Stage assistant ingénieur en développement logiciel

Rapport de stage deuxième année

Elève : Antônio Guilherme Ferreira Viggiano

Tuteur pédagogique : M. Pascal Préa
Tuteur de stage : M. Clemenceau Roberto Da Silva

Stage realisé du 3 Juin au 26 Juillet
(8 semaines)

Remerciements

Je voudrais remercier à tous ceux qui ont contribué pour le bon déroulement de ce stage, en particulier à :

Mme Mirta JUAREZ et Mme Cécile LOUBET – pour les réponses rapides à toute question concernant le stage à l'international.

M. Marco Aurélio RABELO – président-directeur général de Asert, pour être si proche des employés.

M. Clemenceau Roberto DA SILVA – directeur de technologie de l'entreprise, qui m'a bien accueilli par email et personnellement au sein de l'entreprise.

M. Thiago RODRIGUES PEREIRA – architecte logiciel, pour m'avoir formé dans le framework de l'entreprise et m'avoir introduit aux bonnes pratiques en génie logiciel.

Résumé

Ce rapport décrit un stage technique d'assistant ingénieur d'une durée de deux mois au sein de Asert Serviços e Tecnologia da Informação (Asert Services et Technologies de l'Information). Cette entreprise de petite taille située à Goiânia, la capitale de l'état brésilien Goiás, offre des solutions de développement de technologies et externalisation de services informatiques, et est spécialisée en services d'aide médicale.

Le poste qui m'a été affecté a été celui de développeur logiciel dans le Centre de Développement de Services de la société, où j'ai travaille essentiellement avec les langages de programmation Java et Flex. Ces technologies ont été utilisées pour l'élaboration d'un portail web Flash de gestion de personnes adhérentes à la société d'assurance mutuelle Celgmed, une mutuelle de santé destinée aux employés de CELG, la Compagnie Énergétique de Goiás, similaire à Électricité de France S.A.

Le mode de fonctionnement de l'entreprise, sa taille, son marché et ses produits seront également présentés en détail. En outre, une vaste description des processus de développement logiciel, du cycle de vie du produit et de l'architecture logiciel sera abordé à la fin, juste avant le descriptif du produit développé au cours du stage. Les remarques personnelles de l'élève et l'apport de cette expérience seront discutés dans la conclusion de ce rapport.

Abstract

This report describes a technical assistant engineering internship of a two months period at the Asert Serviços e Tecnologia da Informação company (Asert Information Technology Services). This small company is located in Goiânia, the capital of the Brazilian state Goiás, provides development solutions and outsourcing of IT services, and is specialized in medical assistance services.

The position that was assigned to me was that of software development intern in the Development Services Centre of the company, where I worked with the Java and Flex programming languages. These technologies were used for the development of a Flash web portal used to manage people adhering to the Celgmed insurance company, a health insurance for employees of CELG, the Energy Company of Goiás, similar to Electricité de France SA.

The business operation model of the company, its size, its market and its products will be presented in detail further on. In addition, an extensive description of the software development process, product life cycle and software architecture will be discussed at the end, just before the description of the product developed during the internship. Personal observations of the student and the contribution of this experiment will be discussed in the conclusion of this report.

Mots clefs

- Stage assistant ingénieur
- Asert Serviços e Tecnologia da Informação
- École Centrale Marseille
- Externalisation de services
- Systèmes de gestion intégrés
- Audit en services d'aide médicale
- Portail web
- Java
- Flex
- Flash

Glossaire

Eclipse – Environnement de développement intégré multi-langage comprenant un espace de travail de base et une extensible système plug-in de personnalisation de l'environnement. Écrit en Java, il peut être utilisé pour développer des applications en Java et d'autres langages de programmation. Page: 14

ERP – De l'anglais *Entreprise Ressource Planning*, est un progiciel qui intègre les principales composantes fonctionnelles de l'entreprise : gestion de production, gestion commerciale, logistique, ressources humaines, comptabilité, contrôle de gestion. À l'aide de ce système unifié, les utilisateurs de différents métiers travaillent dans un environnement applicatif identique qui repose sur une base de données unique. . Page: 14

Java EE – La plate-forme informatique Java d'Oracle destinée aux entreprises. La plate-forme fournit une API et un environnement d'exécution pour l'élaboration et la mise en œuvre du logiciel d'entreprise, y compris les services de réseau et web, et d'autres applications de réseau à grande échelle, multi-niveaux, évolutive, fiable et sécurisée.. Page: 15

SGBD – Système de Gestion de Base de Données, est un logiciel système destiné à stocker et à partager des informations dans une base de données, en garantissant la qualité, la pérennité et la confidentialité des informations, tout en cachant la complexité des opérations.. Page: 15

Table des matières

Chapitre 1

Introduction

L'École Centrale Marseille propose dans son cursus aux élèves de découvrirent le monde économique et ses contraintes et de bien appréhender la complexité du métier d'ingénieur, au travers d'un stage en entreprise d'une durée de 8 semaines au minimum. Celle-ci est une opportunité extrêmement enrichissante dans plusieurs aspects, complémentaire au stage ouvrier de la première année.

Ce rapport présente les expériences dans l'établissement Asert Serviços e Tecnologia da Informação, une petite société informatique qui propose des solutions de systèmes de gestion intégrés à d'autres entreprises, principalement à celles d'aide médicale.

Le déroulement de ce stage a été du 3 juin 2013 au 26 juillet 2013, avec une durée totale de huit semaines, où j'ai occupé le poste de développeur logiciel dans le Centre de Développement de Systèmes à Asert.

Ce document a pour objectif de montrer de façon détaillée les expériences vécues lors du stage, et pour cella il est divisé en trois parties : la présentation de l'entreprise, la description du poste occupé, avec une mise en évidence du processus de développement logiciel, du cycle de vie du produit et de son architecture, et la conclusion contenant les remarques personnelles de l'élève en ce qui concerne cette expérience.

Chapitre 2

Présentation de l'entreprise

2.1 Historique

Basée dans la capitale de Goiás, Asert a été fondée en 2002 par deux employés de Evoluti Tecnologia e Serviços (Evoluti Technologie et Services) avec l'idée de fournir des systèmes de gestion au secteur de la santé, un domaine qui ne possédait pas les compétences techniques nécessaires à l'intégration de tous ses services. Tout de même, Asert n'a éte qu'une branche rattachée à Evoluti jusqu'en 2008, lorsque l'entreprise a réussi à avoir suffisamment de projets indépendants.

De 2008 à 2010, la société a travaillé fondamentalement avec des systèmes pour des mutuelles de santé. Récemment, suivant les tendances du marché, l'organisation a élargi son champ d'application jusqu'à l'externalisation de tout type de service spécialisé.

2.2 Implantation et Taille

Asert est une micro-entreprise localisée dans la région commerciale de Goiânia, dans un bureau d'approximativement 110 m^2, actuellement avec un effectif de 12 employés. En 2010, grâce à un projet en partenariat avec l'état de Goiás, l'entreprise comptait 200 personnes, y compris des médecins, des auditeurs et consultants, des professionnels de TI, etc. Cependant, à la fin de l'engagement, Asert a dû mettre à terme les contrats de travail à durée déterminée et rester avec son équipe principale.

2.3 Type de structure

L'entreprise se partage en deux secteurs, le Centre de Développement de Systèmes, où quatre développeurs sont gérés par un chef d'équipe, et le Centre Commercial-Financier, où des commerciaux, compteurs et administrateurs sont dirigés par le directeur financier.

L'équipe commerciale se charge de trouver de nouveaux clients et de faire une analyse préalable de la situation de l'entreprise. Le chef de projets du centre de développement fait ensuite l'analyse technique du système de gestion souhaité, en prenant note de toutes les particularités spécifiques aux règles métier (ou "business

rules" en anglais) de l'application. Une fois le besoin du client est complètement identifié, l'équipe de développement décide la structure "physique" du système – quelle base de données relationnelle utiliser, quelle langage de programmation ou technologie la plus adaptée au problème, etc. – et passe à l'étape de codage.

L'équipe financière travaille en parallèle de toutes ces étapes, en analysant les coûts de travail et d'équipements.

2.4 Gamme de produits

Même si Asert est spécialisée en services de santé, le groupe travaille sur deux grandes gamme de projets, ceux de technologies et de l'information et ceux de systèmes intégrés [1].

2.4.1 Technologies de l'information

Dans l'axe des sciences de l'information et de la communication, l'entreprise propose des services de audit et conseil, de développement de systèmes, d'informatique décisionnelle (en anglais *Business Intelligence*) et de gestion.

Audit et conseil

Le rôle du conseil en technologies de l'information est de fournir au client un bilan de la situation actuelle de l'entreprise, d'identifier les problèmes liés à la politique, à la structure, aux procédures et aux méthodes, afin de recommander et d'aider à mettre en œuvre les mesures appropriées aux étapes d'innovation et de croissance de l'entreprise. Après ce diagnostic fondé sur le besoin du client et les résultats attendus, Asert s'occupe du développement de solutions spécialisées et de projets techniques, avec des professionnels qualifiés au travail. Le service de *consulting* comprend :

- Mise à niveau et modernisation de la technologie de la société ;
- Évaluation, sélection et embauche de services et logiciels tiers ;
- Préparation de projets de logiciels ;
- Gestion de projets.

Développement de systèmes

Ce n'est pas toujours que les systèmes de gestion disponibles sur le marché répondent aux demandes des organisations. C'est en vue de cela que Asert propose des systèmes sur mesure, en produisant des outils personnalisés en fonction des besoins des clients, de manière à assurer une plus grande productivité et amélioration de la gestion internet des entreprises.

Asert utilise les plus modernes méthodologies de développement logiciel, utilisant des méthodes de contrôle de version et de testes unitaires afin d'avoir un rendu selon les exigences et les besoins des clients.

Informatique décisionnelle – *Business Intelligence*

Les systèmes d'information sont responsables par la manipulation d'une quantité de données très importante au sein des entreprises. L'informatique décisionnelle, aussi connue par la traduction anglaise "intelligence d'affaires", est le processus de collecte, d'organisation, d'analyse, de partage et de suivi de ces données, afin d'extraire des informations et des indicateurs pour soutenir la gestion de l'établissement. Les services B.I. de Asert incluent :

- Analyse et conception d'environnements de gestion de l'information ;
- Analyse de la qualité des données ;
- Gestion des données ;
- Rendu de rapports dynamiques destinés aux gestionnaires de l'entreprise.

Gestion des T.I.

L'équipe de Asert est axée sur la satisfaction du client, prête à résoudre les difficultés dans l'accès ou l'utilisation des technologies de l'information. Elle fournit également aux utilisateurs un centre d'assistance pour résoudre à tout problème technique lié à ses services, afin de rétablir le fonctionnement normal des activités dès que possible, minimisant ainsi les impacts commerciaux causés par des pannes informatiques.

La gestion des technologies de l'information apporte plusieurs avantages aux organisations, comme la construction d'un lien entre les T.I. et la gestion d'entreprise, la réduction des coûts, l'acheminement des appels aux équipes spécialisées, le soutien aux utilisateurs finaux et l'amélioration de la qualité des services.

2.4.2 Systèmes Intégrés

Les systèmes intégrés de l'entreprise se partagent en deux secteurs : la médecine, celle-ci divisé en Asert Santé et Asert Méd, et l'industrie, avec Asert Cycle.

Asert Santé

Asert Santé est le type de système informatisé pour la gestion des sociétés d'assistance médicale. Développé avec les technologies les plus modernes, le système se traduit par un portail web comme interface d'administration d'utilisateurs et d'adhérents, et de bases de données relationnelles qui manipulent toutes les informations. Ces systèmes sont en conformité avec les spécifications de l'Agence Nationale de Santé brésilienne.

Ces caractéristiques techniques, alliées à une interface conviviale et intuitive, rend le système facile à utiliser, sans demander beaucoup de ressources matériels. Techniquement, le système est divisé en modules indépendants, qui sont intégrés selon la réalité et l'organisation de l'entreprise de santé.

Les particularités de Asert Santé sont :

- Modules intégrés ;
- En accord avec les normes de l'Agence Nationale de Santé brésilienne ;
- Rapidité, efficacité et fiabilité dans les tâches effectuées ;

– Vérification électronique ;
– Normalisation des tables de facturation et des rapports de gestion ;
– Facturation électronique.

Asert Méd

Asert Méd est un outil idéal pour la gestion des cliniques, des hôpitaux, des laboratoires et des bureaux de médecin. Ce système vise à faciliter le service clients, la facturation et la présentation des comptes, adapté aux besoins du client.

Caractéristiques et avantages :

– Modules intégrés ;
– En accord avec les normes de l'Agence Nationale de Santé brésilienne ;
– Enregistrement unique des patients ;
– Assistance aux patients affiliés ou adhérents à une mutuelle de santé ;
– Émission de demandes de visites médicales dans les modèles des normatifs ;
– Contrôle des rapports médicaux personnalisés et de factures de soins médicaux et hospitaliers ;
– Contrôle des stocks multiple (entrepôt, pharmacie, soins infirmiers, etc.) ;
– Gestion financière ;
– Graphiques et rapports de gestion ;
– Dossier médical électronique d'un patient.

Asert Cycle

Asert Cycle est un système développé avec les dernières technologies sur le marché pour la gestion d'industries, permettant de façon innovante le contrôle, le suivi et l'intégration des politiques de management, de production, de commerce et fiscales des entreprises. Ce système se caractérise par :

– Gestion de stock ;
– Suivi du produit dans toutes les étapes de fabrication ;
– Expédition et vente des produits ;
– Émission de la facture ;
– Gestion de la production.

2.5 Marché

2.5.1 Le marché des technologies de l'information et communication au Brésil

Le marché des TIC au Brésil est composé de plus de 90 % de petites et moyennes entreprises et est actuellement en croissance, même si cela n'est pas un secteur traditionnel de l'économie du pays. En 2010, les technologies de l'information et communication ont été responsables par 102,6 milliards de dollars, environ 3% [2] du produit intérieur brut [3] de l'année.

2.5.2 La position de Asert dans le marché

Dans les années 1990 et 2000, le boom des progiciels de gestion intégré (ou en anglais ERP, "Enterprise Resource Planning") a encouragé la création de plusieurs entreprises de de logiciels en tant que service (de l'anglais "Software as a Service", SaaS) dans l'état de Goiás, tels que Canion Software, Interagi Tecnologia, et Apta. Cependant, ce phénomène de haute concurrence a fait tomber les prix des ERPs et a empêché la croissance des petites et moyennes entreprises de technologie.

Ce que l'on constate aujourd'hui est le retour aux systèmes personnalisées. Les clients sont plus exigeantes dans les systèmes adaptés et ne veulent plus de logiciels génériques, souvent critiqués d'être de difficile adaptation et de ne pas représenter l'identité de l'entreprise. Avec ce changement de valeurs du marché, Asert a gagné de nouveaux clients et a assuré sa position en tant qu'entreprise de développement de solutions individualisées.

Au cours du temps, les projets de Asert ont évolué non pas seulement en taille mais aussi en type. Lorsque l'entreprise était rattaché au groupe Evoluti, le premier projet de Asert a été la mise en ouvre d'un système de gestion pour l'association d'assistance médicale "Ipasgo" (Institut de l'assistance publique de l'Etat de Goias). Ensuite, le système s'est adapté à la ville de Palmas, capitale de l'état de Tocantins. Aujourd'hui ces deux projets sont en phase d'assistance – l'étape finale de toutes les solutions d'externalisation de l'entreprise.

Actuellement, l'entreprise subit des transformations structurelles afin de pouvoir agir comme un moteur d'innovation de l'état. En partenariat avec l'institution gouvernementale FAPEC et avec l'institution d'enseignement et de recherche Université de Goias, Asert est en train de développer un système expert pour la gestion sanitaire de réfrigérateurs pour le client Friboi. Toutes les étapes de production, ainsi comme la qualité de la viande bovine seront analysées avec des réseaux bayésiens afin de minimiser les pertes par contamination bactérienne. Ce projet est encore en cours de conception et n'a pas de date prévue de livraison.

Chapitre 3

Deroulement de la mission

La mission qui m'a été affecte a été celle d'aider dans le développement logiciel d'un progiciel de gestion intégré pour le service d'aide médicale CELGMED.

Au cours des deux premières semaines, l'architecte logiciel responsable par le projet m'a formé dans le framework Asert, c'est-a-dire, des fonctions et librairies qui ont été développées au sein de l'entreprise spécifiques a la mise en œuvre des projets.

Encore dans le premier mois de mon stage, des notions d'architecture logicielle m'ont été présentées, en particulier celle des trois couches et la MVC (Model View Controller), qui seront détaillées par la suite.

Après cette période de formation, j'ai pu participer au développement du système ERP. J'étais charge de l'exécution de plusieurs fenêtres, dès la conception de l'interface graphique jusqu'à la création des entités dans la base de donnes et des classes dans le programme.

Le cycle de développement logiciel n'a pas été un thème de grande importance dans ma mission, vu que l'équipe de production était très limitée et le projet était déjà en phase de codage. Tout de même, le chef de projet m'a expliqué les étapes du processus habituel de l'entreprise, présentées dans les pages qui suivent.

Toutes ces étapes de la mission seront décrites avec plus de détails dans ce chapitre.

3.1 Outils de développement

Les solutions Asert sont presque toujours supportées par les mêmes outils de développement logiciel, dans le modèle suivant :

Microsoft Windows – le système d'exploitation utilisé aussi bien pour le développement que pour l'hébergement des applications.

Adobe Flash Builder – aussi connu comme Adobe Flex [4], l'environnement de développement (IDE) construit en tant qu'une couche sur la plate-forme Eclipse destinée au développement d'applications internet pour la plate-forme Adobe Flash.

GlassFish – un serveur d'applications en stage final de développement qui est supporté par Oracle et la communauté GlassFish en utilisant une licence open

source. GlassFish Server [5] est généralement publié avec le soutien de la dernière plate-forme Java EE, bien en avance sur les autres implémentations de serveurs d'applications.

Microsoft SQL Server – système de gestion de base de données relationnelles qui stocke et de récupère les données demandées par d'autres applications logicielles, que ce soit ceux sur le même ordinateur ou celles en cours d'exécution sur un autre ordinateur dans un réseau, comme l'Internet [6].

Hibernate – framework open source gérant la persistance des objets Java en base de données relationnelle. Hibernate apporte une solution aux problèmes d'adaptation entre le paradigme objet et les SGBD en remplaçant les accès à la base de données par des appels à des méthodes objet de haut niveau [7].

Spring – framework libre qui facilite la construction et définition de l'infrastructure d'une application Java, ainsi comme des tests de routine [8]. Il rend possible l'inversion de contrôle, un patron d'architecture qui fonctionne selon le principe que le flot d'exécution d'un logiciel n'est plus sous le contrôle direct de l'application elle-même (et donc du programmeur qui l'a développé) mais du framework ou de la couche logicielle sous-jacente [9].

BlazeDS – technologie permettant de réaliser du « remoting » et du « web-messaging » tout cela basé sur un serveur en Java. Elle permet notamment de récupérer et d'insérer des données en temps réel pour les technologies Adobe Flex et Adobe Integrated Runtime. BlazeDS fournit, en fait, des services qui permettent de lier une application client et une application serveur afin de récupérer, modifier et insérer des données [10].

JasperReports – un outil de *reporting* open source, offert sous forme d'une bibliothèque qui peut être embarquée dans tous types d'applications Java. Il se base sur des fichiers XML (dont l'extension est en général .jrxml) pour la présentation des états, et est souvent couplé à iReport pour faciliter sa mise en œuvre dans une application Java, classique ou orientée web [11].

iReport Designer – concepteur libre de compte rendus et dossiers pour JasperReports. Il est possible de créer des mises layouts très sophistiqués contenant des graphiques, des images, etc. Les données sont accessibles via XML, Hibernate, et d'autres, et les rapports peuvent être publiés à de nombreux formats, dont PDF, XML, CSV, HTML, DOCX et OpenOffice [12].

3.2 Architecture logiciel

Un motif architectural est une solution générale, réutilisable à un problème qui se pose souvent dans l'architecture logicielle dans un contexte donné. Modèles architecturaux sont souvent documentées dans les modèles de conception de logiciels. [13]

Suivant la conception traditionnelle d'architecture, un « style d'architecture logiciel » est une méthode de construction spécifique, caractérisé par les particularités qui le rendent remarquable. Un style architectural définit une famille de systèmes en termes d'un modèle d'organisation structurelle, un vocabulaire de composants et

de connecteurs, avec des contraintes sur la façon dont ils peuvent être combinés. Les styles architecturaux sont des « paquets » réutilisables de décisions de conception et des contraintes qui sont appliquées à une architecture d'induire des qualités souhaitables choisis. Il existe de nombreux modèles et de styles architecturaux reconnus, parmi lesquels : à base de composants ; application monolithique ; en couches ; etc. [14]

L'architecture des systèmes développés à Asert est souvent un intermédiaire entre celle en couches et la MVC (modèle-vue-contrôleur).

3.2.1 L'architecture trois tiers

L'architecture trois tiers, niveaux ou couches est l'application du modèle plus général qu'est le multi-tier [15]. L'architecture logique du système est divisée en trois niveaux ou couches :

La couche de présentation correspondante à l'affichage, la restitution sur le poste de travail, le dialogue avec l'utilisateur.

La couche métier correspondante à la mise en œuvre de l'ensemble des règles de gestion et de la logique applicative.

La couche d'accès aux données correspondante aux données qui sont destinées à être conservées sur la durée, voire de manière définitive.

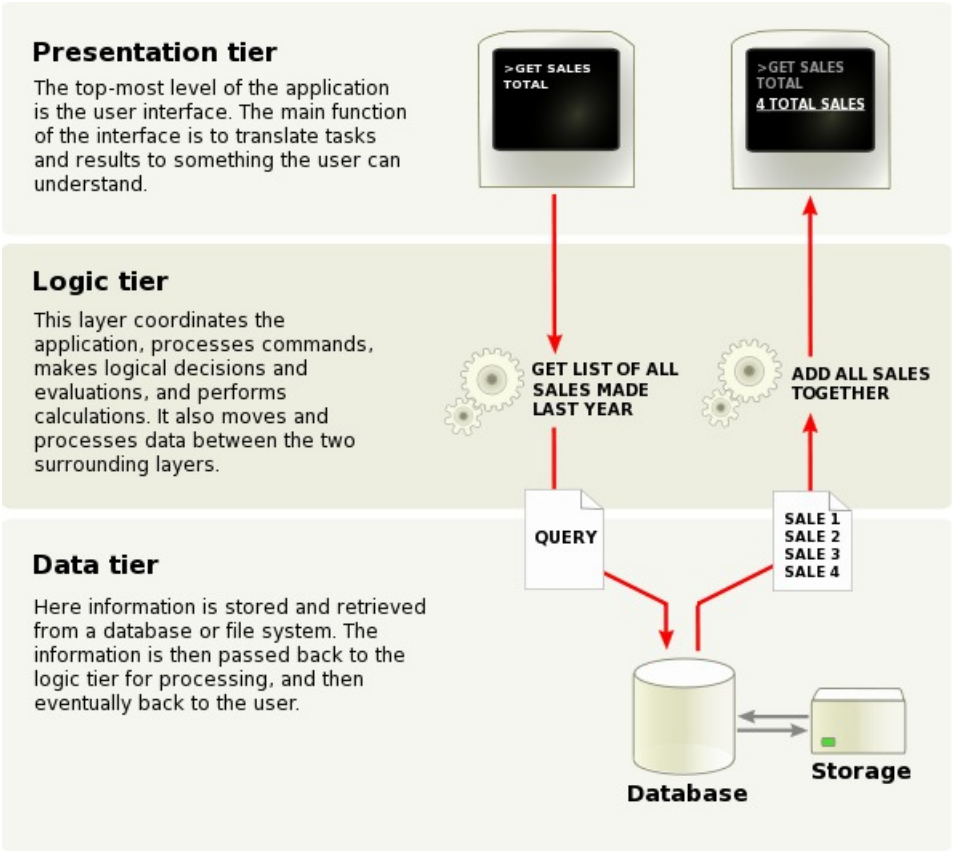

FIGURE 3.1 – Architecture trois couches

Dans cette approche, les couches communiquent entre elles au travers d'un « modèle d'échange », et chacune d'entre elles propose un ensemble de services rendus. Les services d'une couche sont mis à disposition de la couche supérieure. On s'interdit par conséquent qu'une couche invoque les services d'une couche plus basse que la couche immédiatement inférieure ou plus haute que la couche immédiatement supérieure (chaque couche ne communique qu'avec ses voisins immédiats).

Le rôle de chacune des couches et leur interface de communication étant bien définis, les fonctionnalités de chacune d'entre elles peuvent évoluer sans induire de changement dans les autres couches. Cependant, une nouvelle fonctionnalité de l'application peut avoir des répercussions dans plusieurs d'entre elles. Il est donc essentiel de définir un modèle d'échange assez souple, pour permettre une maintenance aisée de l'application.

Ce modèle d'architecture 3-tiers a pour objectif de répondre aux préoccupations tels que l'allègement du poste de travail client, la prise en compte de l'hétérogénéité des plates-formes (serveurs, clients, langages, etc.), l'amélioration de la sécurité des données, en supprimant le lien entre le client et les données, la rupture du lien de propriété exclusive entre application et données et enfin, meilleure répartition de la charge entre différents serveurs d'application [16].

3.2.2 L'architecture Modèle-vue-contrôleur (MVC)

L'architecture MVC est un motif d'architecture qui sépare la représentation de l'information à partir de l'interaction de l'utilisateur avec elle. Le modèle comprend des données d'application, les règles métier, la logique et les fonctions. Une vue peut être n'importe quelle représentation de sortie de données, comme un graphique ou un diagramme. Plusieurs vues des mêmes données sont possibles, comme un graphique à barres pour la gestion et une vue tabulaire pour les comptables. L'entrée de la médiation du contrôleur, le convertir en commandes pour le modèle ou la vue [17].

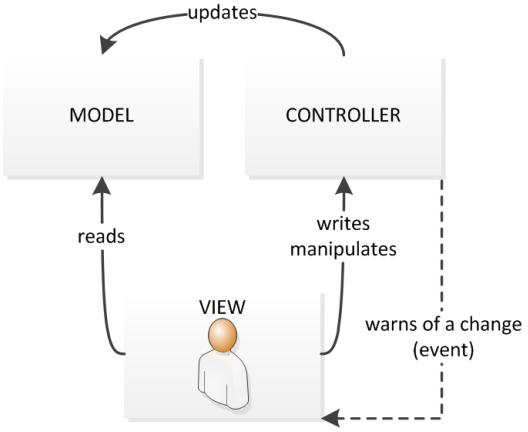

FIGURE 3.2 – Architecture modèle-vue-contrôleur

Souvent confondue avec l'architecture trois couches, la MVC se distingue dans le fait que l'architecture 3-tiers sépare la couche métier de la couche d'accès aux données. Pour qu'une application MVC soit une vraie application 3-tiers, il faut lui

ajouter une couche d'abstraction d'accès aux données de type *DAO* (Data Access Object). Inversement pour qu'une application 3-Tiers respecte MVC il faut lui ajouter une couche de contrôle entre l'interface utilisateur et les règles métier. Loin d'être antagonistes, ces deux pratiques se combinent et sont la fondation de la plupart des frameworks de création d'applications web [18].

3.2.3 Cas pratique dans le système de gestion CELGMED

Dans le système de gestion des adhérents à la société d'assurance des employés de la compagnie énergétique de l'état de Goiás, le système trois couches/MVC est représenté par des classes de Vue, Contrôle, Business et Persistance. D'autres classes de support étaient aussi présentes, mais plus pour une question d'organisation que de style d'architecture, telles que les « Vérificateurs » (utilisées pour assurer le type des variables, intervalle des données numériques ou des dates, etc.) et les « Entités » (par example, une classe Banque est associé en quelque sorte à une entité Rue, Code Postal, etc.).

Comme l'on peut voir, ce style est composé à la fois de classes de Présentation/Vue, de Business/Logique/Modèle, de Données/Persistance et de Contrôle, ce qui assure une gestion de donnés et de développement logiciel plus rigoureux que les deux systèmes séparés : si d'un côté il y a plus d'entités qui détiennent un accès limité aux informations – et donc s'il faut écrire plus de lignes de code –, de l'autre côté cela assure la modularité du programme et la maintenance du code.

De mode général, ce flux de contrôle se passe de la façon suivante :

1. L'utilisateur du programme fait une action dans la **Vue**, ce qui génère un événement
2. Cet événement est capté par Vue et transmis au **Contrôleur** associé
3. Le Contrôleur fait appel au **Business**, qui applique la logique du système selon le cas échéant
   - Les **Vérificateur**s interviennent normalement entre le Contrôleur et le Business, lorsque l'utilisateur fait un action limitée par la conception de la Vue, qui n'a nécessairement pas de rapport avec la logique intrinsèque du système
4. Le Business appelle la **Persistance** pour consulter ou modifier la base de données du programme
5. Une message de confirmation ou d'erreur est transmise dans le flux contraire jusqu'à la Vue et à l'utilisateur.

La Table 3.1 représente ce flux de manière schématique avec l'exemple d'un utilisateur qui veut consulter tous les clients de la base de données.

Flux de contrôle	**Classes concernées**
L'utilisateur clique sur le bouton « Afficher clients »	*Utilisateur* → Vue
La Vue appelle le Contrôleur afin d'interpréter l'événement *clic*	Vue → Contrôleur
Le Contrôleur fait usage du Vérificateur pour voir si toutes les champs de saisie de texte ont été bien remplis (par exemple, n'a-t-il pas mis de caractères de l'alphabet dans le champ « période de consultation » ?) En cas positif, il appelle le Business ; sinon, un message d'erreur se produit et l'utilisateur réécrit les informations.	Contrôleur → (Vérificateur) → Business
Le Business vérifie la logique du système (par example, est-ce que l'utilisateur a le droit d'accès à *tous* les clients de la base de données ?). Ensuite, il appelle la Persistance pour retrouver l'information correspondante	Business → Persistance
La persistance utilise des commandes SQL pour consulter la base de données et retrouver les clients indiqués par le Business (par example, les clients supprimés[1] ne seront pas affichés). La réponse de cette requête est donc renvoyé dans le flux inversé jusqu'à la Vue.	Persistance → ⋯ → Vue
La Vue organise l'information et la rend visible à l'utilisateur, sous forme de présentation mise en forme	Vue → *Utilisateur*

Table 3.1 – Flux de contrôle dans l'architecture des systèmes Asert

3.3 Framework

En dehors des librairies open source et des outils de développement décrits dans la Section 3.1, les projets de l'entreprise étaient développés avec un ensemble cohérent de composants logiciels qui permettaient la réutilisation de code, standardisation du cycle de vie des produits et formalisation de l'architecture logiciel. Cet ensemble a été fait au cours de l'existence de l'entreprise et aujourd'hui compte quelques centaines de fonctions d'aide au développement agile.

Ce framework est utilisé, par example, dans la création de fenêtres graphiques génériques, dans la normalisation de la taille des fenêtres, des couleurs, taille et police des libellés, de la création de boutons qui génèrent toujours le même évènement (p.ex., « Afficher X», « Rechercher Y»), etc.

1. Généralement dans les systèmes d'information, il n'y a pas de suppression *stricto sensu*, car les entrées peuvent être réutilisées ultérieurement. Ce qui est fait est la modification de la valeur indicatrice de suppression : dans la suppression d'un client, sa colonne « Date de suppression » est indiqué avec la date du jour, et lorsque l'on veut consulter tous les clients non supprimés, il suffit d'ajouter à la requête "`WHERE DATE_SUPPRESSION IS NULL`".

En outre que l'organisation interne, la standardisation des entités a aussi une importance dans la communication avec les frameworks et outils externes, tels que Hibernate, par example. Par moyen d'annotations, il est possible de gérer les instances des classes sans effort de la part du programmeur : toutes les *Vue*s commençaient par la lettre V, les *Contrôleur*s par la lettre C, les *Business* par la lettre B, et les *Persistance*s par la P. Ainsi, Hibernate savait que le contrôleur Cbanque était celui qui manipulait la persistance correspondante Pbanque.

La création de fenêtres génériques, à son tour, est possible lorsque l'on se rend compte, par example, que le registre de nouvelles professions et le registre de mots interdits sont similaires dans le fait d'avoir un seul champ de texte, leur description. Donc il suffit que le programmeur rattache les C, B et P dans chaque cas avec la V pour que la fenêtre soit créée.

3.4 Cycle de développement logiciel

Les cycles de développement des systèmes dans l'ingénierie des systèmes sont les processus de création ou de modification de systèmes d'information et les modèles et méthodes utilisés pour développer ces systèmes. En génie logiciel, ce concept sous-tend de nombreux types de méthodologies, et ces méthodes constituent le cadre pour la planification et le contrôle de la création d'un système d'information [19].

Les systèmes informatiques sont complexes et souvent (surtout avec la hausse récente de l'architecture orientée services) relient plusieurs systèmes traditionnels fournis par de différents fournisseurs de logiciels. Pour gérer ce niveau de complexité, un certain nombre de modèles ou de méthodes de gestion de cycle de vie logiciel ont été créés, tels que « cascade », « concourant », « spirale », « développement agile de logiciels », parmi d'autres.

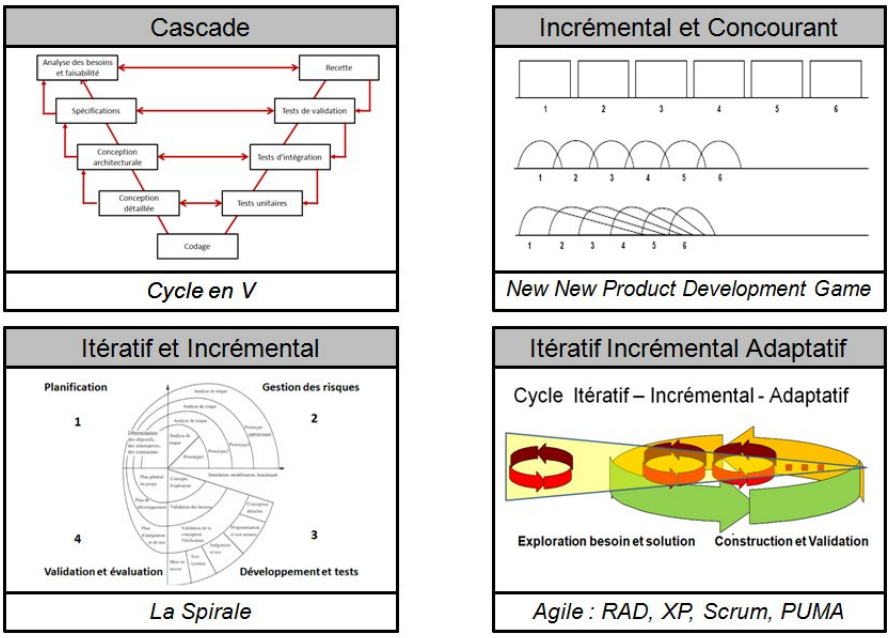

FIGURE 3.3 – Les principales cycles de développement logiciel

3.4.1 Les grandes familles de cycle de développement

Modèle en cascade

Les phases traditionnelles de développement en cascade sont effectuées simplement les unes après les autres, avec un retour sur les précédentes. Le processus de développement utilisant un cycle en cascade exécute des phases qui produisent des livrables définis au préalable, se terminent à une date précise et seulement lorsque les livrables sont jugés satisfaisants lors d'une étape de validation-vérification.

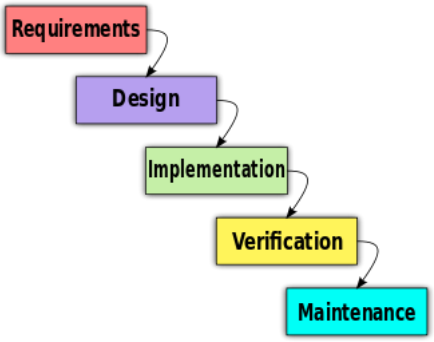

FIGURE 3.4 – Le modèle en cascade

Cycle en spirale

Par l'implémentation de versions successives, le cycle recommence en proposant un produit de plus en plus complet et dur. En effet, le début de chaque itération comprend une phase d'analyse des risques. Ceci est rendu nécessaire par le fait que, lors d'un développement cyclique, il y a plus de risques de devoir défaire à une telle itération ce qu'on a fait à la précédente.

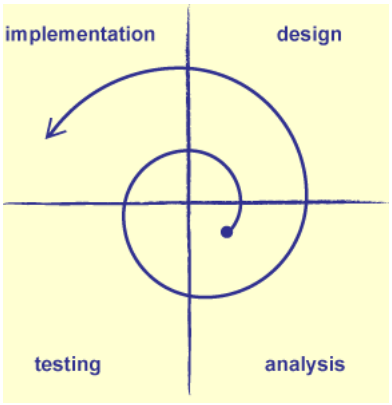

FIGURE 3.5 – Le cycle en spirale

Cycle itératif

Toutes les méthodes Agiles, tels que ASD, FDD, Crystal, Scrum ou *extreme programming*, débutent par des phases séquentielles, courtes mais bien réelles, d'exploration, d'architecture et de planning. Un usage totalement itératif de ces méthodes n'est cependant pas exclu mais ne peut s'appliquer qu'à de très petits projets. En fait, l'idée est de livrer au plus tôt quelque chose qui puisse être testé par le client. On peut en effet réaliser plusieurs itérations sur une documentation telle que l'architecture.

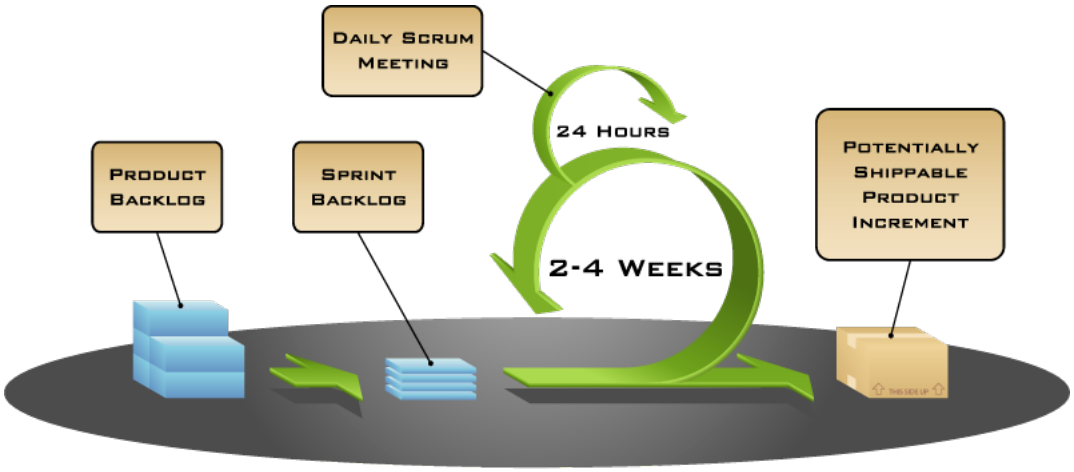

Figure 3.6 – Le processus itératif Agile de type Scrum

3.4.2 Le cycle de développement à Asert

L'entreprise d'accueil ne suit pas l'une des méthodes traditionnelles de processus de développement logiciel. En effet, les directives les plus importantes de toutes les méthodes sont prises en compte lors de la planification du projet, mais la structure globale varie selon le client, la taille et l'équipe.

En général, les étapes initiales des projets commencent très similaires au cycle cascade, avec la spécification des exigences du projet, entre le chef de projet et le client, avec l'aide de l'analyste de systèmes ; la conception des structures du projet, par l'architecte logiciel et l'analyste ; et le codage, par le développeur. Ensuite, le processus devient plus similaire aux méthodes agiles, avec l'intégration de versions préliminaires (alpha, beta, etc.) de la solution et avec des tests hebdomadaires. Finalement, une fois le projet est livré, les routines de maintenance se font périodiquement dans l'année. La documentation de projet n'est pas exhaustive, comme souvent les projets en cascade le demandent, mais un rapport final est livré au cliente avec le logiciel.

Du point de vue d'un projet en cours d'exécution, qui était déjà dans la phase de construction et codage, malheureusement je n'ai pas pu assister à toutes les étapes du cycle de vie du produit. Tout de même, le code du système a été revu plusieurs

fois par le client lors de mon séjour à l'entreprise. De petites modifications ont dû être faites, mais cela n'a pas retardé énormément le développement.

3.5 Développement

Le deuxième mois de mon stage a été concentré sur le développement du système de gestion d'adhérents de CELGMED (Figure 3.7). J'étais chargé de la conception et création de plusieurs entités Java et fenêtres graphiques Flex, et par la liaison avec les bases de données SQL Server déjà existantes.

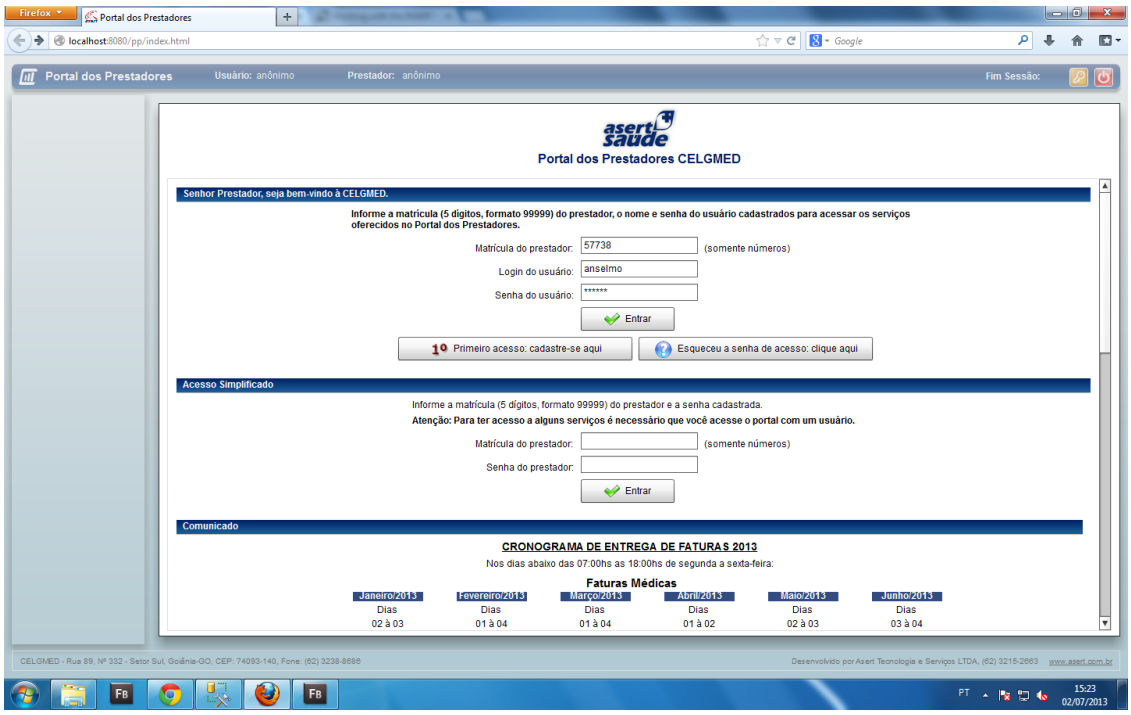

FIGURE 3.7 – Page d'accueil du système d'adhérents CELGMED

L'équipe informatique était composé du chef de projets, d'un architecte logiciel, d'un développeur base de données, d'une analyste de systèmes, d'un développeur logiciel et de deux stagiaires, moi y compris. Du fait de n'être pas une équipe nombreuse, les tâches étaient souvent confondues et le développement était très agile : l'architecte logiciel touchait aussi au code, le développeur logiciel aidait le responsable des bases de données, et les stagiaires les aidait dans toutes les aires de la programmation.

De manière générale, afin de créer les fenêtres graphiques du système de gestion il fallait suivre quelques phases de développement, soit :

1. Création de la fenêtre graphique avec l'aide de l'éditeur visuel Adobe Flash Builder. Cette Vue était normalement associée à un ensemble d'entités plutôt qu'à une seule entité. Dans cette phase, il s'agissait d'une fenêtre de prototype, sans actions ou événements.

2. Vérification de quelles tables de la base de données (qui avaient déjà été créées dans la phase de conception) seraient utilisées dans l'application. Voir Section 3.4.2, Figure 3.8.
3. Création des entités correspondantes aux tables, c'est-à-dire, les classes de Contrôle, Business et Persistance correspondantes. Section 3.3.
4. Codage des interrelations du flux de contrôle décrit dans la Table 3.1. C'était ici que l'on peuplait la Vue avec les actions et événements, ainsi que les classes de *C*, *B* et *P*. Section 3.2.3.
5. Conception du rapport pdf à être généré sur iReport, dans le cas où la fenêtre le demandait, et intégration avec la Vue.
6. Vérification du code par le tuteur.

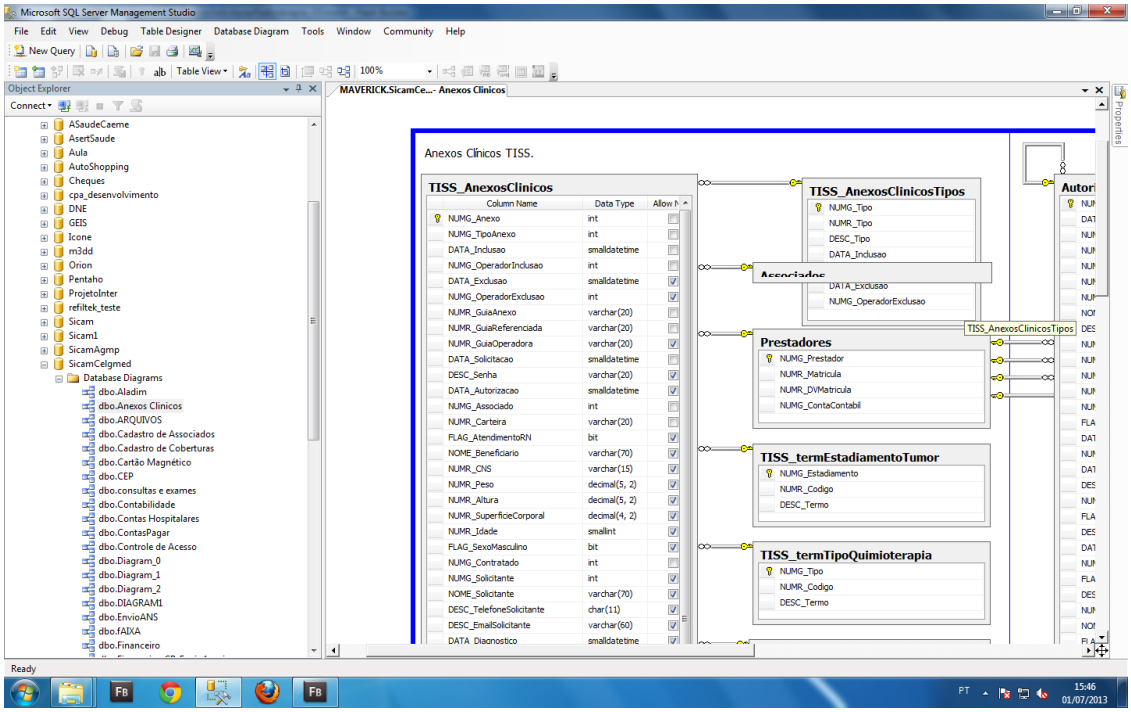

FIGURE 3.8 – Base de données relationnelle SQL Server

Encore une fois, ces phases peuvent être mieux expliquées avec un example pratique que j'ai dû exécuter. Supposons la création d'une fenêtre graphique de demande de radiothérapie par un client de la mutuelle de santé CELGMED (Figure 3.9). Ces étapes deviennent donc :

1. Création de la fenêtre graphique avec l'aide de Flash Builder. Prise en compte des entités qui seraient utilisées : Bénéficiaire (la personne qui a demandé la radiothérapie), Médecin responsable, Maladies diagnostiquées, etc. L'élaboration de cette fenêtre de prototype était très vite, car on prenait souvent une autre fenêtre similaire comme modèle.
2. Vérification des tables à être utilisées : `celgmed.Beneficiaire`, `celgmed.Medecin`, `celgmed.Maladie`, etc.

3. Création des entités correspondantes aux tables : Ebeneficiaire, Emedecin, Emaladie, ainsi que celles correspondantes à la radiothérapie proprement dite, Cradiotherapie, Bradiotherapie, Pradiotherapie.
4. Codage du flux de contrôle. Un clic dans le bouton « Sauvegarder » faisait que la vue appelle le Cradiotherapie, qui à son tour mettait en place le Bradiotherapie pour vérifier la logique associé à une sauvegarde de données, qui ensuite demandait que la Pradiotherapie fasse les altérations dans les tables avec des commandes SQL : "`UPDATE celgmed.Beneficiaire SET VISITE_MEDICALE=1 WHERE NOM_BENEFICIAIRE='Pierre Dubois'`". Le message d'erreur ou succès retournait à la fenêtre et l'utilisateur voyait le résultat de sa commande : "Bénéficiaire mis à jour". Celle-ci était l'étape la plus fastidieuse, car sa sous-division était beaucoup difficile ; la classe P nécessitait de la B, qui nécessitait de la C, qui à son tour était très lié à la V.
5. Conception du rapport PDF à être généré. Il s'agissait de reproduire les mêmes éléments de la fenêtre graphique sous le format JasperReport.
6. Vérification du code par le tuteur.

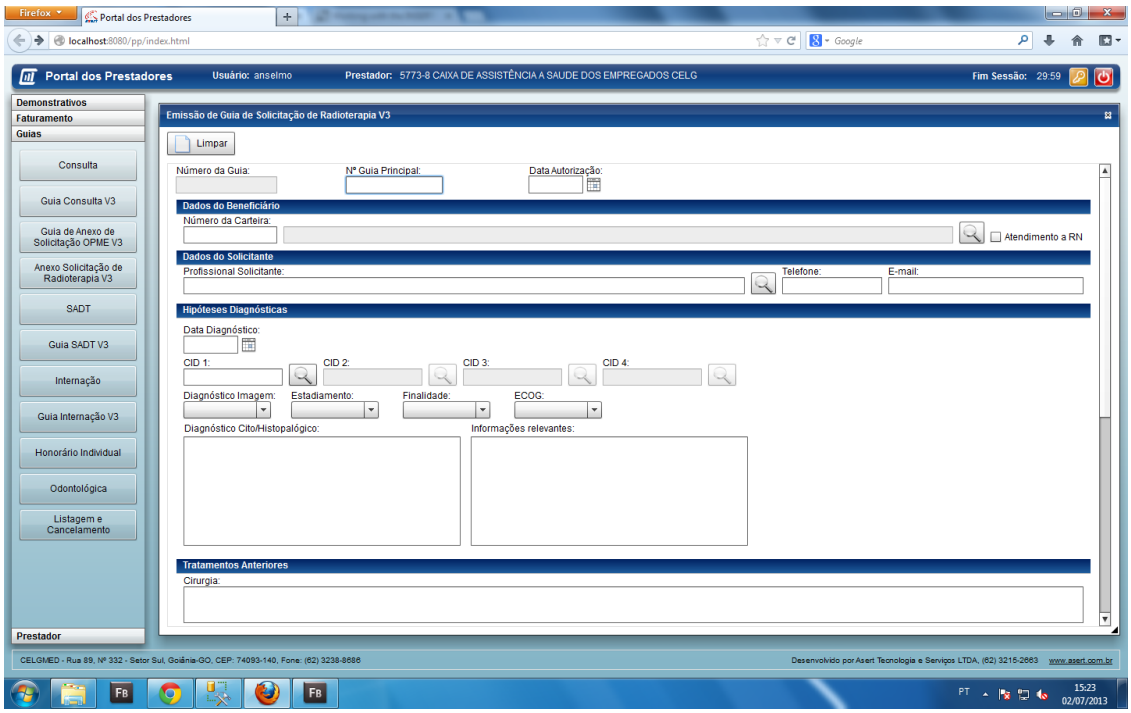

FIGURE 3.9 – Fenêtre de demande de radiothérapie

Le développement de fenêtres plus élaborées, comme celle de la radiothérapie, prenait environ une semaine pour leur réalisation, alors que les fenêtres plus élémentaires, qui ne touchaient qu'à une seule table de la base de données, comme dans l'enregistrement d'une Rue ou d'une Ville, comptaient une journée. Dans ce rythme, j'ai pu assister à la création de plus d'une dizaine de fenêtres complètes au cours du deuxième mois de mon stage.

Chapitre 4

Conclusion

Ce rapport a présente mes expériences professionnelles dans l'établissement Asert Serviços e Tecnologia da Informação, une petite société informatique qui propose des solutions de systèmes de gestion intégrés à d'autres entreprises, principalement à celles d'aide médicale.

L'opportunité de réaliser un stage ingénieur chez une entreprise de développement logiciel a été très enrichissante, et ces huit semaines de travail ont été très productives. Ce stage m'a donné une grande ouverture d'esprit en voyant une équipe très capacité et m'a motivé dans un cadre académique pour produire des technologies appliquées à des problèmes réels du monde corporatif.

L'environnement amical et stimulant favorise un engagement et un approfondissement interdisciplinaire remarquable. Le fait d'être un petit groupe de développement fait qu'il soit nécessaire de croiser de différentes connaissances et penser aux problèmes à partir d'angles inhabituels. Dans ce sens, le bureau crée une atmosphère qui invite les personnes qui travaillent dans des domaines distincts à interagir et se communiquer même dans des circonstances non-professionnelles.

Par rapport aux résultats de mon travail, j'ai pu constater une amélioration de ce que je développais, non pas seulement en question de vitesse de travail mais aussi de qualité de code. À la fin de mon séjour, le développeur responsable ne faisait plus de vérifications, car il était à moi de le faire de façon rigoureuse et progressive.

Le poste que j'ai occupé était sans doute plus proche de celui d'un ingénieur que dans le stage ouvrier à la première année : il y avait des objectifs clairs à remplir, des spécifications, des délais, et des compétences techniques requises. Tout de même, au fur et a mesure que je concevais des fenêtres similaires les unes des autres, j'avais l'impression de ne pas créer de nouvelles fonctionnalités, mais tout simplement de rester sur la méthode de travail ; c'était aux développeurs et à l'architecte d'améliorer le framework de l'entreprise, tandis que j'en étais seulement un utilisateur. J'ai conclu finalement que ce sentiment est normal dans le cadre d'un stage d'apprentissage, vu que j'ai eu très peu de contact avec les bases fondatrices du système, mais cela m'a donne l'envie d'assumer l'un de ces postes dans l'avenir.

La continuation du travail que j'ai effectué a déjà été planifiée et sera finie avant la fin de l'année. L'objectif principal est d'avoir un livrable préliminaire au plus vite possible pour que le client puisse faire des suggestions et l'équipe de développement reformule le système.

Bibliographie

[1] Asert. *Asert*. Mai 2012. URL : http://www.asert.com.br (visité le 10 août 2013).

[2] Wikipedia Brasil. *Evolucao do PIB do Brasil*. Mai 2013. URL : http://pt.wikipedia.org/wiki/Anexo:Evolu\%C3\%A7\%C3\%A3o_do_PIB_do_Brasil (visité le 10 août 2013).

[3] Olhar Digital UOL. *Mercado brasileiro de Tecnologia da Informacao cresce 11% e ultrapassa US$ 100 bilhoes*. Mai 2012. URL : http://olhardigital.uol.com.br/noticia/mercado-brasileiro-de-tecnologia-da-informacao-cresce-11-e-ultrapassa-us-100-bilhoes/26172 (visité le 10 août 2013).

[4] Adobe Flex. *Flex*. Février 2013. URL : http://www.adobe.com/products/flex.html (visité le 15 août 2013).

[5] Oracle. *GlassFish Server*. Août 2013. URL : https://glassfish.java.net/ (visité le 15 août 2013).

[6] Microsoft. *SQL Server*. Janvier 2013. URL : http://www.microsoft.com/en-us/sqlserver/default.aspx (visité le 15 août 2013).

[7] Hibernate. *About Hibernate*. Janvier 2013. URL : http://www.hibernate.org/ (visité le 15 août 2013).

[8] Spring. *Spring Framework*. Janvier 2013. URL : http://projects.spring.io/spring-framework/ (visité le 15 août 2013).

[9] Wikipedia. *Inversion de contrôle*. Juillet 2013. URL : http://fr.wikipedia.org/wiki/Inversion_de_contr%C3%B4le (visité le 15 août 2013).

[10] Wikipedia. *Tutoriel BlazeDS*. Avril 2012. URL : http://www.mti.epita.fr/blogs/2012/04/10/tutoriel-blazeds/ (visité le 15 août 2013).

[11] Wikipedia. *Jasper Reports*. Juillet 2013. URL : http://fr.wikipedia.org/wiki/JasperReports (visité le 15 août 2013).

[12] Jaspersoft Community. *iReport Designer*. Janvier 2013. URL : http://community.jaspersoft.com/project/ireport-designer (visité le 15 août 2013).

[13] Wikipedia. *Software architecture*. Septembre 2013. URL : http://en.wikipedia.org/wiki/Software_architecture (visité le 1er septembre 2013).

[14] Wikipedia. *Software Architecture styles and patterns*. Juin 2013. URL : http://en.wikipedia.org/wiki/Software_Architecture_styles_and_patterns (visité le 1er septembre 2013).

[15] Wikipedia. *Multitier architecture*. Septembre 2013. URL : http://en.wikipedia.org/wiki/Multitier_architecture (visité le 1er septembre 2013).

[16] Wikipedia. *Architecture trois tiers*. Mars 2013. URL : http://fr.wikipedia.org/wiki/Architecture_trois_tiers (visité le 1er septembre 2013).

[17] Wikipedia. *Model–view–controller*. Septembre 2013. URL : http://en.wikipedia.org/wiki/Multitier_architecture (visité le 2 septembre 2013).

[18] Batiste Bieler. *Architectures MVC et 3-Tier*. Juin 2006. URL : http://batiste.dosimple.ch/blog/2006-06-02-1/ (visité le 2 septembre 2013).

[19] Wikipedia. *Systems development life-cycle*. Septembre 2013. URL : http://en.wikipedia.org/wiki/Systems_Development_Life_Cycle (visité le 19 septembre 2013).