 8 : DIAGRAMME D'ACTIVITE CREATION AUTOMATIQUE DU PREMIER EXERCICE

- Un déclenchement automatique par le batch pour vérifier si la date de clôture de l'exercice numéro N est égale à la date système, dans ce cas on crée automatiquement l'exercice N+1 :

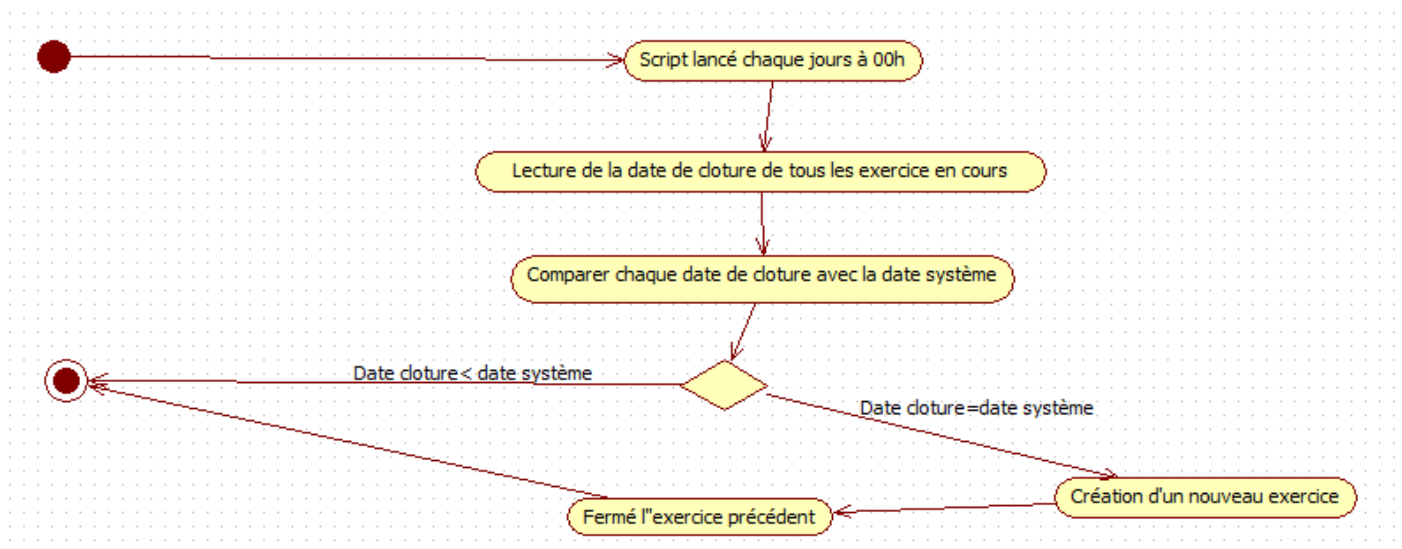


Figure 9 : DIAGRAMME D'ACTIVITE CREATION AUTOMATIQUE DES PROCHAINS EXERCICES

2.2.4 Analyse du cas d'utilisation : Gestion utilisateur

Ce cas d'utilisation permet au gérant d'ajouter, de modifier, archiver ou lister les informations d'un utilisateur.

- **Diagramme de cas d'utilisation :**

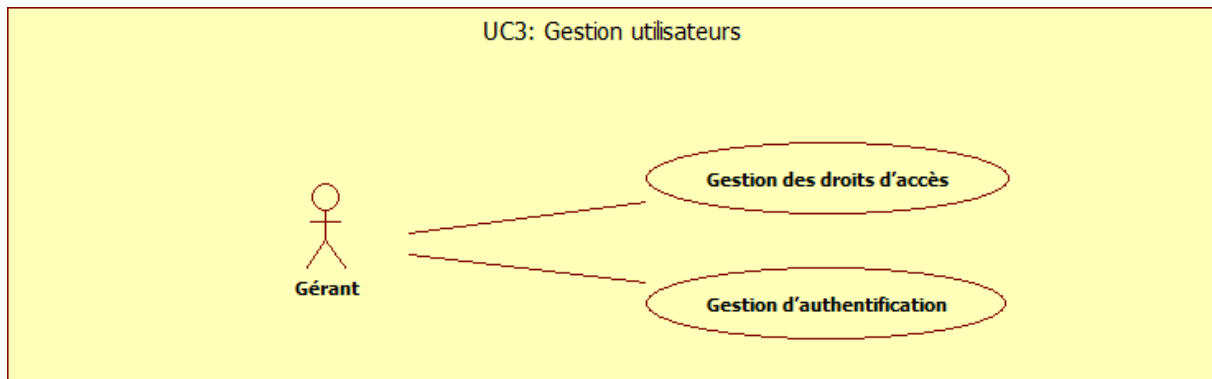


Figure 10 : DIAGRAMME DE CAS D'UTILISATION GESTION UTILISATEUR

- **Tableau de fiche de cas d'utilisation : Gestion d'authentification**

Le cas d'utilisation «*Gestion d'authentification*» permet au gérant d'ajouter ou modifier le compte d'un utilisateur. Au départ le gérant attribue à chaque utilisateur un login et un mot de passe pour qu'il puisse s'authentifier. Ce dernier peut à tout moment faire la modification de son mot de passe.

Tableau 5: FICHE DE CAS D'UTILISATION GESTION AUTHENTIFICATION

Titre	Cas d'utilisation Gestion d'authentification
Version	Version 2.0 du 01/04/2012
Scénario nominal	<ul style="list-style-type: none">• Le gérant enregistre le login et le mot de passe d'un utilisateur.• Message de confirmation de mot de passe s'affiche.• Compte enregistré avec succès.
Acteurs principaux	<ul style="list-style-type: none">• Gérant
Pré condition	Le gérant est identifié et authentifié
Post condition	Création d'un compte utilisateur
Fréquence	Ajout d'un utilisateur

- **Diagramme de séquence pour la gestion d'authentification :**

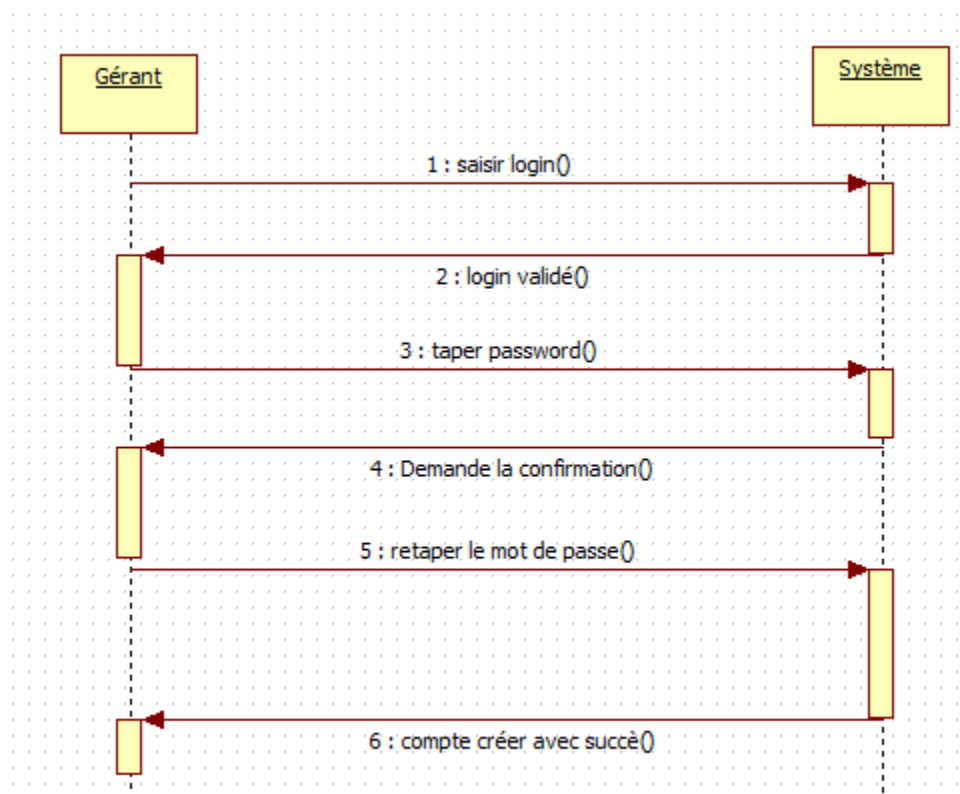


Figure 11 : DIAGRAMME DE SEQUENCE GESTION AUTHENTIFICATION

2.2.5 Analyse du cas d'utilisation : Suivi des dossiers

Le cas d'utilisation « Suivi des dossiers » permet à l'opérateur et au gérant de faire le suivi des différents dossiers comptables à savoir le suivi des déclarations, des journaux et des livres légaux en mentionnant leurs états : en cours client, en cours collaborateur, fait, non fait, terminé, etc.

En effet, après son authentification chaque opérateur doit mettre à jour l'état des dossiers comptables liés au client qui lui est associé. Le gérant peut consulter l'état d'avancement de tous les dossiers et peut aussi changer leurs états. C'est d'ailleurs pour cette raison que nous avons fait une gestion de l'historique de ce suivi pour garder la trace, en cas d'erreurs, de toutes les modifications faites pour chaque dossier en mentionnant l'utilisateur qui a fait cette modification.

- **Diagramme de cas d'utilisation :**

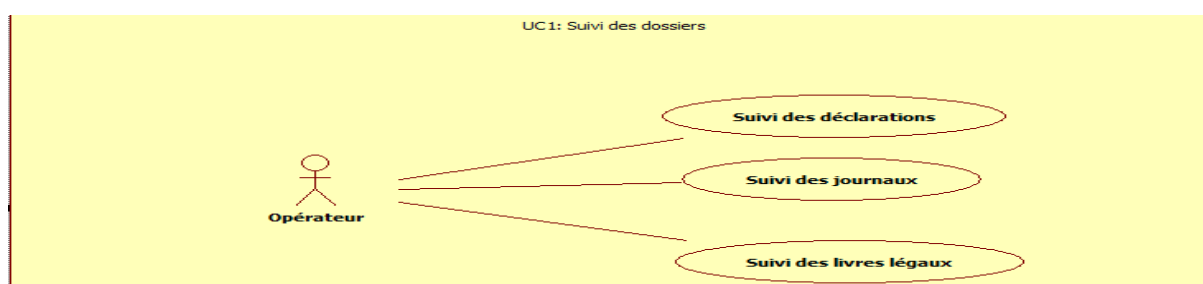


Figure 12 : DIAGRAMME DE CAS D'UTILISATION SUIVI DES DOSSIERS

Chapitre 3 : Analyse et conception

- **Tableau de fiches de cas d'utilisation : suivi des déclarations**

Le cas d'utilisation « suivi des déclarations » permet à l'utilisateur qu'il soit opérateur ou gérant de mettre à jour l'état de chaque déclaration de l'exercice courant d'un client. Une déclaration peut être :

- Non commencée : c'est la valeur par défaut d'une déclaration qui vient d'être créée lors de la création automatique de l'exercice courant.
- En cours client : cet état est attribué à la déclaration si le client concerné n'a pas fourni les pièces nécessaires, ou s'il a fourni des pièces partielles et qu'il lui reste d'autres pièces pour démarrer ses opérations comptables.
- En cours collaborateur : après avoir fourni toutes les pièces nécessaires, la déclaration peut être mise dans l'état « en cours collaborateur ».
- Terminée : c'est l'état de la déclaration après avoir terminé toutes les tâches nécessaires par le collaborateur.
- En instance de signature : pour être signée par le client.
- Signée, par le client.
- En instance de dépôt : chez la secrétaire.
- Envoyée pour le dépôt : Lorsque la déclaration est signée et complète, elle prend le statut « Envoyée pour le dépôt » avec une date de dépôt.
- Classée et archivé : après avoir terminé toutes les tâches de la déclaration celle-ci est classée et archivée avec une date d'archivage.

Tableau 6: FICHE DE CAS D'UTILISATION SUIVI DECLARATION

Titre	Cas d'utilisation suivi des déclarations
Version	Version 2.0 du 01/05/2012
Scénario nominal	<ul style="list-style-type: none">• Choisir un client• Choisir l'exercice courant• Choisir une déclaration• Mettre à jour l'état de la déclaration• Base de données modifiée
Acteurs principaux	<ul style="list-style-type: none">• Opérateur• Gérant
Pré condition	L'utilisateur est identifié.
Post condition	La déclaration de l'exercice courant de ce client est modifiée.
Fréquence	A la demande

- **Diagramme d'état du cas d'utilisation : suivi des déclarations**

La figure ci-dessous présente les différents états qu'une déclaration peut prendre comme il a été mentionné auparavant.

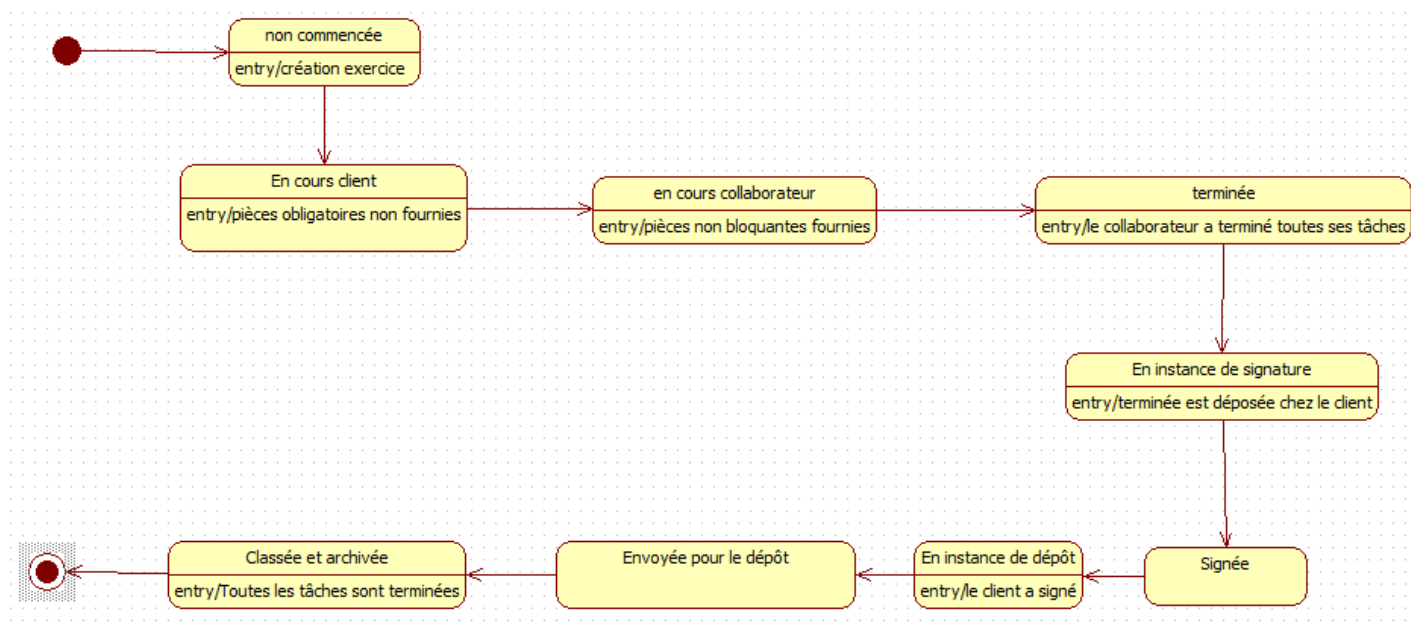


Figure 13 : DIAGRAMME D'ETAT D'UNE DECLARATION

- **Diagramme de composant du cas d'utilisation : Gestion de l'historique**

Le cas d'utilisation « gestion de l'historique » permet au gérant de consulter toutes les modifications faites sur l'état des dossiers comptables. Par exemple, lors du changement d'état d'une déclaration donnée, qui est contenue dans un exercice courant, un traitement automatique se fait permettant au gérant de savoir qui a fait cette modification et quel est le champ affecté dans la déclaration.

En effet, une fonctionnalité apportée par le framework « Envers » se charge de créer une table associée à celle contenant l'état de la déclaration, cette table comporte tous les champs de sa table origine en plus d'un champ appelé « revision », et lors de la modification de l'état d'une déclaration, le script se lance automatiquement indiquant le champ modifié avec son numéro de révision correspondant : si c'est la 1^{ère} révision (modification) ou la 2^{ème} etc.

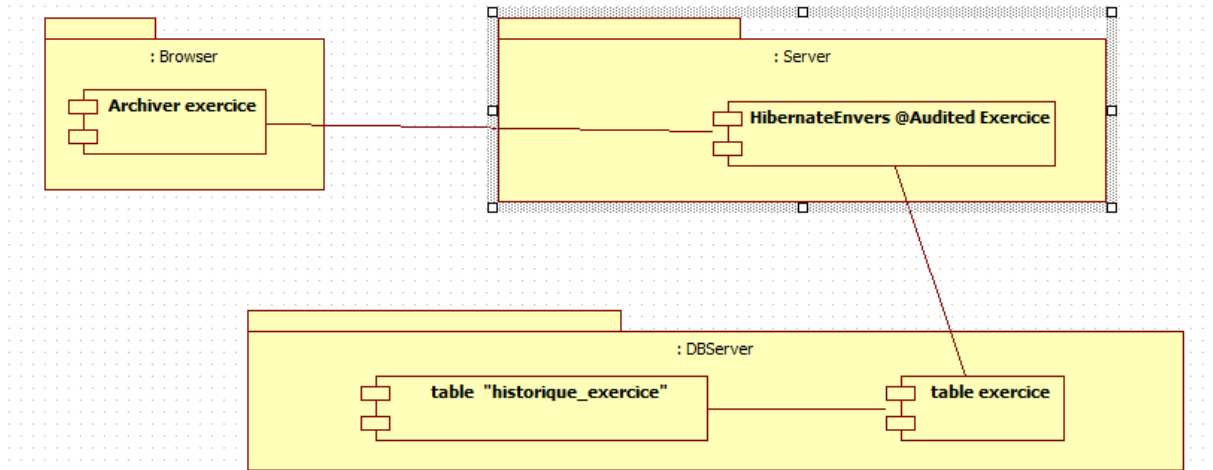


Figure 14 : DIAGRAMME D'HISTORISATION DES EXERCICES

2.2.6 Diagramme de cas d'utilisation global :

La figure suivante représente le diagramme de cas d'utilisation global de notre application :

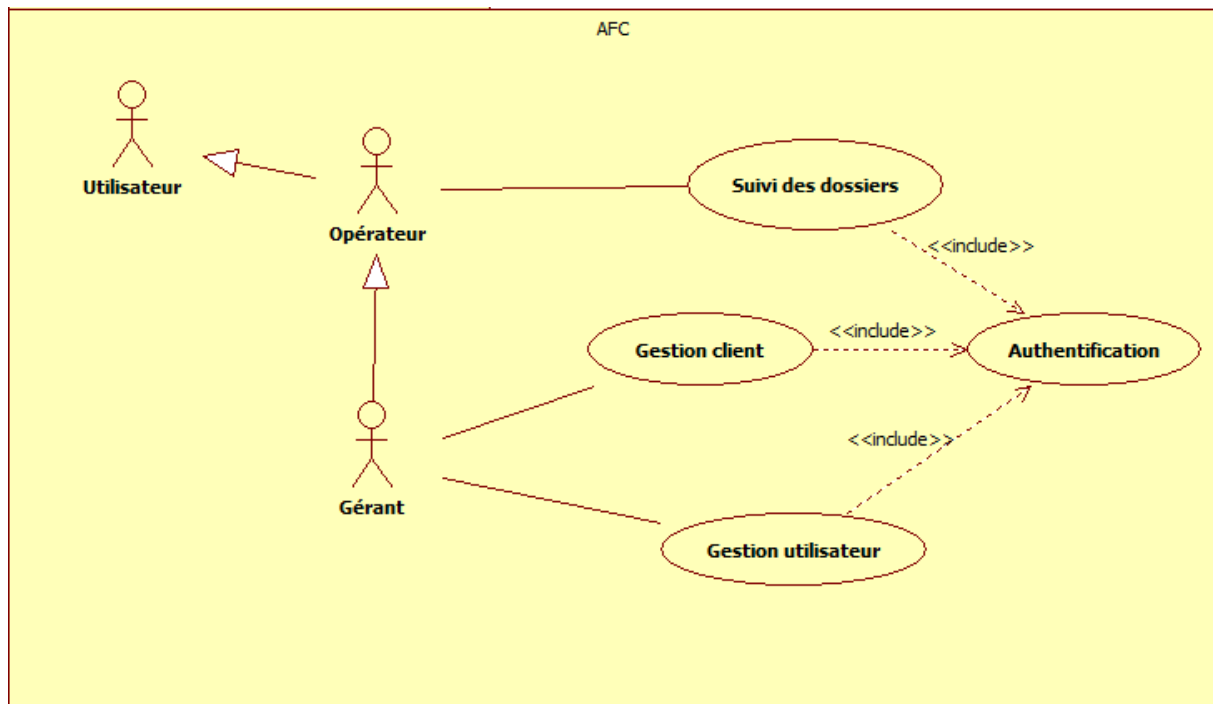


Figure 15 : DIAGRAMME DE CAS D'UTILISATION GLOBAL

Chapitre 3 : Analyse et conception

2.2.7 Diagramme de classes :

La figure ci-dessous représente le diagramme de classes de l'application :

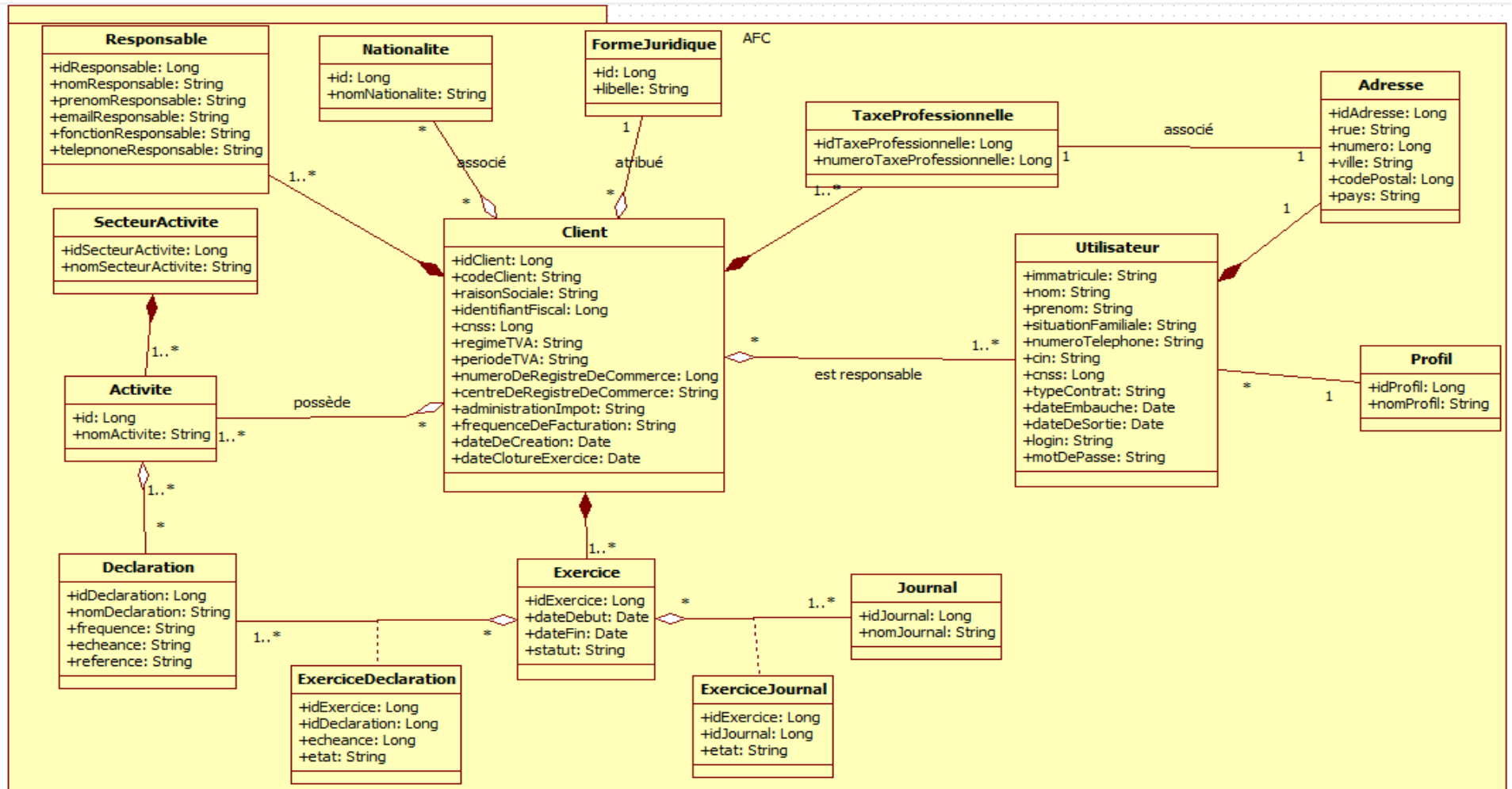


Figure 16 : DIAGRAMME DE CLASSES

Chapitre 3 : Analyse et conception

Ce diagramme illustre l'aspect statique de l'application et se compose des classes fondamentales suivantes :

- **Client** : C'est la classe comportant toutes les informations des clients du cabinet AFC, elle est associée à la plupart des classes :
 - Il est composé de plusieurs « **Responsable** », de plusieurs numéros de « **TaxeProfessionnelle** » et de plusieurs « **Exercice** » ; il a plusieurs « **Nationalite** », une « **FormeJuridique** », et plusieurs « **Activite** » ; il est aussi sous la responsabilité d'un ou plusieurs « **Utilisateur** ».
- **Utilisateur** : elle contient toutes les informations d'un utilisateur de l'application, chaque utilisateur à un profil : soit gérant, opérateur, superviseur ou secrétaire. Un Client peut être associé à plusieurs utilisateurs et celui-ci peut avoir plusieurs clients.
- **Exercice** : Classe définissant un exercice, un client est composé de plusieurs exercices dont un seul est en statut « en cours ».
- **Déclaration** : Classe spécifiant une déclaration, voir en annexe les types de déclaration. Un exercice contient plusieurs déclarations et une déclaration peut appartenir à plusieurs exercices, chaque déclaration d'un exercice est caractérisée par plusieurs échéances, et cela selon la fréquence de cette déclaration, par exemple : un exercice contenant une déclaration de fréquence « trimestrielle » va avoir quatre échéances et pour chacune d'elles (échéance) il y a un état, c'est pour cette raison que nous avons défini une classe d'association « **ExerciceDeclaration** » qui a pour clé primaire les trois champs :
 - idExercice ;
 - idDeclaration ;
 - echeance.
- **Activite** : classe définissant une activité exercée par un client, chaque activité fait partie d'un « **SecteurActivite** ».
- **Journal** : classe définissant un journal, il est créé lors de la création d'un exercice, celui-ci peut avoir plusieurs journaux et un journal peut être dans plusieurs exercices l'état d'un journal « fait » ou « non fait » est persisté dans la classe « **ExerciceJournal** ».

3. Conclusion :

Dans ce chapitre, nous avons vu la démarche de travail utilisée lors de la réalisation de cette application, puis nous avons mené une étude conceptuelle du projet pour finir avec un diagramme de classes général. Dans le prochain chapitre, nous allons faire une étude technique du projet pour finir avec la partie réalisation.

Chapitre 4 :

Etude technique du projet

Dans ce chapitre, nous faisons une étude technique du projet. Nous commençons tout d'abord par mentionner les problèmes rencontrés sans adopter un standard, ensuite nous allons présenter l'architecture adoptée par le groupe pour répondre à ces problèmes ainsi que quelques outils de développement de notre application.

1. Problématique :

« Plus un défaut est détecté tard, plus il coûte cher à corriger » *Loi de Defect Cost Increase*

Avant de se lancer dans le développement de l'application, nous devons préparer l'environnement de travail et se mettre d'accord sur la démarche à suivre avant et après chaque implémentation en s'éloignant au maximum des problèmes suivants :

- Eviter le «ça compile sur mon poste et non sur l'autre».
- Avoir des erreurs sans savoir au niveau de quelle couche.
- Avoir une vision non précise de la manière dont nous allons utiliser le programme avant même d'envisager son implémentation.
- Tomber dans la régression du code.
- Construire un projet complexe, difficile à déployer.

Pour ne pas tomber dans ces situations, nous avons opté pour l'architecture technique suivante :

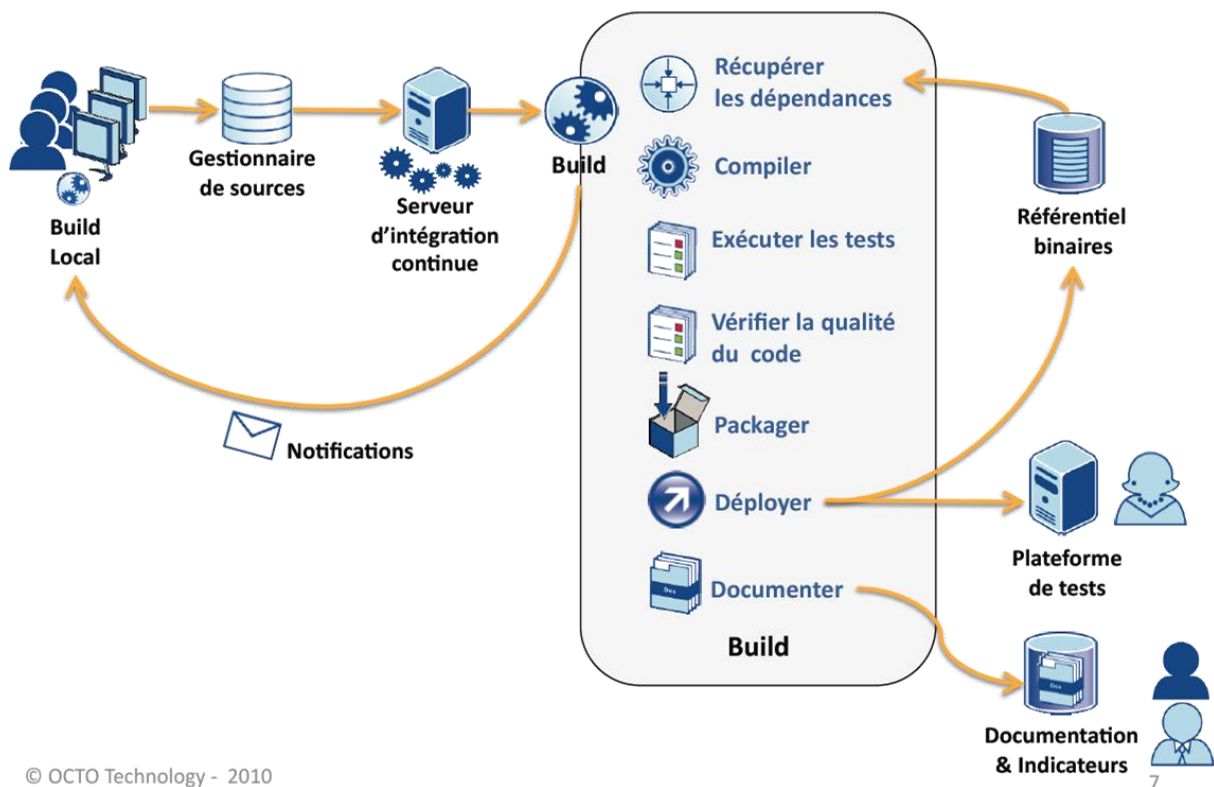


Figure 17 : ARCHITECTURE TECHNIQUE DU PROJET

2. Démarche technique :

2.1 Implémentation:

2.1.1 Problématique

Nous avons rencontré lors de la réalisation de notre projet plusieurs problèmes au niveau du code à savoir :

- La production d'un code n'est pas valide en toutes circonstances.
- Implémenter des fonctionnalités complexes difficiles à tester.
- Ecrire un code, le tester et si notre test échoue on modifie dans le code de l'application.
- La régression du code ce qui augmentera le taux des erreurs.

2.1.2 Solution proposée : TDD (Test Driven Development) :

C'est une méthode de développement logiciel qui préconise d'écrire les tests unitaires avant d'écrire le code source du logiciel. A ce sujet nous testons chaque méthode dao, service avant même d'implémenter ces méthodes en suivant ces étapes:

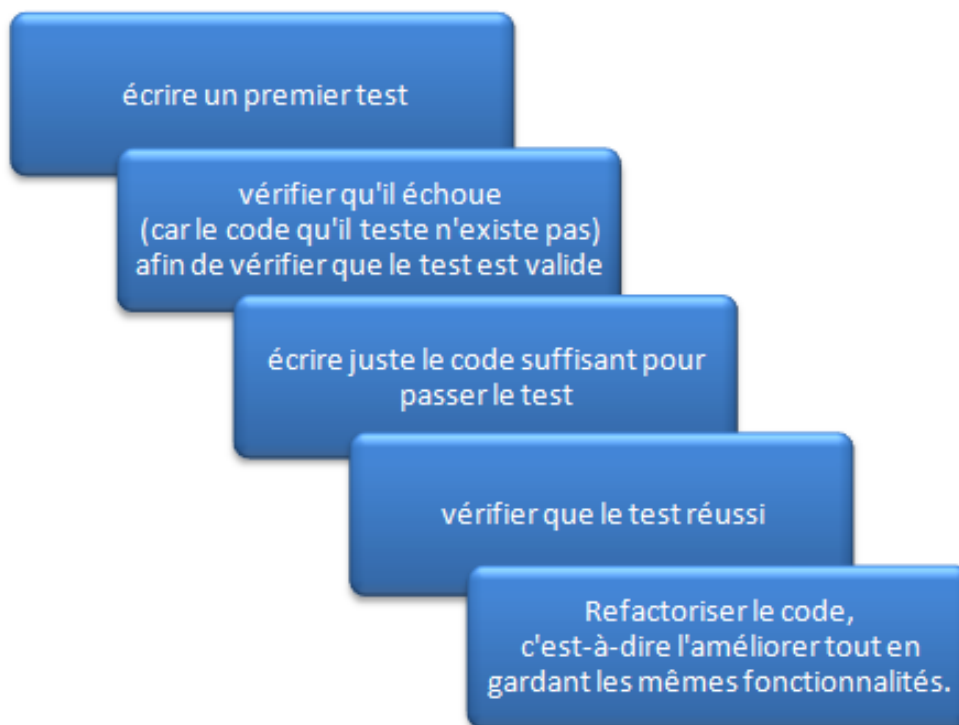


Figure 18 : ETAPES DE LA DEMARCHE TDD

- **Avantages :**
 - Préciser les spécifications du code, son comportement ultérieur en fonction des situations auxquelles il sera exposé (utilisation des mock).
 - Assurer la non régression du code.

Chapitre 4 : Etude technique du projet

- Avoir une vision précise de la manière dont on veut lire le programme avant même d'envisager son implémentation.
- Eviter les erreurs.

2.2 Collaboration:

2.2.1 Problématique :

Puisque nous travaillons en équipe avec la méthode agile, alors nous sommes censés travailler sur la même application et bien sûr :

- Plusieurs personnes peuvent modifier, ajouter du code... dans des parties communes.
- Le besoin de pouvoir récupérer une version précédente.
- Récupérer sur le champ la dernière version de l'application...
- Phase d'intégration complexe avec une équipe de 6 développeurs.

2.2.2 Solution proposée : GIT :

C'est un gestionnaire de code source et versionning (subversion) comme SVN. Il s'est désormais imposé dans ce domaine pour plusieurs raisons :

- Sa rapidité
- Sa robustesse
- Garder le contenu d'un ensemble de répertoire (et non le contenant)
- Garde les révisions du repository et non des fichiers
- Possède une interface rudimentaire
- Très fiable

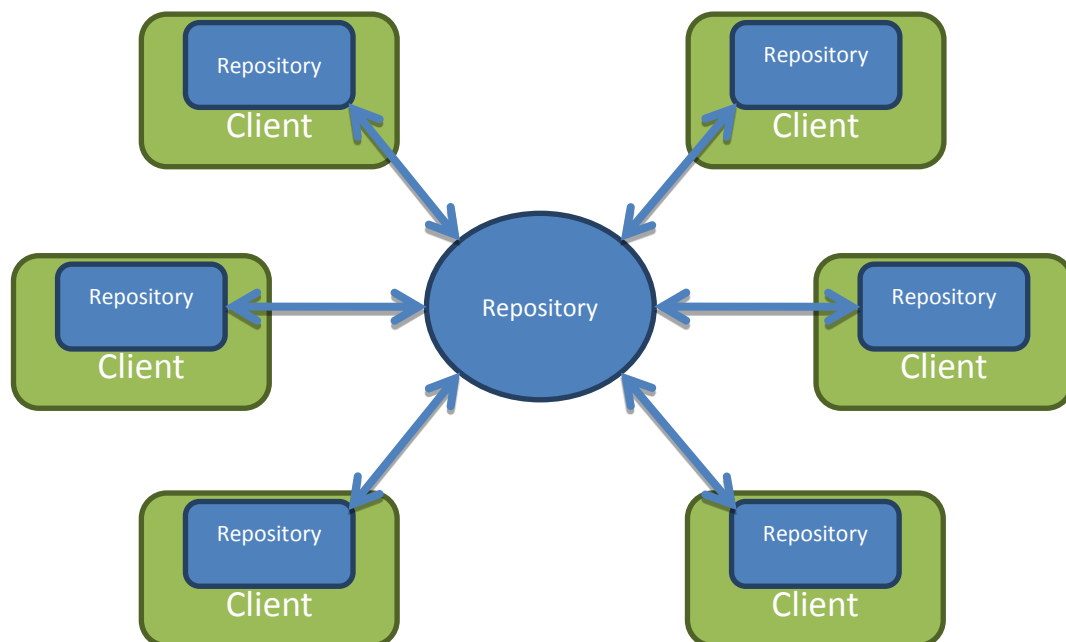


Figure 19 : MODE DE GESTION GIT CHEZ NORSYS

2.3 Intégration continue :

2.3.1 Problématique :

Nous voulons vérifier à chaque modification du code source que le résultat des modifications ne produit pas de régression de l'application AFC.

2.3.2 Solution proposée : Jenkins

A ce sujet, nous avons utilisé l'outil d'intégration continue **Jenkins**, nommé au début « Hudson ».

Intérêts :

- Détecter les problèmes le plus tôt possible : afin d'alerter l'équipe le plus vite possible !
- Générer des compilations régulièrement : afin d'obtenir des versions stables et déployables le plus souvent possible.
- Maintenir un dépôt unique de code versionné en utilisant un outil de build (comme **maven**).
- Tous les développeurs commettent quotidiennement.
- Automatiser les compilations (builds) et les tests.
- Tout commit doit compiler le tronc du code versionné.
- Maintenir une compilation courte en permanence.
- Rendre disponible le résultat du build à tout le monde.
- Automatiser le déploiement (optionnel).

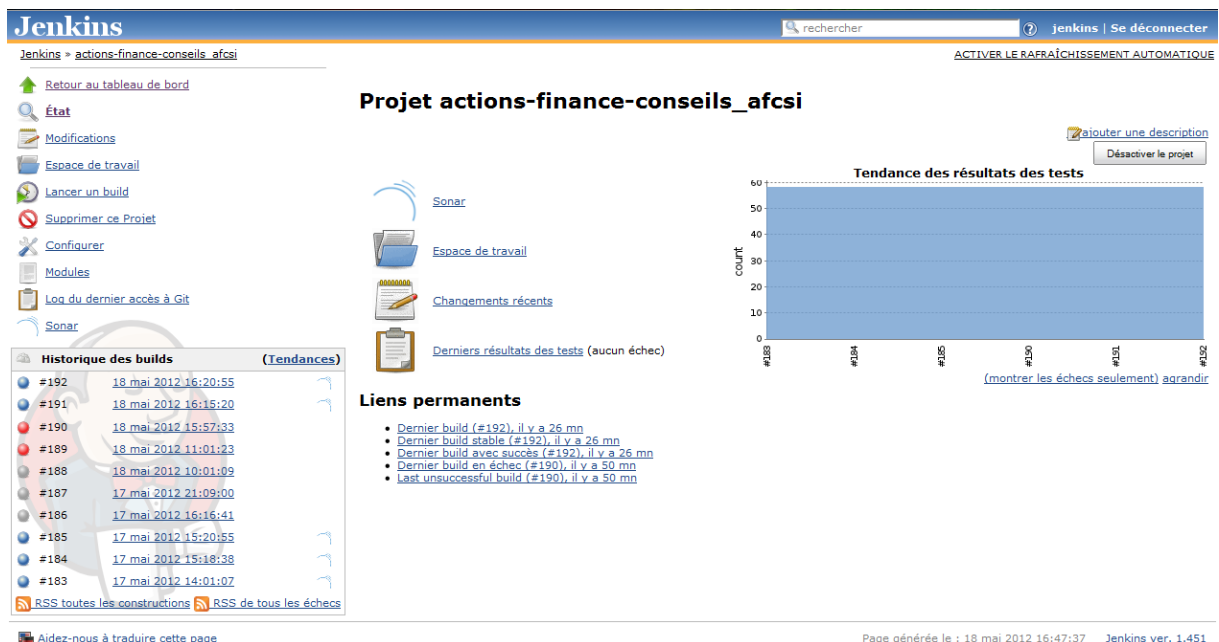


Figure 20 : JENKINS AFC

2.4 Outils de Qualification :

Les systèmes informatiques disposent d'une gestion de la qualité particulière. La qualité des systèmes informatiques intègre au projet de développement une approche permettant de contrôler autant que possible le produit final pour ne pas tomber dans les situations suivantes :

- Mauvaise distribution de la complexité.
- Code dupliqué.
- Mauvais design.
- Existence de bugs potentiels.
- Mauvaise couverture par les tests unitaires,...
- Non-respect des standards de programmation
- Pas ou trop de commentaires

Nous avons installé un ensemble d'outils pour contrôler ces propriétés :

2.4.1 CheckStyle :

Eclipse-cs est un plugin open source qui propose de puissantes fonctionnalités pour appliquer des contrôles sur le respect de règles de codifications.

Pour définir les contrôles réalisés lors de son exécution Eclipse-cs utilise une configuration qui repose sur des modules. Cette configuration est définie dans un fichier XML qui précise les modules utilisés et pour chacun d'entre eux leurs paramètres.



Le plus simple est d'utiliser le fichier de configuration nommé Sun Checks.xml fourni avec Eclipse-cs. Cette configuration propose d'effectuer des contrôles de respect des normes de codification proposée par Sun. Il est aussi possible de définir son propre fichier de configuration, d'ailleurs nous avons défini notre propre fichier checkstyle pour ce projet.

2.4.2 Sonar :

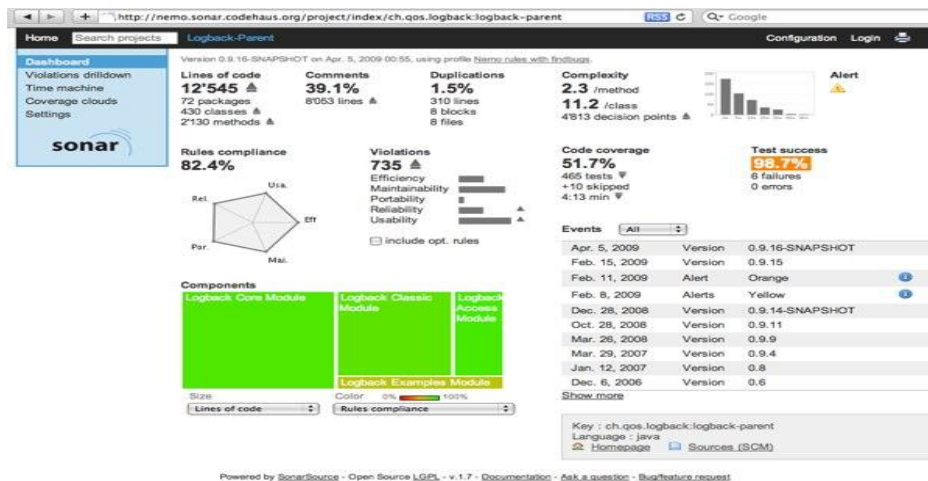


Figure 21 : SONAR

Sonar est un outil de suivi de la qualité d'un logiciel, qui permet via l'analyse du code source et l'exécution des tests de reporter, par exemple :

- les violations des bonnes pratiques de codage
- le niveau de complexité des classes et des méthodes
- le niveau de couverture par les tests
- le niveau de succès des tests
- le niveau de documentation du code...

Nous avons intégré un ensemble de plugins de contrôle dans Sonar :

- **PMD** : Framework qui permet d'analyser le code source Java, contient un certain nombre de règles qui assure la qualité de code (le code inutile, les imbrications trop complexes...)
- **Checkstyle**.
- **FindBugs** : Détecteur de bug, identifie des patterns reconnus comme étant des bugs.
- **Cobertura** : A coupler avec des tests unitaires (JUnit) pour calculer la couverture de test, génère des rapports sous format HTML ou XML.

2.5 Outils de développement :

2.5.1 *SpringSource tool suite (STS) :*



SpringSource Tool Suite™ (STS) offre le meilleur environnement de développement Eclipse-Powered pour construire des applications d'entreprise Spring-powered. STS fournit des outils pour l'ensemble des dernières versions Java entreprise, Spring, Groovy, Grails et d'autres technologies.

Il se caractérise par des cibles de déploiement flexibles comme :

- La prise en charge de tous les serveurs d'applications Java EE les plus courantes
- Le Support avancé pour la SpringSource tc serveur
- La VMware Lab Manager et Workstation pour l'intégration et le déploiement



Spring est un Framework populaire et gratuit et largement déployé. Il aide les développeurs à créer des applications de haute qualité et de manière très rapide. Il assure une programmation cohérente et une configuration du modèle bien comprise et utilisée par des millions de développeurs à travers le monde. Contrairement à la traditionnelle plate-forme Java EE, Spring fournit une gamme de capacités pour la création des applications Java Entreprise, rich Web, et les applications d'intégration d'entreprise qui peuvent être consommés dans un boîtier léger.

2.5.2 *MySQL :*



MySQL, serveur de bases de données SQL Open Source, est développé, distribué et supporté par MySQL AB. MySQL AB est une société commerciale, fondée par les développeurs de MySQL, qui développent leur activité en fournissant des services autour de MySQL.

3. Conclusion :

Dans ce chapitre, nous avons présenté les différents outils et méthodes d'industrialisation utilisées pour concevoir cette application à savoir :

- TDD
- Outils de subversion GIT.
- Les outils d'intégration continue : Jenkins ;
- Les outils de développement : STS et MySQL ;
- Les outils de Qualification : CheckStyle et Sonar.

Dans le chapitre suivant, nous allons présenter l'architecture logique ainsi qu'une partie de la réalisation du projet.

Chapitre 5 :

Réalisation

Ce chapitre présente les deux principaux volets de la réalisation de ce projet : la mise en place d'une architecture logique et l'implémentation du projet.

1. Architecture logicielle :

1.1 Présentation :

Nous sommes arrivés à l'étape de la réalisation dans laquelle les besoins et la conception sont bien déterminés. Il nous reste qu'à définir une architecture logicielle satisfaisante caractérisée par :

- Sa gratuité.
- La haute disponibilité.
- La portabilité.
- La performance.
- Un échange sécurisé.

Après avoir consulté les différents avantages et inconvénients des différentes architectures disponibles nous nous sommes mis d'accord d'utiliser une **architecture JEE trois-tiers**.

1.2 Justification du choix de l'architecture :

1.2.1 Architecture JEE

La plateforme Java EE est la plus riche des plateformes qui offre un environnement standard de développement et d'exécution des applications d'entreprise multi tiers.



En effet, les principaux avantages d'utiliser Java EE (et donc Java) sont la portabilité, l'indépendance, la sécurité et la multitude de bibliothèques proposées.

1.2.2 Architecture Trois-tiers :

L'architecture trois-tiers est un modèle logique d'architecture applicative qui vise, à séparer très nettement trois couches logicielles dont le rôle est clairement défini :

- **la présentation des données** : correspondant à l'affichage, la restitution sur le poste de travail, le dialogue avec l'utilisateur ;
- **le traitement métier des données** : correspondant à la mise en œuvre de l'ensemble des règles de gestion et de la logique applicative c.-à-d. les services;
- et enfin **l'accès aux données persistantes** : correspondant aux données qui sont destinées à être conservées dans une base de données.

Dans cette approche, les couches communiquent entre elles à travers «un modèle d'échange» et elles ont pour objectif :

- Allègement du poste de travail client;
- Prise en compte de l'hétérogénéité des plates-formes (serveurs, clients, langages, etc.) ;

Chapitre 5 : Réalisation

- Amélioration de la sécurité des données, en supprimant le lien entre le client et les données. Le serveur a pour tâche, en plus des traitements purement métiers, de vérifier l'intégrité et la validité des données avant de les envoyer dans la couche de données.
- Et enfin, meilleure répartition de la charge entre différents serveurs d'application.

1.3 Démarche de développement :

Pour développer l'application nous avons utilisé pour chaque couche des outils et des Frameworks, voici une figure illustrant quelques-unes :

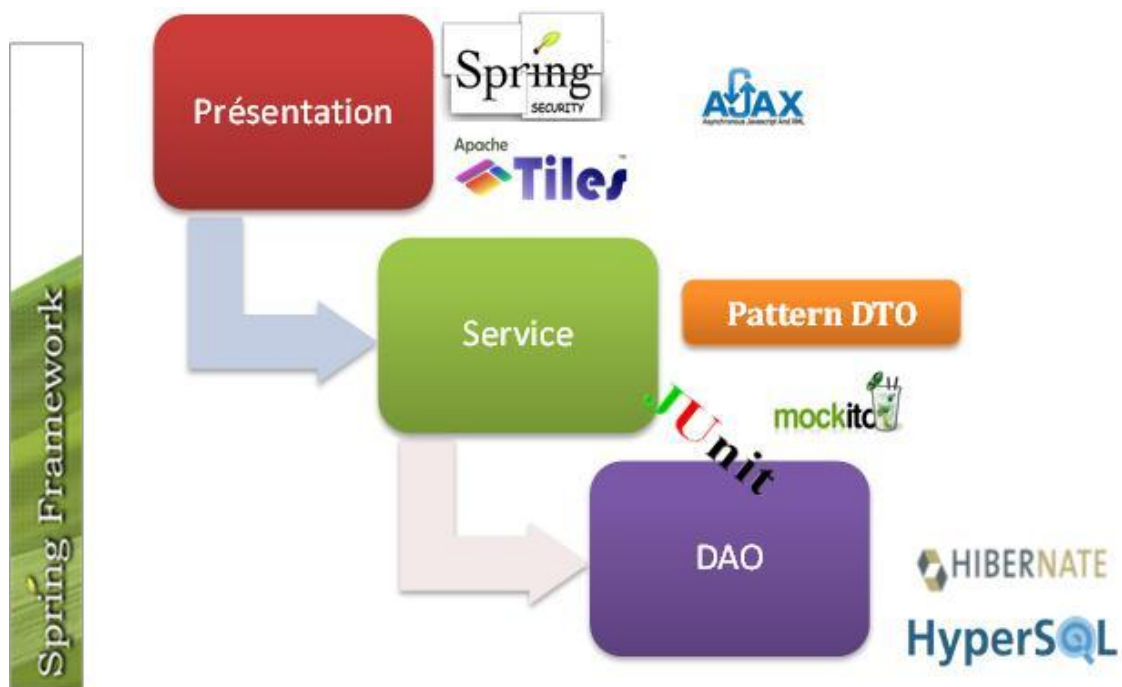


Figure 22 : Architecture logicielle

1.1.1 Couche DAO :

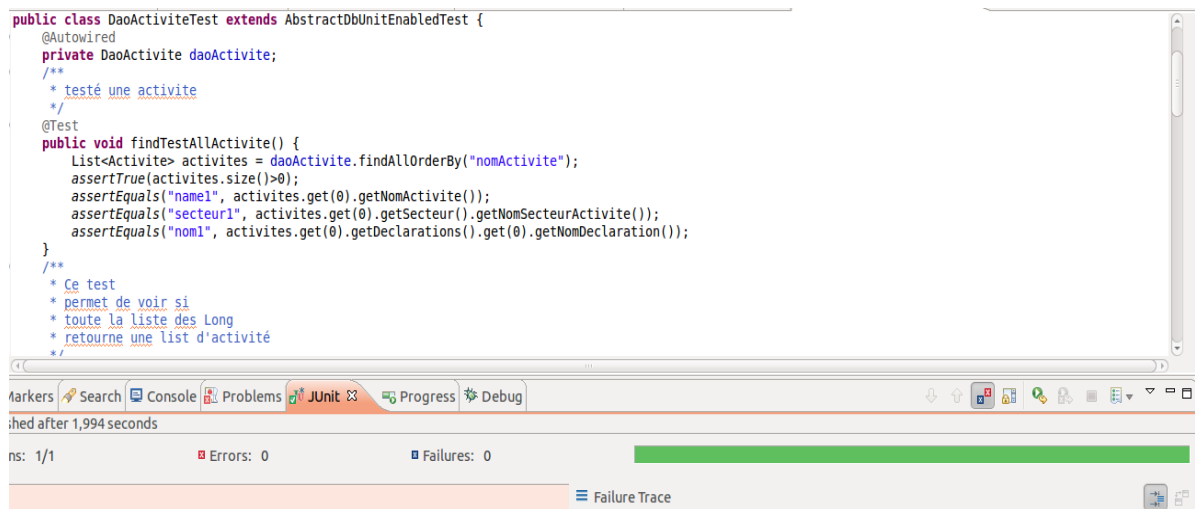
Pour réaliser cette partie, nous avons travaillé avec le Framework **Hibernate** dans la partie implémentation, en effet nous avons réalisé une classe « AbstractDaoJpa » qui généralise l'ensemble des méthodes les plus utilisées dans les classes Dao.


```
public class AbstractDaoJpa<E, PK> implements DaoBase<E, PK> {
    /**
     * objet de type entity manager
     */
    @PersistenceContext
    private EntityManager entityManager;
    /**
     *
     */
    private final Class<? extends E> entityClass;
    /**
     * constructor sans argument
     * @param aEntityClass the aEntityClass to set
     */
    public AbstractDaoJpa(final Class<? extends E> aEntityClass) {
        super();
        this.entityClass = aEntityClass;
    }

    @Override
    public E findByPk(final PK pk) {
        return entityManager().find(this.entityClass, pk);
    }
}
```

Figure 23 : ABSTRACT DAO

Ensuite, nous avons testé cette couche avec le Framework JUnit version 4 et HSQLDB comme SGBD de test :



```
public class DaoActiviteTest extends AbstractDbUnitEnabledTest {
    @Autowired
    private DaoActivite daoActivite;
    /**
     * testé une activite
     */
    @Test
    public void findTestAllActivite() {
        List<Activite> activites = daoActivite.findAllOrderBy("nomActivite");
        assertTrue(activites.size()>0);
        assertEquals("name1", activites.get(0).getNomActivite());
        assertEquals("secteur1", activites.get(0).getSecteur().getNomSecteurActivite());
        assertEquals("nom1", activites.get(0).getDeclarations().get(0).getNomDeclaration());
    }
    /**
     * Ce test
     * permet de voir si
     * toute la liste des Long
     * retourne une list d'activité
     */
}
```

ns: 1/1 Errors: 0 Failures: 0

Failure Trace

Figure 24 : TEST DAO

```
<table name="client_activites">
  <column>id_client</column>
  <column>id</column>
  <row>
    <value>1</value>
    <value>1</value>
  </row>
</table>

<table name="Nationalite">
  <column>id</column>
  <column>nom_nationalite</column>
  <row>
    <value>1</value>
    <value>maroc</value>
  </row>
  <row>
    <value>2</value>
    <value>france</value>
  </row>
</table>
```

Figure 25: HSQL DATA

1.1.2 Couche Service :

Pour simplifier les transferts de données entre les objets d'accès aux données et le contrôleur nous avons utilisé le pattern DTO (Data Transfer Object). Nous avons ensuite implémenté nos services depuis la couche Dao en faisant le mapping avec le DozerBeanMapper :

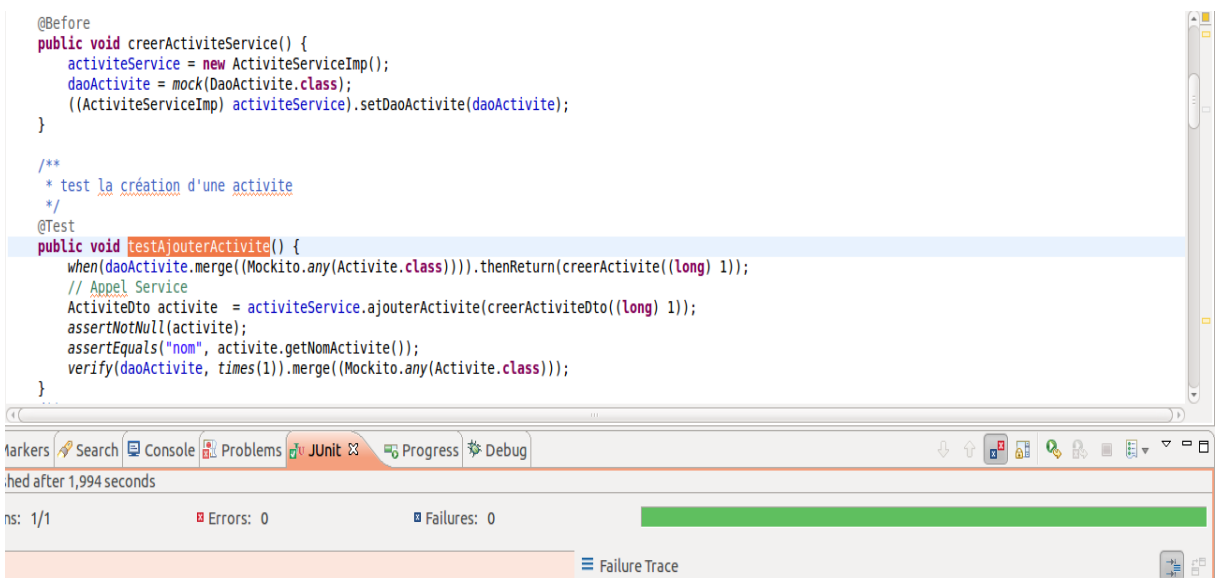
```
@Service
@Transactional
public class ExerciceServiceImpl implements ExerciceService {
    @Autowired
    private DaoExercice daoExercice ;

    @Autowired
    private DaoExerciceDeclarations daoExerciceDeclarations;

    private Mapper mapper = new DozerBeanMapper();
    /**
     * @param daoExercice the daoExercice to set
     */
    public void setDaoExercice(DaoExercice daoExercice) {
        this.daoExercice = daoExercice;
    }
    /** (non-Javadoc)
     * @see fr.norsys.actionsfinanceconseils.afc.service.ExerciceService#findAllExerciceByStatus(java.lang.String)
     */
    @Override
    public List<ExerciceDto> findAllExerciceByStatus(String statut) {
        List<ExerciceDto> exerciceDtos = new ArrayList<ExerciceDto>();
        List<Exercice> exercices = daoExercice.findAllExerciceByStatus(statut);
        for (Exercice exercise : exercices) {
            exerciceDtos.add(mapper.map(exercise, ExerciceDto.class));
        }
        return exerciceDtos;
    }
}
```

Figure 26: DOZER BEAN MAPP

Le Framework Mockito nous a permis de tester notre couche service en créant des doublures des interfaces Dao « *mock(interface.class)* » et en décrivant leurs comportements avec la méthode « *when(doublure.maMethode()).thenReturn(resultat())* » :



```
@Before
public void creerActiviteService() {
    activiteService = new ActiviteServiceImpl();
    daoActivite = mock(DaoActivite.class);
    ((ActiviteServiceImpl) activiteService).setDaoActivite(daoActivite);
}

/**
 * test la création d'une activite
 */
@Test
public void testAjouterActivite() {
    when(daoActivite.merge(Mockito.any(Activite.class))).thenReturn(creerActivite((long) 1));
    // Appel Service
    ActiviteDto activite = activiteService.ajouterActivite(creerActiviteDto((long) 1));
    assertNotNull(activite);
    assertEquals("nom", activite.getNomActivite());
    verify(daoActivite, times(1)).merge(Mockito.any(Activite.class));
}

ns: 1/1
Errors: 0
Failures: 0
```

Figure 27: TEST SERVICE

1.1.3 Couche Présentation :

- Framework Spring MVC :

Dans notre projet, nous avons opté pour le Framework **Spring MVC version 3** afin de réaliser la couche présentation. Ce Framework utilise le pattern MVC (Modele, View, Controller), et offre un ensemble d'annotations (@Controller,..) facilitant le développement de cette partie. La figure ci-dessous résume l'architecture logique du pattern MVC :

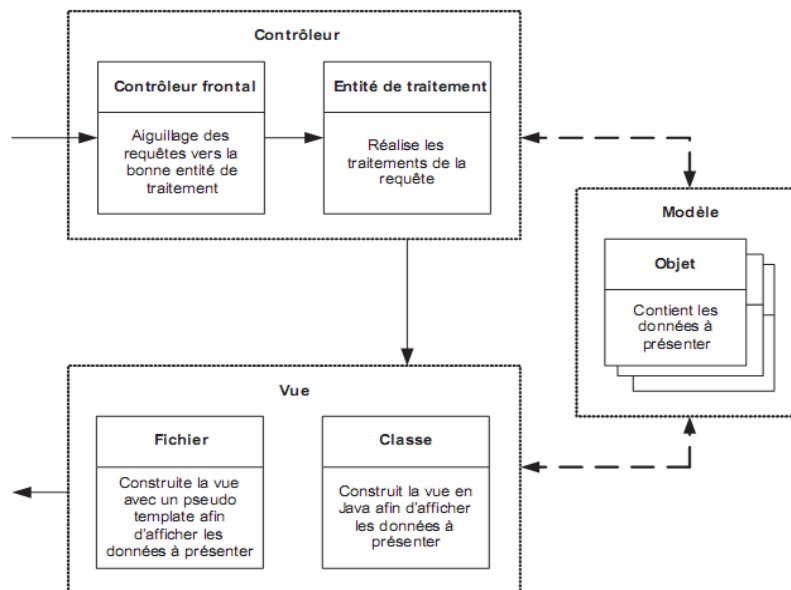


Figure 28 : PATRON MVC

- Framework Spring Security :

Afin d'assurer la sécurité de notre application, nous avons utilisé le Framework Spring Security qui permet de gérer l'accès aux ressources d'une application Java.

La figure ci-dessous est un extrait du fichier de configuration de spring security :

```
http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-3.0.3.xsd"
<global-method-security secured-annotations="enabled">
</global-method-security>
<http use-expressions="true">
  <intercept-url pattern="/authentication.jsp" access="permitAll" />
  <intercept-url pattern="/css/*" access="permitAll" />
  <intercept-url pattern="/css/layout/*" access="permitAll" />
  <intercept-url pattern="/css/theme/*" access="permitAll" />
  <intercept-url pattern="/css/theme/images/*" access="permitAll" />
  <intercept-url pattern="/images/*" access="permitAll" />
  <intercept-url pattern="/js/*" access="permitAll" />
  <intercept-url pattern="/*" access="isAuthenticated()" />
  <form-login login-page="/authentication.jsp"
    default-target-url="/" always-use-default-target="true"
    authentication-failure-url="/authentication.jsp?login_error=1" />
  <logout logout-success-url="/authentication.jsp" />
</http>
```

Figure 29: LAYER SPRING SECURITY

- Tiles :

Tiles est un Framework qui simplifie le développement d'interfaces utilisateurs des applications web. Il permet aux auteurs de définir des fragments de page qui peuvent être assemblés dans une page complète au moment de l'exécution.

La figure ci-dessous est un extrait du fichier de configuration de Tiles :

```
<definition name=".layout" template="/WEB-INF/jsp/layout/layout.jsp">
  <put-attribute name="header" value="/WEB-INF/jsp/layout/header.jsp" />
  <put-attribute name="menu" value="/WEB-INF/jsp/layout/menu.jsp" />
  <put-attribute name="content" value="${content}" />
</definition>

<definition name=".index" extends=".layout">
  <put-attribute name="content" value="/WEB-INF/jsp/layout/content.jsp" />
</definition>

<definition name=".pageAccueil" template="/WEB-INF/jsp/accueil/accueil.jsp">
</definition>

<definition name=".formPage" template="/WEB-INF/jsp/accueil/Personne/formPage.jsp">
</definition>
```

Figure 30: LA STRUCTURE DE L'APPLICATION AFC PAR TILES

- Constante et internalisation :

Afin de rendre l'application flexible pour toute éventuelle maintenance, nous avons créé des fichiers de paramétrages des différents messages et labels qui s'affichent à l'utilisateur, voici un extrait d'un fichier de message utilisé :

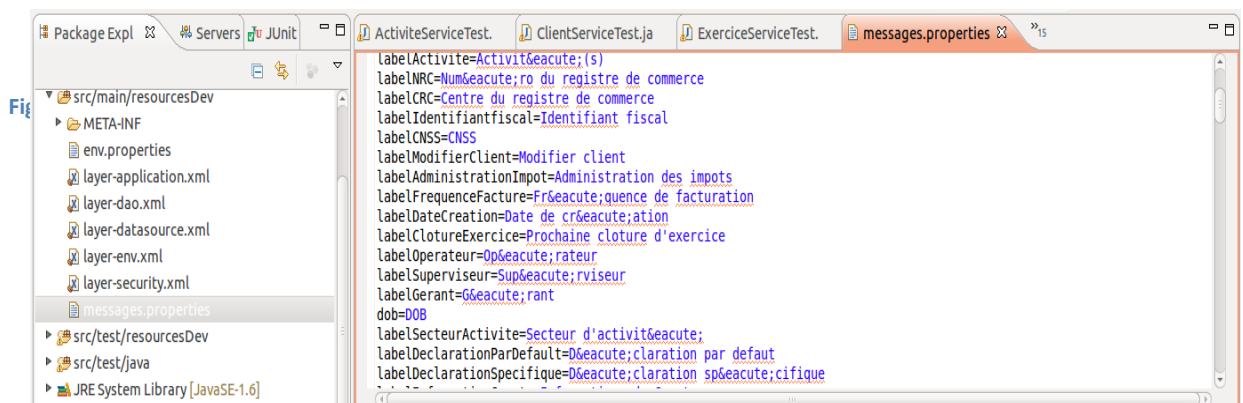


Figure 32: FICHIER D'INTERNATIONALISATION

2. Réalisation :

2.1 Présentation

Nous avons choisi des captures d'écran de certaines fonctionnalités de l'application AFC afin d'illustrer notre travail pendant cette période. Nous présentons la page d'accueil qui est composée d'un menu à gauche décrivant les fonctionnalités primordiales de l'application (client, utilisateur, exercice) et un autre menu dans le header pour les champs paramétrés (déclarations, activité, secteur d'activité, profil).

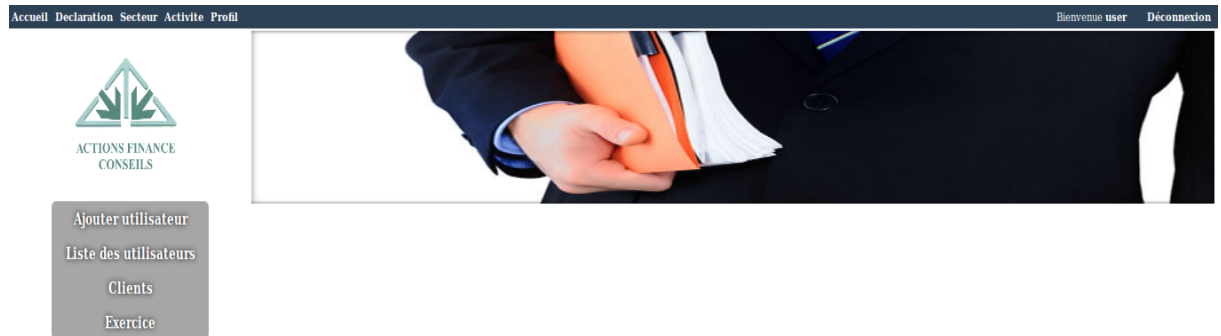


Figure 33 : PAGE D'ACCUEIL

2.2 Authentification

Nous avons utilisé le Framework Spring security pour sécuriser l'application. La gestion des droits d'accès permet à l'utilisateur de visualiser uniquement sa partie par exemple le gérant à un accès sur toutes les fonctionnalités de l'application.



Actions Finance Conseils

Connexion

Identifiant

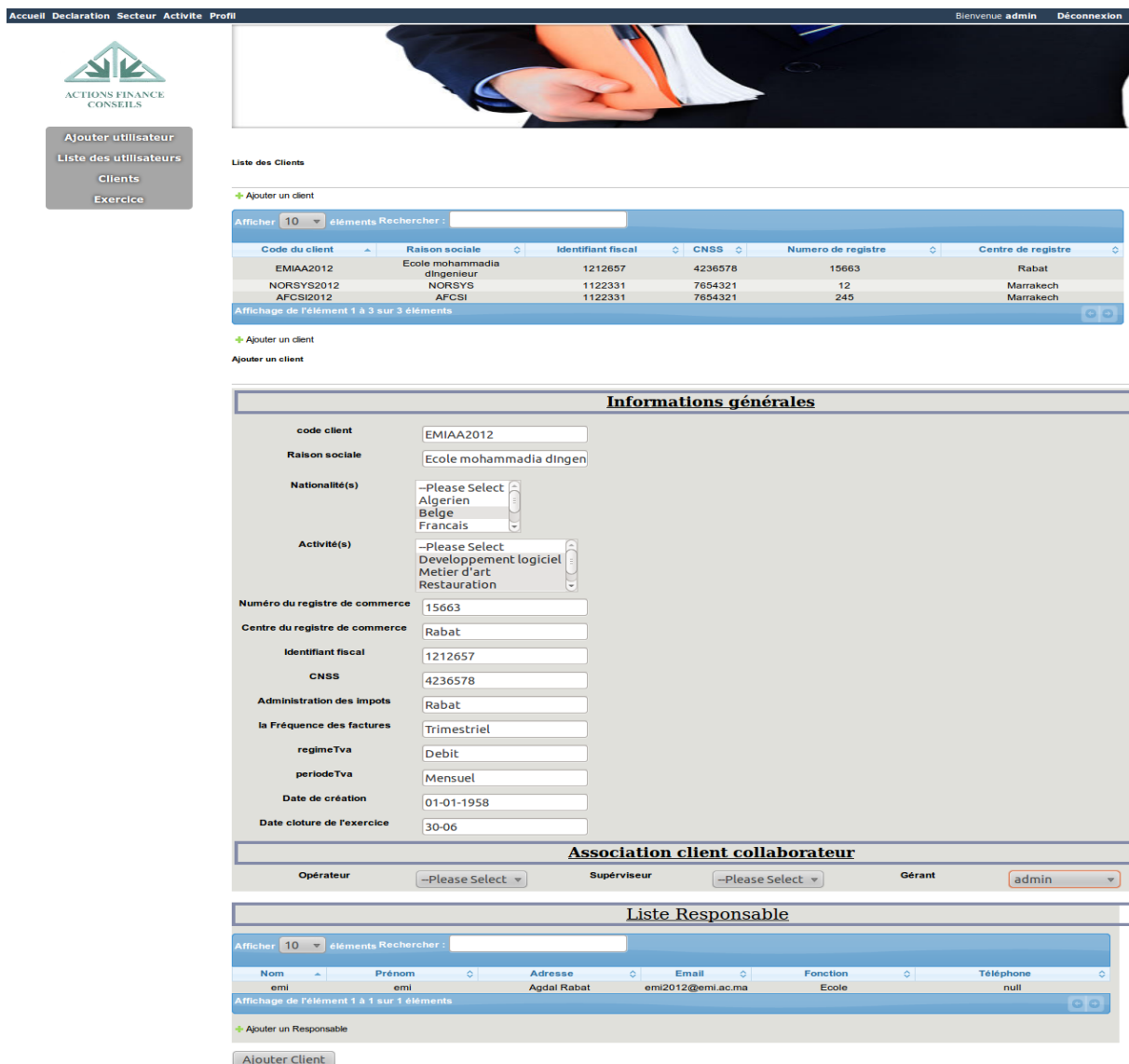
Mot de passe

Valider Annuler

Figure 34 : PAGE D'AUTHENTIFICATION

2.3 Gestion client

L'utilisation de Tiles et AJAX facilite l'aspect visualisation des différentes fonctionnalités du client à savoir lister, rechercher, modifier, ajouter, et archiver Tous ses éléments sont présentés dans une seule page web.



Accueil Declaration Secteur Activite Profil Bienvenue admin Déconnexion

Ajouter utilisateur
Liste des utilisateurs
Clients
Exercice

Liste des Clients

Ajouter un client

Afficher 10 éléments Rechercher :

Code du client	Raison sociale	Identifiant fiscal	CNSS	Numero de registre	Centre de registre
EMIAA2012	Ecole mohammadia d'ingenieur	1212657	4236578	15663	Rabat
NORSYS2012	NORSYS	1122331	7654321	12	Marrakech
AFGSI2012	AFCISI	1122331	7654321	245	Marrakech

Affichage de l'élément 1 à 3 sur 3 éléments

Ajouter un client

Ajouter un client

Informations générales

code client: EMIAA2012

Raison sociale: Ecole mohammadia d'ingen

Nationalité(s): -Please Select, Algerien, Belge, Francais

Activité(s): -Please Select, Developpement logiciel, Metier d'art, Restauration

Numéro du registre de commerce: 15663

Centre du registre de commerce: Rabat

Identifiant fiscal: 1212657

CNSS: 4236578

Administration des impots: Rabat

la Fréquence des factures: Trimestriel

regimeTva: Debit

periodeTva: Mensuel

Date de création: 01-01-1958

Date cloture de l'exercice: 30-06

Association client collaborateur

Opérateur: -Please Select, Supérieur: -Please Select, Gérant: admin

Liste Responsable

Afficher 10 éléments Rechercher :

Nom	Prénom	Adresse	Email	Fonction	Téléphone
emi	emi	Agdal Rabat	emi2012@emi.ac.ma	Ecole	null

Affichage de l'élément 1 à 1 sur 1 éléments

Ajouter un Responsable

Ajouter Client

Figure 35 : LISTE DES CLIENTS ET SON FORMULAIRE D'AJOUT

Chapitre 5 : Réalisation

Le contrôle des champs est effectué par un validator Hibernate qui explique pour chaque champ la valeur attendue si non l'action appliquée n'est pas mise en place.


CNSS	<input type="text" value="765432"/>	<i>doit être plus grand que 1000000</i>
Administration des impots	<input type="text" value="Marrakech"/>	
la Fréquence des factures	<input type="text" value="Mensuel"/>	
regimeTva	<input type="text" value="debit"/>	
periodeTva	<input type="text" value="Mensuel"/>	
Date de création	<input type="text" value="-05-1994"/>	<i>Invalid Date.</i>
Date cloture de l'exercice	<input type="text" value="20-07"/>	

Figure 36 : VALIDATOR CLIENT

2.4 Gestion des exercices :

Un utilisateur qu'il soit gérant ou opérateur peut consulter la liste des exercices d'un client. Pour le faire, il doit cliquer dans le menu à gauche sur le lien « exercice » puis choisir un client depuis la liste déroulante. Un tableau s'affiche contenant la liste des exercices. L'utilisateur peut sélectionner ensuite un exercice pour afficher sa liste de déclarations. La figure ci-dessous montre la liste des déclarations de l'exercice courant du client NORSYS :

Accueil Declaration Secteur Activite Profil Bienvenue admin Déconnexion



Ajouter utilisateur
Liste des utilisateurs
Clients
Exercice

Choisir un client

Raison sociale

La liste des exercices du client NORSYS

Date debut exercice	Date cloture exercice	Statut
21-07-2012	20-07	curent

Affichage de l'élément 1 à 1 sur 1 éléments

Liste des Déclarations

Modifier

Nom de D?claration	Fr?quence	Ech?ance	R?ference
TVA	Trimestriel	Avant le 20 du mis qui suit le trimestre d'clar	ADC100F-10E
Taxe sur les produits d'action	Mensuel	Le mois qui suit le mois de paiement des dividendes	TPA

Affichage de l'élément 1 à 2 sur 2 éléments

Figure 37 : LISTE DES DECLARATIONS DE L'EXERCICE COURANT DU CLIENT NORSYS

Chapitre 5 : Réalisation

L'utilisateur peut aussi modifier les déclarations d'un exercice donné en ajoutant de nouvelles déclarations ou en supprimant quelques unes. La figure ci-dessous montre le formulaire responsable de cette modification :

The screenshot shows the user interface of the 'ACTIONS FINANCE CONSEILS' application. At the top, there is a navigation bar with 'Accueil', 'Déclaration', 'Secteur', 'Activité', and 'Profil' on the left, and 'Bienvenue admin' and 'Déconnexion' on the right. Below the navigation bar is a logo for 'ACTIONS FINANCE CONSEILS' and a sidebar menu with 'Ajouter utilisateur', 'Liste des utilisateurs', 'Clients', and 'Exercice'. The main content area is titled 'Choisir un client' and shows a dropdown menu for 'Raison sociale' with 'NORSYS' selected. Below this is a section titled 'La liste des exercices du client NORSYS' which contains a table with columns for 'Date debut exercice', 'Date cloture exercice', and 'Statut'. The table shows one row with values '21-07-2012', '20-07', and 'curent'. Below the table is a section titled 'Modifier un exercice' with input fields for 'Date debut exercice' (21-07-2012), 'Date cloture exercice' (20-07-2013), and 'Statut' (curent). There is also a dropdown menu for 'Declarations' with options: '-Please Select', 'TVA', 'IS', and 'Taxe de promotion touristique'. A 'Modifier' button is located at the bottom of this section.

Figure 38 : MODIFICATION DES DECLARATIONS DE L'EXERCICE COURANT DU CLIENT NORSYS

3. Conclusion :

Nous avons vu dans ce chapitre l'architecture logicielle que nous avons utilisée pour réaliser l'application AFC, puis nous avons présenté certaines pages que nous avons développées, Dans ce qui suit, nous allons faire une conclusion générale et perspectives ensuite vous trouverez la liste des annexes.

CONCLUSION GENERALE ET PERSPECTIVES :

La réalisation de ce travail a permis de concrétiser les objectifs fixés au début du projet. En effet, notre mission a consisté en l'étude, la conception et la réalisation des briques fonctionnelles d'un système d'information de l'entreprise Actions-Finance-Conseils, en passant par les différentes étapes du cycle de développement d'un projet depuis la rédaction du cahier des charges jusqu'à l'intégration dans le système.

Pour réaliser ce projet, nous avons adopté la méthode de développement Scrum. Nous avons commencé par une étude de l'existant avant d'aborder les deux branches fonctionnelle et technique du cycle de développement. Au terme de ces deux branches, nous avons établi une architecture fonctionnelle et logicielle répondant aux besoins et aux exigences du système. Lors de la phase de conception, nous avons présenté les différents diagrammes UML pour chaque Sprint pour mieux comprendre la relation entre les différents objets du projet. Enfin, nous avons mis en œuvre notre solution de gestion client et suivi de document.

Ce stage que nous avons effectué au sein du groupe NORSYS nous a donné l'occasion de faire le lien entre nos connaissances académiques et le monde professionnel. D'une part, il nous a permis de développer nos compétences techniques, d'approfondir nos connaissances théoriques et les mettre en pratique. D'autre part, l'environnement de travail nous a permis d'améliorer notre savoir-faire et notre rigueur et aussi notre professionnalisme. Cette expérience a aiguisé nos capacités d'analyse et de synthèse et nous avons eu l'opportunité de renforcer nos connaissances concernant la méthode agile Scrum et l'architecture J2EE.

Il était également une occasion pour nous de consolider et de forger nos compétences en matière de développement informatique.

En perspective pour ce projet, AFC envisage d'étendre l'application en développant également une fonctionnalité de statistique qui permettra de faire une analyse décisionnelle pour la répartition de ces clients selon les secteurs et aussi selon chaque ville.

Bibliographie

[1] Claude Aubry. SCRUM Le guide pratique de la méthode agile la plus populaire, 2010 DUNOD.

[2] Gary Mak. Spring par l'exemple, 2008 PEARSON.

Webographie

[site1] : <http://static.springsource.org/> : dernière date de consultation : 30-05-2012, site officiel du Framework Spring STS.

[site2] : <http://tiles.apache.org/index.html> dernière date de consultation : 30-05-2012, site officiel du Framework Tiles qui simplifie le développement des interfaces des applications web en créant des Templates pour chaque page.

[site3] : <http://www.vaannila.com/hibernate/hibernate-tutorial/hibernate-tutorial.html> : dernière date de consultation 30-05-2012, site présentant des Tutoriels pour la manipulation des annotations Hibernate.

[site4] : <http://logging.apache.org/log4j/> : dernière version 30-05-2012, site officiel du Framework Log4j, c'est une bibliothèque de journalisation pour Java. Apache log4j est un projet de l'Apache Software Foundation qui sert à garder la trace de l'application lors de son exécution.

[site5] : <http://courses.coreservlets.com/Course-Materials/dojo.html> : dernière version 30-05-2012, site offrant des cours et des tutoriels dans différents Frameworks spécialement pour Ajax.

[site6] : <http://hibernate.org/> : dernière date de consultation 30-05-2012, site officiel du Framework Hibernate.

[site7] : <http://checkstyle.sourceforge.net/releasenotes.html> : dernière date de consultation 30-05-2012, site officiel du Plugin CheckStyle.

[site8] : <http://netapsys.developpez.com/tutoriels/java/tests-junit4-spring/> : dernière date de consultation 30-05-2012, un simple tutoriel illustrant les différents tests unitaires JUnit 4.

[site9] : <http://docs.oracle.com/javaee/5/api/javax/persistence/package-summary.html> : dernière date de consultation 30-05-2012, site officiel du Framework Hibernate.

[site10] : <http://196.217.240.202:8080/icescrum/> : dernière date de consultation 30-05-2012, site de l'outil IceScrum de Norsys dont nous travaillons pour gérer le projet.

ANNEXES :

ANNEXE 1 : schématisation du cahier de charge.

ANNEXE 2 : Extrait des comptes rendus.

ANNEXE 3 : Qualification actuelle du projet.

ANNEXE 1

Schématisation du Cahier des charges pour la conception d'un système d'information chez A.F.C

CLIENT

- Code client
- Raison sociale
- Forme juridique
- Nationalités des associés
- Numéro (s) de la taxe professionnelle
- Numéro et Centre de RC
- L'identifiant fiscal
- Numéro d'affiliation CNSS
- Activité (s)
- Secteur d'activité
- Régime TVA : débit ou encaissement
- Période TVA : mensuelle ou trimestrielle
- Date de création
- Date de clôture de l'exercice
- Contacts (Nom et prénom ,fonction)
- Administration des impôts
- Montant annuel des honoraires
- Fréquence de facturation

COLABORATEUR

- N°d'immatriculation
- Nom
- Prénom
- Situation familiale
- Adresse
- N° téléphone
- N° de CIN
- N° de CNSS
- Type de contrat
- Date d'embauche
- Date de sortie
- Congés
- Absences
- Retards
- Observations

Etapes de traitement d'une déclaration

- Le client n'a pas fournis les pièces,
- Le client a fournis les pièces partielles (Etat bloquant)
- Le client a fournis les pièces partielles (Etat non bloquantes)
- Le client a fournis toutes les pièces,
- La déclaration en cours,
- La déclaration arrêtée,
- La déclaration en instance de signature,
- La déclaration est signée : en instance d'envoi,
- La déclaration en cours de dépôts par xxxx,
La reçu de dépôt de déclaration est classée et archivé,

ANNEXE 1

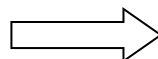
➤ Listes des déclarations

DECLARATIONS	FREQUENCES	ECHEANCE	REFERENCES
TVA : MENSUELLE TRIMESTRIELLE	Mensuelle ou Trimestrielle	Avant le 20 du mois qui suit le mois ou le trimestre déclaré	MODEL ADC100F-10E MODEL ADC080F-10E
CNSS : Bordereau des salaires Bordereau de paiement	Mensuelle	Avant le 10 du mois qui suit le mois déclaré	BORDEREAU DE CNSS
AMO : Bordereau de paiement	Mensuelle	Avant le 10 du mois qui suit le mois déclaré	BORDEREAU D'AMO
Impôt sur le revenu : IR	Mensuelle	Fin su mois déclaré	MODEL ADP050F-09I
Cotisation minimale : CM	Annuelle	Fin de l'exercice	MODEL RSP 010F-08E
IS : Déclaration du Résultat Fiscal Bordereau – Avis de versement	Annuel Trimestrielle	Fin de l'exercice Fin de trimestre	MODEL ADM020F-07E MODEL RSM010F-07E
Déclaration du revenu global des personnes physiques	Annuelle	Avant fin février	MODEL ADP010F-10E
Impôt sur le Revenu: déclaration des traitements et salaires	Annuelle	Avant fin février	MODEL ADC040F-10E
Taxe professionnelle, Taxe de services communaux : déclaration des éléments imposables (TP)	Annuelle	Avant le 31 janvier	MODEL ADC062F- 08E
Taxe de promotion touristique (TPT)	Trimestrielle	Avant la fin du mois suivant le trimestre	MODEL TPT
Taxe de séjour (TS)	Trimestrielle	Avant la fin du mois suivant le trimestre déclaré	MODEL TS
Taxe début de boisson (TDB) : Déclaration de paiement Déclaration annuelle	Trimestrielle et annuelle	Avant la fin du mois suivant le trimestre déclaré Avant le 31 janvier	MODEL TDB

Taxe sur les produits d'action (TPA) Déclaration de versement Déclaration annuelle	Mensuelle Annuelle	Le mois qui suit le mois de paiement des dividendes Avant le 3ème mois qui suit la clôture de l'exercice	MODEL TPA
Assemblée générale ordinaire	Annuelle	Dans les six mois qui suivent la clôture de l'exercice	
Dépôt Etat de synthèse au tribunal	Annuelle	Dans les 30 jours qui suivent la tenue de l'AGO	

➤ **Suivi des AGO**

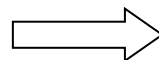
- PV AGO



- Non encore fait
- Fait (date)
- Remis au client le
- Retourné par le client le

➤ **Mise à jour de la saisie par exercice et par mois**

- Journal Achat
- Journal Vente
- Journal Caisse
- Journal Banque
- Journal Opérations diverses

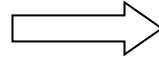


- Non encore fait
- Fait (date)

ANNEXE 1

➤ Editions et mises à jour annuelles : (fréquence : annuelle).

- Grand livre
- Balance
- Journaux auxiliaires
- Journal général
- Mises à jour livre journal
- Mises à jour livre d'inventaires
- Mises à jour livre des assemblées
- Mise à jour du registre des réunions du CA
- Mises à jour registre des transferts d'actions



- Non encore fait
- Fait (date)

➤ Règles de gestion :

A. D'une entreprise cliente :

- Chaque client est associé à un code unique.
- Chaque client est identifié par un identifiant fiscal unique.
- Un client peut avoir un ou plusieurs numéros de taxe professionnelle.
- Chaque numéro de taxe professionnelle correspond à une adresse.

ANNEXE 1

- Pour chaque client il y'a des responsables à contacter (nom/prénom/fonction/ numéro de téléphone).
- Pour chaque client on veut avoir un historique d'événements correspondant à une date et un commentaire.

B. D'un collaborateur :

- Un collaborateur peut s'absenter à une date donnée.
- L'absence peut avoir une justification ou non.
- Un collaborateur peut être en retard à une date donnée pour une durée déterminée.
- Le retard peut avoir une justification ou non.
- Un collaborateur a droit à un congé caractérisé par la date de début et la date de fin du congé, et le type de congé.
- Pour chaque collaborateur il pourrait y avoir des observations concernant des événements à des dates données.

C. Suivi des déclarations :

- Chaque dossier correspond à une entreprise.
- Chaque dossier est composé par plusieurs exercices.
- Chaque dossier est géré par un gérant, supervisé par un superviseur et traité par un opérateur.
- Chaque dossier contient plusieurs journaux à saisir chaque mois, dès qu'un journal est saisi sa date d'établissement doit être mentionnée.
- Les journaux correspondant à chaque dossier sont : achats, ventes, banques, caisses, opérations diverses...
- Un sauvegarde concerne un client pendant un exercice

D. Suivi des dossiers juridiques :

- L'élaboration d'un PV AG pour chaque dossier dans les six mois qui suivent chaque exercice.
- La mise à jour des livres légaux concerne chaque client à la fin de chaque exercice

ANNEXE 2

➤ Extrait des comptes rendus :

Marrakech, le 19/03/2012

COMPTE RENDU 1^{ERE} REUNION

Responsable de projet :

- ⊕ Karim Tammar : Chef de projet
- ⊕ Vincent Groux : Responsable de formation & directeur du projet
- ⊕ Abdeljalil Aitikkene : développeur
- ⊕ Adil Jirari : développeur

Sujet : Découvrir L'environnement de travail au sein d'AFC

Dans la réunion que nous avons effectuée avec M. Mohamed ASSA et M. Nouredine HALOUI le 19/03/2012 et qui a durée de 17h à 19h 15, cette visite a pour objectifs :

✓ **Présentation de l'entreprise AFC :**

Nous avons commencé par un aperçu de l'espace de travail pour comprendre le processus de suivi des dossiers depuis l'accueil du client jusqu'à l'archivage.

Présentation des différents services tel que :

- ⊕ Service production
- ⊕ Service secrétariat
- ⊕ Service d'archivage ce dernier est découpé en deux :
 - l'archivage des dossiers d'exercice courant
 - l'archivage des dossiers d'exercices n-1

✓ **Présentation du logiciel utilisé par les collaborateurs:**

M. Assa nous a expliqué les méthodes de travail ainsi les logiciel utilisés tel que « Omni comptable » qui fournit divers fonctionnalités afin de faciliter le travail des utilisateurs.

La majorité des informations sont stockées sur des fichiers de type Excel, ce qui pose des problèmes en terme de sécurisation des données, d'informations non centralisées, non normalisées : chaque gestionnaire a son propre format de fichier de stockage. L'ensemble de ces raisons a créé le besoin de développer un Système d'Information pour le suivi des clients.

Nous avons clôturé ce meeting par la programmation d'une réunion pour le démarrage du premier Sprint qui aura lieu le Vendredi 23-03-2012 à 17h00, chez Norsys Afrique.

ANNEXE 3

➤ Qualification du projet AFC par Sonar

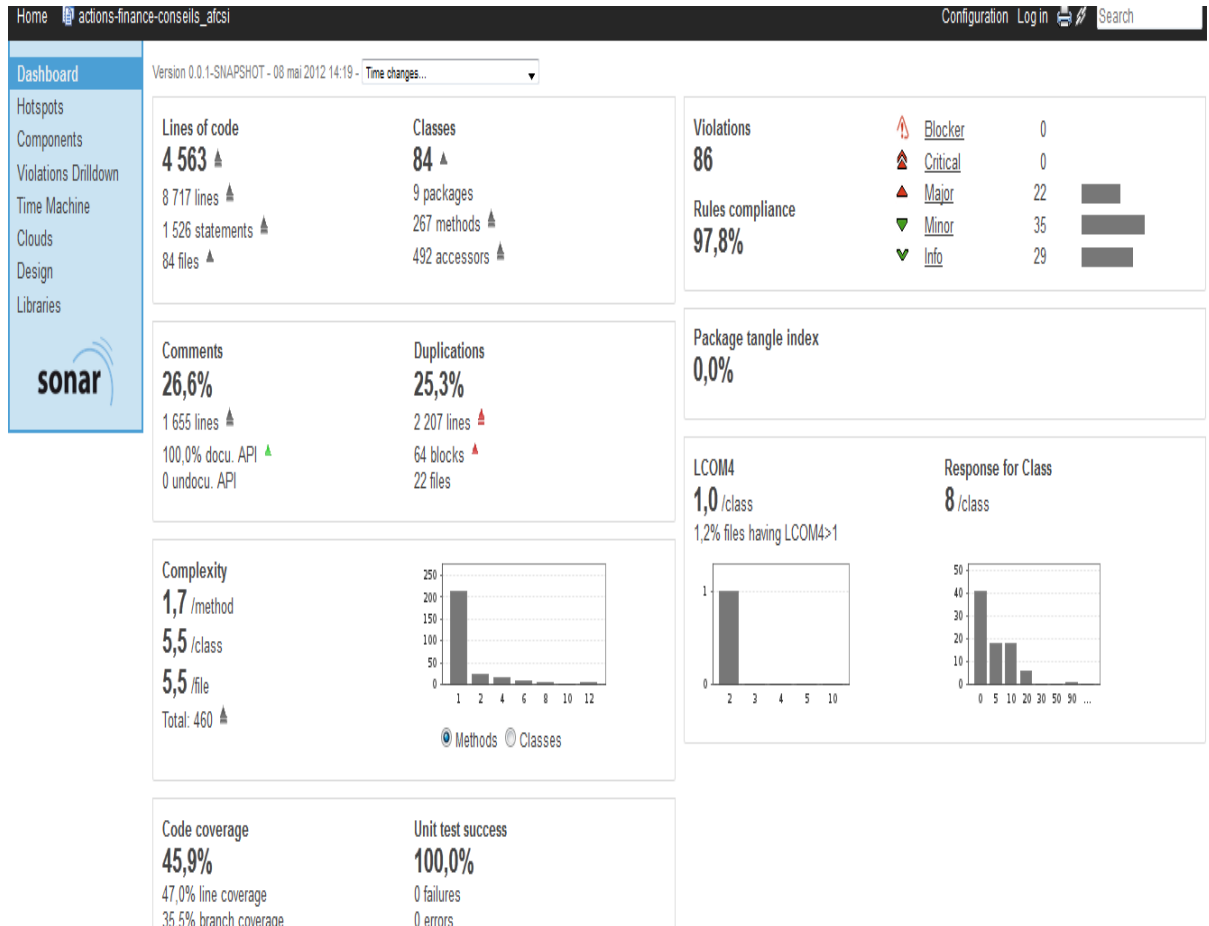


Figure 39 : STATISTIQUE SONAR DU PROJET AFC

C'est une snapshot (20/04/2012) de l'outil de qualification SONAR appliqué à notre projet AFC qui contrôle le rendement de notre projet au niveau :

- Violation : 86
- Commentaire : 26,6%
- Nombre de classe : 84
- Couverture des tests : 45,9%.
- Réussite des tests unitaires : 100%