IMPLEMENTATION D'UNE APPLICATION INFORMATIQUE DISTRIBUEE POUR LA GESTION DE PRESTATIONS DE L'OFFICE CONGOLAIS DE CONTROLE - DIOR

Joseph Loleke Ongembe*

**Assistant de deuxième mandat , Département : Informatique de Gestion, Institut Supérieur Pédagogique de Lubutu*

**Corresponding Author:*

AOUT 2023

SIGLES ET ABBREVIATIONS

- API : Applicative Programing Interface. L'interface de programmation applicative.
- CPM : Critical Path Method
- CRUD : Create Read Update Delete.
- Dior : Direction Orientale
- EJB : Enterprise JavaBeans
- JAX-RS : Java Api for RESTful Web Services.
- MVC : Modele vue Controleur
- OCC: Office Congolais de Contrôle
- ORM : (Object-Relational Mapping) Mapping Objet relationnel.
- PDM : Precedence Diagramming Method
- REST : Representational state transfer
- SQL: (Structured Query Language) Langage de requête structuré.
- UML: (Unified Modeling Language) Langage de Modélisation Unifié
- UP: (Unified Process) Processus Unifié.

Resume

De nos jours, une des tendances les plus en vue et qui concerne tous les secteurs de développement, est l'information. Depuis l'apparition de l'informatique et son introduction dans le monde économique, les entreprises et entités publiques aspirent à optimiser et à rendre fiable la gestion de leur structure interne. Dans cet article, nous faisons le point sur les étapes indispensables dans la construction d'une application distribuée qui repose sur des architectures logicielles adaptées. L'OCC/ Dior étant l'une des grandes entreprise dans la province, doit miser sur les nouvelles technologies pour améliorer la qualité, la rapidité, la sécurité de la transmission des rapports des services rendu à ses clients et avoir un outil capable de rentabiliser sa gestion actuelle qui est encore manuelle.

L'objectif de cet article est de définir les étapes analytiques et conceptuelles d'une application informatique distribuée reposant sur la technologie REST coté serveur consommée par une application mobile sous Android permettant à cette entreprise de disposer d'un système adapté à ses besoins et qui soit directement lié au reste de son équipement informatique.

Abstract

Now a days, one of the most envy trend and which concern all the development sectors, is the information. Since the appearance of IT and its introduction into the economic world, companies and public entities aspire to optimize and make reliable management of their internal structure. In this article, we take stock of the construction of a distributed application that is based on adapted software architects. The OCC being one of the large companies in the province, must be based on new technologies to improve the quality, speed, security of transmitting services rendered to its customers and a tool capable of profitable its current management which is still manual. The objective of this article is to define the analytical the analytical and conceptual steps of a distributed computer application based on the server side restructuring technology consume by a mobile application under Android allowing this company to have a system adapted to its needs and that is directly related to the REST of its computer equipment.

I. INTRODUCTION

Toute organisation possède un ou plusieurs systèmes d'information qui sont susceptibles de fournir des rapports d'information de gestion à divers paliers organisationnels. Ces systèmes sont utilisé depuis toujours et ont pour but, de recueillir au cours des opérations de l'organisme, des données qui seront utilisées à la préparation des rapports rédigé pour répondre au besoin de la gestion.

Sur cet, un mot retient notre attention, c'est l'information, l'information et l'informatique ne sont pas deux mots synonymes. L'information est l'ensemble de fait, de notions' d'éléments de connaissance ou de jugement, possédé a un moment donné, sur un sujet déterminé et faisant l'objet d'une communication, d'une interprétation ou d'un traitement.[1]

L'informatique est une science de traitement rationnel, notamment par la machines automatique de l'information considéré comme support de connaissance humaines et de communication dans le domaine technique, économique et sociaux.[2] Alors le système d'information est l'ensemble d'information circulant dans l'entreprise ou dans une organisation.

L'Office Congolais de Contrôle /Kisangani, *« une entreprise à caractère technique et scientifique créé par l'ordonnance loi n°74/013 du 10/01/1974 après dissolution de la société congolaise de surveillance (SCS) qui fut une filiale de la société de surveillance dont le siège se trouve à Genève (en Suisse).»,* prestataire de services seule habile à contrôler la qualité, la conformité et la quantité des produits en République Démocratique du Congo, ne dispose pas d'un système automatisé pouvant faciliter ces différentes tâches qui jusqu'à l'heure de la rédaction de cet article recourt aux moyens traditionnels de collecte, nous comprenons par-là, l'utilisation des supports papiers.

Cette tâche faisant partie du Système de traitement transactionnel comme l'ont défini K. Laundon et al. « Les Systèmes de traitement transactionnels sont la concrétisation des Systèmes Opérants. Ils exécutent et enregistrent les transactions quotidiennes et routinières associées aux évènements quotidien tels que la saisie des bons de commandes ou le calcul des tournées de livraison de la flotte de camions[3] », exige à cette entreprise de mettre en place une infrastructure logicielle et matérielle adéquate susceptible de prendre en charge la mobilité des agents qui doivent échanger les données ainsi prélevées avec le reste de l'infrastructure informatique de manière sure et valide.

Ceci étant, le grand problème s'enregistre dans le cadre de traitement relative aux clients et leurs marchandises au niveau de poste de contrôle qui s'effectue manuellement et engendre la lenteur dans la transmission des données vers les organes décideurs.

Eu égard à ce qui précède, nous nous sommes posé une question dans le but d'améliorer l'actuelle gestion : Quelle solution capable de facilité la transmission des données des prestations en temps réel entre les postes de contrôle et les Services des traitements centralisé ? Partant de cette question, la solution idéale compte tenu de problème soulevées sera l'ajout d'une couche logicielle à l'infrastructure informatique déjà existante. Sur ce, la solution pouvant prendre en compte la mobilité des agents dans les postes de contrôle pour intégration solide avec le reste de l'infrastructure logicielle serait l'implémentation d'une nouvelle couche logicielle mobile sous Android reposant sur la technologie REST[4] facilitant les échanges entre les agents et le serveur d'application qui est hébergé au sein de l'entreprise. Cette solution permettrait à l'OCC / Dior d'améliorer les différentes tâches liées à la gestion des prestations des services rendus dans les postes de contrôle et de lui épargner du traitement manuel des données et du risque de perte de données sensibles.

Le principal objectif de cette étude est d'expliquer les différentes phases du développement d'une application informatique distribuée basée sur une architecture en 3 couches dont chacune des couches correspond à une brique logicielle bien identifiée Pour atteindre cet objectif, nous faisons recourt à la méthode UP (Unified Process) « Processus Unifié » en français, qui est un processus de développement logiciel construit sur UML ; il est itératif et incrémental, centré sur l'architecture, conduit par les cas d'utilisation et piloté par les risques[5]. UP utilise UML pour la modélisation du système. UML se définit comme un langage de modélisation graphique et textuel destiné à comprendre et à décrire des besoins, spécifier, concevoir des solutions et communiquer des points de vue[6].

Pour estimer le cout du développement de cette application, nous avions utilisé la méthode COCOMO[7] (acronyme de l'anglais Constructive COst MOdel) qui est un modèle permettant de définir une estimation de l'effort à fournir dans un développement logiciel et la durée que ce dernier prendra en fonction des ressources allouées.

---

[1] Rolland HURTUBISE, Informatique et information, Paris, 1976, p.8

[2] WWW.GOOGLE.COM, Informatique

[3] Kenneth L., Jane L., Eric F., Serge C. et Sophie C., management des systèmes d'information, 13ième édition, Nouveaux Horizons, 2013, page 51.

[4] Antonio Goncalves, Java EE 6 et GlassFish 3, Pearson Education France, 2010, ISBN : 978-2-7440-4157-0, p.

[5] Pascal Roques et Franck Vallée, UML 2 en action, De l'analyse des besoins à la conception 4e édition, Eyrolles 2007, p.12.

[6] Pitman, N., *UML 2 en concentré.* O'Reilly 2006, p. 15

[7] https://fr.wikipedia.org/wiki/Constructive_Cost_Model consulté le 10 juin 2023 à 20h42

Les techniques d'observation, d'interview et documentaires ont été utilisées pour nous permettre de nous imprégner de la situation réelle qui prévalait au sein de cette entreprise. Une attention particulière a ainsi été faite sur l'architecture logicielle qui ne tient plus compte de la grandeur de l'entreprise. Pour bien cerner le problème, nous avions subdivisé cette étude en 2 sections dont la première se focalise sur la considération générale, explicitant au clair les notions essentielles des architectures logicielles existantes et les tendances technologiques actuelles. La seconde section s'intéresse premièrement à l'analyse fonctionnelle et organique de l'actuel système allant de poste de prestation jusqu'à la direction qui est le centre de copulation des informations. En second lieu, se présente la conception de la nouvelle architecture logicielle en justifiant les raisons sur lesquelles s'est fondée se choix.

II. CONSIDERATION GENERALE.

Cette étape nous permet de définir le contexte d'étude, l'approche de résolution ainsi que la gestion de notre projet.

II.1. DEFINITION DES CONCEPTS

II.1.1.IMPLEMENTATION:

La mise en place ou mise en œuvre est le fait de mettre en place. En ingénierie et plus particulièrement en informatique, la mise en œuvre désigne la création d'un produit fini à partir d'un document de conception, d'un document de spécification, voire directement depuis un cahier des charges. L'utilisation de l'anglicisme « implémentation », de l'anglais *toimplement*, est courante (et acceptée) et reflète la volonté de ne pas tomber dans la traduction ambiguë que serait l'utilisation du verbe *implanter*. Cependant, selon certaines personnes, cet anglicisme est impropre car souvent Ingénierie et informatique utilisé à tort pour qualifier des actions très différentes.

II.1.2. SYSTEME DISTRIBUE

Les systèmes distribués sont pour la plupart des entités à couplage lâche communiquant entre elles pour accomplir certaines tâches. L'un des modèles les plus courants en informatique distribuée est constitué par les appels de procédures distants RPC[8].

Plusieurs architectures existent dans le monde de développement de logiciels. Nous les présentons comme suit :

• **Architecture simple tiers ou application monolithique**

Une application monolithique est une architecture constituée d'un seul bloc et s'exécutant sur
une seule machine. Ces applications sont généralement utilisées dans le domaine du temps réel ou bien au sein d'applications demandant de grandes performances. Elles restent également omniprésentes dans le monde du grand public, étant utilisées en standalone (de manière autonome) sur les machines personnelles[9]. Cette architecture a été plus utilisée surtout dans le développement des applications bureautiques et de gestion liée aux petites entreprises. Elles s'occupent l'essentiel du parc logiciel fonctionnant dans nos entreprises locales congolaises. Cette architecture monolithique est appelée simple tiers car toutes les fonctionnalités sont comprises dans une seule couche logicielle[10]. *Cette architecture a les inconvénients suivants : évolution difficile, maintenance lourde, partage difficile des données, etc*.

Ce qui a poussé les architectes de systèmes logiciels à penser à d'autres solutions stimulées par l'essor des réseaux et surtout de l'internet.

• **Architecture Client Serveur**

Nous devons comprendre une architecture Client-Serveur comme étant une architecture qui est bâtie sur deux couches dont l'une représente la partie interface utilisateur appelée autrement Front-End et l'autre encapsule la partie métier et/ou base de données désignée par Back-End.

Ces deux couches communiquent à travers un réseau informatique qui définit le protocole adapté à cette communication. La raison de cette approche est de centraliser les données afin de permettre à plusieurs utilisateurs d'y accéder simultanément. Les données peuvent ainsi être partagées entre plusieurs utilisateurs de l'application. Cette architecture est communément appelée client-serveur, qui dans notre approche peut être représentée en deux tiers[11].

La couche « client » est souvent de type « léger » c'est-à-dire utilisant un navigateur web.

Certains langages de programmation tels que « Java » et « C# » offrent la possibilité d'utiliser les interfaces graphiques faisant appels aux ressources du système d'exploitation pour les construire, on les appelle « client lourd ».
La couche « serveur » quant à elle, se charge de fournir les ressources demandées par le client.

[8] Robert ENGLANDER, Java et SOAP, O'Reilly, 2009, page 1.

[9] Frédéric Chuong, Olivier Corgeron Cyril Joui, Jean-Baptiste Renaux et Maxime Vialette, op. cit., p.2

[10] http://remy-manu.developpez.com/introjava2ee consulté le 23 avril 2023

[11] http://remy-manu.developpez.com/introjava2ee, consulté le 28 avril2023

Cette solution apporte plus de flexibilité et permet une bonne maintenance de l'application. Cependant, le développement de ce genre d'application nécessite la création d'un protocole de communication entre le client et le serveur. Ce protocole étant souvent propriétaire, l'évolution de ces applications doit souvent être faite par les créateurs[12].

Un des inconvénients de l'architecture deux-tiers est que la logique chargée de la manipulation des données et de l'application des règles métiers afférentes est incluse dans l'application elle-même. *Cela pose problème lorsque plusieurs applications doivent partager l'accès à une base de données*[13].

• **Architecture trois-tiers**
Ce modèle est une évolution du modèle d'application client-serveur défini précédemment.
L'architecture trois-tiers est donc divisée en trois niveaux ou rangées :
- Tiers client qui correspondent à la machine sur laquelle l'application cliente est exécutée.
- Tiers métier qui correspondent à la machine sur laquelle l'application centrale est exécutée.
- Tiers accès aux données qui correspondent à la machine gérant le stockage des données[14].

La séparation entre le client, l'application et le stockage, est le principal atout de ce modèle.

Toutefois, dans des architectures qui demandent de nombreuses ressources, il sera assez limité.

En effet, aucune séparation n'est faite au sein même de l'application, qui gère aussi bien le logique métier que la logique fonctionnelle ainsi que l'accès aux données.

Dans le cadre de notre étude, notre solution logicielle se basera sur cette architecture qui sera facilement implémentée par le trio Swing, EJB et JPA respectivement couche vue, couche métier et couche modèle.

• **Architecture Multi-tiers**
Dans le cadre d'applications beaucoup plus importantes, l'architecture trois-tiers montre ses limites. L'architecture multi-tiers est simplement une généralisation du modèle précédent qui prend en compte l'évolutivité du système et évite les inconvénients de l'architecture trois-tiers vus précédemment.

Dans la pratique, on travaille généralement avec un tiers permettant de regrouper les logiques métiers de l'entreprise. L'avantage de ce système, c'est que ce tiers peut être appelé par différentes applications clientes, et même par des applications classiques, de type fenêtrées, qui ne passent donc pas par le serveur Web.

II.I.3. GESTION
Le mot « gestion » est un terme générique, important à définir afin de répondre précisément à l'attente de l'entreprise.
Selon le Larousse[15], la gestion est définie comme l'action et/ou la manière de gérer, d'administrer, de diriger, d'exécuter, d'organiser quelque chose. Elle doit mener à des actions qui s'appliquent à une activité courante de l'entreprise.

Selon Peter Drucker[16], théoricien américain du management, la gestion est l'art de prendre une décision rationnelle et informée. La décision se fait donc à partir d'une analyse complète et réfléchie. Gérer s'oriente plus vers gouverner une organisation en précisant les buts et les moyens pour les atteindre.

II.I.4. PRESTATION DES SERVICES
Le terme prestation de service désigne une activité qui ne concerne aucune livraison de biens matériels. Elle met en avant une compétence précise exécutée par le prestataire au service d'une entreprise moyennant une rémunération[17].

III. ANALYSE ET CONCEPTION
Dans cette étape, nous présentons d'abord l'analyse de besoin dans toute ces facette et décrivons la conception à faire pour atteindre notre objectif fixé.

III.1. ANALYSE
C'est la phase d'analyse des besoins fonctionnels et techniques qui nous permet de déterminer les éléments constitutifs de notre future application. Les spécifications fonctionnelles ont pour objectif de décrire précisément l'ensemble des fonctions d'un logiciel ou d'une application, et de fixer ainsi le périmètre fonctionnel du projet[15]. La rédaction des spécifications est basée sur l'expression des besoins du client, généralement regroupés et reformulés dans un cahier des charges. La spécification d'une fonction va donc décrire dans le détail les services qu'elle va fournir à l'application ou à l'utilisateur. Hors que les spécifications

[12] Frédéric Chuong, Olivier Corgeron Cyril Joui, Jean-Baptiste Renaux et Maxime Vialette, op. cit., p.2
[13] http://remy-manu.developpez.com/introjava2ee consulté le 30 avril 2023
[14] Frédéric Chuong, Olivier Corgeron Cyril Joui, Jean-Baptiste Renaux et Maxime Vialette, op. cit, p.3
[15] https://www.lecoindesjeux.com/rediger-une-specification-fonctionnelle-detaillee/ consulté le 14 juin 2023

III.2. SPECIFICATIONS FONCTIONNELLES

Le système que nous projetons mettre en place à l'OCC/KISANGANI doit avoir les fonctionnalités suivantes :

- La gestion des marchandises avec les opérations de base (CRUD : Create Read Update Delete).
- La gestion des clients avec les opérations de base (CRUD : Create Read Update Delete).
- Elaboration de rapport d'activité,
- Gestion des comptes utilisateurs,
- L'automatisation des opérations de facturation,
- La recherche rapide des informations sur les postes d'activité.

Dans cette étude, nous avons pu déterminer les acteurs suivants :

- Administrateur système,
- Chef de poste,
- Chef de service,
- Chef de Division,
- Directeur Provincial.

Cette partie représente un point de vue « fonctionnel » de l'architecture système. A partir des cas d'utilisation, nous serons en contact permanent avec les acteurs du système en vue de définir les limites de celui-ci, et ainsi éviter de trop s'éloigner des besoins réels de l'utilisateur final. Les cas d'utilisation constituent un moyen de recueillir et de décrire les besoins des acteurs du système. Ils peuvent être aussi utilisés ensuite comme moyen d'organisation du développement du logiciel, notamment pour la structuration et le déroulement des tests du logiciel[16].

Diagramme de cas d'utilisation de notre analyse

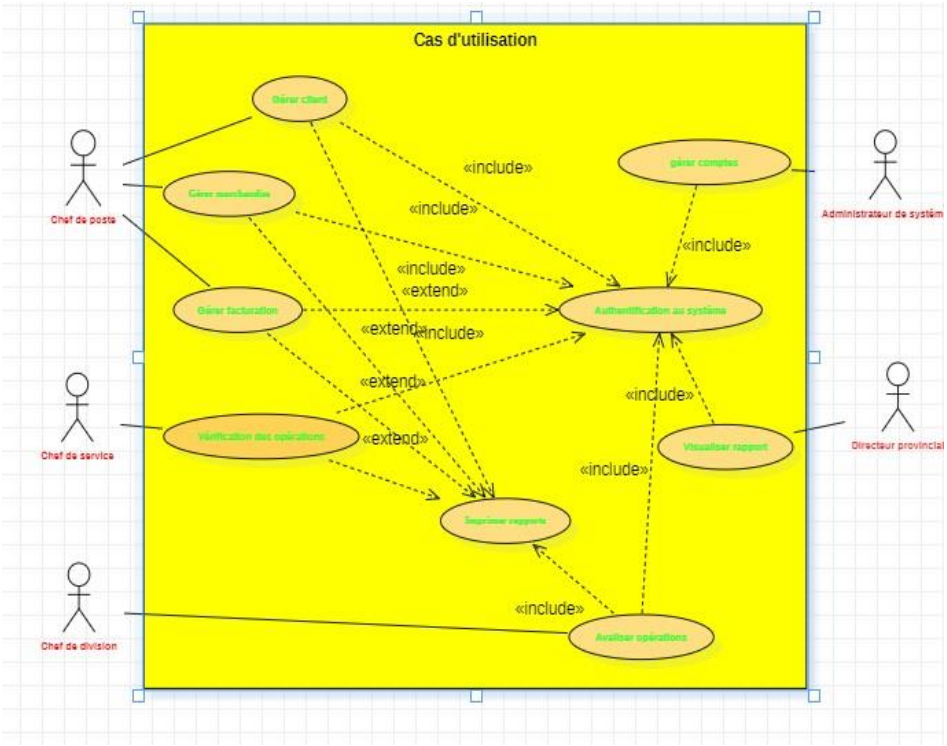


Figure I : *Diagramme de cas d'utilisation*

Description textuelle de diagramme de cas d'utilisation

- Cas d'utilisation Gere client

o Résumé : ce cas d'utilisation permet d'enregistrer, modifier, afficher, et supprimer un client dans la base de données. o Acteurs : Chef de poste. o Dates : le 11 juin 2023 o Responsable : LOLEKE. o Version : 1.0

- Cas d'utilisation Gérer compte

o Résumé : Ce cas permet d'enregistrer, modifier, afficher et supprimer les comptes utilisateurs dans la base de données. o Acteurs : Administrateur. o Dates : le 11 juin 2023. o Responsable : LOLEKE o Version : 1.0

- Cas d'utilisation Visualiser rapport

[16] Joseph Gabay et David Gabay, UML 2. Mise en œuvre guidée avec études de cas : ANALYSE ET CONCEPTION, Dunod, Paris, 2008, p. 61

o Résumé : ce cas d'utilisation permet de visualiser les données synthétisées qui ont été produites au cours d'une activité.
o Acteurs : Directeur provincial o Dates : le 11 juin 2023 o Responsable : LOLEKE o Version : 1.0

- Cas d'utilisation Authentification o Résumé : ce cas permet de vérifier l'authenticité de l'utilisateur dans le système.

o Acteurs :
✦ Principale : Administrateur,
✦ Secondaire : Chef de poste, Chef de division, chef de service et Directeur provincial.
o Dates : le 11 juin 2023 o Responsable : LOLEKE o Version : 1.0

- Cas d'utilisation Aviser les opérations o Résumé : ce cas permet d'accepter ou non les opérations effectuées.

o Acteurs : Chef de division o Dates : le 11 juin 2023 o Responsable : LOLEKE o Version : 1.0

- Cas d'utilisation Gérer marchandises

o Résumé : ce cas permet d'enregistrer, modifier, afficher, et supprimer les marchandises dans la base de données. o Acteurs : Chef de poste. o Dates : le 11 juin 2023 o Responsable : LOLEKE o Version : 1.0

- Cas d'utilisation Gérer Facturation

o Résumé : ce cas permet d'enregistrer, modifier, afficher, et supprimer la facture dans la base de données.
o Acteurs : Chef de poste. o Dates : le 11 juin 2023 o Responsable : LOLEKE o Version : 1.0

- Cas d'utilisation Imprimer rapport o Résumé : ce cas permet d'imprimer le rapport d'activité.

o Acteurs :
✦ Principale : Administrateur,
✦ Secondaire : Chef de poste, Chef de division, chef de service et Directeur provincial.
. Dates : le 11 juin 2023
o Responsable : LOLEKE
o Version : 1.0

III.3. SPECIFICATIONS TECHNIQUES

L'application que nous mettons en place est une application qui se repose sur une architecture distribuée multi couche qui permet de gérer une entité. De manière théorique, une application distribuée est une application découpée en plusieurs unités. Chaque unité peut être placée sur une machine différente, s'exécuter sur un système différent et être écrite dans un langage différent. La plateforme de développement utilisée pour la réalisation de notre application est JEE ou JavaEE 7 (Java Enterprise Edition dans sa version 7) qui est l'extension serveur de la plateforme JSE (Java Standard Edition) de SUN.

C'est une plateforme de développement qui permet de développer des applications Web composées de Servlet et JSP et des applications métiers à base d'EJB.

Notre application étant assez simple, nous avons choisi d'utiliser MySQL comme serveur de base de données.
Le mapping Objet/Relationnel (mapping O/R) consiste à réaliser la correspondance entre le modèle de données relationnel et le modèle objets de la façon la plus facile possible.

Différentes solutions peuvent être utilisées pour la persistance des objets en Java. Notre choix est porté sur la solution JPA qui est l'API officielle d'Oracle avec son implémentation EclipseLink[17][18].

Un serveur d'application met en œuvre toutes les spécifications prévues par Java EE. Nous savons déjà que Java EE permet de fabriquer des applications Web à l'aide de servlet et de pages xHTML, le tout orchestré par la technologie JSF. Par ailleurs, nous découvrons maintenant que Java EE intègre les EJB[22].

Le serveur d'application sur lequel sera déployé l'application, une fois achevée, est GlassFish Server Open Source dans sa version 5.0. C'est un serveur d'application facile à utiliser, rapide et faisant parti des leaders du marché.

III.4. ESTIMATION DE CALCUL DU COUT DE DEVELOPPEMENT LOGICIEL

L'estimation est une activité complexe qui implique différentes pratiques, de nombreux acteurs (client, directeur, chef de projet, développeurs, etc.) et un environnement dynamique et incertain (dû aux changements des exigences du client, à l'évolution des compétences des développeurs, à la disponibilité des ressources, etc.). Ces éléments sont des facteurs d'échec principaux de l'estimation de l'effort, coûts et durée de projets logiciels[19] .

Le calcul des coûts consiste en un exercice exigeant et délicat qui sera affiné pendant toute la phase préparatoire du projet. La principale source de difficultés est liée à l'estimation d'un produit nouveau, encore mal défini et qu'il faudra pourtant

[17] https://www.oracle.com consulté le 24 juin 2023

[18] Frédéric Chuong, Olivier Corgeron,Cyril Joui, Jean-Baptiste Renaux et Maxime Vialette, « EJB : Des concepts à l'écriture du code, Guide du développeur », Laboratoire SUPINFO des technologies Sun, édition Dunod, page 231

[19] Safae Laqrichi. Approche pour la construction de modèles d'estimation réaliste de l'effort/coût de projet dans un environnement incertain : application au domaine du développement logiciel, thèse de doctorat, Université de Toulouse, Informatique, délivré par l'Ecole des Mines d'Albi-Carmaux, 2015, page 13

chiffrer. Le maître d'œuvre doit connaître le coût du projet avant d'être trop engagé dans sa réalisation, de manière à pouvoir réorienter ses choix, ou renoncer à son projet[20].

Comme énoncé à l'introduction de cet article, nous avons recours à la méthode d'estimation de cout COCOMO qui s'appuie uniquement sur la taille estimée du logiciel et sur le type de logiciel à développer. Nous allons estimer également le cout des matériels associés à cette application. Ainsi, pour le faire, nous imaginons l'architecture matérielle de notre application pour déterminer les équipements qui seront utilisés.

a. Architecture du système proposé

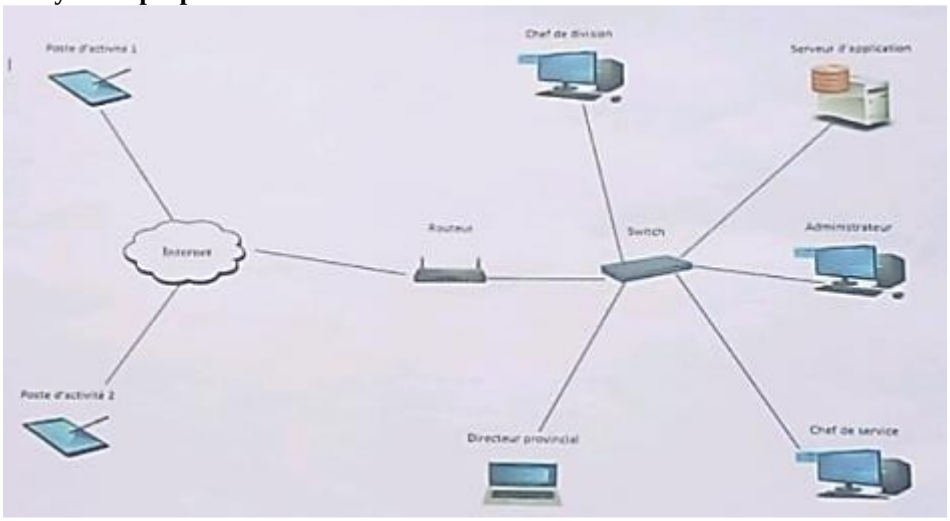


Figure 2 : *Architecture du système.*

Ce scénario permet de mettre en place une architecture réseau basée sur 2 topologies en étoile représentant chacune un réseau à part entière. Le premier réseau regroupe les terminaux mobiles des chefs de poste qui accèdent par internet au routeur de l'organisation et le deuxième réseau relie autour d'un switch le poste du chef de service, le chef de division, le Directeur provincial, le poste du serveur d'application et de bases de données et le poste pour l'administrateur du système.

a. Besoins matériels et logiciels

Les besoins matériels et logiciels de ce scénario sont présentés dans le tableau suivant :

Catégorie	Désignation	Quantité	Prix unitaire FC	Prix total FC	Commentaire
Matériels et logiciels	Ordinateur de bureau	3	584 500	1 753 500	A acheter
	Ordinateur portable	1	1 500 000	1 500 000	A acheter
	Switch	1	125 000	125 000	A acheter
	Serveur (Application et Base de données)	1	5.000.000	5.000.000	A acheter
	Tablette Android	2	400.500	801 000	A acheter
	Câble UTP	1 rouleau (300m)	250 000	250.000	A acheter
	Imprimante en Laser	2	350.000	700.000	A acheter
	GlassFish server 4.0				Gratuit
	NetBeans 8.2				Gratuit
	JDK 8				Gratuit
	Oracle DataBase XE 11g				Gratuit
	JavaSE et JavaEE				Gratuit
	Antivirus	1	250 000	250 000	A acheter
	Cout total de matériels et logiciels			10 379 500	

Tableau 1 : *Besoins matériels et logiciels.*

[20] Gérard CASANOVA - Denis ABÉCASSIS, op. cit., page 11

b. Cout de développement et formation des utilisateurs

Un projet de type mode semi-détaché convient au mieux pour ce scénario. On estime le nombre de lignes de notre code source à 8KLoc qui représente approximativement 8000 de lignes de codes. Nous servant de la formule fournie par COCOMO nous pouvons ainsi estimer le cout du développement de l'application et le cout global.

Cout de développement :

Intitulé	**Formule**	**Valeurs**
Effort à consentir	Effort = $3.0*(8)^{1.12}$	30HM
Temps de développement	TDev=$2.5*(30)^{0.35}$	8.22 Mois
Nombre moyen des personnes	Effort / TDev	4 personnes
Cout financier en FC	Effort * Salaire moyen congolais[21]	**5640000 FC**

Tableau 2 : *Cout de développement logiciel.*

Cout global:

Désignation	**Montant en Francs Congolais**
Cout total des matériels et logiciels	10 379 500
Cout de développement	**5640000**
Cout de formation des utilisateurs	800000
Cout total du scénario	**16819500**

Tableau 3 : *Cout de total de l'implémentation.*

III. CONCEPTION

• Diagramme de classes

Le diagramme de classe constitue l'un des pivots essentiels de la modélisation avec UML. En effet, ce diagramme permet de donner la représentation statique du système à développer. Cette représentation est centrée sur les concepts de classe et d'association. Chaque classe se décrit par les données et les traitements dont elle est responsable pour elle-même et vis-à-vis des autres classes[22].

La classe peut donc être considérée comme le modèle, le moule ou la notice qui va permette la construction d'un objet. Nous pouvons encore parler de type (comme pour une donnée).

On dit également qu'un objet est l'instance d'une classe (la concrétisation d'une classe). Les classes identifiées lors de l'étude des cas d'utilisation, puis réparties dans les catégories, sont simplement des classes candidates pour l'analyse objet. Il convient désormais de les examiner de manière détaillée, d'en éliminer certaines, ou au contraire d'en ajouter d'autres.

Cette activité de validation est itérative ; l'affinement des associations, ainsi que l'ajout des attributs et des opérations, vont nous fournir de précieuses informations.[23]

Règles de gestion :

- Le chef de poste gère un ou plusieurs clients,
- Le chef de poste gère un ou plusieurs marchandises,
- Le chef de poste gère un ou plusieurs factures,
- L'administrateur gère un ou plusieurs utilisateurs,
- Le chef de service vérifie un ou plusieurs opérations,
- Le chef de division avalise un ou plusieurs rapports d'activités,
- Le directeur provincial visualise un ou plusieurs rapports.

Le diagramme de classes correspondant à notre étude se présente comme suit :

[21] Salaire moyen = SMIG * nb Jours. (DECRET MIN. N°18/017 DU 22 MAI 2018, page 5). Dans notre cas le salaire moyen est égal à 188000Fc.

[22] **Joseph Gabay et David Gabay**, op. cit., page 17

[23] Pascal Roques et Franck Vallée, op. cit., p. 133

Figure 3 : *Diagramme de classes*

• Diagramme de séquences

Le diagramme de séquence décrit la dynamique du système. À moins de modéliser un très petit système, il est difficile de représenter toute la dynamique d'un système sur un seul diagramme.

Aussi la dynamique globale sera représentée par un ensemble de diagrammes de séquence, chacun étant généralement lié à une sous fonction du système.

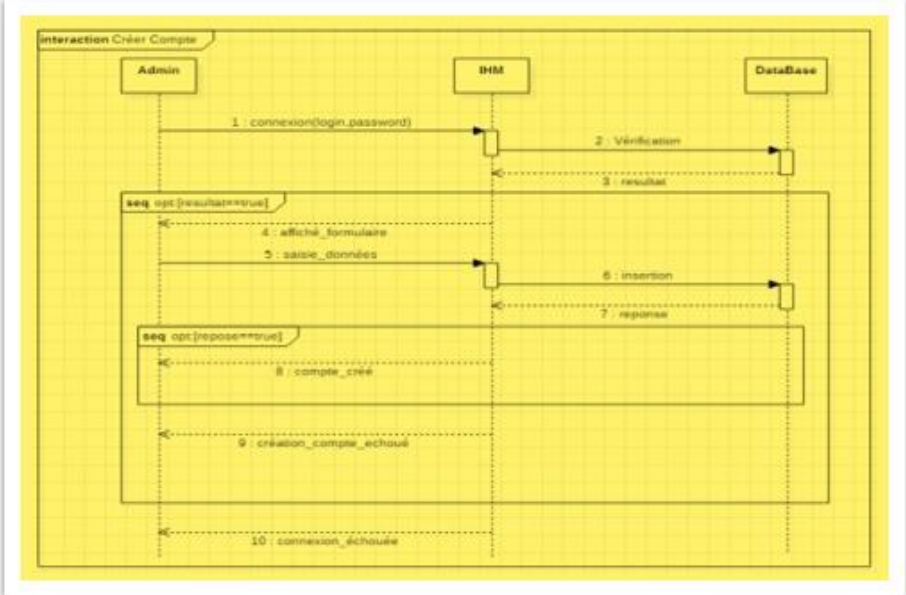
Le diagramme de séquence décrit les interactions entre un groupe d'objets en montrant, de façon séquentielle, les envois de message qui interviennent entre les objets. Le diagramme peut également montrer les flux de données échangées lors des envois de message.

Pour interagir entre eux, les objets s'envoient des messages. Lors de la réception d'un message, un objet devient actif et exécute la méthode de même nom. Un envoi de message est donc un appel de méthode.[24]

• Diagramme de séquence gérer compte

Dans ce diagramme de séquence nous avions réparti les opérations en 4 taches : Créer compte, modifier compte, supprimer compte et afficher compte.

Diagramme de séquence Création Compte Utilisateur.



[24] Fien VAN DER HEYDE et Laurent DEBRAUWER, UML 2 Initiation, exemples et exercices corrigés 2ième édition, Editions ENI, p.46

Figure 4 : *Diagramme de séquence création compte utilisateur*

Diagramme de séquence Modification compte utilisateur

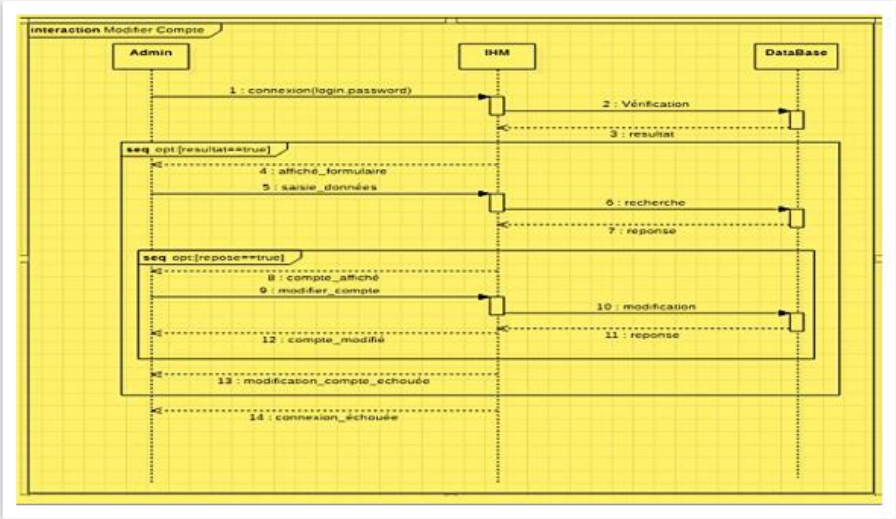


Figure 5 : *Diagramme de séquence modification compte utilisateur*

Diagramme de séquence « Afficher comptes »

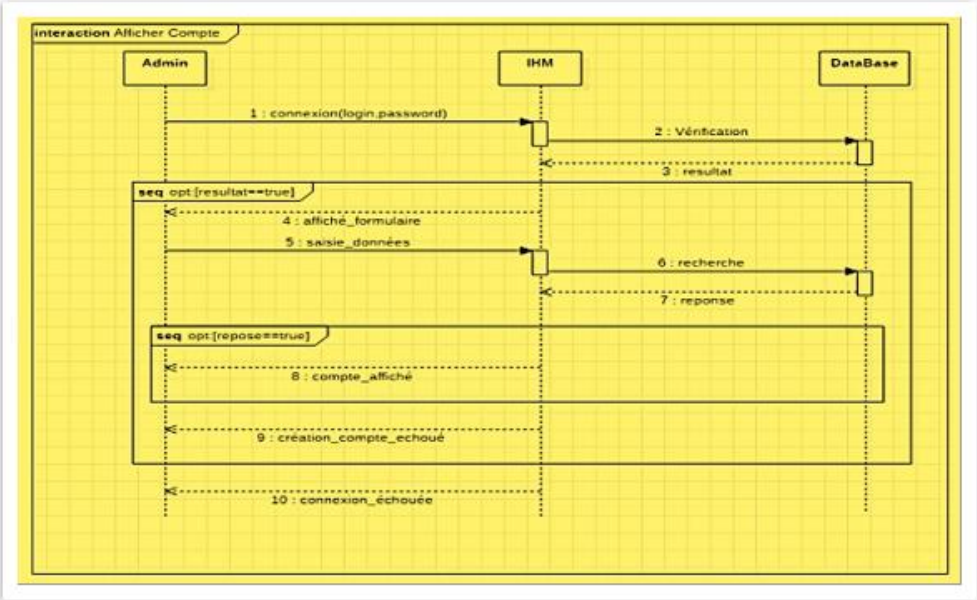


Figure 6 : *Diagramme de séquence Affiche compte utilisateur*

Diagramme de séquence « Supprimer compte »

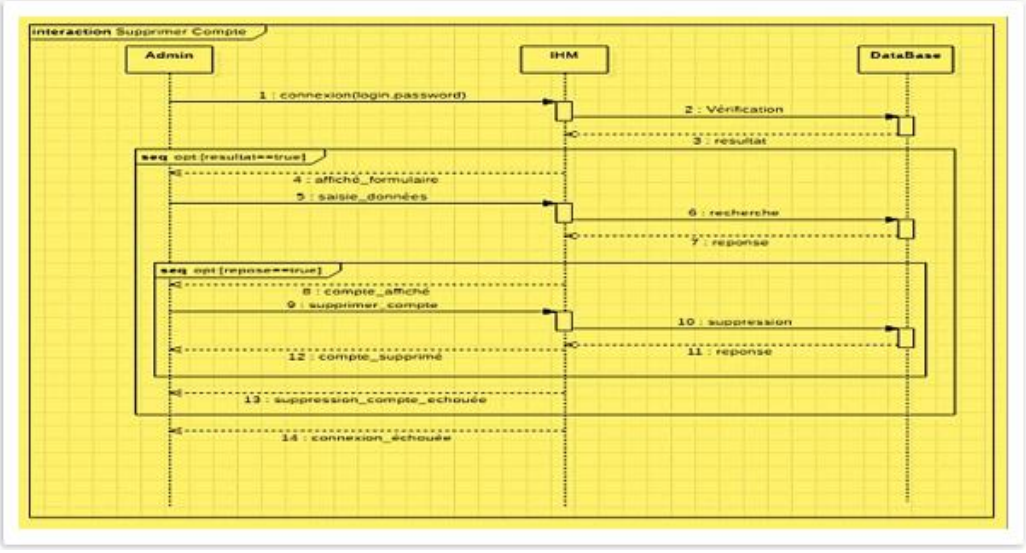


Figure 7 : *Diagramme de séquence supprimer compte utilisateur*

Diagramme de séquence « S'authentifier »

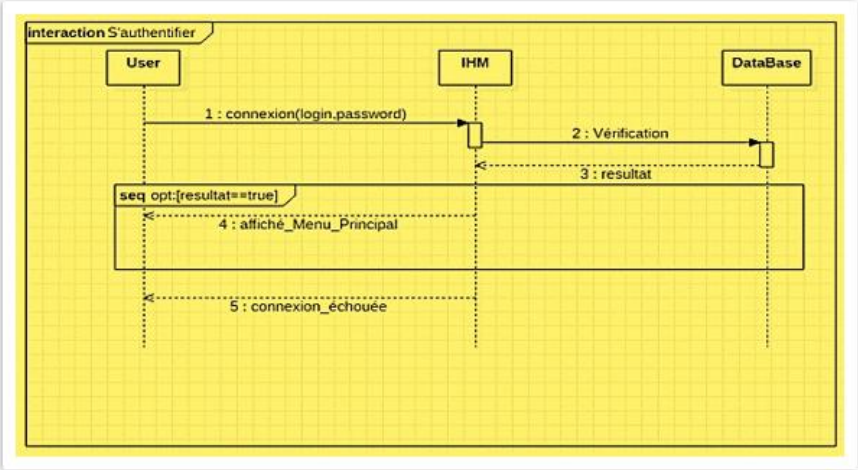


Figure 8 : *Diagramme de séquence Authentification compte utilisateur*

• **Diagramme de séquence gérer client**

Dans ce diagramme de séquence, nous avons 4 opérations de base à accomplir :
Création, modification, affichage et suppression du client.

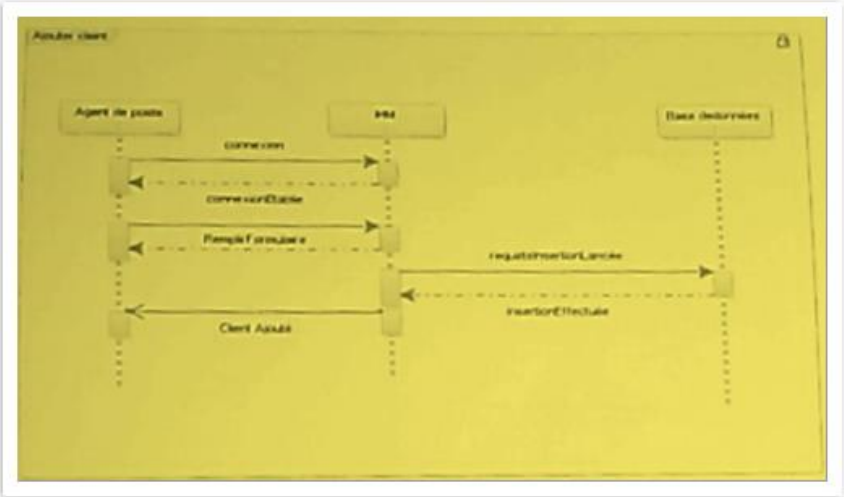


Figure 9:*Diagramme de séquence création client*

Diagramme de séquence « modification client »

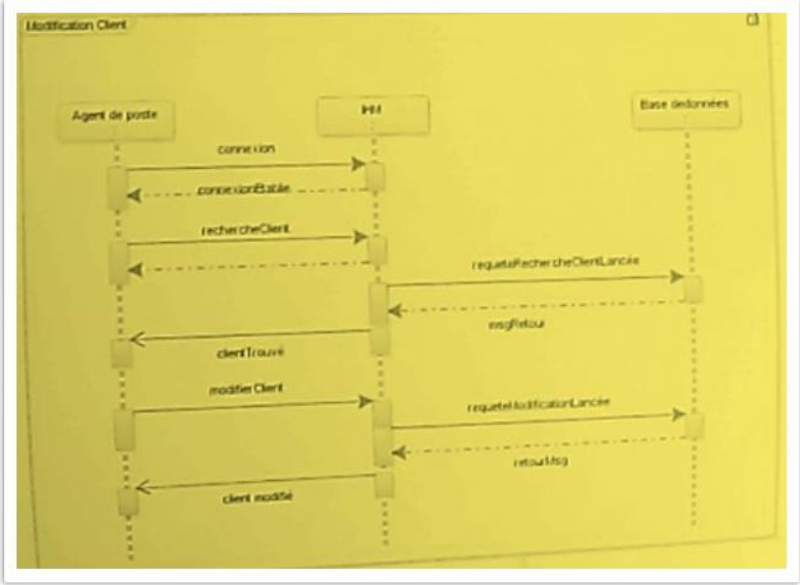


Figure 10 : Diagramme de séquence modification client

Diagramme de séquence « suppression client »

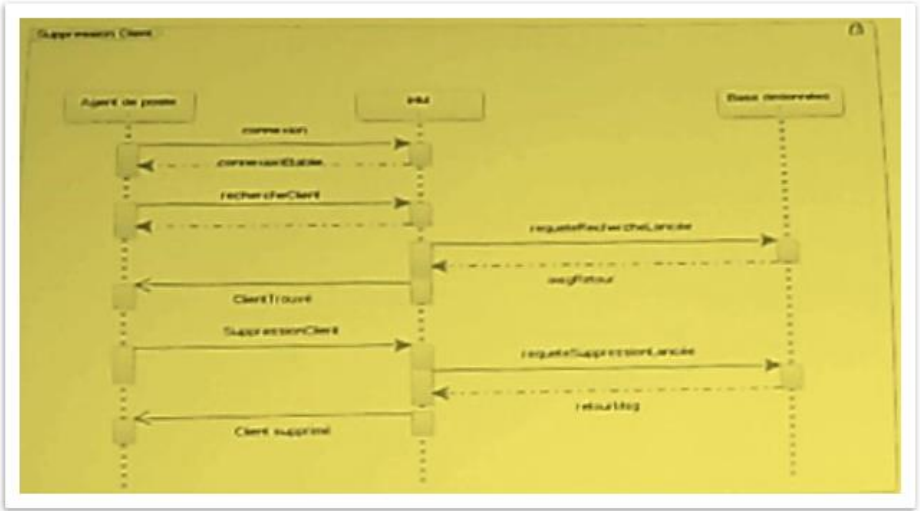


Figure 11 : Diagramme de séquence supprimer client

Diagramme de séquence « Affichage client »

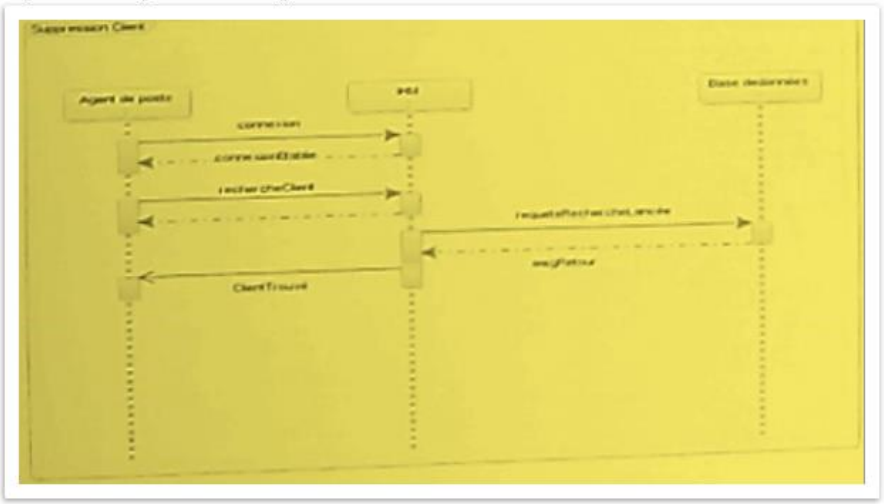


Figure 12 : Diagramme de séquence Afficher client

• **Digramme de séquence Gérer Marchandise**

Dans ce diagramme de séquence, nous avons 4 opérations de base à accomplir :
Création, modification, affichage et suppression de marchandise.

Diagramme de séquence « Ajout marchandise »

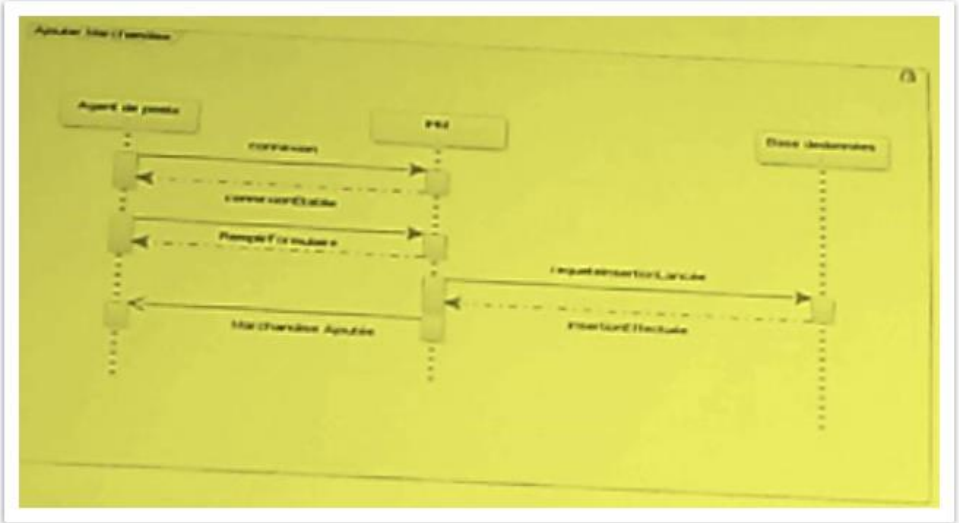


Figure 13 : Diagramme de séquence Ajout marchandise

Diagramme de séquence « Modification marchandise »

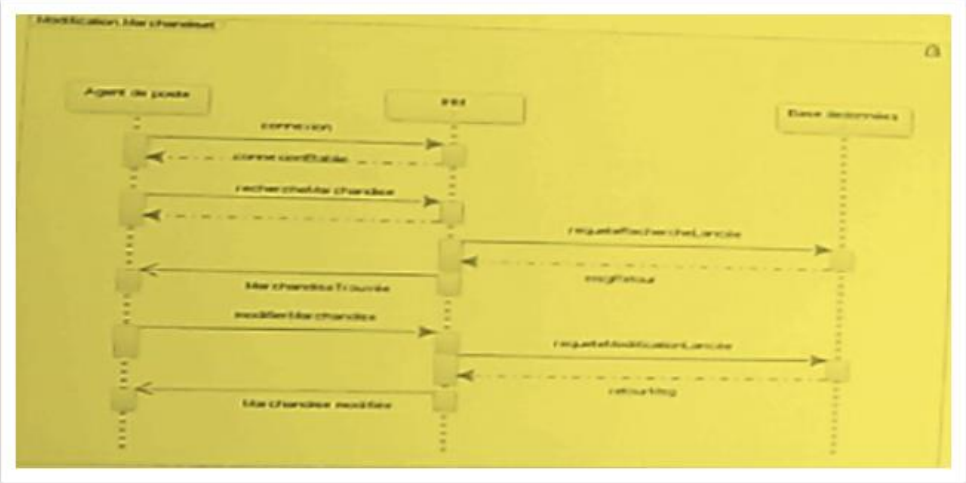


Figure 14 : *Diagramme de séquence modification merchandise*

Diagramme de séquence « Suppression marchandise »

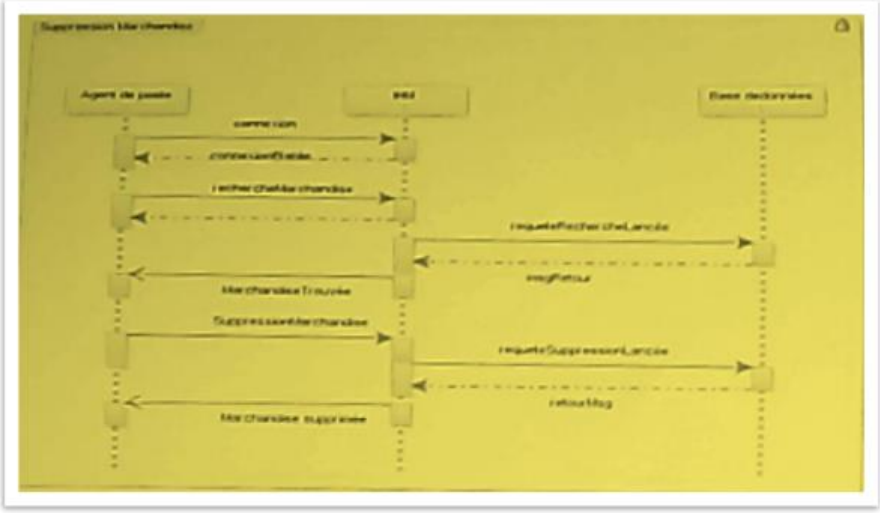


Figure 15 : *Diagramme de séquence supprimer marchandise* Diagramme de séquence « Affichage marchandise »

Diagramme de séquence « Affichage marchandise »

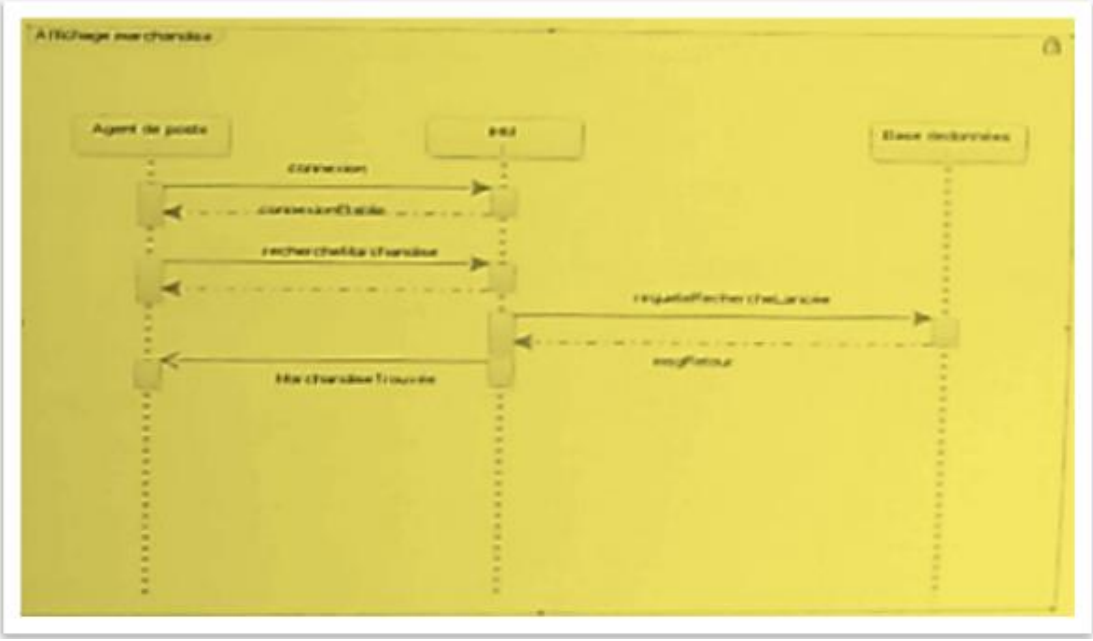


Figure 16 : *Diagramme de séquence Affiche marchandise*

Diagramme de séquence « Vérification des opérations »

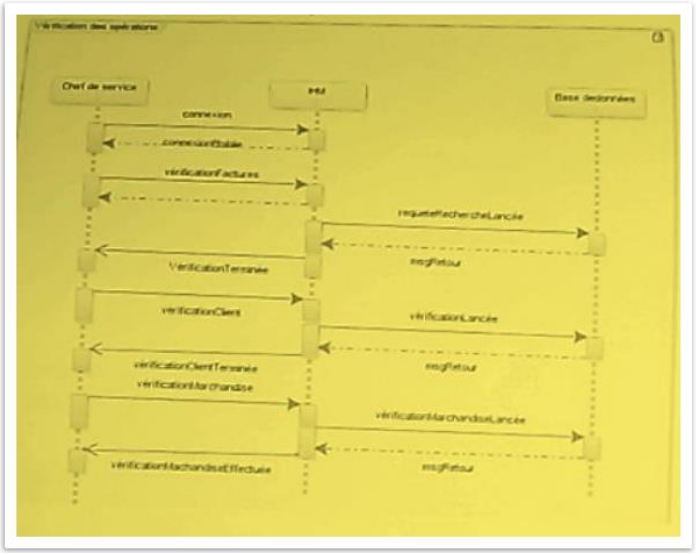


Figure 17 *: Diagramme de séquence vérification des opérations*

• **Digramme de séquence Gérer Facturation**

Dans ce diagramme de séquence, nous avons 4 opérations de base à accomplir :
Création, modification, affichage et suppression de facture.

Diagramme de séquence « Ajout facture »

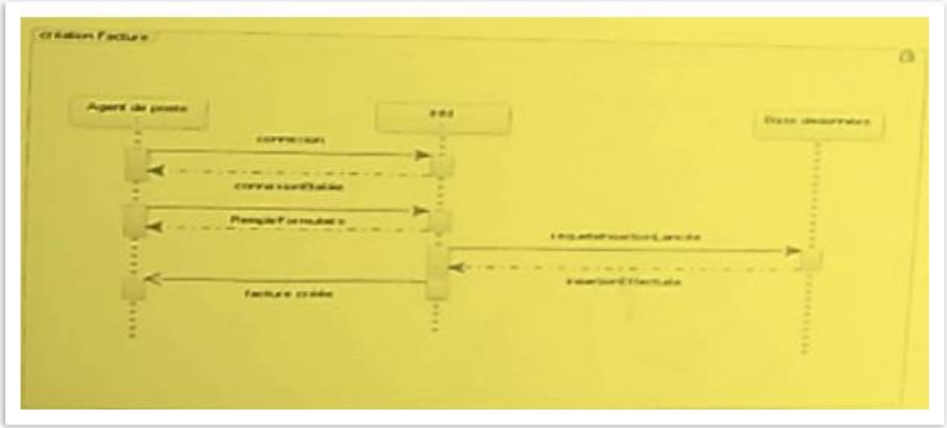


Figure 18 : Diagramme de séquence Ajout facture

Diagramme de séquence « Modification facture »

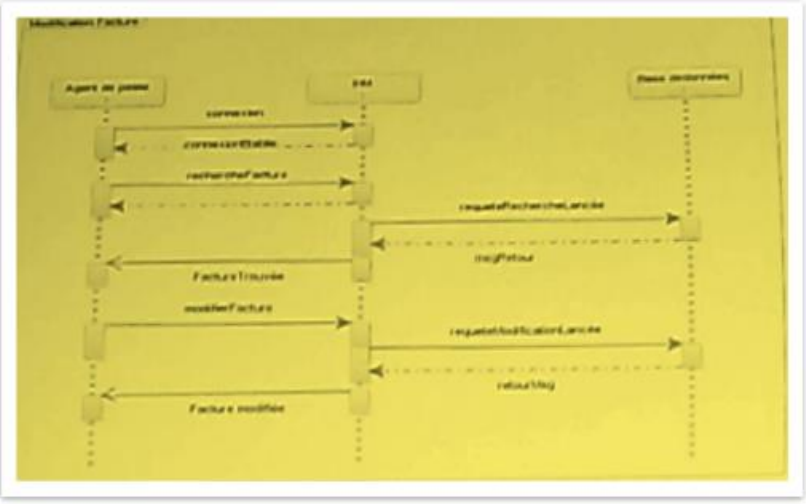


Figure 19 : *Diagramme de séquence modification facture*

Diagramme de séquence « Suppression facture »

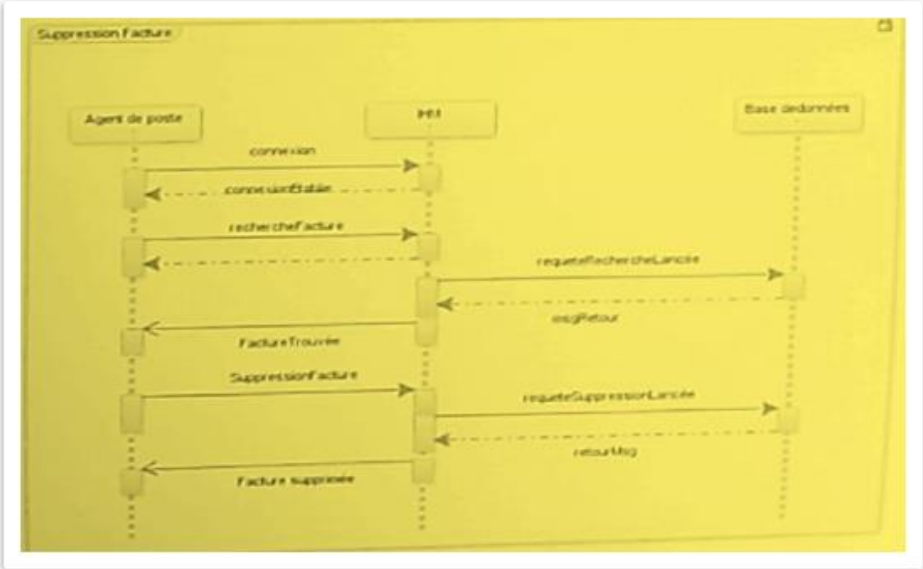


Figure 20 : *Diagramme de séquence supprimer facture*

Diagramme de séquence « Affiche facture »

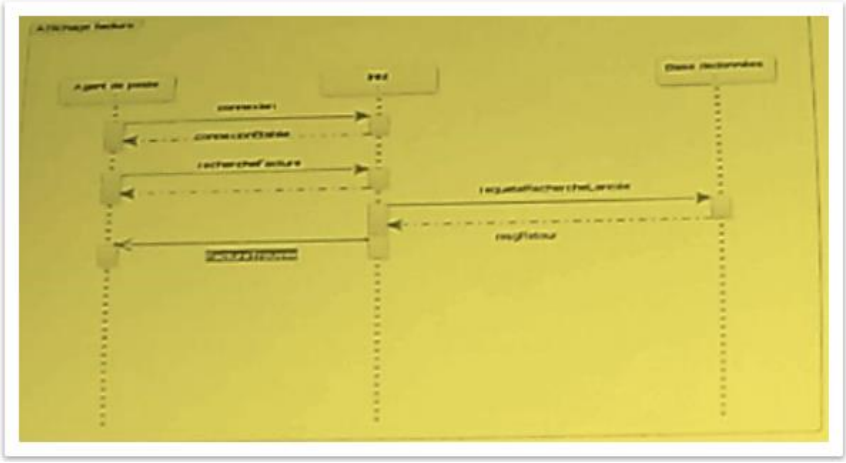


Figure 21 : Diagramme de séquence Affiche facture

Diagramme de séquence « Avaliser les opérations »

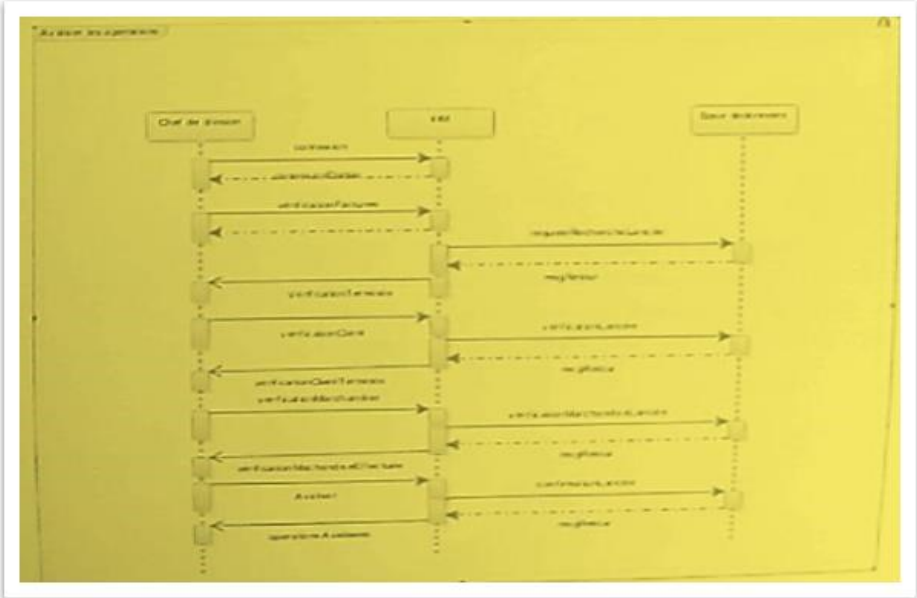


Figure 23 : *Diagramme de séquence Avaliser les opérations*

Diagramme de séquence visualiser rapport

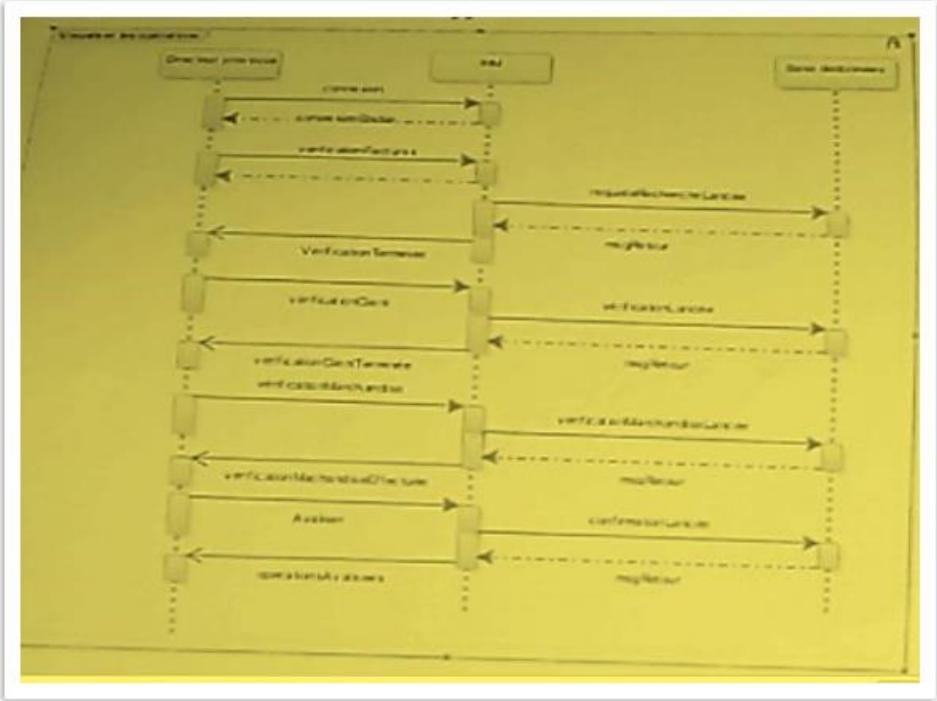


Figure 24 : *Diagramme de séquence Visualiser les rapports*

I.V. PRESENTATION DES TECHNOLOGIES

Nous présentons les outils technologiques qui constituent le socle de notre application finale. Il faut se rappeler que l'application que nous présentons repose sur une architecture distribuée et elle est de type 3 couches. Ainsi nous présentons la partie serveur qui constitue le pilier de notre application globale implémentée en Java EE 7 et la partie client permettant aux utilisateurs locaux ou distants d'interagir avec le back-end. Cette dernière solution possède plusieurs technologies pouvant accéder au serveur, il agit de clients mobiles implémentés en Android, des clients web représenté en JSF et du client riche desktop JavaFx.

IV.1. APPLICATION COTE SERVEUR OU BACKEND (JAX-RS)

Nous avions évoqué une application distribuée à plusieurs couches (3-tiers) reposant sur la plateforme Java EE. La plateforme Java EE offre plusieurs possibilités pour développer une application distribuée. En ce qui nous concerne, nous avions opté pour un service web de type REST qui permet un accès hétérogène à divers clients tels qu'Android, Java SE, JSF, etc.

OUTILS TECHNOLOGIQUES UTILISES

REST (Representational State Transfert) est un type d'architecture reposant sur le fonctionnement même du web, qu'il applique aux services web. Pour concevoir un service REST, il faut bien connaître tout simplement le protocole HTTP (Hypertext Transfert Protocole), le principe des URI (Uniform Resource Identifiers) et respecter quelques règles. Il faut raisonner en termes de ressources.

Dans l'architecture REST, toute information est une ressource et chacune d'elles est désignée par une URI - généralement un lien sur le Web. Les ressources sont manipulées par un ensemble d'opérations simples et bien définies. L'architecture client-serveur de REST est conçue pour utiliser un protocole de communication sans état - le plus souvent HTTP. Ces principes encouragent la simplicité, la légèreté et l'efficacité des applications.

L'implémentation REST en Java est JAX-RS. JAX-RS est une collection d'interfaces et d'annotations Java qui simplifient le développement des applications REST côté serveur. En utilisant la technologie JAX-RS, les applications REST (Representational State Transfer) sont plus simples à développer et consommer comparé aux autres types de systèmes répartis49.

Pour réaliser notre application, nous avions eu recourt aux outils suivants :

- NetBeans 8.2
- Java EE 7
- JPA 2.0
- Glassfish Serveur 5
- JAX-RS 2.0

- MySQL Serveur 5
- BootFaces 1.1
- Postman.

IV.2. APPLICATION COTE CLIENT OU FRONT-END (Android, JavaFx et JSF)

La partie client de notre système distribué constitue le point d'interaction avec l'utilisateur final. Elle est essentielle pour faciliter l'utilisation harmonieuse des échanges de données avec le back-end. Selon l'architecture qui a été définie dans cette étude, notre front-end offre plusieurs types d'accès à notre service web implémenté ci-haut.

Parmi les différentes technologies clientes existantes, nous optons les trois types suivants :
Un client mobile sous Android ;

Un client léger web avec JSF ; Un client riche avec JavaFx.

Pour mieux gérer les ressources aux différents utilisateurs, seul le client riche implémentera toutes les fonctionnalités de l'application. Une série d'opérations doit être définie dans les différents clients.

CONCLUSION

Enfin, nous sommes au terme de notre article, l'idéal au départ était d'apporter notre contribution en mettant en place un système informatisé distribué dans la gestion des prestations des services de l'Office Congolais de Contrôle/Direction Orientale. Avant d'aborder le corps de notre article, nous somme parti d'un constat sur la gestion manuelle des prestations qui poser d'énormes difficultés dans la transmissions des donnés par des postes de travail à la direction Provinciale.

Compte tenu de toutes ces difficultés, nous nous sommes posé cette question qui constituait notre problématique : Quelle solution capable de facilité la transmission des données des prestations en temps réel entre les postes de contrôle et les Services des traitements centralisé ? Partant de cette question, la solution idéale compte tenu de problème soulevées sera l'ajout d'une couche logicielle à l'infrastructure informatique déjà existante. Sur ce, la solution pouvant prendre en compte la mobilité des agents dans les postes de contrôle pour intégration solide avec le reste de l'infrastructure logicielle serait l'implémentation d'une nouvelle couche logicielle mobile sous Android reposant sur la technologie REST facilitant les échanges entre les agents et le serveur d'application qui est hébergé au sein de l'entreprise. Cette solution permettrait à l'OCC / Dior d'améliorer les différentes tâches liées à la gestion des prestations des services rendus dans les postes de contrôle et de lui épargner du traitement manuel des données et du risque de perte de données sensibles Pour mener bien notre recherche, nous avons eu recours aux méthodes et techniques suivantes :

- La méthode UP : nous a permis d'analyser le système existant et de concevoir le nouveau système en se servant des diagrammes suivants : - diagramme de cas d'utilisation, - diagramme de séquence ; diagramme de classe.
- COCOMO : nous a aidés à faire l'estimation de l'effort fournir pour ce projet et la durée tenant compte des moyens alloués pour ce travail.

Les techniques utilisées : - la documentation: qui nous a permis la récolte et l'analyse des données à partir des textes, ouvrages et documents qui cadrent avec notre étude.

- l'interview : nous a aidé à recueillir les informations de personnels les mieux concernés par notre travail.

Pour atteindre l'objectif que nous nous sommes fixés dans cette étude, nous avions subdivisé ce travail en quatre grandes parties hormis l'introduction et la conclusion.

La première partie qui s'est intéressé à la définition des concepts de base en vue d'éclairer le lecteur sur la terminologie et les architectures utilisées. La seconde consistait à faire l'analyse fonctionnelle et technique de l'application à mettre en place en fixant les fonctionnalités nécessaires attendues et à poser les bases technologiques de l'ensemble de l'architecture logicielle à construire. Dans cette section, nous avions également fait usage de la méthode COCOMO qui nous a permis d'estimer le cout de cette réalisation informatique. La troisième partie abordait la conception du nouveau système en faisant usage de la méthode UP utilisant le langage UML pour capturer les besoins utilisateurs par la présentation de séquence et de classes. La dernière quant à elle s'est basée sur la présentation des technologies indispensable à la mise en œuvre de cette application.

Les différentes étapes définies dans cette étude ont constitué le cadre idéal pour développer une application informatique distribuée composée des plusieurs couches logicielles. Cette démarche ainsi adoptée permet à tout informaticien de se lancer dans la construction des applications reposant sur des architectures logicielles distribuées et facilite le choix des technologies à adopter. Compte tenu de cette clarification et de cette démarche, nous pouvons en toute humilité confirmer notre hypothèse et affirmer haut que l'objectif que nous nous sommes fixés a été atteint.

Aux autorités de l'Office Congolais de Contrôle, nous leur demandons d'utiliser cette solution pour améliorer la gestion des prestations et la transmission des données en temps réel avec des postes de contrôle et la Direction Provinciale.

Ce travail est une œuvre humaine qui ne manque pas d'imperfections et des remarques, c'est ainsi que nous sollicitons indulgence à tous nos lecteurs et sommes disposé à recevoir dans un esprit scientifique leurs suggestions pour améliorer d'avantage notre recherche.

BIBLIOGRAPHIE

Ouvrages

[1]. Antonio Goncalves, Java EE 6 et GlassFish 3, Pearson Education France, 2010, ISBN : 978-2-74404157-0.
[2]. Fien VAN DER HEYDE et Laurent DEBRAUWER, UML 2 Initiation, exemples et exercices corrigés 2ième édition, Editions ENI.
[3]. Frédéric Chuong, Olivier Corgeron Cyril Joui, Jean-Baptiste Renaux et Maxime Vialette, op. cit., p.2
[4]. Gérard CASANOVA - Denis ABÉCASSIS, Gestion de projet - calculs des coûts, Université de Lorraine, 2014, p.11
[5]. Goncalves A., les Cahiers du Programmeur : Java EE 5, éditions Eyrolles, 2007, ISBN : 978-2-21212038-7
[6]. Joseph Gabay et David Gabay, UML 2. Mise en œuvre guidée avec études de cas : ANALYSE ET CONCEPTION, Dunod, Paris, 2008
[7]. Kenneth L., Jane L., Eric F., Serge C. et Sophie C., management des systèmes d'information, 13ième édition, Nouveaux Horizons, 2013.
[8]. M.Bigaud, JP Bourey, H. Camus, D. Corbeel, conception des systèmes d'information : Modélisation de données, étude des cas, Edition TECHNIP, Paris, 2006, page 14.
[9]. Pascal Roques et Franck Vallée, UML 2 en action, De l'analyse des besoins à la conception 4e édition, Eyrolles 2007.
[10]. Pitman, N., *UML 2 en concentré.* O'Reilly 2006.
[11]. Robert ENGLANDER, Java et SOAP, O'Reilly, 2009, page 45.
[12]. Safae Laqrichi. Approche pour la construction de modèles d'estimation réaliste de l'effort/coût de projet dans un environnement incertain : application au domaine du développement logiciel, thèse de doctorat, Université de Toulouse, Informatique, délivré par l'Ecole des Mines d'Albi-Carmaux, 2015.

Webographie

[1]. http://remy-manu.developpez.com/introjava2ee
[2]. https://fr.wikipedia.org/wiki/Constructive_Cost_Model
[3]. https://www.lecoindesjeux.com/rediger-une-specification-fonctionnelle-detaillee/
[4]. https://www.oracle.com