N° d'ordre : 16/DESS - TNSI Année Universitaire : 2003 / 2004

UNIVERSITE D'ANTANANARIVO

ECOLE SUPERIEURE POLYTECHNIQUE

MEMOIRE DE FIN D'ETUDES

en vue de l'obtention

du Diplôme d'Etudes Supérieures Spécialisées
en Technologies Nouvelles des Systèmes d'Information

par : **RANDRIANAVALOTIANA Rivo Jean Michel**

Mise en place d'une architecture J2EE appliquée au calcul de prime d'assurance automobile

Soutenu le mardi 05 Avril 2005 à 10 h 30 mn devant la Commission d'Examen composée de :

Président :

M. RANDRIAMITANTSOA Paul Auguste

Examinateurs :

1. Mme RABEHERIMANANA Lyliane
2. M. RATSIMBA Joël
3. M. RANDRIANARIJAONA Lucien Elino

Directeur de mémoire :

M. RAMANANTSIZEHENA Pascal

REMERCIEMENTS

Je tiens à remercier tous ceux qui ont participé à l'élaboration de ce mémoire de DESS sur les Technologies Nouvelles des Systèmes d'Information. Mes remerciements les plus sincères s'adressent à :

Monsieur RANDRIANOELINA Benjamin, Professeur, Directeur de l'Ecole Supérieure Polytechnique d'Antananarivo, pour la période d'études, d'avoir accepté ma candidature ;

Monsieur RANDRIAMITANTSOA Paul Auguste, Professeur, qui me fait l'honneur de présider les membres du jury de ce mémoire ;

Monsieur RAMANANTSIZEHENA Pascal, Professeur, Directeur de ce mémoire, qui, malgré ses lourdes responsabilités, m'a toujours prodigué ses conseils. Je tiens à lui adresser toute ma gratitude ;

Monsieur RAMIANDRISON Jimmy, Directeur Général de la Société d'assurance ARO, qui a accepté ma demande d'approfondir mes connaissances en informatique à l'ESPA ;

Monsieur RATSIMBA Joël, Chef du Département Système d'Information, qui a accepté de m'encadrer durant le travail ;

Monsieur RANDIMBISON Josoa, Auditeur Informatique et Organisationnel, qui a bien voulu accepter de m'introduire dans le Service Informatique;

Monsieur RABAKO Andriarimalala Tovohery, Administrateur de données, et Monsieur RAHARISON Hagalalaina, Responsable Service Etudes et Organisation, de m'avoir aidé et donné toutes les informations nécessaires pour l'accomplissement de mon travail ;

Mes plus vifs remerciements s'adressent également à tous les enseignants et personnel administratif de l'ESPA, sans leur soutien et encadrement, ma formation n'aurait pas atteint ce niveau ;

Je n'oublierai pas :

Tout le personnel de l'assurance ARO pour leur sympathie ;

Toute ma famille qui m'a toujours soutenu durant toutes mes études ;

Tous ceux qui, de près ou de loin, m'ont aidé dans l'élaboration du présent mémoire.

TABLE DE MATIERES

NOMENCLATURE

APACHE - serveur web APACHE
Le serveur web le plus populaire. Un produit de Apache Software Foundation.

API - Application Programming Interface
Ensemble de routines standards, accessibles au moyen de l'interface utilisateur graphique du système d'exploitation, et qui sont destinées à faciliter au programmeur le développement d'applications.

APPLET - APPLET JAVA
Petite application écrite en langage Java et qui, insérée dans un document web, exécute ses objets multimédias en présence d'un navigateur compatible, et ce, directement sur l'ordinateur de l'internaute, peu importe le système d'exploitation utilisé.

BES - Borland Enterprise Server
Une infrastructure pour J2EE, services web, et applications CORBA de Borland Software Corporation.

BMP - Bean-Managed Persistence
Un mécanisme dans lequel le transfert de données entre une variable d'une entity bean et le gestionnaire de ressource est géré par l'entity bean.

CA - Certification Authority
Tiers de confiance qui crée des certificats composés d'assertions sur divers attributs, et les associe à une entité et/ou leurs clés publiques.

CMP - Container Managed Persistence
C'est une expression utilisée dans le EJB pour désigner l'interaction automatique avec les bases de données. Les conteneurs J2EE (ou les serveurs d'applications) enregistrent automatiquement les EJB dans la base de données.

CONTAINER
Une entité qui fournit la gestion des cycles de vie, de la sécurité et du moteur d'exécution de services des composants J2EE. Chaque type de container (EJB, Web, JSP, servlet, applet et application client) fournit aussi des services spécifiques au composant concerné.

CORBA - Common Object Request Broker Architecture
Standard définissant l'architecture que doivent adopter les systèmes d'exploitation et les applications pour rendre possible la communication entre des objets provenant d'environnements différents.
Le standard CORBA n'est pas une norme au sens strict du terme, car ses spécifications n'ont pas été entérinées par un organisme officiel de normalisation. Il a été défini et adopté de façon consensuelle par un ensemble d'entreprises regroupées sous le nom de Object Management Group ou OMG.

EAR - Enterprise Archive file
Un archive JAR qui contient une application J2EE.

EJB - Enterprise JavaBeans ou enterprise bean
La technologie EJB est le côté serveur de l'architecture à composant pour J2EE. La technologie EJB permet le développement rapide et simplifié des applications distribuées, transactionnelles, sécurisées et portables basées sur la technologie Java.

HTML - Hyper Text Markup Language
HTML est un langage de description à balises. Il permet de créer des documents hypertextes utilisés dans le web. Les balises spécifient aux navigateurs comment afficher les données.

HTTP - HyperText Transfert Protocol
Protocole utilisé pour transférer des documents hypertextes ou hypermédias entre un serveur web et un client web.

HTTPS
Version http sécurisée sur SSL.

IDL - Interface Definition Language
Langage développé selon la norme CORBA, et qui permet de définir l'interface d'accès à un objet, indépendamment de la façon dont cet objet est écrit.

IETF - Internet Engineering Task Force
Groupe de travail qui développe les nouveaux standards pour l'internet.

IIOP - Internet Inter-ORB Protocol
Protocole de communication permettant d'intégrer et de mettre en réseau des applications de provenances diverses.

ITU-T - International Telecommunication Union Telecommunication standardization sector
Anciennement le CCITT (Consultative Committee for International Telegraph and Telephone), une organisation mondiale de standardisation de technologies de télécommunication

J2EE Java 2 Platform, Enterprise Edition
Plateforme Java spécialisée dans le développement et le déploiement, côté serveur, des applications complexes destinées aux entreprises, à l'intérieur d'une architecture multitiers.
Développée à partir de la plateforme J2SE, la plateforme J2EE simplifie la programmation des applications, notamment par l'utilisation des composants modulaires standardisés (comme les composants logiciels EJB) et par la manipulation automatique de nombreuses fonctions touchant le comportement des applications.

J2SE - Java 2 Platform, Standard Edition
Plateforme java de base conçue pour tourner sur des ordinateurs de bureau, et fournissant les outils nécessaires au développement rapide, au déploiement et à l'exécution, côté client, d'applets et de programmes écrits en langage Java.

JAVA
Le java est un langage de développement écrit par James Gosling, produit par la société SUN et lancé le 23 mai 1995. C'est un langage de programmation orienté réseau et objet. Sa syntaxe est dérivée de celle du C++. Son principal avantage est d'être entièrement portable.

JAR - Java Archive
Un fichier dont le format est indépendant du plateforme permettant à plusieurs fichiers être intégrés sur à seul.

JavaBeans - JavaBeans component model
Modèle de composant logiciel développé à partir du langage Java, transférable et indépendant de plateforme, qui permet de créer des composants logiciels dynamiques, réutilisables et interactifs.

JDBC - Java Database Connectivity
JDBC est un API java permettant aux programmes java d'exécuter des requêtes SQL. Il permet aux programmes java d'interagir avec n'importe quelle base de données compatibles avec SQL.

JDK - Java Development Kit
Outils de développement Java constitués de la documentation et des modules permettant de produire des programmes Java. Le JDK développé par Sun comprend entre autres un compilateur Java, un débogueur Java ainsi qu'un visualiseur d'applet.

JMS - Java Messaging Service
Un API pour invoquer des opérations dans le système de messagerie d'entreprise.

JNDI - Java Naming and Directory Interface
Fournit un interface de nommage multiple et de service répertoire unifié pour que les composants des applications utilise les nommages et les services répertoires.

JSP - JavaServer Page
Une technologie web extensible qui utilise des données statiques, des éléments JSP et des objets Java côté serveur pour générer un contenu dynamique pour un client.

MAC - Message Authentication Code
Fonction de hachage à sens unique dépendant d'une clé, nécessitant l'utilisation d'une clé identique pour vérifier le hachage.

MD5 - Message Digest (5^e version) algorithm
Fonction de hachage à sens unique conçue par Ron Rivest, dépendant d'une permutation aléatoire d'octets pour créer une empreinte de 128 bits.

Module Web
Un unité déployable consistant en un ou plusieurs composants web, autres ressources et un descripteur de déployement d'application web contenus dans des hiérarchies de répertoires et de fichiers respectant le format standard des applications web.

MULTITIERS – architecture multitier
Architecture dérivée du modèle client-serveur, dans laquelle on peut insérer, au besoin, des niveaux logiciels supplémentaires entre le niveau client et le niveau serveur, permettant ainsi à des entreprises d'étendre leurs activités sur le web.

ODBC - Open DataBase Connectivity
Interface développée par Microsoft en 1992, qui permet l'accès systèmes de bases de données.

ODMG - Object Database Management Group
Consortium créé pour établir des standards pour les bases de données orientées objet.
L'ODMG définit des interfaces destinées aux bases de données.

OMG – Object Management Group
Un consortium qui conçoit et maintient les spécifications d'interopérabilité des applications entreprise.

OQL - Object Query Language
Langage d'interrogation orienté objet utilisé dans les bases de données.
OQL est un langage développé par l'ODMG et est un surensemble de SQL.

RMI - Remote Method Invocation protocol
Protocole de communication qui permet à des composants Java de s'exécuter à distance dans un environnement distribué. Un protocole développé par Javasoft de Sun.

SDK - Software Development Kit
Ensemble d'outils logiciels conçus pour aider les programmeurs à créer des programmes destinés à tourner sur une plateforme donnée.

SERVLET
Programme java qui utilise des modules supplémentaires figurant dans l'API Java, qui s'exécute dynamiquement sur le serveur web et permet l'extension des fonctions du serveur.

SGML - Standard Generalized Markup Language
Langage normalisé permettant de décrire les relations entre le contenu d'un document informatique et sa structure, et non son aspect typographique.

SHA-1 - Secure Hash Algorithm
Même fonction que MD5.

SQL - Structured Query Language
Langage d'interrogation, de mise à jour et de gestion des bases de données relationnelles.

SSL - Secure Sockets Layer protocol
Protocole permettant la transmission sécurisée de formulaires sur le web, notamment lors des transactions commerciales en ligne, nécessitant l'utilisation d'une carte de crédit.

TLS - Transport Layer Security
Le successeur du protocole SSL, créé par l'IETF pour l'encryption et l'authentification des communications sur des réseaux TCP/IP.

TOMCAT
Tomcat est un container de JSP/Servlet de référence officiel choisi par Sun. Les dernières versions de Tomcat supportent toujours les dernières spécifications des JSP et servlets.

UML - Unified Modeling Language
Langage de modélisation objets, permettant de déterminer et de présenter les composants d'un système objet lors de son développement, ainsi que d'en générer la documentation.
Le langage UML est le résultat de la fusion des travaux de Rumbaugh, Booch et Jacobson.

URL - Uniform Resource Locator address
Adressage standard de n'importe quel document, sur n'importe quel ordinateur en local ou sur internet.

W3C - World Wide Web Consortium
Organisme officiellement chargé de la normalisation de tout ce qui concerne le web, et particulièrement des évolutions du langage HTML.

WAR - Web application Archive file
Un fichier JAR qui contient un module web

WS - Java Web Services
Les Web Services sont des applications d'entreprise basés sur web. L'échange de données avec le client se base sur les standards XML et les protocoles de transports.

X-509
Norme technique des certificats numériques de l'ITU-T. La norme X.509 v3 s'applique à des certificats comportant ou pouvant comporter des extensions.

XML - eXtensible Markup Language
Evolution du langage SGML permettant aux concepteurs de documents HTML de définir leurs propres marqueurs, dans le but de personnaliser la structure de données qu'ils comptent présenter.

INTRODUCTION

L'utilisation des architectures client-serveur dans un contexte d'une entreprise à ramification nationale et surtout internationale engendre des coûts exorbitants d'exploitation et de maintenance si le nombre des utilisateurs augmente. C'est la limite du système, et ce d'autant plus, si on envisage la perspective d'intégrer directement le public dans le système.

C'est de ces limites que le système multitiers a vu le jour. L'apparition de J2EE, la version entreprise de Java, a fait tourner les développeurs vers le système multitiers. Avec J2EE, les problèmes posés par les systèmes d'exploitation ne sont plus que de douloureux souvenirs.

Le présent mémoire montre la conception et la mise en place d'une architecture multitiers avec J2EE. Cette technologie est testée pour le calcul de prime d'assurance de la société d'assurance Aro Antsahavola. C'est un projet pour la société d'évaluer les capacités réelles de la technologie ainsi que son intégration dans le système d'information.

En tant que projet, ce livre est structuré en trois parties et chaque partie détaille une phase de projet, à savoir la phase de lancement, la phase d'exécution et la phase de clôture.

La phase de lancement se limite aux documents nécessaires, à la planification, aux outils et matériels nécessaires et aux diverses études de faisabilité.

La phase d'exécution détaille les exécutions planifiées dans la phase de lancement. On y verra les divers outils de conception et de mise en œuvre comme l'UML et MERISE.

La phase de clôture donne les divers bilans : le bilan technique, le bilan de l'organisation, le bilan économique, le bilan qualité et les livrables.

PARTIE I - PHASE DE LANCEMENT DU PROJET

Cette phase comprend :

- Le cadre logique du projet
- La définition du cadre contractuel du projet
- Les documents nécessaires
- Le cahier de charge
- L'identification du maître d'œuvre et du maître d'ouvrage
- La mise en place de l'organisation et la constitution des divers comités
- L'établissement du planning du projet

• CHAPITRE I - CADRE CONTRACTUEL DU PROJET

I.1 – CADRE LOGIQUE

LOGIQUE D'INTERVENTION	INDICATEURS OBJECTIF VERIFIABLES	SOURCES ET MOYENS DE VERIFICATION	HYPOTHESES ET RISQUES
FINALITE DU PROJET : - Mise en place d'une architecture J2EE appliquée au calcul de prime d'assurance automobile	- Intégration du TIC au développement durable - Sensibilisation au TIC par les autorités	- Articles de journaux relatifs aux discours sur le TIC - Atelier de sensibilisation au TIC - Critère de choix d'une architecture appropriée	- Pas de nouvelles lois interdisant le TIC - Les efforts du gouvernement sur la promotion du TIC continuent
BUTS DU PROJET : - Mise en place de la base de données - Mise en place d'un serveur d'application et d'un serveur web - Conception et implémentation de la base, des applications et des interfaces - Test	- BES, Jbuilder, MS SQLServer prêt à installer - Lieu d'exploitation prêt - Estimation et planification effectuées - Diagramme de Gantt	- Cahier de charge - Résultat d'étude d'opportunité - Note d'estimation et planification - Bon de sortie des PC	- ARO ne renonce pas au projet - Pas de grève - Pas de rupture de stocks - Les utilisateurs ne sont pas hostiles au projet
RESULTATS : - PC serveurs configurés et prêts - BES, JBuilder, MS SQLServer installés et configurés - Applications développées - Tests effectués	- Serveurs et PC de développement - CD d'installation	- PV d'installation et de configuration - PV de test - PV de livraison - Journal de développement	- Documents perdus ou volés - Destruction du local en cas de force majeure - Matériel grillé par tension
ACTIVITES : - Se procurer des outils d'exploitation (pc serveur, Borland BES et Jbuilder) - Installer et paramétrer les outils - Développer les applications - Effectuer des tests	**MOYENS :** - 01 personnel développeur - 02 PC serveurs	**COUTS :** - Personnel : 100.000 Ar/mois - Mat. Informatiques : 2.000.000 Ar	

I.2 - OBJET DU PROJET

Le projet est un projet pilote de mise en place d'une architecture multitiers basée sur la technologie J2EE [1].

- Centraliser les données et les applications pour le calcul d'assurance automobile
- Centraliser les données et les applications
- Permettre aux utilisateurs en province d'utiliser les applications et d'accéder aux bases de données hébergées au serveur central.
- Mettre en place une application robuste, évolutive, et performante
- Réduire les taux de ressaisie et de contrôle supplémentaire à la réception des données au siège
- Réduire les interventions et les maintenances à distance

I.3 - MOTIVATION DU PROJET

Plus le volume des applications augmente, plus la difficulté de la maintenance augmente, surtout quand il s'agit de plusieurs versions et de plusieurs types d'applications.

Mais force est de constater que si les applications concernent plusieurs postes repartie géographiquement dans différentes villes et utilisant différents systèmes d'exploitation, la maintenance devient un cauchemar pour le Responsable informatique. Elle exige ainsi plus de personnel, plus de différentes sortes de compétences, plus de machines puissantes, plus de différentes versions de système d'exploitation, donc plus d'argent.

Donc les objectifs du projet est de démontrer la possibilité de centraliser des applications en un seul endroit, à travers une application de calcul de prime, en faisant des économies et en gardant la même efficacité. L' autre objectif aussi est de démontrer la puissance, la sécurité, la portabilité et la fiabilité inégalable de la technologie J2EE à travers des applications d'entreprise.

I.4 - EXIGENCES DE L'APPLICATION

Le calcul de prime d'assurance auto permet aux commerciaux d'ajouter, de mettre à jour, de consulter et de supprimer des informations sur les clients et permettre ainsi des calculs et héberger l'application au serveur central.

Nous utiliserons des interfaces web pour permettre de manipuler les entrées des informations par les clients. Les données sont stockées dans une base de données MS SQL Server 2000 consultable en plusieurs formulaires via le site web de la compagnie à la place de l'ancienne application basée sur client-serveur.

L'application doit fournir les fonctionnalités suivantes :

- Chaque entrée par table correspond à une information donnée représentée par un objet auto, avenant, risque, client.
- Chaque police a un ou plusieurs avenants et chaque avenant appartient à une et une seule police.
- Chaque police concerne un (mono véhicule) ou plusieurs véhicules (flotte) et chaque véhicule appartient à une et une seule police.
- Chaque véhicule fait l'objet d'un ou plusieurs avenants et chaque avenant concerne un et un seul véhicule.
- Un client peut avoir une ou plusieurs polices et chaque police appartient à un et seul client.

- Un client peut avoir un ou plusieurs véhicules et un véhicule appartient à un et un seul client.
- Un véhicule est destiné à une et une seule utilisation et une utilisation peut correspondre à plusieurs véhicules.

I.5 - MODULES UTILISES PAR L'APPLICATION

Avant de commencer le projet, nous séparons le projet en différents modules pouvant être construits séparément et indépendamment.

On a défini trois modules :

- Module stockage : permet le stockage et l'extraction des données dans la base des données.
- Module calcul : procède aux calculs des primes.
- Module manipulation : permet la manipulation des données
- Module web : sert d'accès à distance aux informations pour les commerciaux.

Diviser notre application en différents modules fournit plusieurs avantages avant, pendant et après le développement. Avec cette structure, un travail en équipe est possible car chaque module est indépendant si on respecte les contraintes d'interaction.

Cet approche est très bénéfique en donnant aux modules une modularité parfaite. En d'autres termes, des améliorations sur un module sont possibles sans toucher aux conceptions des autres équipes.

• CHAPITRE II - LES DOCUMENTS NECESSAIRES

Les documents suivants sont nécessaires:
- Résultats d 'étude d'opportunité
- Critères de choix d'une architecture appropriée
- Assurance automobile.
- Tarification
- J2EE [1].
- Serveur d'application BES 5.1 [2].
- Cahier de charges

II.1 - I.B.1 - RESULTATS D'ETUDE D'OPPORTUNITE

II.1.1 - PLACE DE L'AUTO

La branche auto représente 30% du chiffre d'affaire de la société d'assurance ARO. Et par rapport aux autres sociétés d'assurance, ARO représente 45% du marché rien que sur l'assurance auto.

Au vu de ces chiffres, nous constatons que cette branche occupe une part importante de l'activité de la société et qu'elle mérite de faire l'objet d'améliorations et de suivi de très près.

II.1.2 - HISTORIQUE DE L'APPLICATION

L'application pour le calcul de prime automobile (mono véhicule et flotte) utilisée par la société d'assurance ARO remonte aux environs de 1994. Elle a été conçue par deux Ingénieurs informaticiens et sa conception a duré environ six mois. Elle a été réalisée sous CLIPPER.

Avec l'expansion de la société à travers le territoire de Madagascar, les agences ont besoin de l'application et la société est forcée d'en installer une par agence. La ville d'Antananarivo même comporte plusieurs agences et chaque agence a sa propre application.

Récemment, la société a fait l'acquisition d'une nouvelle application nommée win@pass écrit en visual basic. Cette nouvelle application utilise l'architecture client-serveur. Actuellement, la société ARO commence à l'exploiter dans ses agences. Etant écrit sous visual basic, l'application reste très liée au système d'exploitation windows.

II.1.3 - CONTRAINTES RENCONTREES

Les contraintes suivantes sont relevées :
- Taux d'intervention journalier très élevé, toutes interventions confondues.
- Nombre de personnel informatique insuffisant,
- Nécessité d'un logiciel de télé maintenance PCAnywhere pour la maintenance,
- Portabilité des applications limitée au système d'exploitation Windows seulement,
- Nécessité des machines puissantes pour faire tourner les applications,
- Aucune mobilité des clients,
- Coût d'extension très élevé (acquisition nouvelle machine puissante, déplacement, …),
- Durée d'extension de plusieurs jours pour une agence,
- Installation des applications très difficile,
- Nécessité de déplacement en cas de panne du PCAnywhere,

II.1.4 - OPPORTUNITES POUR LA NOUVELLE TECHNOLOGIE

La technologie J2EE [1] permet de :

- centraliser les applications,
- limiter en un seul endroit la maintenance (plus besoin de télé maintenance),
- garder constantes les charges de travail du personnel chargé pour la maintenance,
- garder constant le nombre de personnel qualifié nécessaire,
- éliminer les problèmes d'installation et d'adaptation,
- rester indépendant du système d'exploitation utilisé,
- rester indépendant de la version du système d'exploitation utilisé,
- ne pas exiger des machines puissantes pour les clients,
- offrir une disponibilité 7/7 et 24/24 en fonction du besoin,
- offrir une parfaite extensibilité en fonction des charges,
- rendre le coût d'extension nul,
- réduire le délai d'extension à quelques minutes,
- garantir la récupérabilité d'une session en cas de panne du serveur,
- offrir aux clients une mobilité parfaite.

II.2 - CRITERES DE CHOIX D'UNE ARCHITECTURE APPROPRIEE

La sélection d'une architecture appropriée amène les responsables à considérer plusieurs aspects importants qui ont des avantages , des inconvénients et qui nécessitent des compromis.

II.2.1 - PERFORMANCE, DISPONIBILITE, EXTENSIBILITE

Les applications d'entreprise sont particulièrement sensibles aux questions de disponibilité et de performance, surtout dans des situations critiques où les serveurs sont très surchargés.

II.2.1.1 - MAINTENANCE ET MISE A JOUR

Contrairement aux logiciels de distribution, qui sont développés à partir des plans de réalisation, les applications conçues pour être utilisées dans une entreprise ont typiquement évoluées en permanence [2].

Si l'application tient le succès d'une affaire, elle fera l'objet de corrections de bugs et d'améliorations. Dans une telle situation, la modularité et la flexibilité de la conception doivent être les aspects de l'application permettant des modifications indépendantes de la logique de l'application elle-même [2].

II.2.1.2 - FACTEUR DE RISQUE CONNUE

Si on construit une application ayant une mission très critique, tâcher de dépenser plus de temps à concevoir le traitement des transactions et le développement d'une architecture qui réduisent les risques d'interruptions dans le programme s'avère souvent être l'aspect le plus compliqué et le plus consommateur de temps de la conception et du test de l'application [2].

II.2.2 - CONSIDERATION TECHNIQUE

II.2.2.1 - COMPLEXITE ET LIMITE

Si l'application doit assurer les traitements des données de multiples sources, l'extraction des données, la gestion d'une transaction complexe, il est clair qu'une architecture très sophistiquée est au menu, c'est le modèle multitiers [1, 2].

Autrement dit, s'il n'y a que quelques étapes concernées, le modèle orienté web peut être utilisé, ce qui élimine les complexités et peut réduire le temps de développement requis pour finir le projet.

II.2.2.2 - REUTILISABILITE

L'application utiliserait-elle des composants qui existent déjà ? Dans le cas de développement d'une application faisant partie d'une large série de projets, le temps dépensé au développement de composant se paiera à long terme [2].

II.2.2.3 - DUREE DE VIE ET POSSIBILITE DE MODIFICATION

Qu'est-ce qui se passera si les exigences doivent changer durant la vie de l'application ? Une longue durée de vie avec possibilité de fréquentes modifications exige une application à haut degré de flexibilité et à architecture modulaire [2].

Pourtant, une application destinée à des utilisations temporaires ne devra pas bénéficier de la complexité accrue d'une architecture basée sur des composants couplés.

II.2.3 - CONSIDERATION ORGANISATIONNELLE

II.2.3.1 - TAILLE DE L'EQUIPE ET COMPETENCE

Quelle est la taille de l'équipe ? Une personne ou une large équipe de développement ?

L'équipe de développement est-elle composée de débutants ou de vétérans saisonniers ?

Une large équipe avec des compétences complémentaires tend à favoriser le modèle incorporant les EJB [1].

La possibilité de partager l'application en composants discrets favorise la division du travail, la spécialisation des développeurs, et une meilleure gestion de l'équipe [2].

II.2.3.2 - TEMPS ET ARGENT

Combien de temps et d'argent sont-ils alloués au projet ? Des niveaux de complexité accrue signifient généralement plus de temps, et dans le cas des EJB [1], plus d'argent. La complexité et le temps sont des compromis , mais il faut aussi considérer les dépenses de maintenance. Il est absurde de concevoir une application rigide et difficile à maintenir dans le but de gagner du temps et d'argent si on est constamment obligé de consacrer des ressources de développement pour maintenir le projet dans le futur [2].

II.2.3.3 - CONTROLE DES ATOUTS ET RESSOURCES

Combien de contrôle a-t-on sur les ressources importantes pour le projet ? Si l'application accède à des bases de données existant déjà et hors contrôle, l'architecture avec des couches additionnelles d'abstraction est nécessaire afin d'épargner aux développeurs des interfaces variables et disparates [2].

II.3 - ASSURANCES AUTOMOBILES

II.3.1 - MODULARITES PRATIQUES

II.3.1.1 - GARANTIE DE BASE

- Risque A : Responsabilité Civile
- Risque B : Dommages éprouvés par le véhicule
- Risque C : Incendie du véhicule
- Risque D : Vol du véhicule [23]

II.3.1.2 - GARANTIES FACULTATIVES

- BG : Bris de Glace
- GEMP : Grèves, Emeutes et Mouvements Populaires
- DPN : Dommages aux pneumatiques
- PJ : Privation de Jouissance
- IDR : Indemnisation Directe et Recours [23]

II.3.2 - DISPOSITIONS COMMUNES A LA TARIFICATION

Les dispositions, ci-après, sont applicables à toutes les catégories tarifaires, à l'exception du Tarif 4 [23] :

II.3.2.1 - RESPONSABILITE CIVILE (A)

La prime Responsabilité civile a pour base les éléments ci-dessous fixés d'après déclaration expresse du Souscripteur et constatés dans la carte grise du véhicule :

- Usage du véhicule
- Puissance fiscale en CV
- Source d 'énergie
- Nombre de place gratuites ou payantes
- Poids du véhicule : PTC
- Remorque

L'assurance du recours des tiers incendie est incluse dans la garantie Responsabilité Civile sans surprime.

II.3.2.2 - DOMMAGES EPROUVES PAR LE VEHICULE (B)

La prime « Dommages » est fonction :

- De la catégorie tarifaire à laquelle appartient le véhicule
- De la valeur du véhicule qui est celle déclarée par le Souscripteur, à laquelle devra s'ajouter la valeur des transformations
- Eventuellement, de la valeur de la remorque ou de la semi-remorque qui s'ajoute intégralement à celle du véhicule tracteur.

II.3.2.3 - INCENDIE DU VEHICULE (C)

La prime « Incendie » est fonction :

- De la catégorie tarifaire à laquelle appartient le véhicule ;
- De la valeur du véhicule, qui est celle déclarée par le Souscripteur, à laquelle devra s'ajouter la valeur des transformations ;
- Eventuellement, de la valeur de la remorque ou de la semi-remorque qui s'ajoute intégralement à celle du véhicule tracteur ;

Le risque « incendie » des véhicules en circulation ou immobilisés dans un garage peut être couvert suivant le tarif propre à chaque catégorie.

II.3.2.4 - VOL DE VEHICULE (D)

La prime « Vol » est fonction :

- De la catégorie tarifaire à laquelle appartient le véhicule ;
- De la valeur du véhicule déclarée par le Souscripteur augmentée de la valeur des accessoires hors série dont les caractéristiques et la valeur respectives sont à préciser ;

A noter qu'en aucun cas, la valeur des accessoires hors série ne doit pas dépasser 25% de la valeur assurée du véhicule. Eventuellement, de la valeur de la remorque ou de la semi-remorque.

Le risque « Vol » des véhicules en circulation ou immobilisés dans un garage peut être couvert suivant le tarif propre à chaque catégorie.

II.3.2.5 - BRIS DE GLACES

La prime est fonction de la valeur de l'ensemble des glaces du véhicule, y compris la glace du toit si celle-ci est prévue par le constructeur.

Toutefois, la valeur des glaces ne doit pas dépasser 20% de la valeur assurée du véhicule.

Cette extension de garantie ne peut être accordée qu'associée aux Risques Incendie et Vol (C, D)

II.3.2.6 - DOMMAGES RESULTANT DES GREVES, EMEUTES ET MOUVEMENTS POPULAIRES (Extension sur B – C – D – BG)

Cette extension de garantie peut être octroyée moyennant surprime et sous réserve de la souscription des garanties de base [23].

Dans le cas où les garanties de base comportent une franchise, cette même franchise est applicable au présent risque.

En cas de franchise forfaitaire, sur le risque B uniquement, il y a lieu d'appliquer la réduction prévue à cet effet.

II.3.2.7 - DOMMAGES AUX PNEUMATIQUES

Cette extension de garantie ne peut être accordée sans la souscription de la garantie « Dommages éprouvés par le véhicule » et ce, moyennant surprime.

II.3.2.8 - PRIVATION DE JOUISSANCE

Cette garantie est exclusivement réservée aux véhicules [23] :

- A 4 roues de la catégorie 1
- Des catégories 2 et 3 dont le PTC est inférieur ou égal à 3,500 T et à titre gratuit

La prime est fonction du montant de l'indemnité journalière choisie par l'Assuré.

II.3.2.9 - INDEMNISATION DIRECTE ET RECOURS

La garantie « Indemnisation directe et Recours » peut être accordée même avec la garantie « Responsabilité Civile » seule [23].

La prime nette annuelle afférente à cette garantie est fixée forfaitairement pour toutes les catégories de véhicules (sauf pour la catégorie 4)

II.4 - TARIFICATION

Dans le cas où doivent être effectuées plusieurs opérations de réduction ou de majoration, le calcul de la prime sera obtenu par opérations successives et non par addition des taux de majoration ou de réduction [23].

II.4.1 - TARIF 1 : PROMENADE ET AFFAIRE

- Véhicules privés utilisés tant pour l'exercice de la profession que pour la promenade.
- Véhicules à 4 roues
- Véhicules à carrosserie « tourisme » jusqu'à 10 places : CI – Berline – Break - Familial – Limousine – Voiture privée – Station wagon.
- Véhicules à 4 roues d'un poids inférieur ou égal à 150 kg dont le nombre de places n'excède pas deux et dont la conduite ne nécessite pas la possession d'un permis de conduire.

II.4.2 - TARIF 2 : TRANSPORT DE PERSONNES

- Véhicules à carrosseries autres que celles prévues au Tarif 1 et dont l'usage est destiné au transport de personnes à titre gratuit.
- Véhicules de toutes carrosseries confondues et dont l'usage est destiné au transport de personnes à titre payant.

II.4.3 - TARIF 3 : TRANSPORT DE MARCHANDISES

Véhicules de toutes carrosseries confondues et dont l'usage est destiné au transport de marchandises à titre gratuit et payant.

II.4.4 - TARIF 4 : VEHICULES CONFIES AUX GARAGISTES

- Véhicules de toutes carrosseries confondues confiés aux garagistes, concessionnaires et/ou vendeurs de véhicules circulant sous « W » :
- Véhicules à 4 roues.
- Véhicules motorisés à 2 ou 3 roues
- Ou véhicules à 4 roues dont le poids à vide est inférieur ou égal à 150 kg.

La prime par police sera fixée en fonction du nombre de cartes roses délivrées au Souscripteur.

II.4.5 - TARIF 5 : ENGINS DE CHANTIER

Engin de chantier dont la conduite ne nécessite pas la possession d'un permis de conduire : élévateur – buldozer – répandeuse – nivelleuse – compacteur – scraper – autoscraper – chargeuse – grue – grue mobile – bétonneuse – compresseur – pelle mécanique – excavatrice – rouleau compresseur – etc

II.4.6 - TARIF 6 : VEHICULES DE TYPES SPECIAUX

Véhicules de types spéciaux ne rentrant dans aucune des catégories 1 à 5 :

- Ambulances – corbillards et fourgons funéraires
- Véhicules circulant sur aérodromes – tracteurs – dépannage
- Véhicule de collectivité publiques : camions ou bennes utilisés pour ordures, véhicules d'incendie et de secours, goudronneuse, voiture de vidange.

II.5 - J2EE JAVA 2, ENTERPRISE EDITION

II.5.1 - MODELE D'APPLICATION J2EE

La spécification J2EE [1] n'oblige pas le modèle d'application à 2 ou 3 ou multitiers [1]. L'essentiel est qu'il est important d'utiliser des outils appropriés pour un problème donné.

II.5.1.1 - MODELE AVEC CLIENT DE TYPE APPLICATION

Suivant un modèle de programmation J2EE, on a deux types de clients :

- Client application en interaction directe avec un serveur java EJB [1].

Le serveur EJB est hébergé dans un EJB container. Le protocole utilisé peut-être RMI-IIOP [1] et le serveur EJB accède aux bases de données.

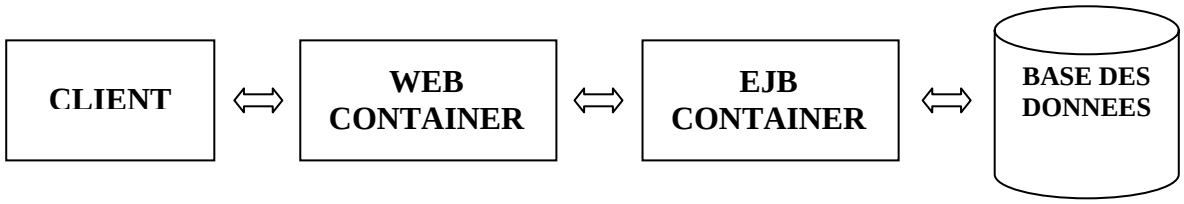
- Client application java accédant directement les bases de données.

Dans ce modèle, la présentation et la logique d'application sont localisées chez le client et regroupées en une seule application. Ce modèle est essentiellement connu sous le nom de client-serveur.

II.5.1.2 - MODELE D'APPLICATION ORIENTEE WEB

Ce modèle d'application multitiers orientée web est très repandu. La présentation et la logique d'application sont essentiellement hébergées dans le web container. On peut supposer ici que l'accès aux bases de données se fait avec ODBC [1, 9 , 10 11].

II.5.1.3 - MODELE D'APPLICATION MULTITIERS



Le web container héberge exclusivement la logique de présentation. Les pages JSP assurent les contenus dynamiques des pages web destinés aux clients.

Le EJB container héberge la logique d'application qui d'une part, répond aux requêtes venant du web tier et d'autre part, accède aux bases de données.

L'isolation du front-end et du back-end définit la puissance de ce modèle [1].

II.5.2 - APPLICATIONS DISTRIBUEES MULTITIERS

La plate-forme J2EE utilise un modèle d'application distribuée. La logique d'application est divisée en composants selon la fonction, et plusieurs composants de l'application qui constituent l'application J2EE sont installés sur différentes machines dépendant du tiers de l'environnement

multitiers J2EE auxquels le composants appartiennent. La figure 1.1 montrent deux(02) applications multitiers J2EE séparées en tiers décrits sur la liste suivante [1]:

- Les composants Client-tier tournant sur les machines clients ;
- Les composants Web-tier tournant sur le serveur J2EE ;
- Les composants Business-tier tournant sur le serveur J2ee ;
- Les composants EIS-tier (Enterprise Information System) tournant sur le serveur EIS.

Bien que l'application J2EE peut consister en trois ou quatre tiers, les applications multitiers J2EE sont considérées comme étant trois tiers parce qu'elles sont distribuées sur trois différentes machines : les machines clients, les serveurs J2EE, et les bases de données ou les « legacy » machines à l'arrière. Les applications trois tiers possédant cette architecture étendent le modèle deux tiers standard client serveur en plaçant de(s) serveur(s) d'application « multithread » entre le client et le stockage situé à l'arrière.

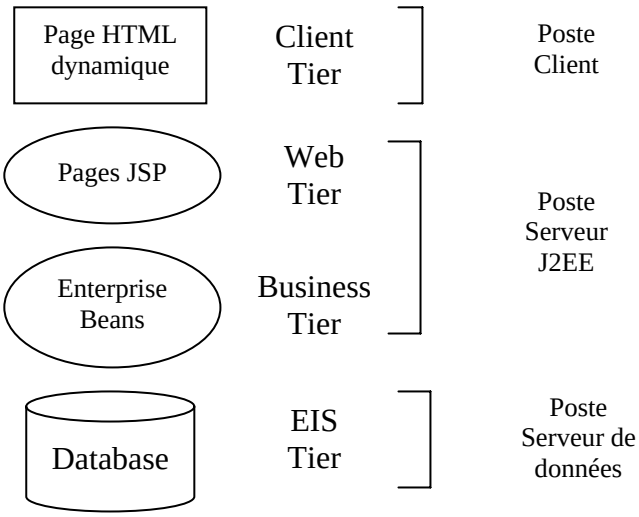


Figure 1.1 Application multitiers

Les applications J2EE sont composées des composants. Un composant J2EE est une unité du logiciel utilitaire indépendante communicant avec d'autres composants. Les spécifications J2EE définissent les composants suivants :

- Les applications clients et les applets sont des composants tournant sur les machines clients.
- Les composants à base de la technologie de Java servlet et JSP sont des composants Web tournant sur les serveurs.
- Les composants EJB sont des composants métiers tournant sur le serveur

II.6 - SERVEUR D'APPLICATION BES 5.1

II.6.1 - CARACTERISTIQUES

Le BES 5.1 permet de [2]:

- Développer et déployer des applications web utilisant JavaServer Pages et Servlets
- Déployer des applications exigeant CORBA pour communiquer avec les objets non java

- Construire une application distribuée java implémentant la plateforme J2EE 1.3

Pour remplir ses fonctions, BES 5.1 inclut [2]:
- Le serveur web Apache version 2.035
- Le web container Tomcat version 4.0.3
- Le Java Session Service (JSS) pour stocker les informations de session pour récupération au cas où le conteneur tombe en panne
- Le IIOP Connector permettant au serveur web Apache et au web container Tomcat de communiquer via le protocole IIOP
- Le ORB visibroker, le principal ORB industriel
- Le naming service, qui est une implémentation de JNDI
- Le plug-in IIOP pour CORBA qui prend en charge les requêtes Apache au IDL et CORBA via IIOP et les transportent comme une requête http. Ainsi, les objets java, C++, CORBA peuvent être utilisés avec les requêtes http
- Le EJB container fournit tous les services nécessaires pour l'application pour être complètement portable et compatible avec J2EE 1.3
- Le sonic MQ, le JMS utilisé par le BES
- Le Borland Security Service qui est la sécurité du BES

II.6.2 - EXIGENCE EN RESSOURCES

Pour installer le BES 5.1 sur les plateformes Windows, Borland recommande les exigences suivantes [2]:
- Windows NT 4.0, service pack 6, Workstation et serveur
- Windows 2000, sevice pack 2, professional et serveur
- Pentium III 600MHZ
- RAM 128 MO minimum, 160 MO ou plus recommandé.
- Espace libre disque dur 500 MO
- JavaSoft JDK 1.3.1
- Le JDK version 1.3.1 est inclus avec l'installeur de BES 5.1. il est vivement recommandé d'utiliser le JDK inclus dans l'installeur.

II.6.3 - LES INTERFACES DE GESTION DU BES 5.1

Les interfaces de gestion les plus utilisées sont [2]:
- Borland Management Console
- Deployment Descriptor Editor

II.6.4 - BORLAND MANAGEMENT CONSOLE

BES fournit une interface utilisateur, le Borland Management Console, qui a pour fonction de contrôle centralisé du serveur [2].

Le console permet de :
- Voir les serveurs dans le réseau
- Démarrer et arrêter les serveurs et services
- Changer les configurations d'un serveur
- Gérer les services et outils permettant de construire, déployer et gérer des applications d'entreprise basées sur des composants
- Voir les modules déployés
- Fixer les propriétés de déploiement
- Contrôler la performance

II.6.5 - DEPLOYMENT DESCRIPTOR EDITOR

Le Borland Management Console inclut un DDEditor (Deployment Descriptor Editor) qui peut être utilisé pour modifier et ajouter les informations de déploiement d'un fichier d'un archive J2EE [1, 2].

Les objets pouvant être édités sont les descripteurs (créant XML standard et XML propriétaire Borland) :
- Application.xml pour EARs
- Web.xml pour WARs
- Ejb-jar.xml pour EJB 2.0 JAR

La liste des archives comprend :
- EARs pour les applications J2EE 1.3
- WARs pour les servlets
- JARs pour les EJB 2.0

Le Deployment Descriptor Editor peut être utilisé sur une large variété d'archives et de fichiers xml.

• CHAPITRE III - CAHIER DE CHARGES

On élabore le cahier de charges relatif aux besoins du maître d'ouvrage.

Quelle que soit la taille du projet, l'élaboration de ce documents de bases est indispensable pour faciliter le contrôle de réalisation.

III.1 - OBJECTIFS

L'objectif du projet est d'optimiser les applications, les données et les ressources dans le domaine de la branche assurance automobile en mettant en évidence les points suivants :

- Centraliser les données et les applications
- Permettre aux agences d'utiliser directement les applications et les données centralisées
- Mettre en place une application robuste, évolutive et performante
- Réduire les taux de ressaisie et de contrôle supplémentaire à la réception des données au siège
- Réduire les domaines d'intervention et de maintenance à distance
- Utiliser la technologie J2EE à cette fin

III.2 - MAITRE D'OEUVRE

Le maître d'œuvre et le maître d'ouvrage doivent séparer les différentes étapes de réalisation ainsi que le système de contrôle et les contraintes avant, pendant et après la réalisation du projet.

III.3 - PRESENTATION DU PROJET

Cette partie relate le cadre logique donnant explicitement toutes les finalités de l'étude. On y identifie les auteurs concernés dans les départements et les composants respectifs.

L'interface utilisateur final doit répondre à la nature de la demande du point de vue consultation, mise à jour et calcul.

III.4 - DISPOSITIFS MIS EN OEUVRE

Cette étape met en évidence les procédés, les ressources matérielles, humaines pour la réalisation du projet. Les différentes méthodologies utilisées sur les différentes étapes : conception, réalisation, y sont également dressées sans oublier le planning de l'étude, les réalisations journalières, date de début et de fin du projet pour le bon déroulement du projet.

III.5 - ANALYSE DE L'EXISTANT

L'analyse de l'existant est demandée non seulement pour le démarrage mais aussi pour pouvoir proposer des solutions permettant d'atteindre l'objectif final du projet.

III.6 - OBJECTIFS DU NOUVEAU SYSTEME

Le nouveau système doit faire apparaître :

- la critique de l'existant
- les objectifs et les principaux résultats proposés et attendus.
- Les différentes options possibles
- Le cadre organisationnel des solutions à rechercher selon l'option retenue.

III.7 - PRESENTATION DES SOLUTIONS

On demande dans cette étape :

- le cadre organisationnel proposé,
- La description fonctionnelle du nouveau système de telle façon qu'un utilisateur quelconque comprenne facilement l'exploitation du nouveau système,
- L'optimisation du choix technique en terme de matériels de communication, et support logiciel attendus
- L'établissement d'un bilan économique en terme financier, en affectation des ressources et d'échéances.

III.8 - ENJEUX ET RISQUES LIES AU PROJET

Les enjeux de ce projet sont nombreux :

- L'utilisation des technologies modernes fait augmenter la confiance de la clientèle
- La technologie permet une ouverture au client pour ne plus avoir à se déplacer
- Les investissements ultérieurs relatifs à l 'extension restent très faibles
- L'utilisation des produits est pratiquement gratuite
- L'augmentation des performances des informaticiens (plus de télémaintenance)
- La mobilité des utilisateurs (commerciaux) est assurée
- La mobilité des clients est garantie

Toutefois, les risques liés au projet ne demeurent pas pour autant négligeables :

- Les aléas naturels pouvant causer des dégâts sur les serveurs ou les moyens de transmission
- Les grèves entourant l'exploitation et la transmission des données.
- Les vols organisés des ressources (informatiques ou divulgation des secrets)

A voir de très près, ces risques sont purement des cas extrêmes dont la probabilité reste très faible. En effet, il est inconcevable que les sociétés prestataires de services restent les mains croisées devant la situation. Par ailleurs, la situation, si ce cas se présente, est parfois générale, et donc entraîne une paralysie commune.

III.9 - MOYENS NECESSAIRES

Les moyens nécessaires sont :

- Borland Jbuilder 7.0
- BES 5.1
- Microsoft SQL Server 2000
- MS Project 2000
- Microsoft Windows XP
- Ordinateur Pentium IV 1.7 GHz, RAM 256 MO minimum
- Internet Explorer 5.0 minimum

III.10 - ORGANISATION ENVISAGEE

Dans le projet doit figurer l'organisation suivante

- Nomination du Chef de projet
- Constitution du comité de pilotage
- Constitution des comité des utilisateurs
- Constitution de l'équipe de projet
- Etablissement du planning du projet
- Etablissement de la charte de projet
- Tenue d'un tableau de bord

- Tenue de réunions des divers comités
- Etablissement d’un compte rendu à chaque réunion

III.11 - CHARTE PROJET

La charte de projet est un document qui précise l'engagement du service vis à vis des utilisateurs en terme de cible fonctionnelle, qualité de service, lotissement, budget... Elle constitue un contrat entre le projet et ses commanditaires (services utilisateurs, direction, ...).

La charte de projet est un document primordial pour le lancement du projet car toute ambiguïté à ce niveau pourrait avoir des conséquences très lourdes pour la suite des opérations (développement en double, problèmes de communication entre systèmes d'information, mauvaise couverture fonctionnelle de la cible, dépassement des charges et délais, difficultés de prise de décision et de validation...).

La charte de projet n'est en général pas mise à jour en cours de projet.

Les principales informations que l'on y trouve sont l'organigramme du projet et pour chaque membre du projet :

- le nom de la personne concernée, le projet et le nom du responsable
- les missions générales attribuées à la personne
- la durée (dates de début et fin) et la charge (en % ou m.h)
- les visas des responsables hiérarchiques concernés

III.12 - IDENTIFICATION DU MAITRE D’OUVRAGE ET DE MAITRE D’OEUVRE

III.12.1 - LE MAITRE D’OUVRAGE

L’assurance ARO, Antsahavola en tant que demandeur du projet est le patron du projet

III.12.2 - LE MAITRE D’OEUVRE

Le maître d’œuvre : Rivo Randrianavalotiana, Personnel du département informatique ARO, les enseignants encadreurs du DESS TNSI.

- # CHAPITRE IV - PLANIFICATION DU PROJET

C’est un document établi en début du projet qui est mis à jour à chaque fin de période.
17 février 2003 – 14 mars 2003

- Test de fonctionnalité du serveur d’application BES 5.1

17 mars 2003 – 11 avril 2003

- Conception d’une interface de mise à jour d’une base de données
- Conception d’une application de mise à jour d’une base de données

14 avril 2003 – 23 mai 2003

- Conception d’une module de calcule de prime d’assurance automobile
- Conception d’interface d’accès au module de calcul

26 mai 2003 – 06 juin 2003

- Mise en place d’une application de mise à jour d’une base de données assurance auto
- Couplage de l’application de mise à jour avec celle du module de calcul de prime
- Conception des interfaces utilisatrices attrayantes

PARTIE II - PHASE D'EXECUTION DU PROJET

Cette phase a pour objectifs de préparer et exécuter les différentes étapes du cycle de développement du système. L'identification des étapes est primordiale afin de maîtriser la qualité, les coûts et les délais du projet.

Dans cette phase, on aborde successivement :

- Préparation
- Réalisation
- Utilisation de la méthode UML pour la conception
- Base des données
- Plan du site

• CHAPITRE I - PREPARATION

Pour atteindre l'objectif fixé, cette phase nécessite plusieurs réunions périodiques.

Il y a :

- Les réunions
  - Les réunions périodiques fixes de l'équipe ;
  - Les réunions de suivi du projet ;
  - Les réunions du comité ;
- Les livrables suivants:
  - Le journal de bord ;
  - Le compte-rendu de réunion ;
  - Le plan d'assurance et contrôle qualité
- Installation de l'environnement méthodologique et technologique

I.1 - LES REUNIONS

I.1.1 - LES REUNIONS DE L'EQUIPE

Mettre en œuvre des réunions périodiques fixes avec l'équipe du projet pour favoriser la communication / information entre tous les membres, faire le point d'avancement

I.1.2 - PREREQUIS

Le Journal de bord du projet et le planning remis à jour doivent être apportés à la réunion.

Il est conseillé de tenir les réunions si possible le même jour de la semaine à une heure et un lieu fixe.

I.1.3 - REUNION DE SUIVI DU PROJET

Mettre en oeuvre des réunions périodiques de suivi de projet au niveau coordination des projets du service, afin de s'assurer du bon déroulement des projets et prendre les décisions relevant de la direction du service.

I.1.4 - PREREQUIS

Le chef de projet doit apporter à la réunion :

- le planning à jour
- le journal de bord
- le tableau de suivi des risques
- un récapitulatif sur l'état du budget
- les fiches d'affectation des ressources

I.1.5 - LES DIFFERENTS POINTS A VOIR A LA REUNION

- Examiner l'avancement du projet (suivi planning, consommation du budget)
- Examiner l'affectation des ressources
- Passer en revue les différents risques identifiés par le chef de projet
- Prendre les décisions nécessaires
- Rédiger un compte-rendu de réunion

Remarques :

- La périodicité de ces réunions est généralement mensuelle, mais modulable en fonction des besoins des projets.

- D'autres intervenants peuvent participer à la réunion (membres de l'équipe projet, direction, assurance qualité, ...) sur demande du chef de projet ou de la coordination des projets.

I.1.6 - REUNIONS DES COMITES

Mettre en oeuvre des réunions périodiques fixes avec les comités de projet (pilotage et des utilisateurs) pour les informer de l'avancement et proposer / valider les orientations du projet

Cette réunion est dirigée par le Chef de Projet.

Organisation :
- préparer un ordre du jour et convoquer les participants à la réunion au moins 02 jours avant celle-ci
- tenir la réunion en suivant l'ordre du jour prévu
- rédiger un compte-rendu de réunion.

La périodicité varie en fonction des besoins du projet, mais il est suggéré une fois par mois.

Il est souhaitable que toute réunion du comité de pilotage (où des décisions seront prises quant aux propositions formulées), soit précédée d'une réunion du comité des utilisateurs (au cours de laquelle des besoins seront exprimés)

Les comptes-rendus de réunion doivent être diffusés systématiquement aux deux comités (pilotage et des utilisateurs) afin que l'information circule correctement de l'un à l'autre.

I.2 - LIVRABLES

I.2.1 - JOURNAL DE BORD

Le journal de bord permet de noter les actions à mener dans le cadre du projet (indépendamment des tâches attribuées à chacun), le responsable de l'action, ainsi que la date prévue de réalisation. Lorsque l'action est réalisée, la date réelle de réalisation est renseignée.

Le journal de bord est initialisé en début de projet et complété par le chef de projet au fur et à mesure des réunions :
- les actions devant être réalisées à la date de réunion sont passées en revue, leur date de réalisation est renseignée
- les nouvelles actions à effectuer sont écrites à la date de la réunion, en précisant qui en est responsable et leur date de réalisation prévisionnelle.

Un journal de bord se structure comme suit :
- numéro chronologique et date d'identification
- énoncé du problème ou de l'information
- énoncé de l'action ou de la décision prise
- responsable de l'action (initiales)
- date de réalisation prévue puis réelle

Ce document est saisi sous format électronique et est conservé dans le dossier de suivi du projet.

I.2.2 - COMPTE RENDU DE REUNION

Un compte-rendu de réunion est rédigé après une réunion, pour garder une trace des décisions et discussions ayant eu lieu pendant la réunion.

- Le compte-rendu de réunion doit être diffusé au minimum à l'ensemble des participants. Il peut faire l'objet d'une révision sur demande de l'un des participants.
- Le compte-rendu de réunion d'un comité de projet doit être approuvé par le président du comité avant d'être diffusé à l'ensemble des membres du comité.
- Le compte-rendu doit avoir le plan suivant :
  - Le compte-rendu doit reprendre les points de l'ordre du jour dans le même ordre.
  - Il est conseillé que cet ordre du jour soit si possible le même d'une réunion à l'autre.
  - Pour une réunion de l'équipe projet , on y retrouve : point d'information du chef de projet, point d'avancement, revue des actions réalisées et à faire du journal de bord, problèmes, questions diverses.
  - Pour une réunion de suivi du projet , on y trouve : point d'avancement (planning, budget), revue de l'affectation des ressources, examen des risques divers.
  - Pour une réunion des comités, on y trouve : point d'avancement, proposition et validation des orientations du projet, le compte-rendu de réunion des comités suit l'ordre du jour défini préalablement.

Remarques :

Les comptes-rendus de réunion interne à l'équipe projet peuvent être manuscrits. Ceux des comités de projet doivent être saisis sous format électronique. Ils sont tous conservés dans le dossier de suivi de projet.

I.2.3 - PLAN D'ASSURANCE ET CONTROLE QUALITE - PACQ

Le plan d'assurance et contrôle qualité est un document qui précise les éléments permettant de s'assurer de la mise en oeuvre et de l'efficacité des activités prévues pour obtenir la qualité requise [AFNOR/Z67 - 100-3]

Le PACQ est initialisé en début de projet au cours de la phase de lancement. C'est un document indispensable à tout projet.

Il peut être remis à jour à la fin d'une étape afin de préciser les règles applicables à l'étape suivante.

Les principales informations que l'on trouve dans le PACQ sont :

- objectifs qualité du projet
- environnement méthodologique (conduite de projet, démarche de développement, outils, normes, règles et standards)
- rôles et responsabilités
- modalités de gestion de la documentation
- modalités de gestion de configuration
- modalités de gestion des modifications
- contrôle des fournisseurs
- suivi et évaluation de la qualité

I.3 - INSTALLATION DE L'ENVIRONNEMENT METHODOLOGIQUE ET TECHNOLOGIQUE

Ceci concerne :

- La mise en œuvre des outils ou documents nécessaires au niveau méthodologique tels qu'ils ont été prévus en phase de préparation. Ex : préparer les fichiers contenant les plans types des documents techniques, installer les outils MS-PROJECT et autres.
- La mise en œuvre des environnements de développement, de test, de gestion de configuration, spécifiques au projet
- La mise à la disposition de la documentation d'utilisation des outils

I.3.1 - PREREQUIS

Plan d'Assurance Qualité (PAQ) mis à jour, complet et approuvé pour l'étape à venir

I.3.2 - ORGANISATION

- Préparer les formulaires et les plans types des documents à produire au cours de l'étape
- Les mettre à disposition des équipes (sur INTRANET)
- Installer et tester les outils supports de la méthodologie choisie
- Former les équipes à la méthodologie et aux outils
- Installer les logiciels de base nécessaires au développement
- Installer et tester les outils de développement choisis
- Former les équipes à l'environnement de développement

I.3.3 - ARCHITECTURE

L'architecture du système s'inscrit dans un schéma multitiers composé des éléments suivants :

- Serveur de base de données SQL server sous WinXP,
- Serveur web Apache-tomcat sousWindows,
- Serveur d'application BES 5.1
- Clients Windows sur lequel devra être installé Internet Explorer

• CHAPITRE II - REALISATION

II.1 - TEST DE FONCTIONNALITE DU SERVEUR D'APPLICATION BES 5.1

17 février 2003 – 14 mars 2003

Le serveur d'application doit être testé sur toutes ces fonctionnalités :

- Fonctionnalité du serveur web Apache-Tomcat.
- Fonctionnalité des conteneurs pour les EJB.
- Fonctionnalité de la liaison serveur d'application et base de données.

Ce test est nécessaire avant de démarrer le projet pour avoir l'habitude des interfaces et les fonctions du logiciel. C'est un point très important puisqu'il dicte la vitesse d'exécution du projet.

II.2 - CONCEPTION DES INTERFACES ET DES APPLICATIONS DE MISE A JOUR D'UNE BASE DE DONNEES

17 mars 2003 – 11 avril 2003

II.2.1 - IMPLANTATION DE LA BASE DE DONNEES SUR SQL SERVER

Pour pouvoir tester l'accès à la base de données, pour pouvoir travailler sur la partie sécurité de la base, nous avons dû faire la mise en place des bases de données.

En Annexe, nous montrons la structure des bases de données et les explications relatives à chaque base.

II.2.2 - CREATION MANUELLE DES INFORMATIONS SUR LES CLIENTS VERS LA BASE SQL SERVER

Il nous faut passer par cet étape pour pouvoir faire la simulation du logiciel. Nous n'avons pas des données réelles pour tester le fonctionnement du logiciel.

II.2.3 - REALISATION D'UNE INTERFACE HOMME MACHINE CONSULTABLE PAR INTERNET

C'est la partie visible par l'utilisateur. c'est le masque de saisie et l'écriture des différents scripts pour pouvoir manipuler les données.

II.3 - CONCEPTION DU MODULE DE CALCUL

14 avril 2003 – 23 mai 2003

II.3.1 - REALISATION DU MODULE DE CALCUL

Le module de calcul assure la fonction calcul seulement. Tous les algorithmes de calcul y sont placés.

II.3.2 - CONCEPTION DES INTERFACE POUR MODULE DE CALCUL

C'est la partie visible par l'utilisateur. c'est le masque de saisie et l'écriture des différents scripts pour pouvoir manipuler les données.

II.4 - CONCEPTION FINALE

26 mai 2003 – 06 juin 2003

II.4.1 - MISE EN PLACE D'UNE APPLICATION DE MISE A JOUR D'UNE BASE DE DONNEES AUTOMOBILE

- Conception des applications de mise à jour des données clients (sélection, mise à jour, insertion, suppression).
- Cette conception concerne les tables client, automobile, et avenant

II.4.2 - COUPLAGE DE L'APPLICATION DE MISE A JOUR AVEC CELLE DU MODULE DE CALCUL DE PRIME

Le couplage du module de calcul avec les applications de mise à jour des données client nécessite une très grande attention puisqu'il peut y avoir des incompatibilités des données. Il faut que les deux applications arrivent à se voir et à travailler ensemble.

II.4.3 - EMBELLISSEMENT DES INTERFACES UTILISATEURS

C'est la partie finale du projet. Il consiste à embellir les interfaces utilisateurs pour les rendre attrayantes.

Les embellissements consiste à ajouter :

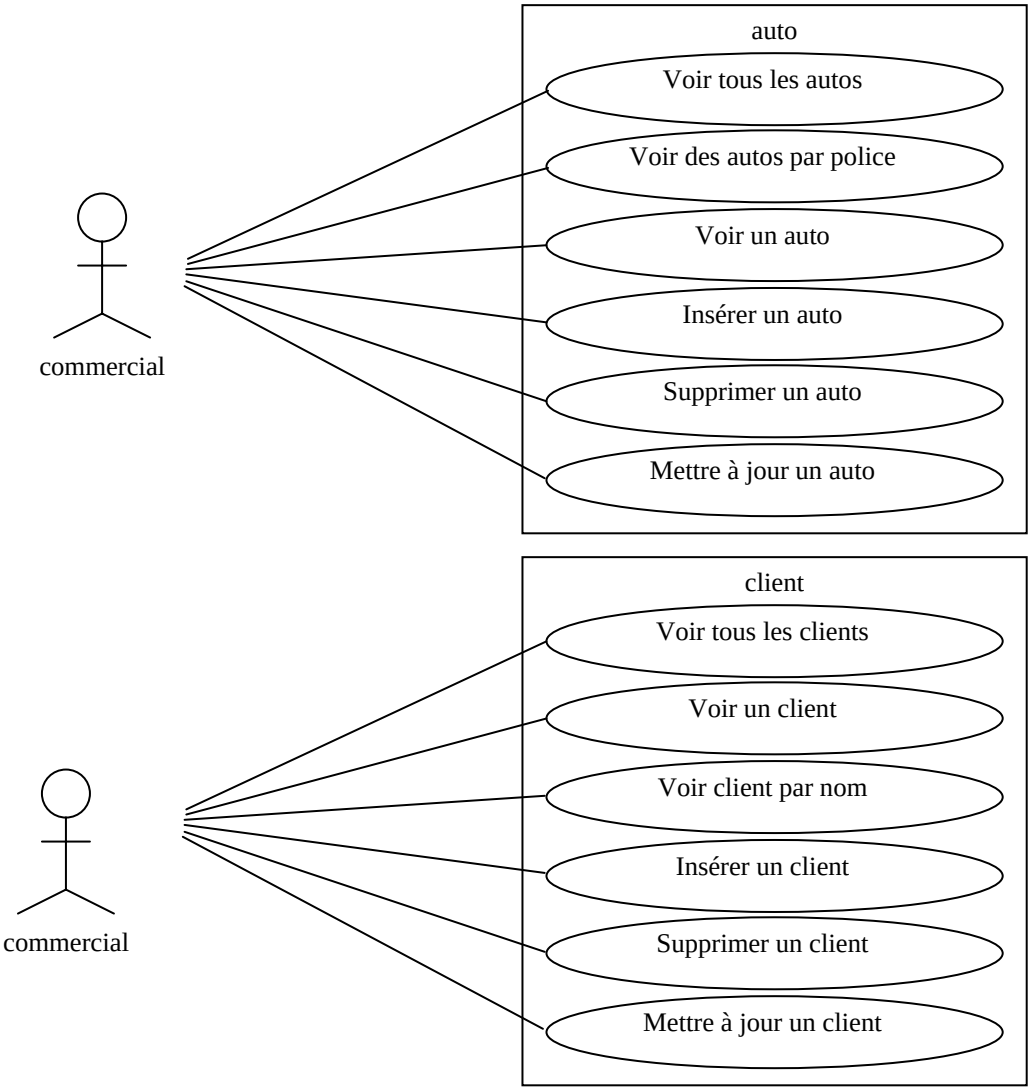
- Des scripts pour les changements de couleur au passage de la souris sur les boutons
- Des script pour un menu dynamique qui suit la page
- Des images de fonds

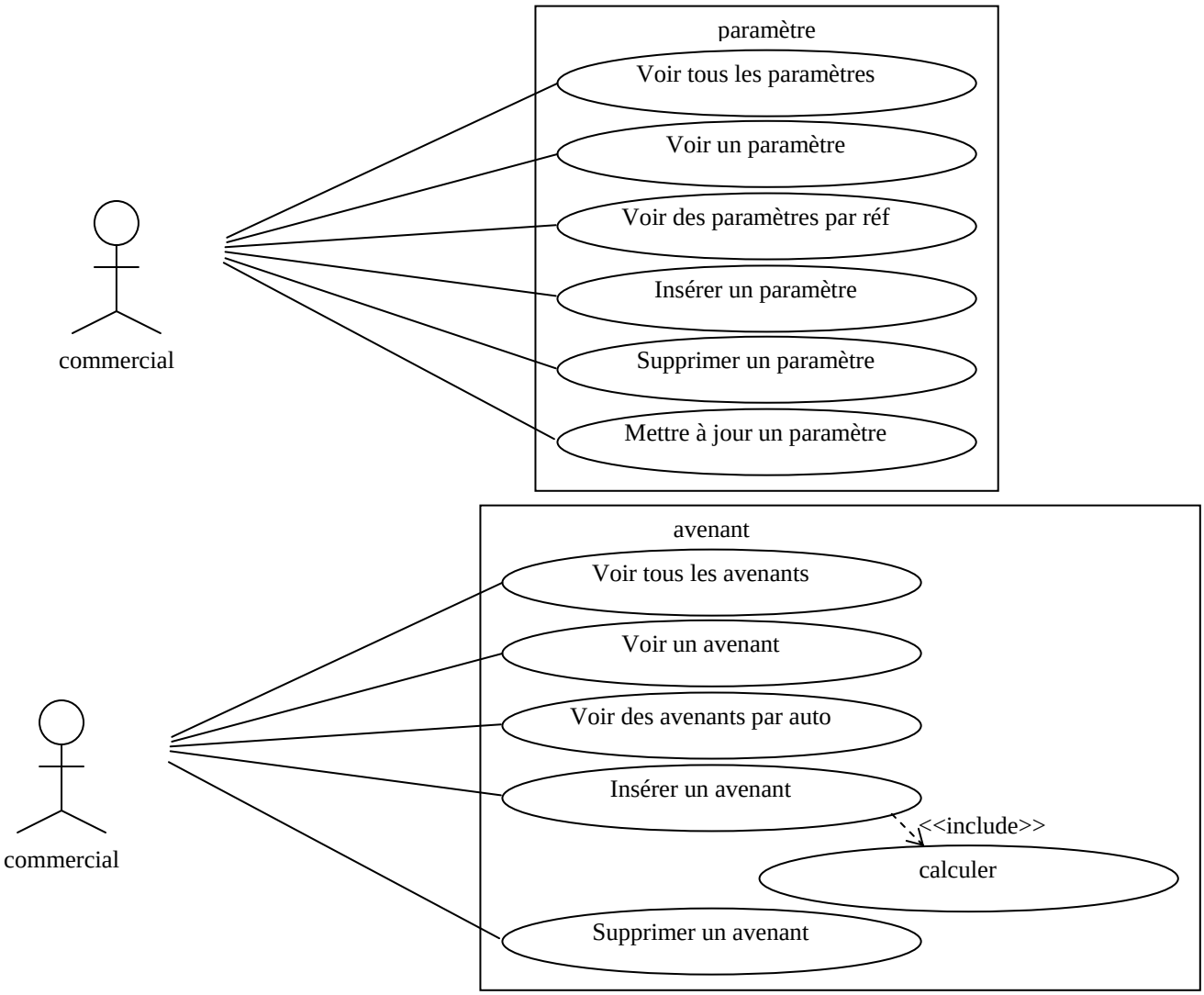
- **CHAPITRE III - UTILISATION DE LA METHODE UML POUR LA CONCEPTION**

On détaille ici [29, 30, 31, 32]:

- Les diagrammes d'utilisation du système en entier
- Auto
- Client
- Paramètres
- Avenants
- Cas d'utilisation calculer
- Diagramme de composants
- Modules
- Déploiement

III.1 - LES DIAGRAMMES D'UTILISATION DU SYSTEME EN ENTIER





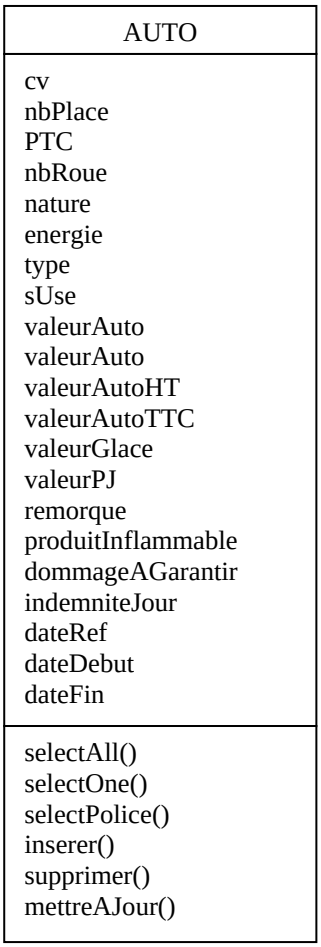
Les cas d'utilisation ci-dessus nous montrent l'acteur qui est le commercial qui effectue la sélection, l'insertion, la suppression, et mise à jour de l'auto, client, avenant et les paramètres.

Passons successivement en détail par regroupement auto, client, avenant et paramètre.

III.2 - CAS D'UTILISATION : AUTO

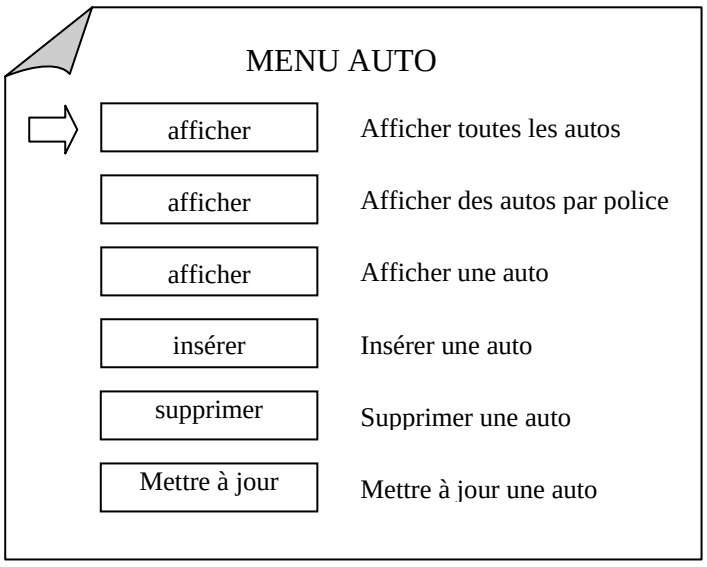
III.2.1 - DIAGRAMME DE CLASSE

Le diagramme de classe des cas d'utilisation du groupe auto reste le même et se différencie seulement des méthodes appelées détaillées ci-après.



- selectAll() pour voir toutes les autos.
- selectOne() pour voir une auto.
- selectPolice pour voir des autos par police.
- inserer() pour insérer une auto.
- supprimer() pour supprimer une auto.
- mettreAJour() pour mettre à jour une auto.

LA PAGE DE MENU : menuAuto.jsp



III.2.2 - CAS D'UTILISATION : VOIR TOUTES LES AUTOS

Le commercial peut voir en ligne la liste de toutes les autos stockés dans la base des données. Ceci détaille les caractéristiques de chaque auto.

III.2.2.1 - SCENARIO

- Le commercial accède au 'menu auto'
- Le commercial appuie sur le bouton 'afficher toutes les autos'
- Le système appelle la classe auto et en instancie un objet appelé auto correspondant
- Le système appelle la méthode selectAll de l'objet auto
- L'objet effectue l'extraction des données et les passe à la page autoall.jsp
- Le commercial obtient la liste des autos
- Le commercial navigue dans la liste
- Pour quitter, le commercial appuie sur le bouton 'retour menu'

III.2.2.2 - MAQUETTE

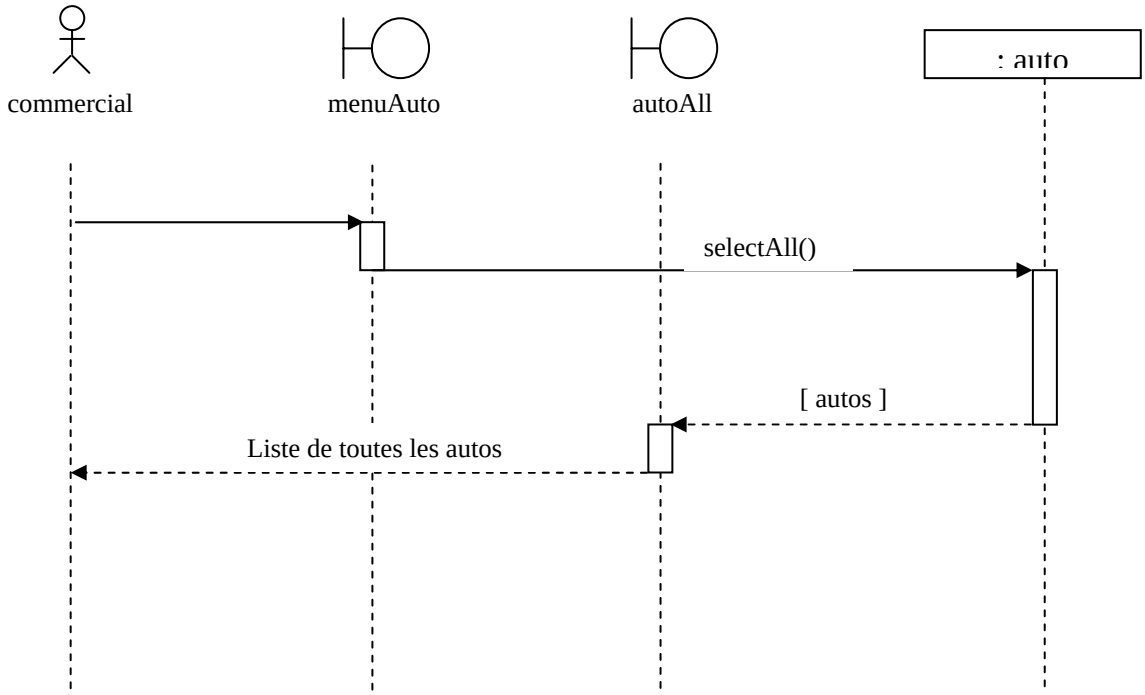
autoAll.jsp

LISTE DE TOUTES LES AUTOS

N° auto	N° client	N° police	Usage	CV	PTC	Nb place	Nb roue	Valeur	Valeur HT	Valeur glace	Energie
5286TR		EG1256	March	50	15	3	4	80000000	75000000	900000	Gas oil
2568TP		EG1257	March	50	15	3	4	80000000	75000000	900000	Gas oil
0244TL		EC5617	Plaisir	28	2	8	4	65000000	60000000	1000000	Essence
0249TL		EC5617	plaisir	28	2	8	4	65000000	60000000	1000000	Essence

Retour menu

III.2.2.3 - DIAGRAMME DE SEQUENCE



III.2.3 - CAS D'UTILISATION : VOIR DES AUTOS PAR POLICE

Le commercial peut voir la liste des autos par police.

III.2.3.1 - SCENARIO

- Le commercial accède au 'menu auto'
- Le commercial appuie sur le bouton 'afficher les autos par police'
- Le système appelle la page de saisie de critère de sélection police.
- Le système appelle la classe auto et en instancie un objet appelé auto correspondant
- Le système appelle la méthode selectPolice de l'objet auto
- L'objet effectue l'extraction des données et les passe à la page autoPolice.jsp
- Le commercial obtient la liste des autos pour la police spécifiée
- Le commercial navigue dans la liste
- Pour quitter, le commercial appuie sur le bouton 'retour menu'

III.2.3.3 - MAQUETTE

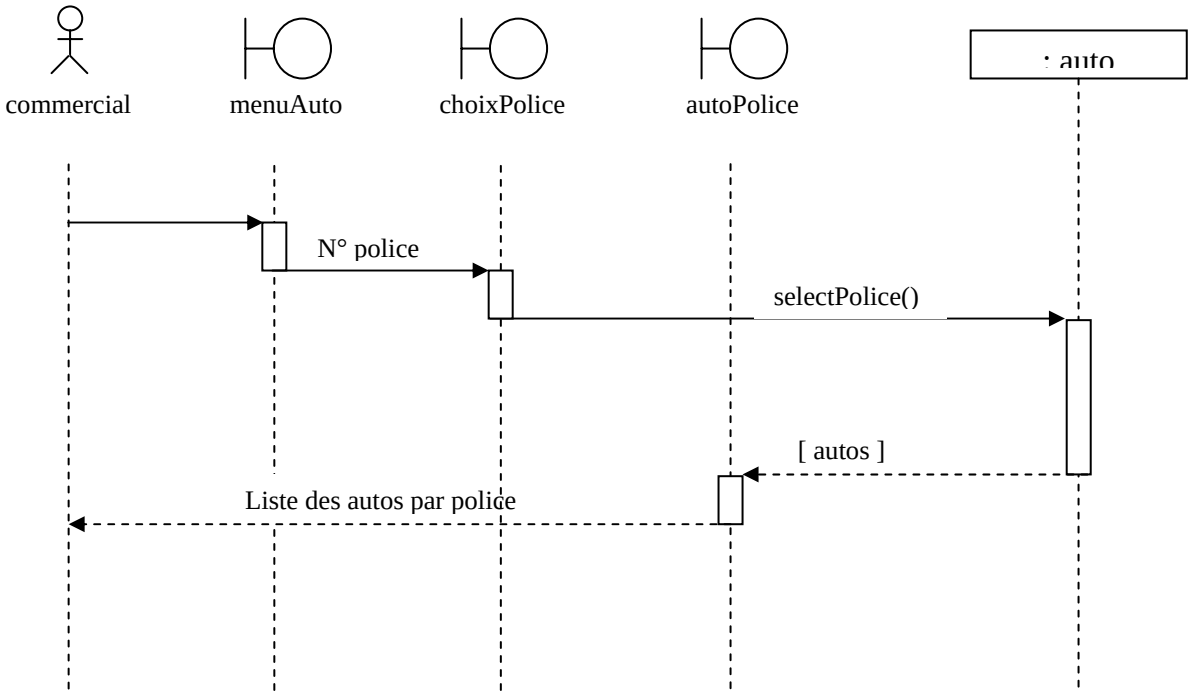
autoPolice.jsp

LISTE DES AUTOS POUR POLICE **EC5617**

N° auto	N° client	N° police	Usage	CV	PTC	Nb place	Nb roue	Valeur	Valeur HT	Valeur glace	Energie
0244TL		EC5617	Plaisir	28	2	8	4	65000000	60000000	1000000	Essence
0249TL		EC5617	plaisir	28	2	8	4	65000000	60000000	1000000	Essence

Retour menu

III.2.3.3 - DIAGRAMME DE SEQUENCE



III.2.4 - CAS D'UTILISATION : VOIR UN AUTO

Le commercial peut voir les informations sur une auto particulier.

III.2.4.1 - SCENARIO

- Le commercial accède au 'menu auto'
- Le commercial appuie sur le bouton 'afficher un auto'
- Le système appelle la page de saisie de critère de sélection auto.
- Le système appelle la classe auto et en instancie un objet appelé auto correspondant
- Le système appelle la méthode selectOne de l'objet auto
- L'objet effectue l'extraction des données et les passe à la page autoOne.jsp
- Le commercial obtient les informations de l 'auto spécifiée
- Pour quitter, le commercial appuie sur le bouton 'retour menu'

III.2.4.2 - MAQUETTE

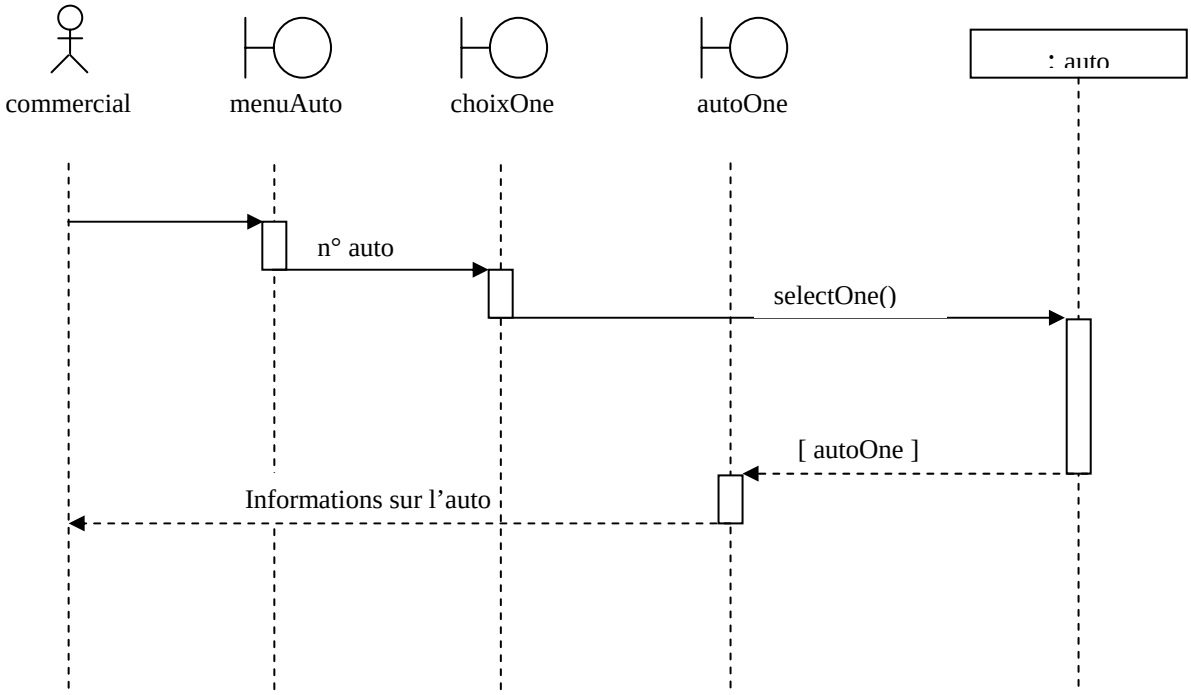
autoOne.jsp

INFORMATION SUR L' AUTO 0244TL

N° auto	N° client	N° police	Usage	CV	PTC	Nb place	Nb roue	Valeur	Valeur HT	Valeur glace	Energie
0244TL		EC5617	Plaisir	28	2	8	4	65000000	60000000	1000000	Essence
0249TL		EC5617	plaisir	28	2	8	4	65000000	60000000	1000000	Essence

Retour menu

III.2.4.3 - DIAGRAMME DE SEQUENCE



III.2.5 - CAS D'UTILISATION : INSERER UNE AUTO

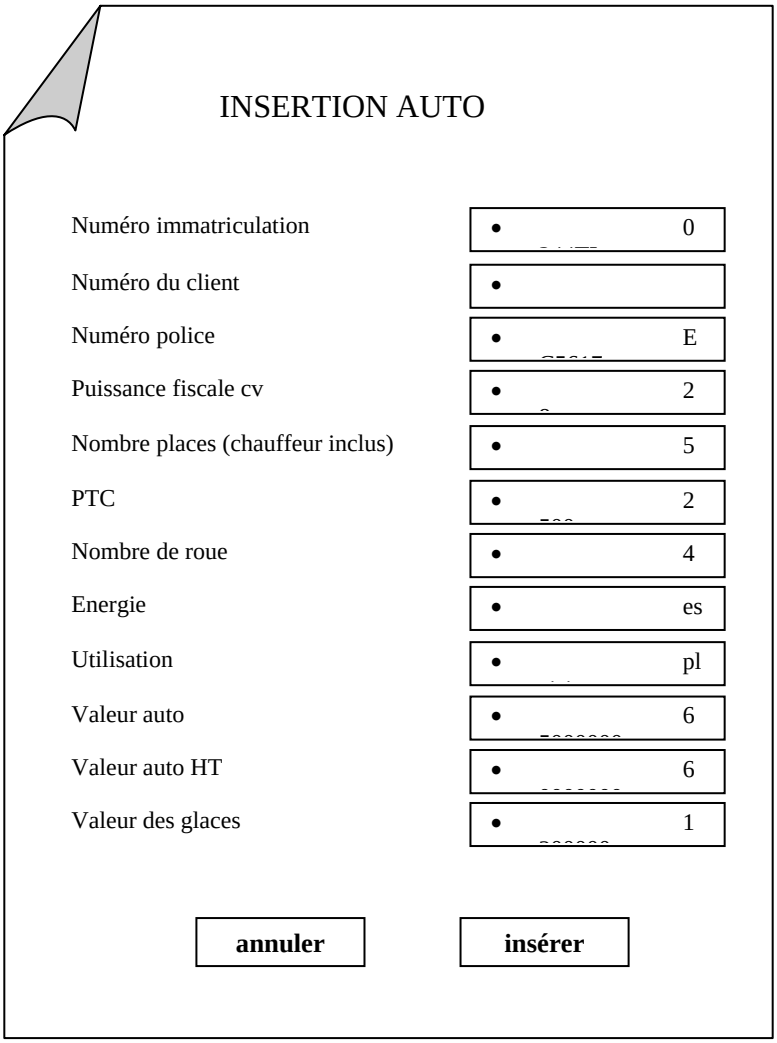
Le commercial peut insérer des informations sur une auto donnée.

III.2.5.1 - SCENARIO

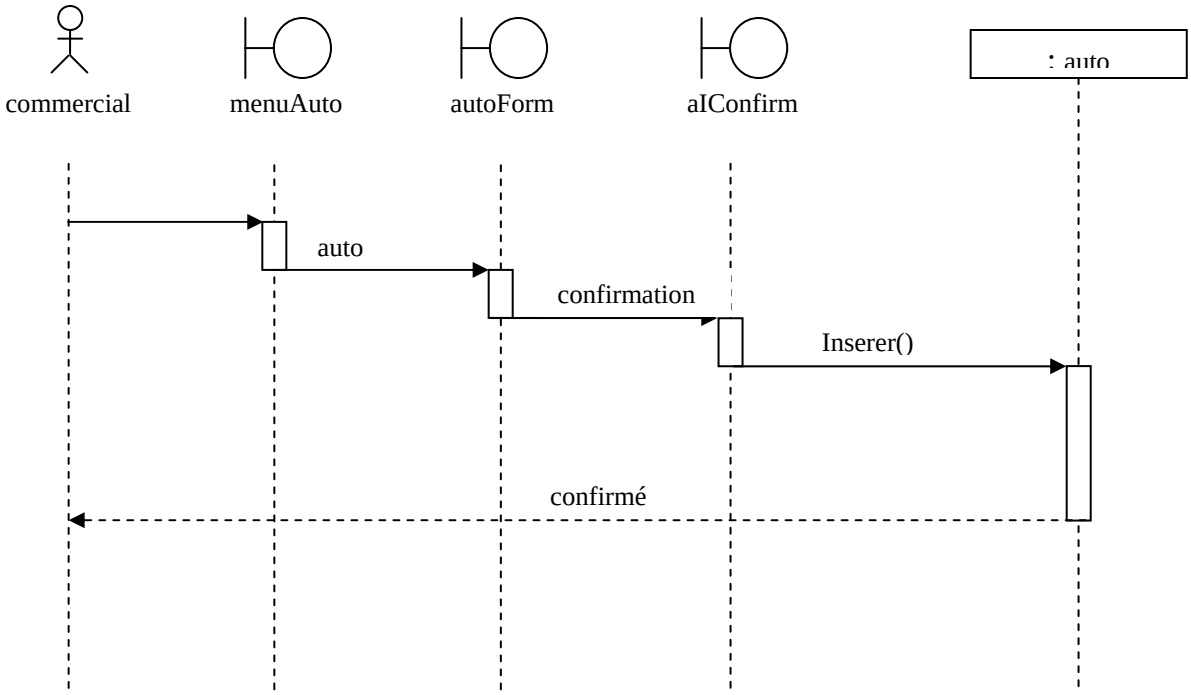
- Le commercial accède au 'menu auto'
- Le commercial appuie sur le bouton insérer une auto
- Le système appelle la page de saisie des informations sur l'auto à insérer.
- Le commercial remplit le formulaire d'insertion.
- Le commercial appuie sur le bouton insérer.
- Le système appelle la page de confirmation d'insertion.
- Le commercial appuie sur le bouton insérer.
- Le système appelle la classe auto et en instancie un objet appelé auto correspondant
- Le système appelle la méthode inserer() de l'objet auto.
- Le système appelle le 'menu auto' après insertion.
- Le commercial est ainsi informé de l'insertion.

III.2.5.2 - MAQUETTE

autoForm.jsp & aIConfirm.jsp



III.2.5.3 - DIAGRAMME DE SEQUENCE



III.2.6 - CAS D'UTILISATION : SUPPRIMER UNE AUTO

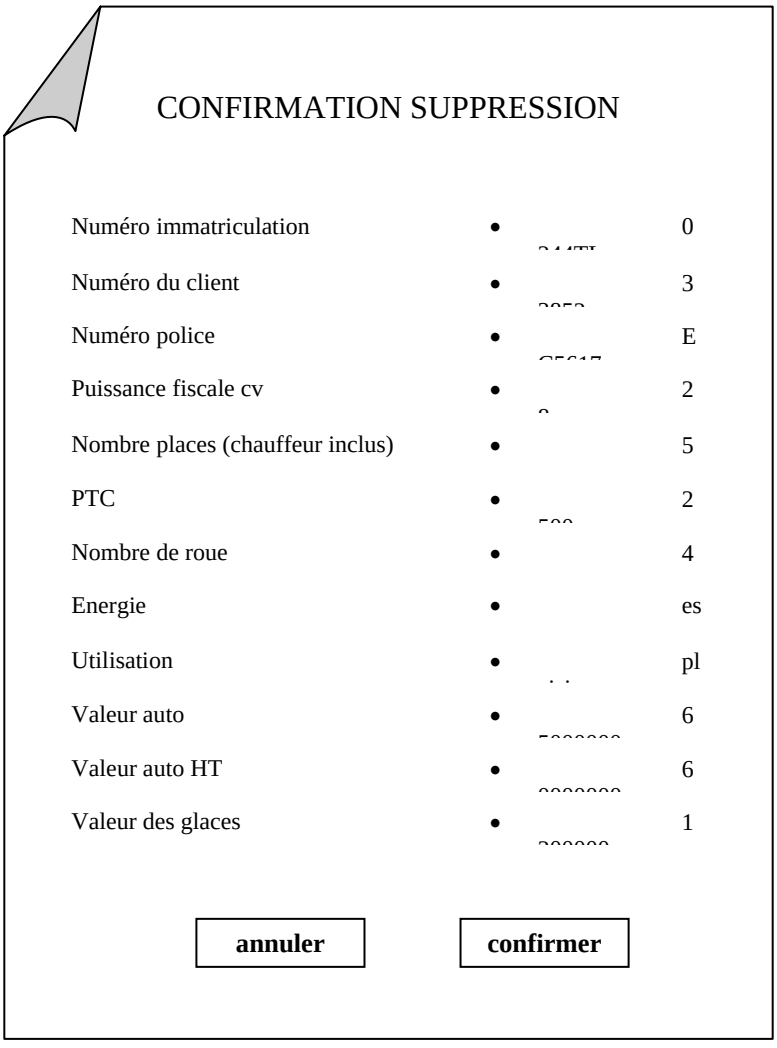
Le commercial peut supprimer un enregistrement sur une auto donnée.

III.2.6.1 - SCENARIO

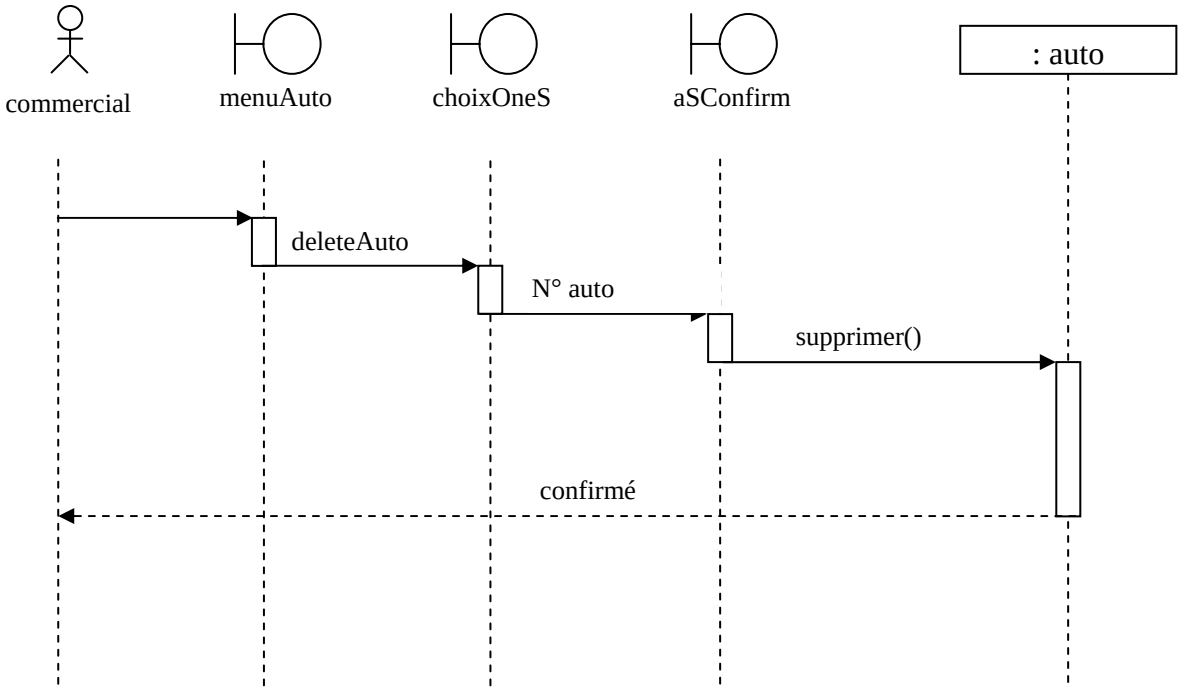
- Le commercial accède au 'menu auto'
- Le commercial appuie sur le bouton 'supprimer une auto'
- Le système appelle la page d'affichage de l'auto à supprimer.
- Le commercial appuie sur le bouton ''supprimer.
- Le système appelle la page de confirmation de suppression.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le menu auto après suppression.
- Le commercial est ainsi informé de l'insertion.

III.2.6.2 - MAQUETTE

aSConfirm.jsp



III.2.6.3 - DIAGRAMME DE SEQUENCE



III.2.7 - CAS D'UTILISATION : METTRE A JOUR UNE AUTO

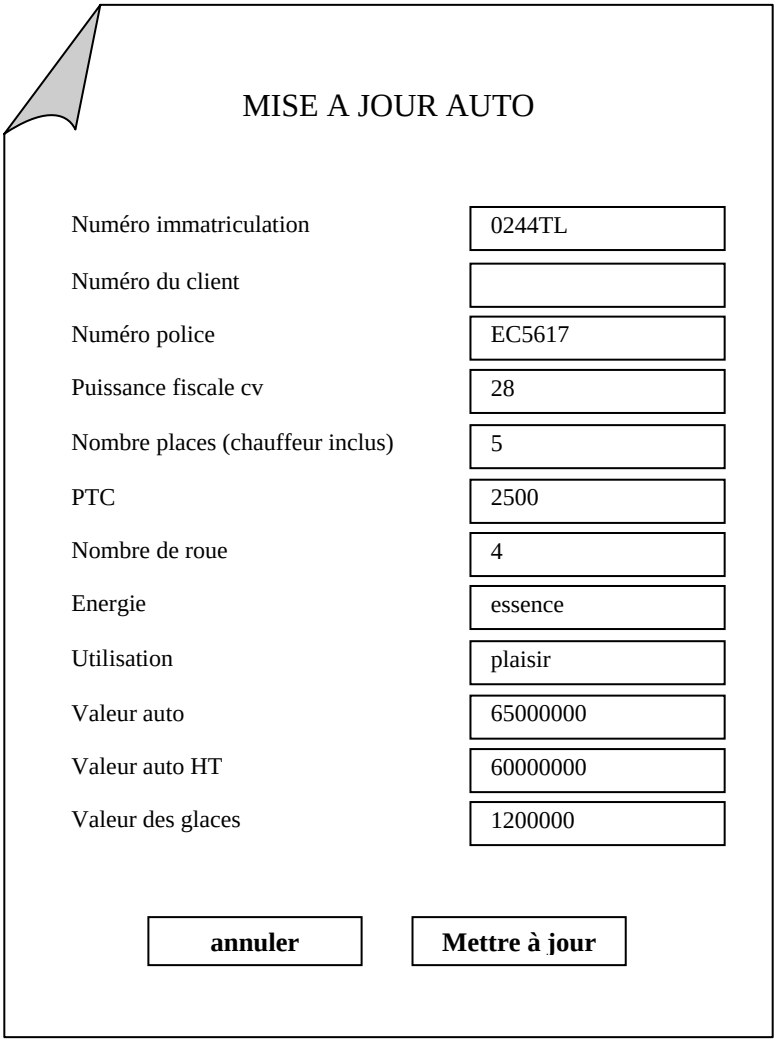
Le commercial peut mettre à jour un enregistrement sur une auto donnée.

III.2.7.1 - SCENARIO

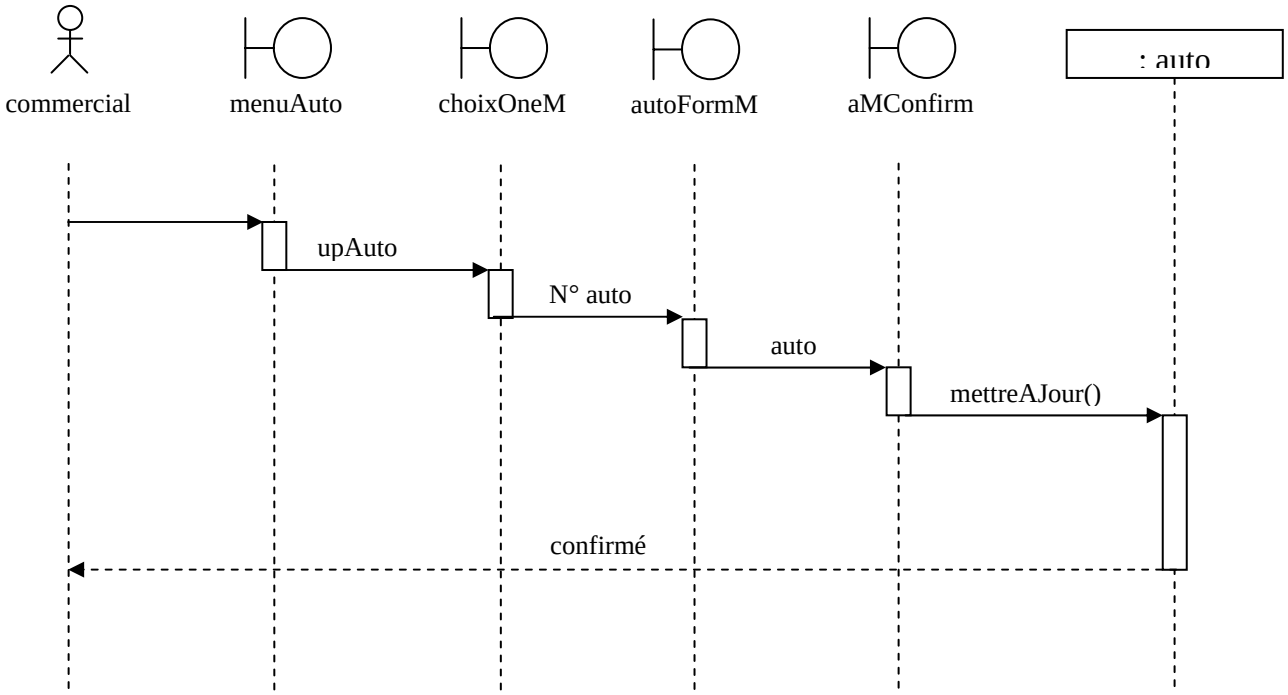
- Le commercial accède au 'menu auto'
- Le commercial appuie sur le bouton 'mettre à jour une auto'
- Le système appelle la page d'affichage de l'auto à mettre à jour.
- Le commercial appuie sur le bouton 'mettre à jour'.
- Le système appelle la page de confirmation de le mise à jour.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le menu auto après mise à jour.
- Le commercial est ainsi informé de la mise à jour.

III.2.7.2 - MAQUETTE

autoFormM.jsp



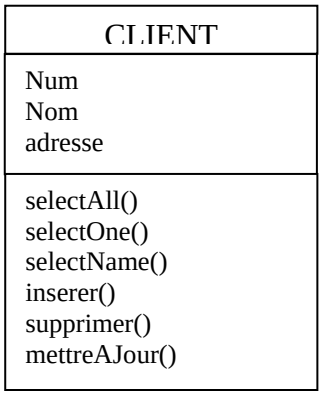
III.2.7.3 - DIAGRAMME DE SEQUENCE



III.3 - CLIENT

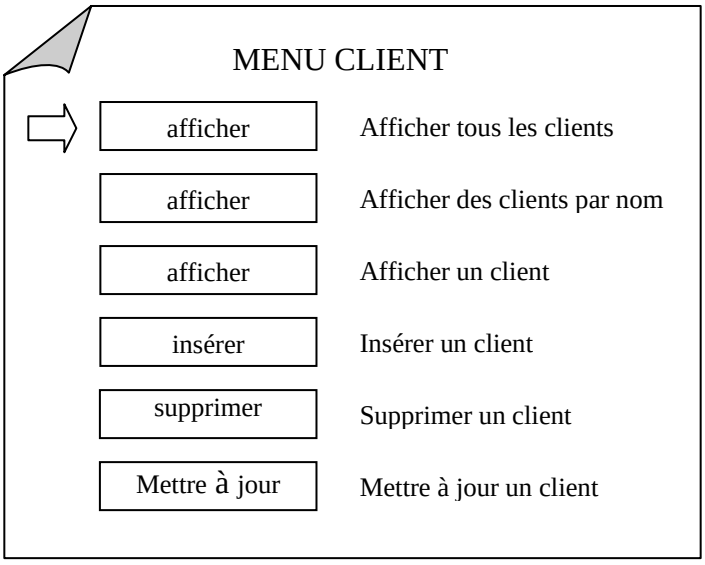
III.3.1 - DIAGRAMME DE CLASSE

Le diagramme de classe des cas d'utilisation du groupe client reste le même et se différencie seulement des méthodes appelées détaillées ci-après.



- selectAll() pour voir tous les clients.
- selectOne() pour voir un client.
- selectName pour voir des clients par nom.
- inserer() pour insérer un client.
- supprimer() pour supprimer un client.
- mettreAJour() pour mettre à jour un client.

LA PAGE DE MENU CLIENT : menuClient.jsp



III.3.2 - CAS D'UTILISATION : VOIR TOUS LES CLIENTS

Le commercial peut voir en ligne la liste de tous les clients stockés dans la base des données. Ceci détaille les informations sur chaque client.

III.3.2.1 - SCENARIO

- Le commercial accède au 'menu client'.
- Le commercial appuie sur le bouton 'afficher tous les clients'.
- Le système appelle la classe client et en instancie un objet appelé client correspondant.
- Le système appelle la méthode selectAll de l'objet client.
- L'objet effectue l'extraction des données et les passe à la page clientAll.jsp.
- Le commercial obtient la liste des clients.
- Le commercial navigue dans la liste.
- Pour quitter, le commercial appuie sur le bouton retour menu.

III.3.2.2 - MAQUETTE

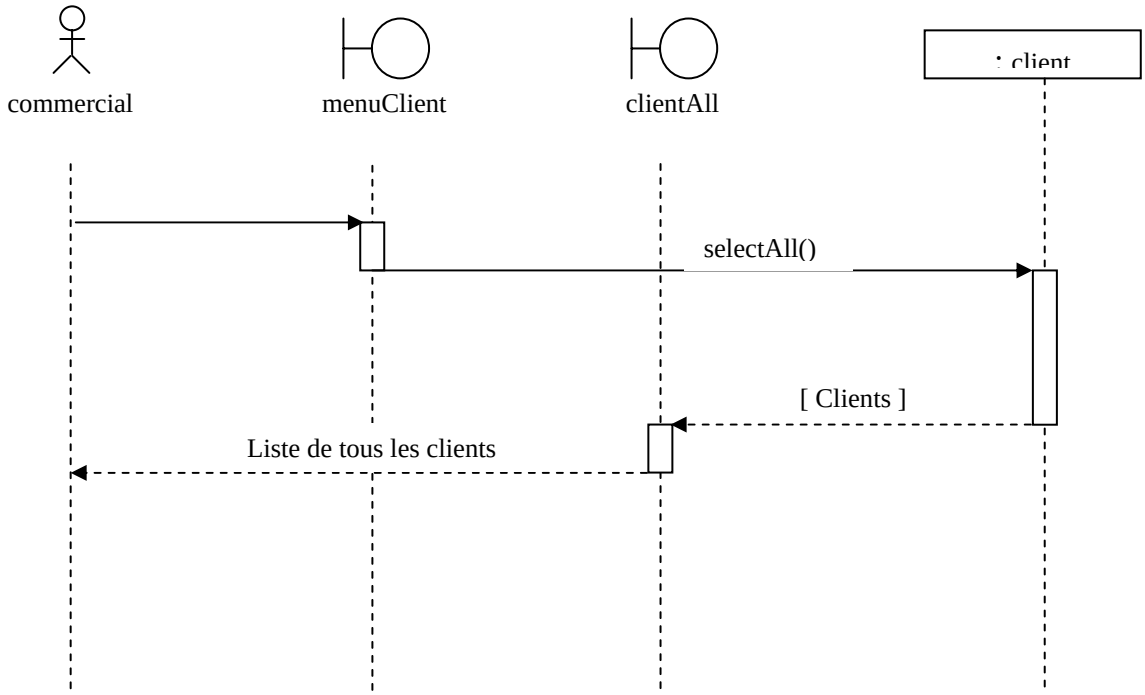
clientAll.jsp

LISTE DE TOUS LES CLIENTS

NUMERO	NOM	ADRESSE
20032	BOURGOIN	ANTSIRANANA 520
20531	LEONG	SAMBAVA
30631	AMIDJY	ANTSIRANANA
33852	MOUSSETANSIR	ANTSIRANANA

Retour menu

III.3.2.3 - DIAGRAMME DE SEQUENCE



III.3.3 - CAS D'UTILISATION : VOIR DES CLIENTS PAR NOM

Le commercial peut voir la liste des clients par nom.

III.3.3.1 - SCENARIO

- Le commercial accède au 'menu client'.
- Le commercial appuie sur le bouton 'afficher les clients par nom'.
- Le système appelle la page de saisie de critère de sélection nom.
- Le système appelle la classe client et en instancie un objet appelé client correspondant.
- Le système appelle la méthode selectName de l'objet client.
- L'objet effectue l'extraction des données et les passe à la page clientName.jsp
- Le commercial obtient la liste des clients pour le nom spécifié.
- Le commercial navigue dans la liste.
- Pour quitter, le commercial appuie sur le bouton 'retour menu'.

III.3.3.2 - MAQUETTE

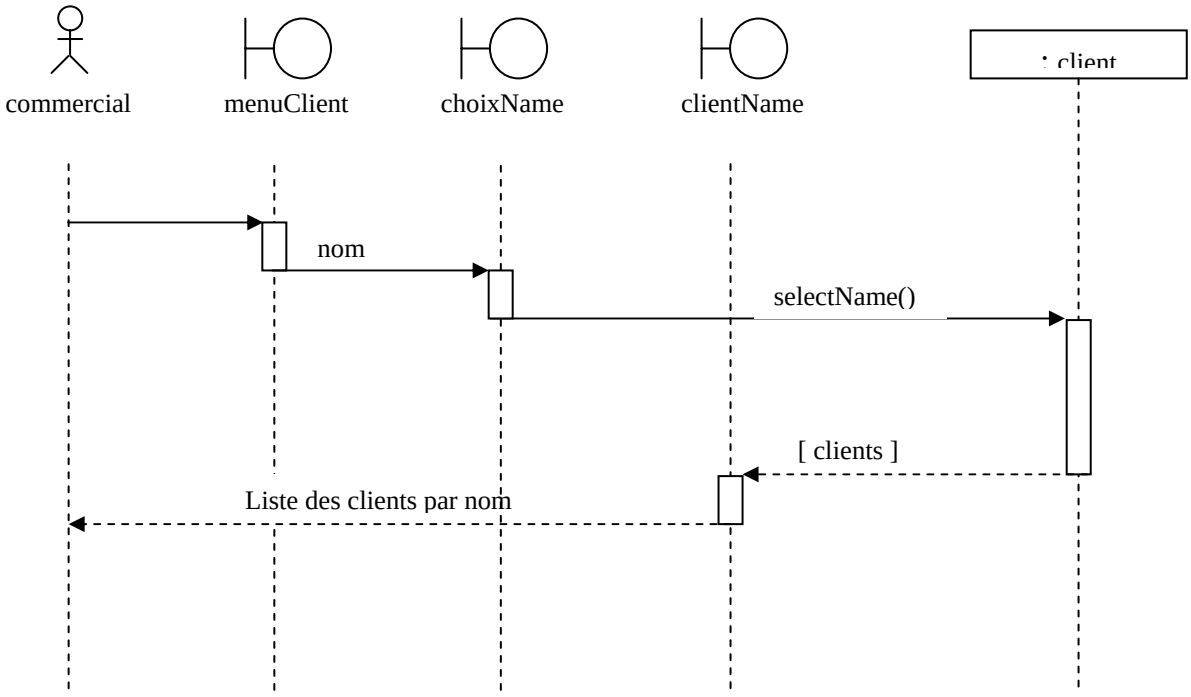
clientName.jsp

LISTE DES CLIENTS LEONG

NUMERO	NOM	ADRESSE
20531	LEONG	SAMBAVA

Retour menu

III.3.3.3 - DIAGRAMME DE SEQUENCE



III.3.4 - CAS D'UTILISATION : VOIR UN CLIENT

Le commercial peut voir les informations sur un client.

III.3.4.1 - SCENARIO

- Le commercial accède au 'menu client'
- Le commercial appuie sur le bouton 'afficher un client'
- Le système appelle la page de saisie de critère de sélection client.
- Le système appelle la classe client et en instancie un objet appelé client correspondant
- Le système appelle la méthode selectOne de l'objet client
- L'objet effectue l'extraction des données et les passe à la page clientOne.jsp
- Le commercial obtient les informations du client spécifié
- Pour quitter, le commercial appuie sur le bouton 'retour menu'

III.3.4.2 - MAQUETTE

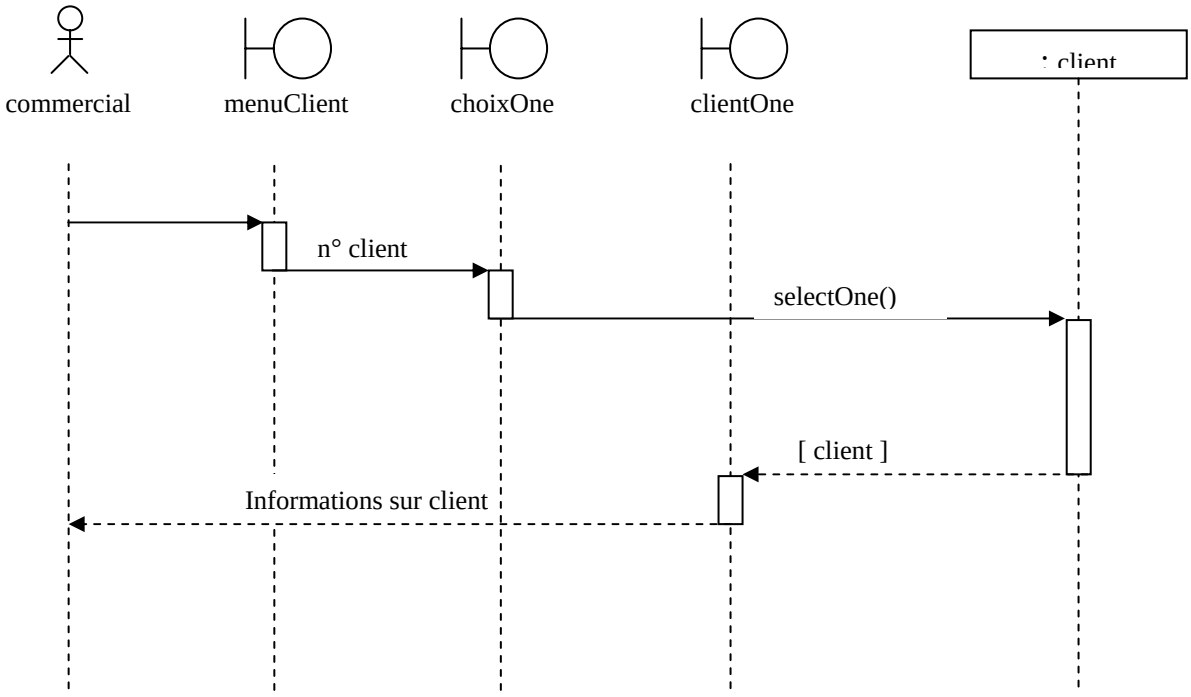
clientOne.jsp

INFORMATION SUR LE CLIENT N° 20531

NUMERO	NOM	ADRESSE
20531	LEONG	SAMBAVA

Retour menu

III.3.4.3 - DIAGRAMME DE SEQUENCE



III.3.5 - CAS D'UTILISATION : INSERER UN CLIENT

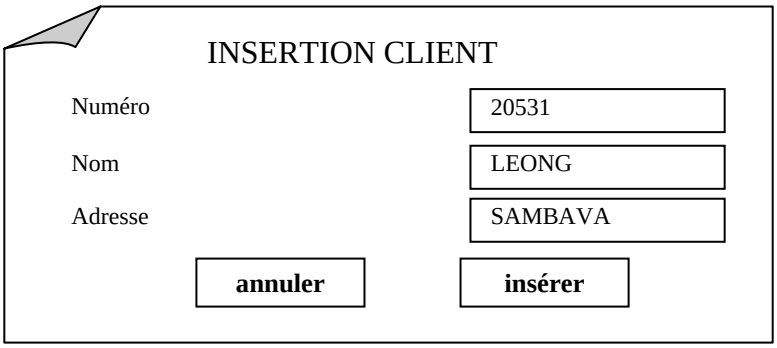
Le commercial peut insérer un enregistrement sur un client donné.

III.3.5.1 - SCENARIO

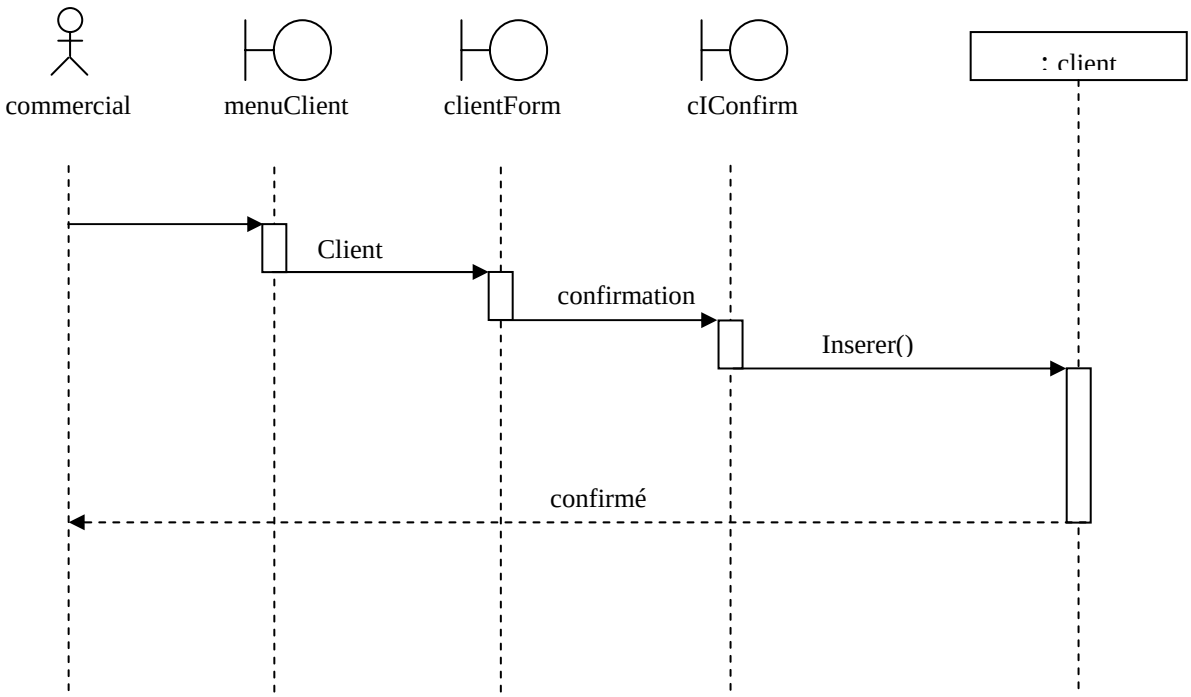
- Le commercial accède au 'menu client'
- Le commercial appuie sur le bouton 'insérer un client'
- Le système appelle la page de saisie des informations sur le client à insérer.
- Le commercial remplit le formulaire d'insertion.
- Le commercial appuie sur le bouton 'insérer'.
- Le système appelle la page de confirmation d'insertion.
- Le commercial appuie sur le bouton 'insérer'.
- Le système appelle la classe client et en instancie un objet appelé client correspondant
- Le système appelle la méthode inserer() de l'objet client.
- Le système appelle le 'menu client' après insertion.
- Le commercial est ainsi informé de l'insertion.

III.3.5.2 - MAQUETTE

clientForm.jsp & cIConfirm.jsp



III.3.5.3 - DIAGRAMME DE SEQUENCE



III.3.6 - CAS D'UTILISATION : SUPPRIMER UN CLIENT

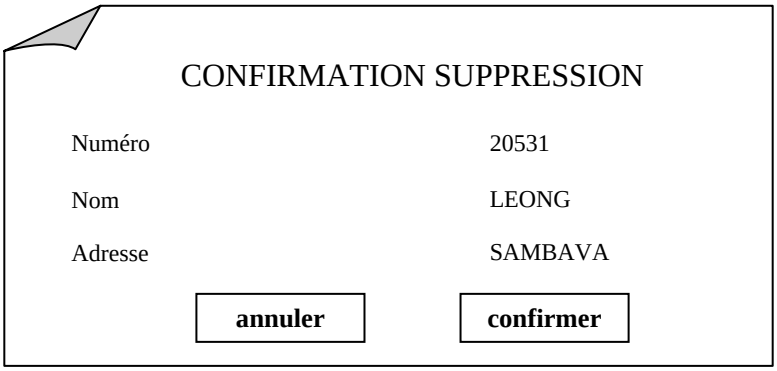
Le commercial peut supprimer un enregistrement sur un client donné.

III.3.6.1 - SCENARIO

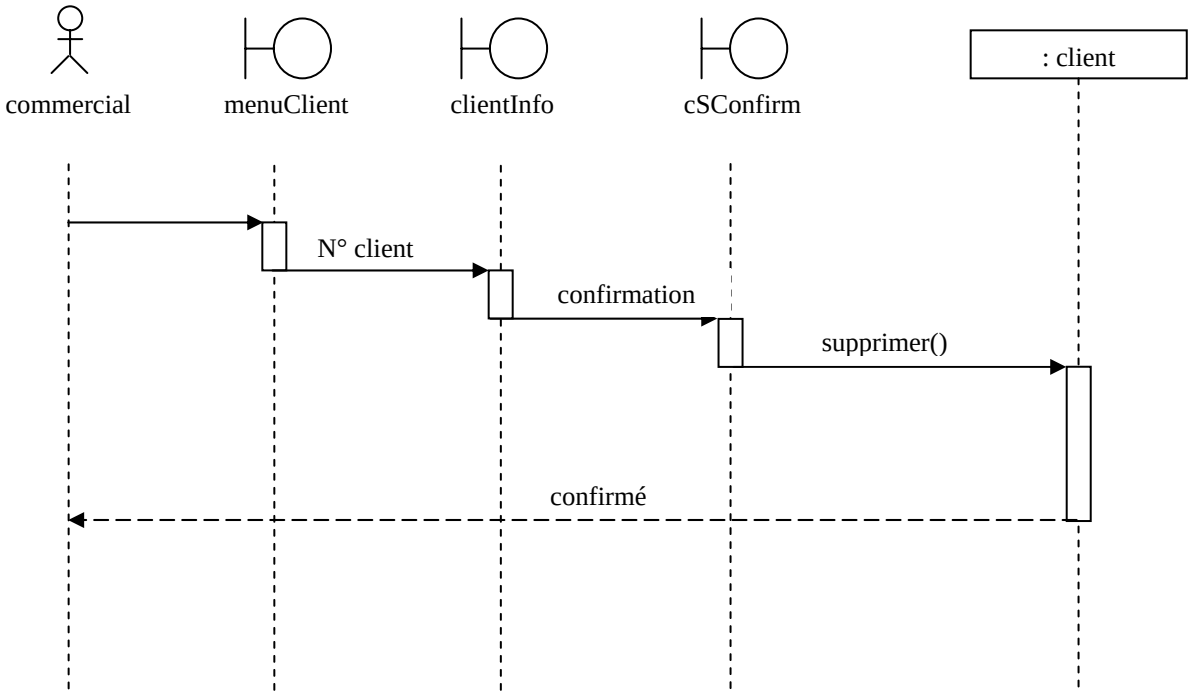
- Le commercial accède au 'menu client'
- Le commercial appuie sur le bouton 'supprimer un client'.
- Le système appelle le page de critère de suppression.
- Le commercial saisit le numéro du client à supprimer.
- Le commercial appuie sur le bouton 'supprimer'.
- Le système appelle la page d'affichage du client à supprimer.
- Le commercial appuie sur le bouton 'supprimer'.
- Le système appelle la page de confirmation de suppression.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le 'menu client' après suppression.
- Le commercial est ainsi informé de la suppression.

III.3.6.2 - MAQUETTE

cSConfirm.jsp



III.3.6.3 - DIAGRAMME DE SEQUENCE



III.3.7 - CAS D'UTILISATION : METTRE A JOUR UN CLIENT

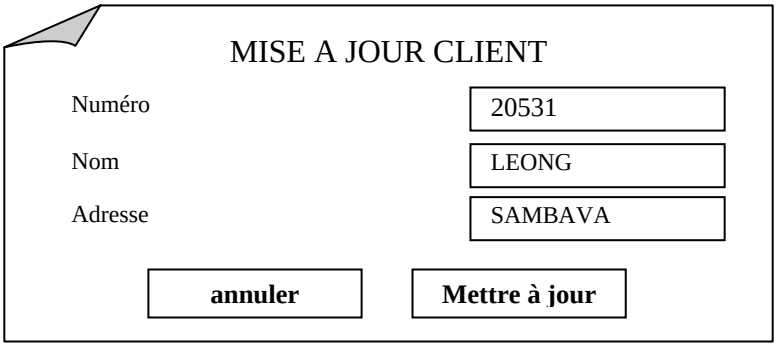
Le commercial peut mettre à jour un enregistrement sur un client donné.

III.3.7.1 - SCENARIO

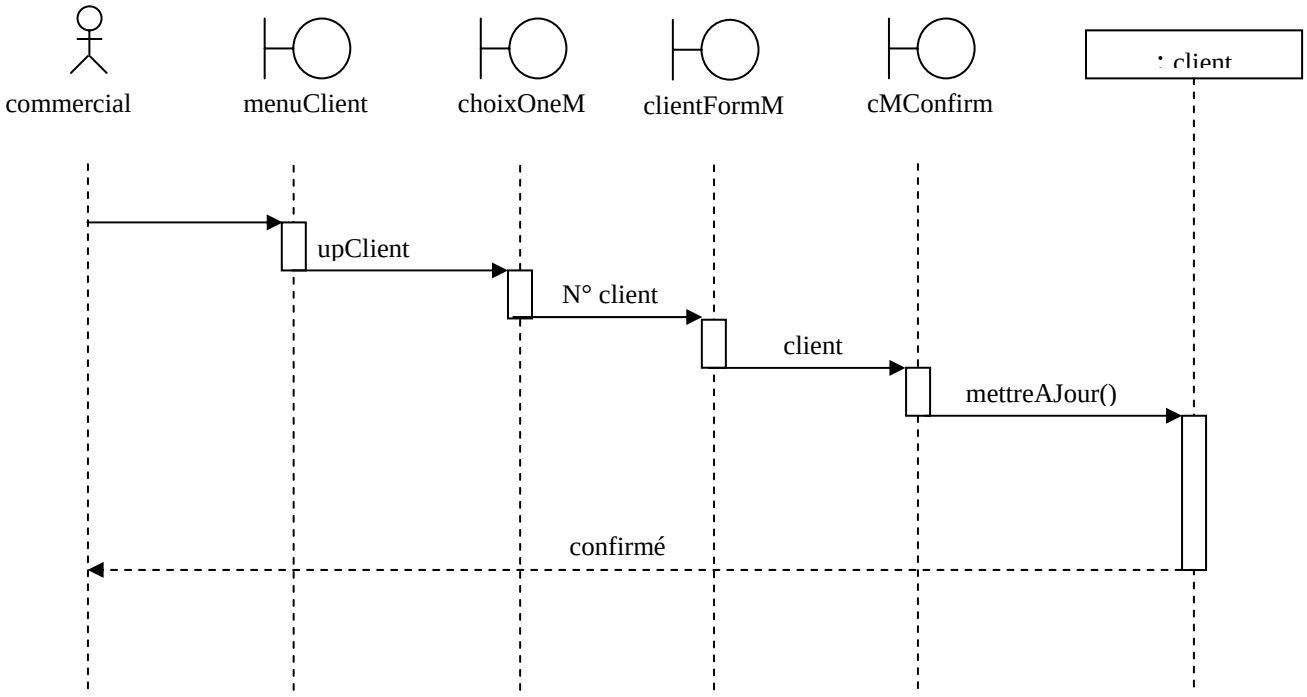
- Le commercial accède au 'menu client'.
- Le commercial appuie sur le bouton 'mettre à jour un client'.
- Le système appelle la page de critère de sélection de mise à jour.
- Le commercial saisit le numéro du client à mettre à jour.
- Le système appelle la page d'affichage du client à mettre à jour.
- Le commercial appuie sur le bouton 'mettre à jour'.
- Le système appelle la page de confirmation de la mise à jour.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le 'menu auto' après mise à jour.
- Le commercial est ainsi informé de la mise à jour.

III.3.7.2 - MAQUETTE

clientFormM.jsp & cMConfirm.jsp



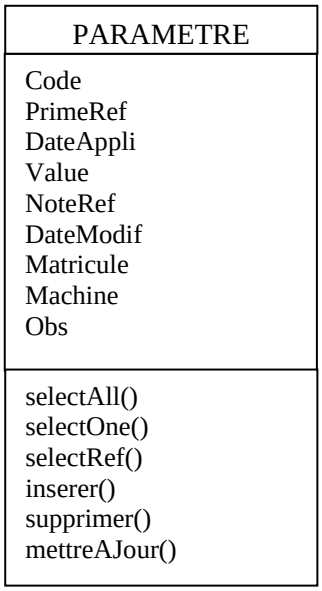
III.3.7.3 - DIAGRAMME DE SEQUENCE



III.4 - PARAMETRES

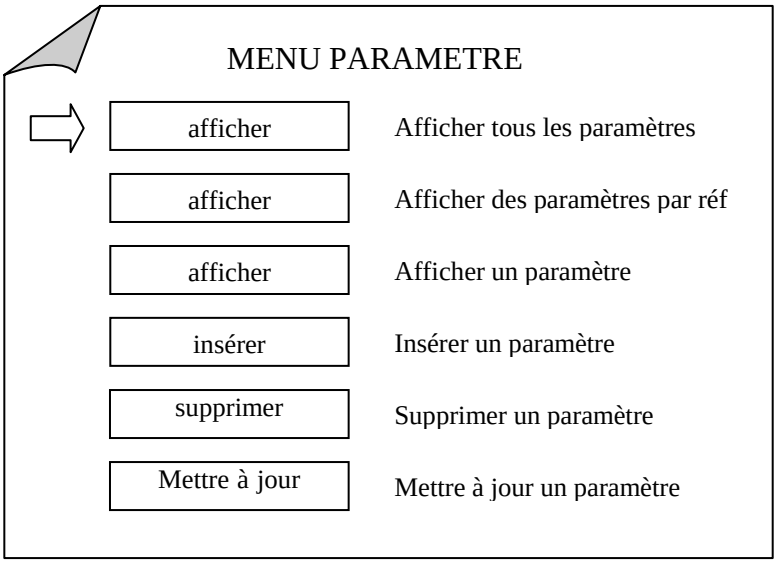
III.4.1 - DIAGRAMME DE CLASSE

Le diagramme de classe des cas d'utilisation du groupe paramètre reste le même et se différencie seulement des méthodes appelées détaillées ci-après.



- selectAll() pour voir tous les paramètres.
- selectOne() pour voir un paramètre.
- selectRef pour voir des paramètres par réf.
- inserer() pour insérer un paramètre.
- supprimer() pour supprimer un paramètre.
- mettreAJour() pour mettre à jour un paramètre.

LA PAGE DE MENU PARAMETRES : menuParam.jsp



III.4.2 - CAS D'UTILISATION : VOIR TOUS LES PARAMETRES

Le commercial peut voir en ligne la liste de tous les paramètres stockés dans la base des données. Ceci détaille les informations sur chaque paramètre.

III.4.2.1 - SCENARIO

- Le commercial accède au 'menu paramètre'.
- Le commercial appuie sur le bouton 'afficher tous les paramètres'.
- Le système appelle la classe paramètre et en instancie un objet appelé paramètre correspondant.
- Le système appelle la méthode selectAll de l'objet paramètre.
- L'objet effectue l'extraction des données et les passe à la page paramAll.jsp.
- Le commercial obtient la liste des paramètres.
- Le commercial navigue dans la liste.
- Pour quitter, le commercial appuie sur le bouton 'retour menu'.

III.4.2.2 - MAQUETTE

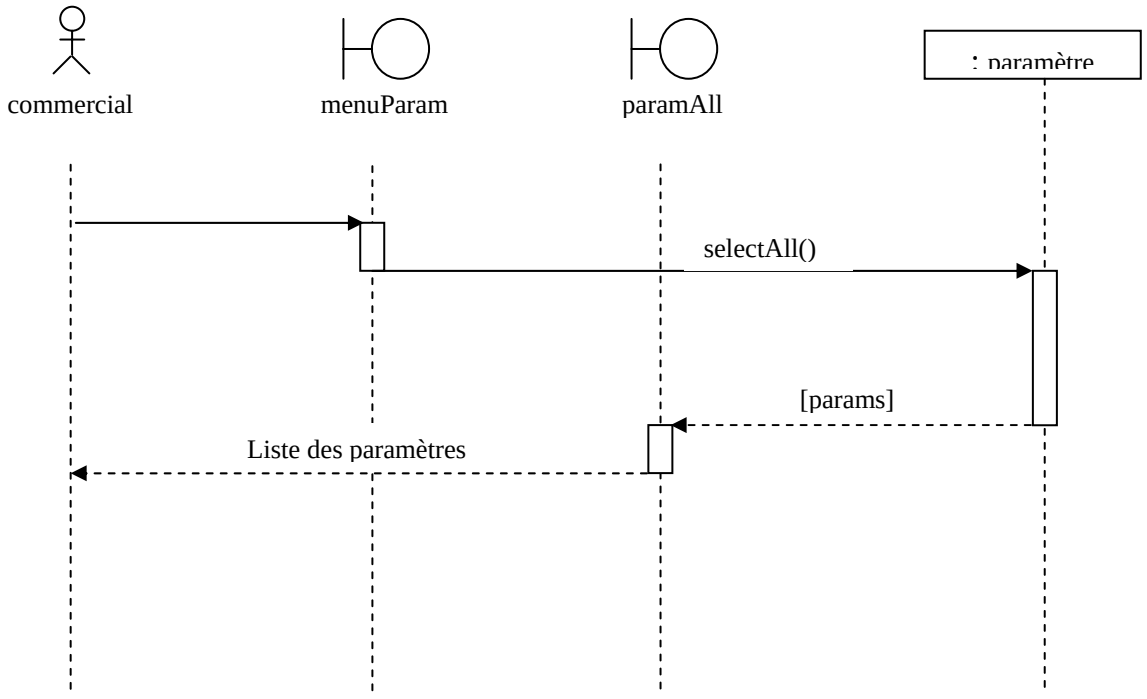
paramAll.jsp

LISTE DE TOUS LES PARAMETRES

Code	*Réf*	Date application	Valeur	Réf note	Date modification	Matricule	Machine	Observation
1	21aa1	12/02/02	113000					
2	21ba1	12/02/02	142000					
3	21ca1	12/02/02	160000					
4	21da1	12/02/02	194000					

Retour menu

III.4.2.3 - DIAGRAMME DE SEQUENCE



III.4.3 - CAS D'UTILISATION : VOIR DES PARAMETRES PAR REF

Le commercial peut voir la liste des paramètres par référence.

III.4.3.1 - SCENARIO

- Le commercial accède au 'menu paramètre'.
- Le commercial appuie sur le bouton 'afficher les paramètres par référence'.
- Le système appelle la page de saisie de critère de sélection nom.
- Le système appelle la classe paramètre et en instancie un objet appelé paramètre correspondant.
- Le système appelle la méthode selectRef de l'objet paramètre.
- L'objet effectue l'extraction des données et les passe à la page paramRef.jsp
- Le commercial obtient la liste des paramètres pour la référence spécifiée.
- Le commercial navigue dans la liste.
- Pour quitter, le commercial appuie sur le bouton retour menu.

III.4.3.2 - MAQUETTE

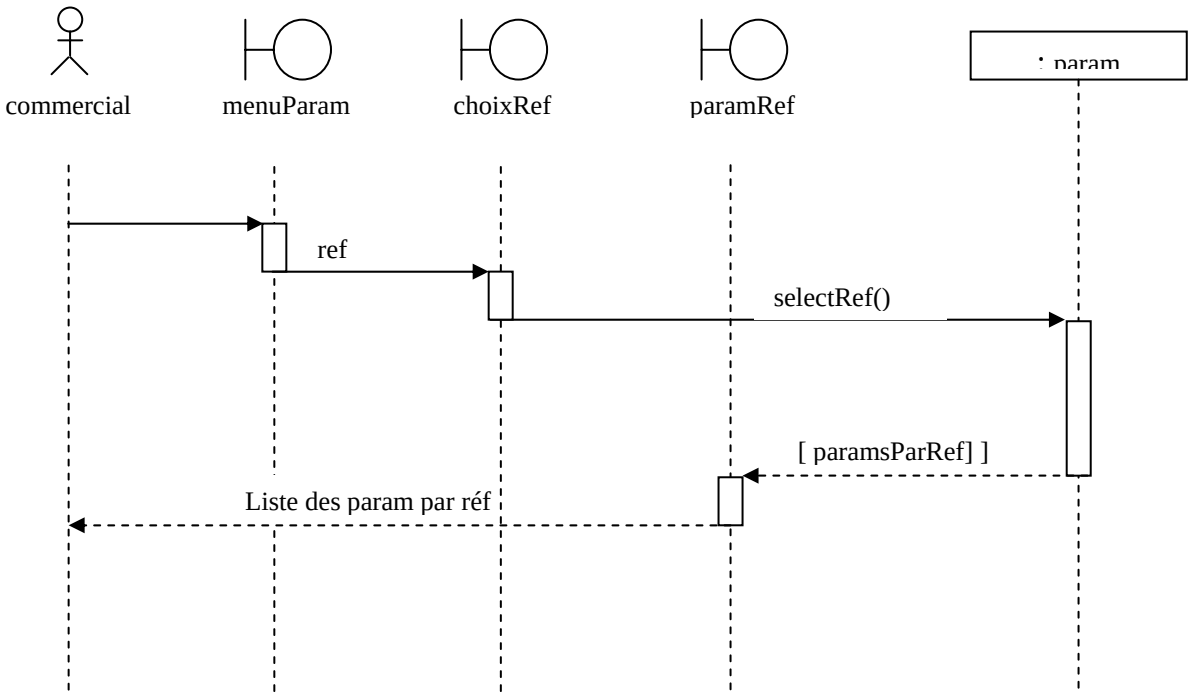
paramRef.jsp

LISTE DES PARAMETRES POUR 21aa1

Code	*Réf*	Date application	Valeur	Réf note	Date modification	Matricule	Machine	Observation
1	21aa1	12/02/02	113000					
190	21aa1	12/03/02	142000					

Retour menu

III.4.3.3 - DIAGRAMME DE SEQUENCE



III.4.4 - CAS D'UTILISATION : VOIR UN PARAMETRE

Le commercial peut voir les informations sur un paramètre.

III.4.4.1 - SCENARIO

- Le commercial accède au 'menu paramètre'.
- Le commercial appuie sur le bouton 'afficher un paramètre'
- Le système appelle la page de saisie de critère de sélection paramètre.
- Le commercial saisit le code d'un paramètre à visualiser.
- Le système appelle la classe paramètre et en instancie un objet appelé paramètre correspondant
- Le système appelle la méthode selectOne de l'objet paramètre
- L'objet effectue l'extraction des données et les passe à la page paramOne.jsp
- Le commercial obtient les informations du paramètre spécifié
- Pour quitter, le commercial appuie sur le bouton retour menu

III.4.4.2 - MAQUETTE

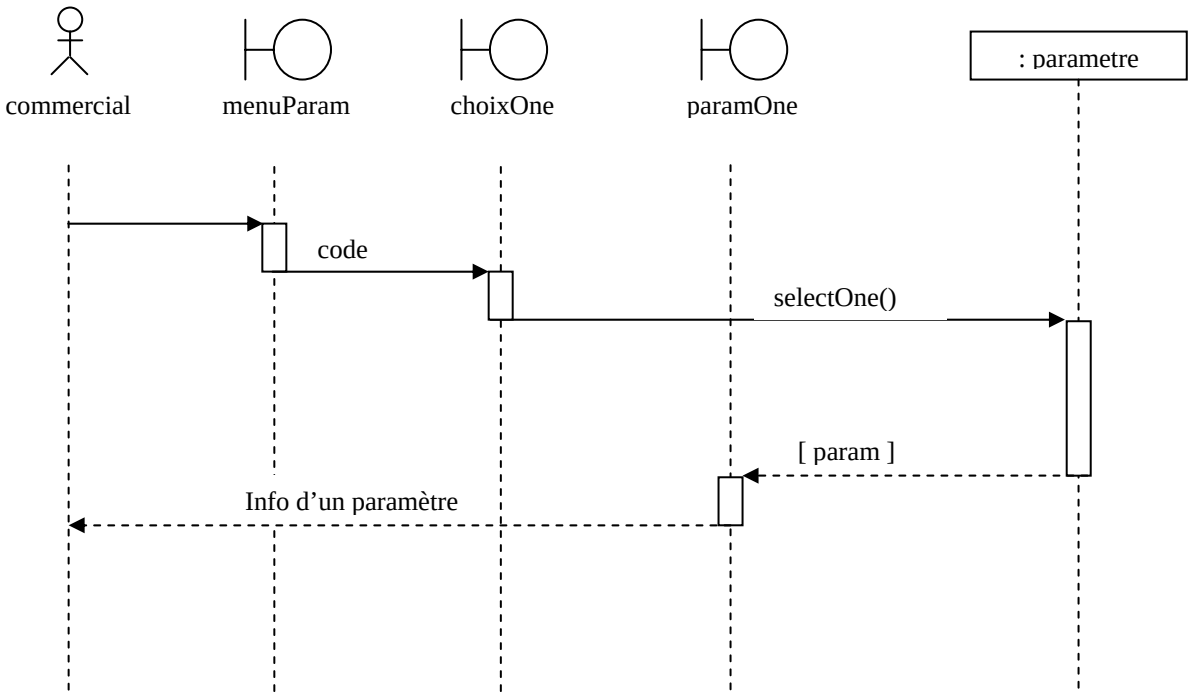
paramOne.jsp

INFORMATION SUR CODE 1

Code	*Réf*	Date application	Valeur	Réf note	Date modification	Matricule	Machine	Observation
1	21aa1	12/02/02	113000					

Retour menu

III.4.4.3 - DIAGRAMME DE SEQUENCE



III.4.5 - CAS D'UTILISATION : INSERER UN PARAMETRE

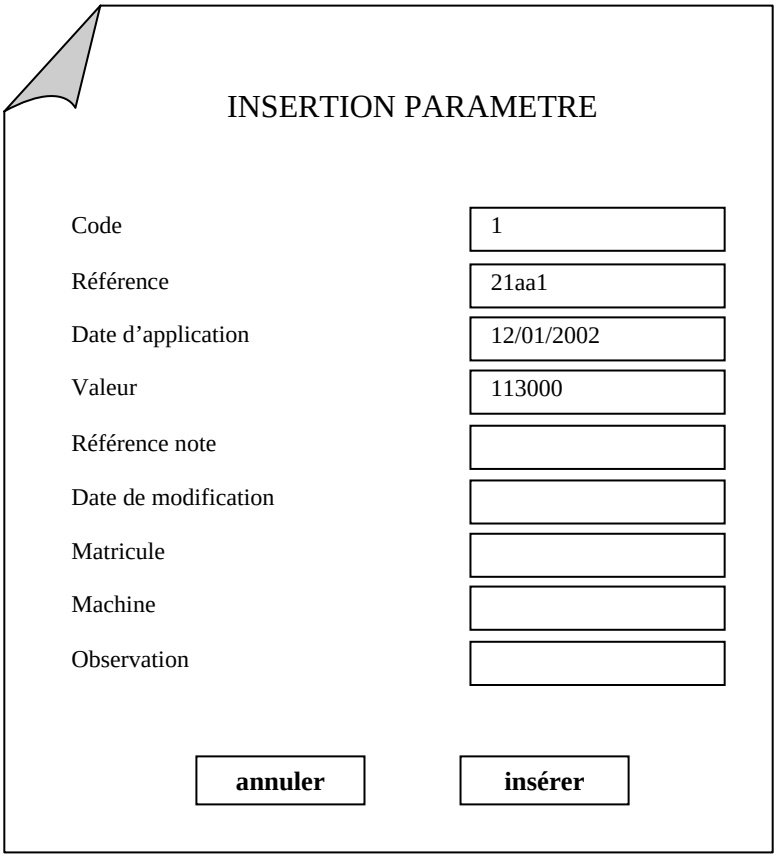
Le commercial peut insérer un paramètre.

III.4.5.1 - SCENARIO

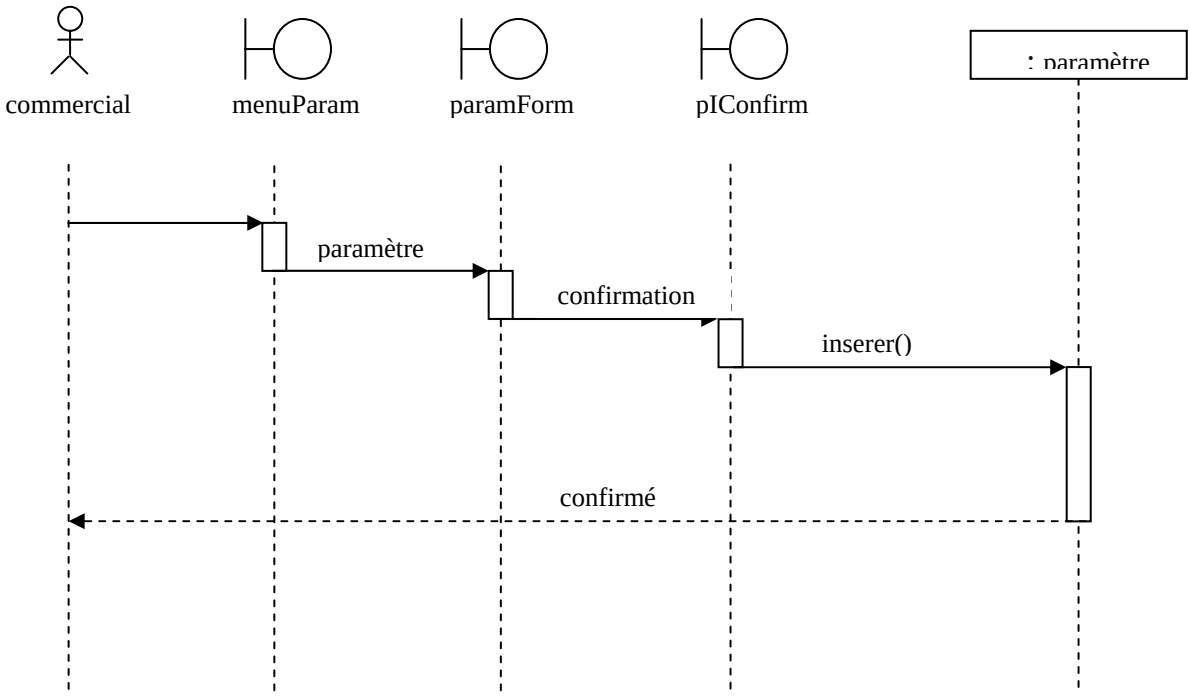
- Le commercial accède au 'menu paramètre'
- Le commercial appuie sur le bouton 'insérer un paramètre'
- Le système appelle la page de saisie des informations sur le paramètre à insérer.
- Le commercial remplit le formulaire d'insertion.
- Le commercial appuie sur le bouton 'insérer'.
- Le système appelle la page de confirmation d'insertion.
- Le commercial appuie sur le bouton insérer.
- Le système appelle la classe paramètre et en instancie un objet appelé paramètre correspondant
- Le système appelle la méthode inserer() de l'objet paramètre.
- Le système appelle le menu paramètre après insertion.
- Le commercial est ainsi informé de l'insertion.

III.4.5.2 - MAQUETTE

paramForm.jsp & pIConfirm.jsp



III.4.5.3 - DIAGRAMME DE SEQUENCE



III.4.6 - CAS D'UTILISATION : SUPPRIMER UN PARAMETRE

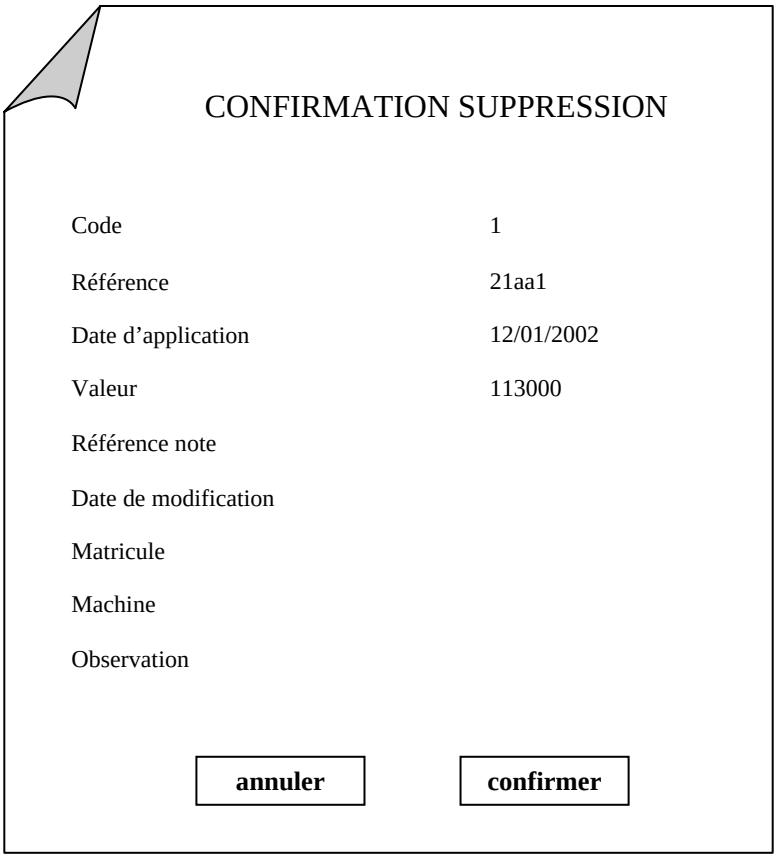
Le commercial peut supprimer un paramètre donné.

III.4.6.1 - SCENARIO

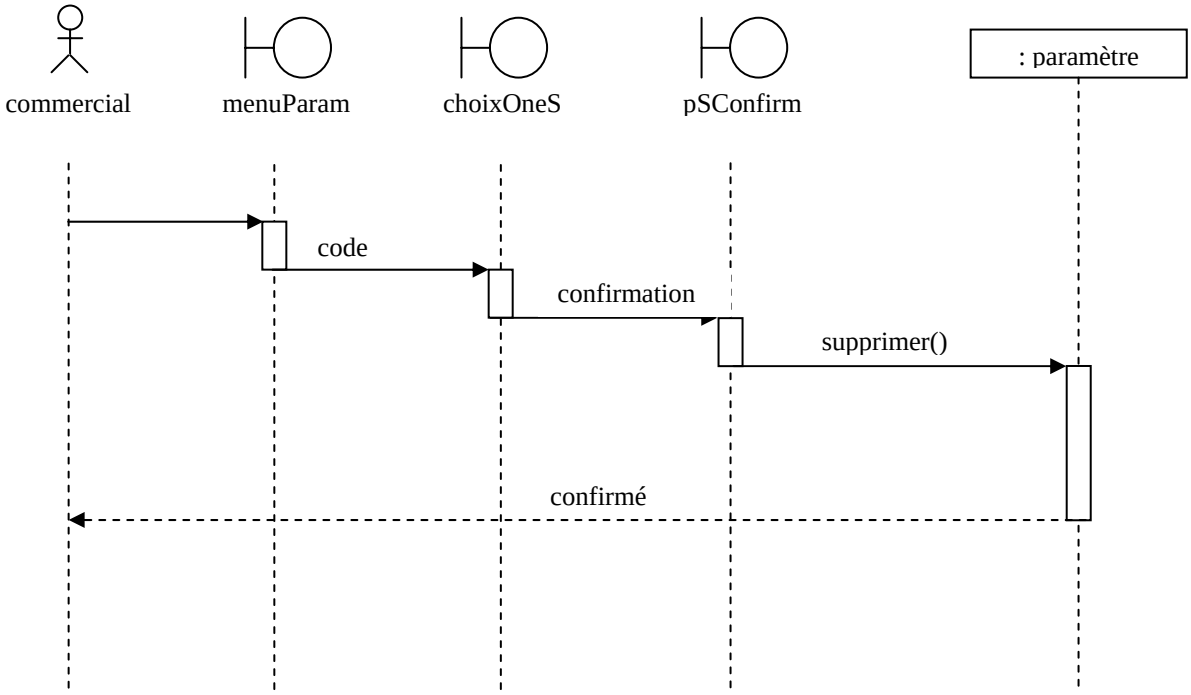
- Le commercial accède au 'menu paramètre'.
- Le commercial appuie sur le bouton 'supprimer un paramètre'.
- Le système appelle le page de critère de suppression.
- Le commercial saisit le code du paramètre à supprimer.
- Le commercial appuie sur le bouton 'supprimer'.
- Le système appelle la page d'affichage du paramètre à supprimer.
- Le commercial appuie sur le bouton 'supprimer'.
- Le système appelle la page de confirmation de suppression.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le menu paramètre après suppression.
- Le commercial est ainsi informé de la suppression.

III.4.6.2 - MAQUETTE

aSConfirm.jsp



III.4.6.3 - DIAGRAMME DE SEQUENCE



III.4.7 - CAS D'UTILISATION : METTRE A JOUR UN PARAMETRE

Le commercial peut mettre à jour un paramètre donné.

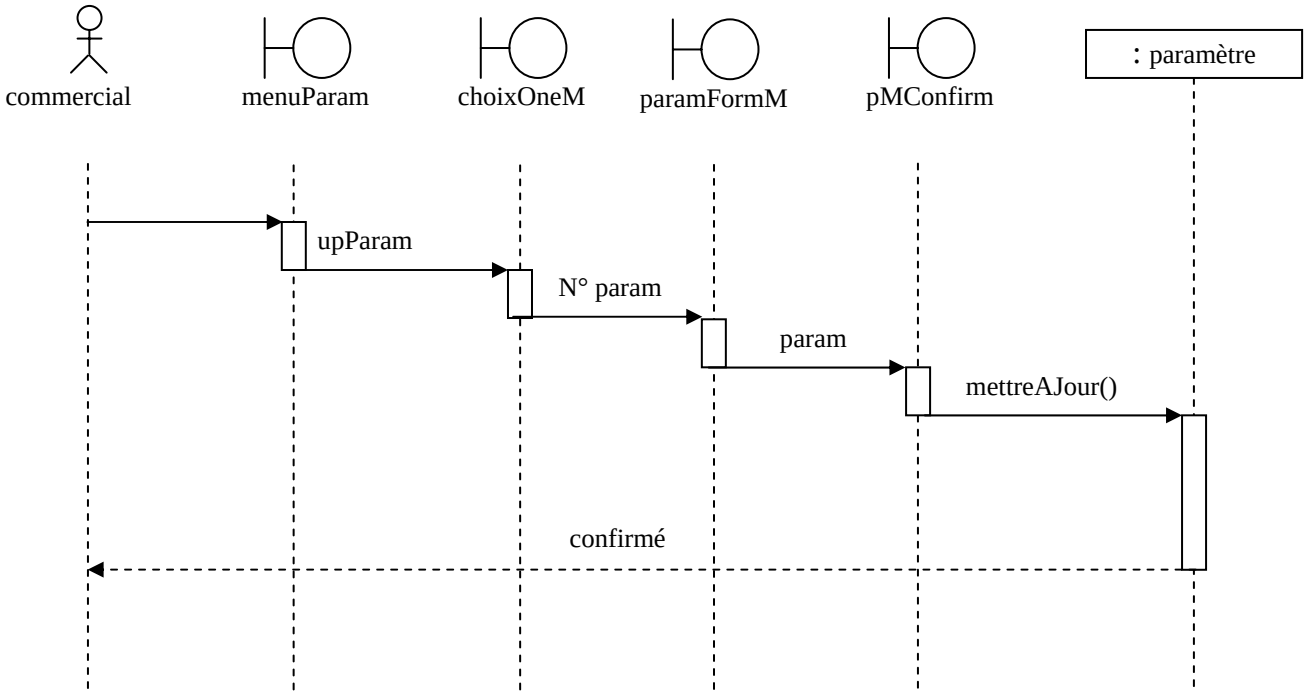
III.4.7.1 - SCENARIO

- Le commercial accède au 'menu paramètre'.
- Le commercial appuie sur le bouton 'mettre à jour un paramètre'.
- Le système appelle la page de critère de sélection de mise à jour.
- Le commercial saisit le numéro du paramètre à mettre à jour.
- Le système appelle la page d'affichage du paramètre à mettre à jour.
- Le commercial appuie sur le bouton 'mettre à jour'.
- Le système appelle la page de confirmation de la mise à jour.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le menu paramètre après mise à jour.
- Le commercial est ainsi informé de la mise à jour.

III.4.7.2 - MAQUETTE

paramFormM.jsp & pMConfirm.jsp

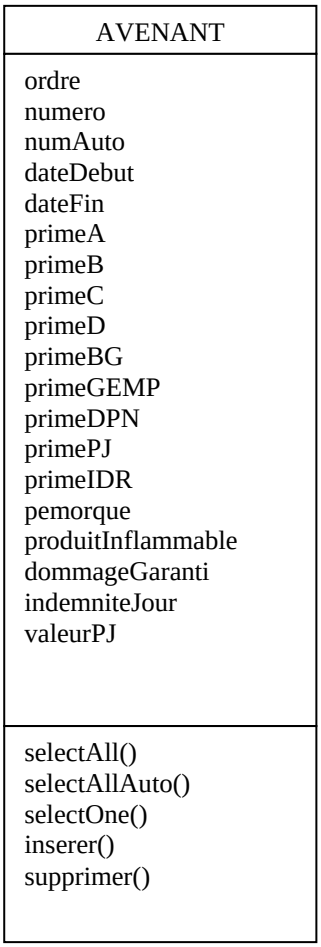
III.4.7.3 - DIAGRAMME DE SEQUENCE



III.5 - AVENANT

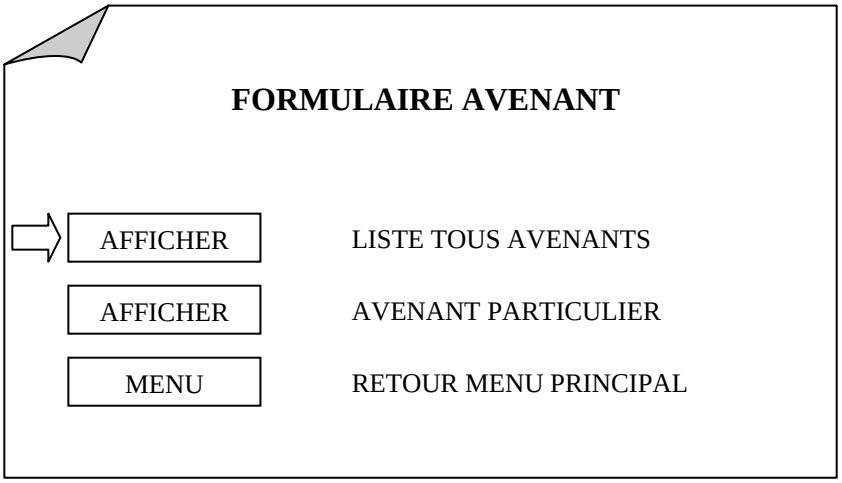
III.5.1 - DIAGRAMME DE CLASSE

Le diagramme de classe des cas d'utilisation du groupe auto reste le même et se différencie seulement des méthodes appelées détaillées ci-après.



- selectAll() pour voir tous les avenants.
- selectOne() pour voir un avenant.
- inserer() pour insérer un auto.
- supprimer() pour supprimer un auto.

LA PAGE DE MENU DE AVENANT : menuavenant.jsp



III.5.2 - CAS D'UTILISATION VOIR TOUS LES AVENANTS

Le commercial peut voir en ligne la liste de tous les avenants stockés dans la base des données. Ceci détaille les caractéristiques de chaque avenant.

III.5.2.1 - SCENARIO

- Le commercial accède au 'menu avenant'
- Le commercial appuie sur le bouton 'afficher liste tous avenants'
- Le système appelle la classe Avenant et en instancie un objet appelé avenant correspondant
- Le système appelle la méthode selectAll de l'objet avenant
- L'objet effectue l'extraction des données et les passe à la page avenantall.jsp
- Le commercial obtient la liste des avenants
- Le commercial navigue dans la liste
- Pour quitter, le commercial appuie sur le bouton retour menu

III.5.2.2 - MAQUETTE

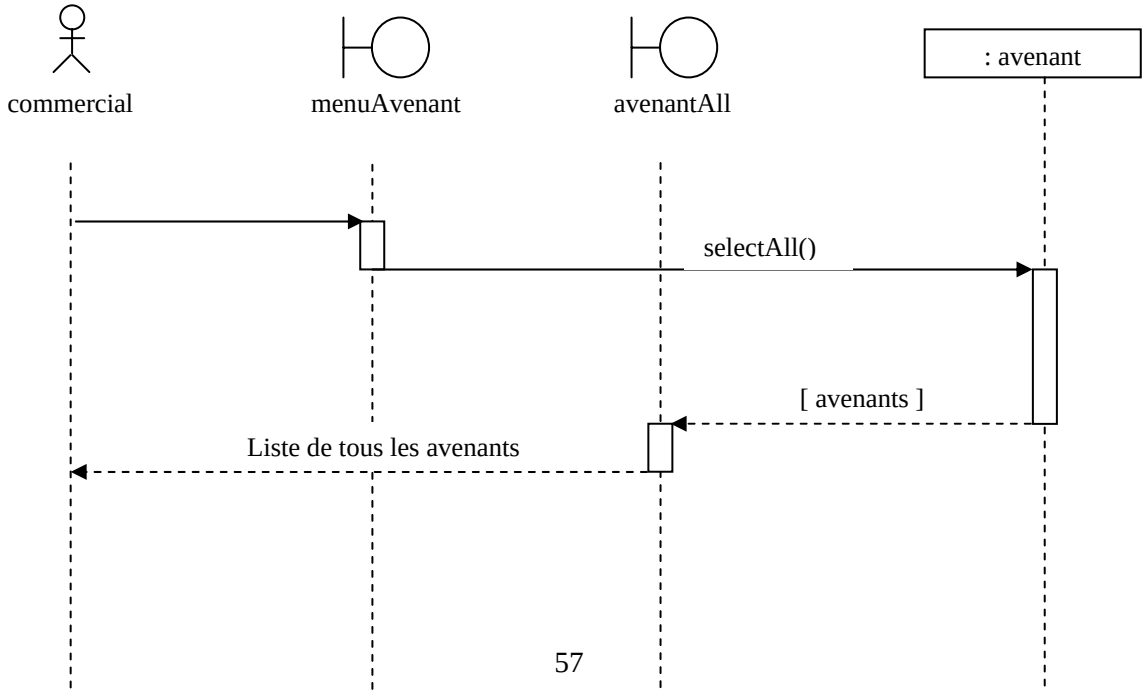
Avenantall.jsp

LISTE DES AVENANTS

ORDRE	NUMERO	DATE DEBUT	DATE FIN	PRIME A	PRIME B	PRIME C	PRIME D	PRIME BG	PRIME DPN
8	1	12/01/03	12/01/04	1513951	2252903	450581	450581	291976	50065
9	2	12/01/03	12/01/04	1513951	2252903	450581	450581	291976	50065
10	26	12/01/03	12/01/04	771995	2202839	375484	375484	272952	50065

MENU PRINCIPAL MENU AVENANT

III.5.2.3 - DIAGRAMME DE SEQUENCE



III.5.3 - CAS D'UTILISATION : VOIR DES AVENANTS PAR AUTO

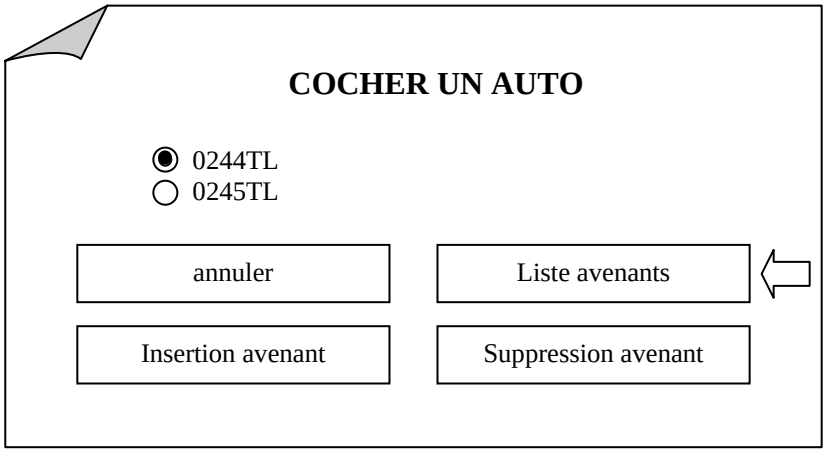
Le commercial peut voir la liste des avenants par auto.

III.5.3.1 - SCENARIO

- Le commercial accède au 'menu avenant'
- Le commercial appuie sur le bouton 'afficher avenant particulier'
- Le système appelle la page de critère de choix police.
- Le commercial saisit le numéro de police et appuie sur le bouton 'afficher'.
- Le système appelle la classe avenant et en instancie un objet appelé avenant correspondant
- Le système appelle la méthode privée selectListAuto de l'objet avenant
- L'objet effectue l'extraction des données et les passe à la page de critère de choix des véhicules
- Le commercial coche le véhicule intéressé et appuie sur le bouton 'liste avenants'.
- Le système appelle l' objet avenant
- Le système appelle la méthode selectAllAuto de l'objet avenant
- L'objet effectue l'extraction des données et les passe à la page Avenantlisteparauto.jsp
- Le commercial obtient la liste des avenants pour le véhicule spécifié
- Le commercial navigue dans la liste
- Pour quitter, le commercial appuie sur le bouton 'retour menu'

III.5.3.2 - MAQUETTE

Avenantchoixauto.jsp



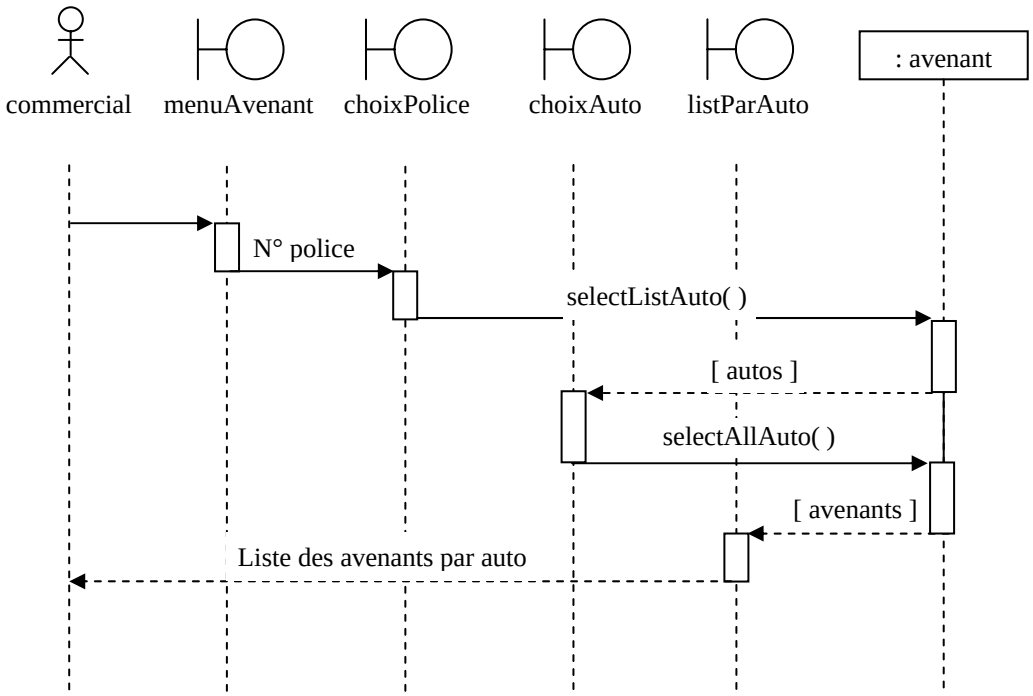
Avenantlisteparauto.jsp

LISTE DES AVENANTS DES POUR UN VEHICULE

ORDRE	NUMERO	DATE DEBUT	DATE FIN	PRIME A	PRIME B	PRIME C	PRIME D	PRIME BG	PRIME DPN
1	1	12/04/02	12/04/03	0	0	0	0	0	0
6	2	12/01/03	12/01/04	0	0	0	0	0	0
7	3	12/01/03	12/01/04	0	0	0	0	0	0

MENU PRINCIPAL MENU AVENANT

III.5.3.3 - DIAGRAMME DE SEQUENCE



III.5.4 - CAS D'UTILISATION : INSERER UN AVENANT

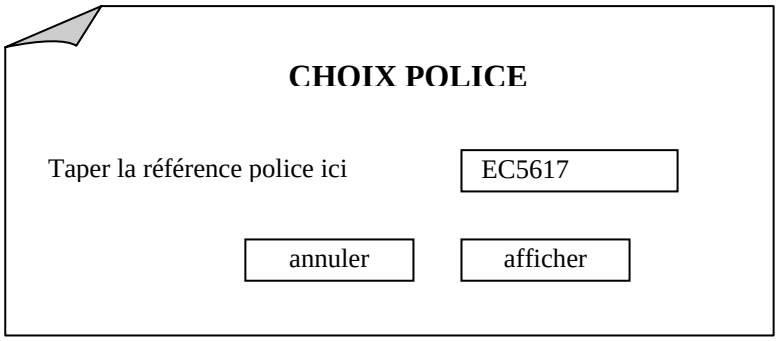
Le commercial peut insérer des informations sur un avenant donné.

III.5.4.1 - SCENARIO

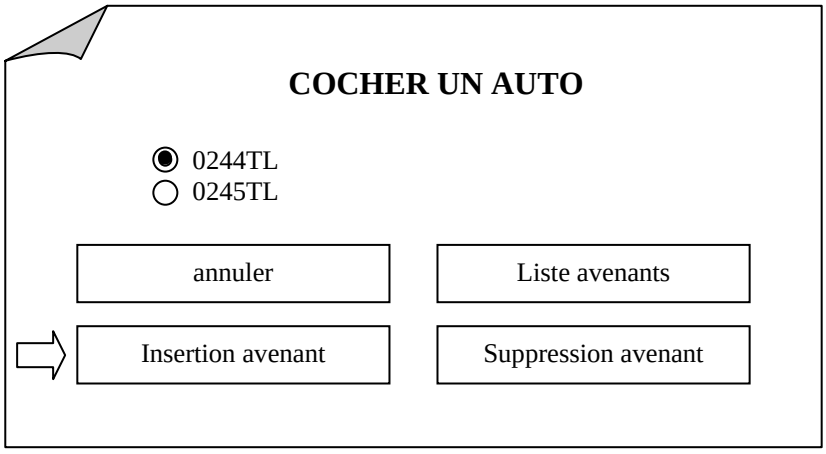
- Le commercial accède au 'menu avenant'
- Le commercial appuie sur le bouton 'avenant particulier'
- Le système appelle la page de saisie de critère police
- Le commercial saisit la référence de la police et appuie sur 'afficher'.
- Le système affiche la liste des véhicules correspondant à la police.
- Le commercial choisit un véhicule et appuie sur le bouton 'insérer un avenant'.
- Le système appelle la page de saisie des informations sur l'avenant à insérer.
- Le commercial remplit le formulaire d'insertion.
- Le commercial appuie sur le bouton 'insérer'.
- Le système appelle la classe Calcul et en instancie un objet appelé calcul correspondant.
- Le système appelle la page de confirmation d'insertion.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle la classe auto et en instancie un objet appelé avenant correspondant
- Le système appelle la méthode inserer() de l'objet avenant.
- Le système appelle le menu avenant après insertion.
- Le commercial est ainsi informé de l'insertion.

III.5.4.2 - MAQUETTE

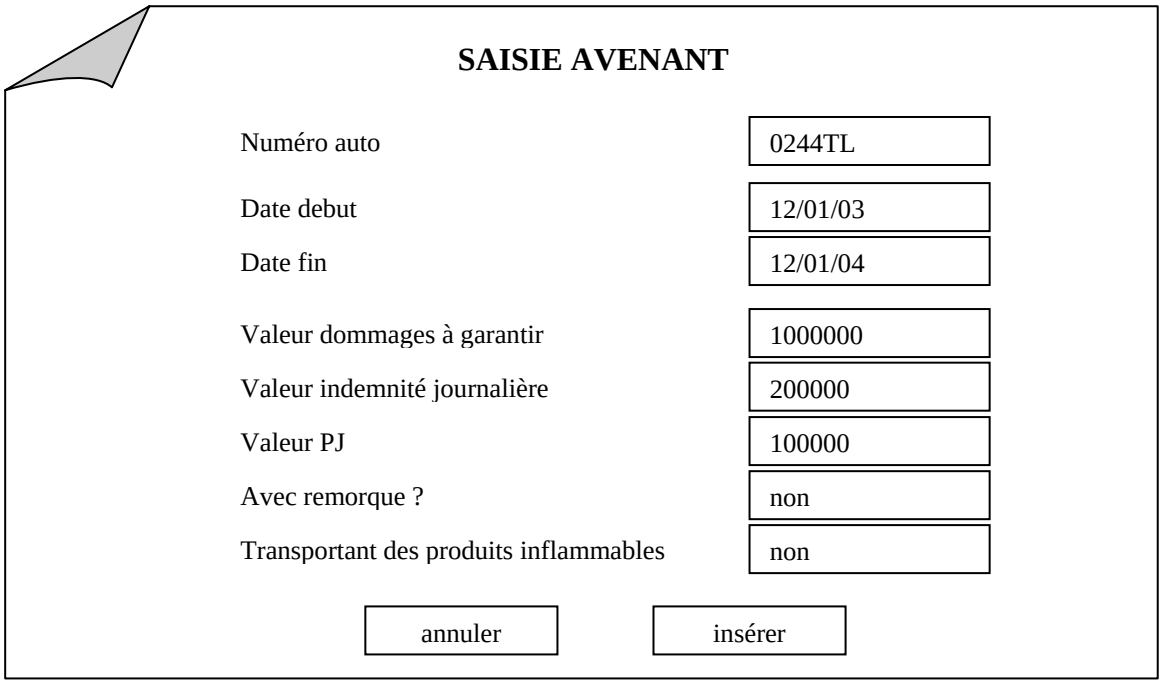
Avenantchoixpolice.jsp



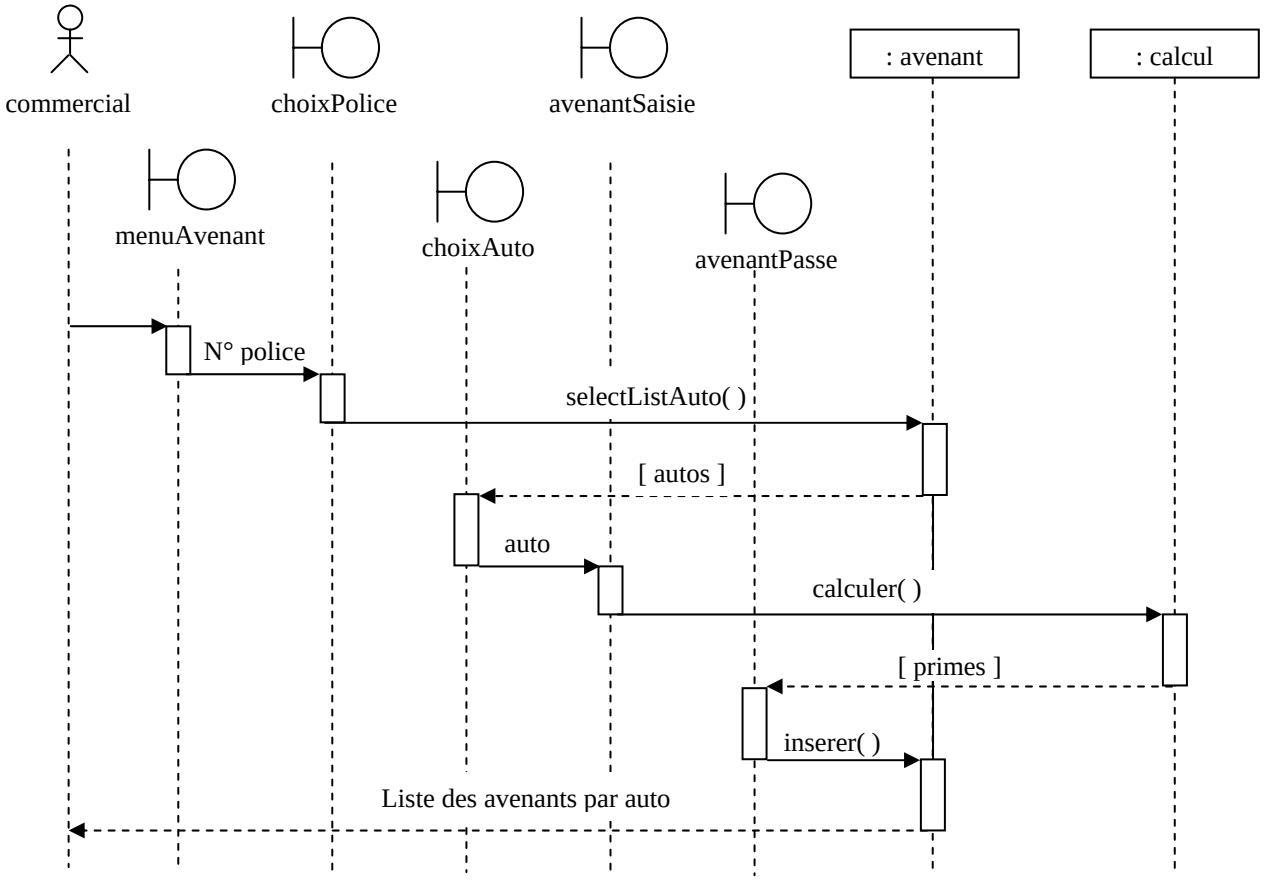
Avenantchoixauto.jsp



avenantsaisie.jsp & Avenantpasse.jsp



III.5.4.3 - DIAGRAMME DE SEQUENCE



III.5.5 - CAS D'UTILISATION : SUPPRIMER UN AVENANT

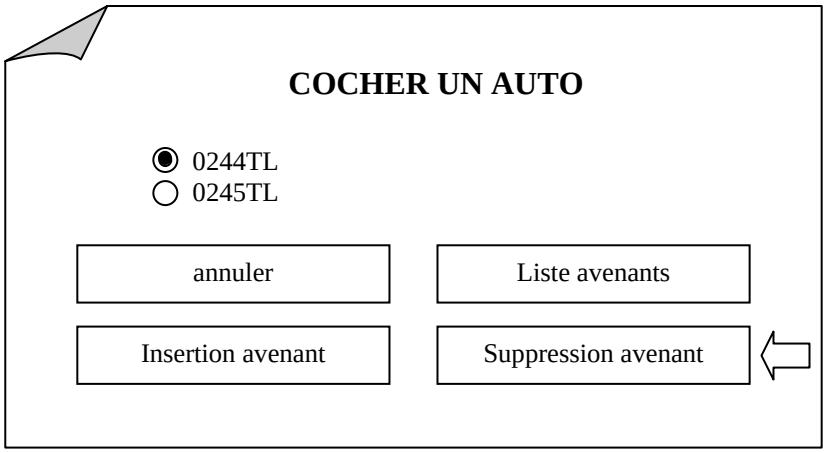
Le commercial peut supprimer un enregistrement sur un auto donné.

III.5.5.1 - SCENARIO

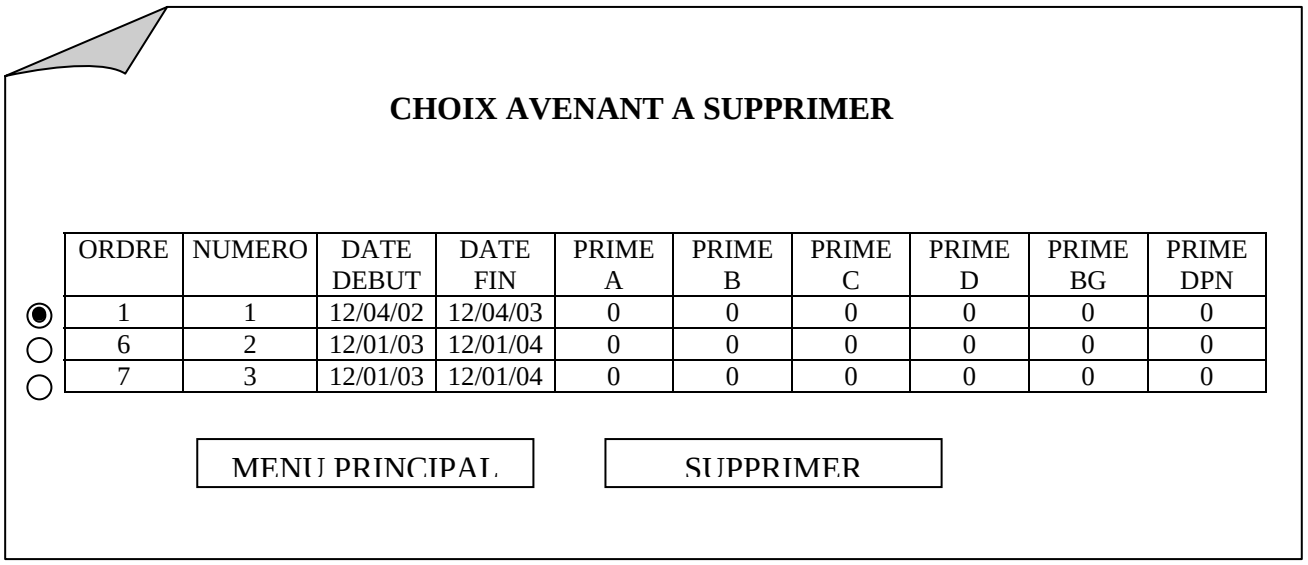
- Le commercial accède au 'menu avenant'.
- Le commercial appuie sur le bouton 'avenant particulier'.
- Le système appelle la page de saisie de critère police.
- Le commercial saisit la référence de la police et appuie sur le bouton 'afficher'.
- Le système affiche la liste des véhicules correspondant.
- Le commercial appuie sur le bouton 'suppression avenant'.
- Le système affiche la liste des avenants correspondant.
- Le commercial choisit un avenant et appuie sur le bouton 'supprimer'.
- Le système affiche l'avenant à supprimer.
- Le commercial appuie sur le bouton 'supprimer'.
- Le système affiche la page de confirmation de suppression.
- Le commercial appuie sur le bouton 'confirmer'.
- Le système appelle le menu avenant après suppression.
- Le commercial est ainsi informé de la suppression.

III.5.5.2 - MAQUETTE

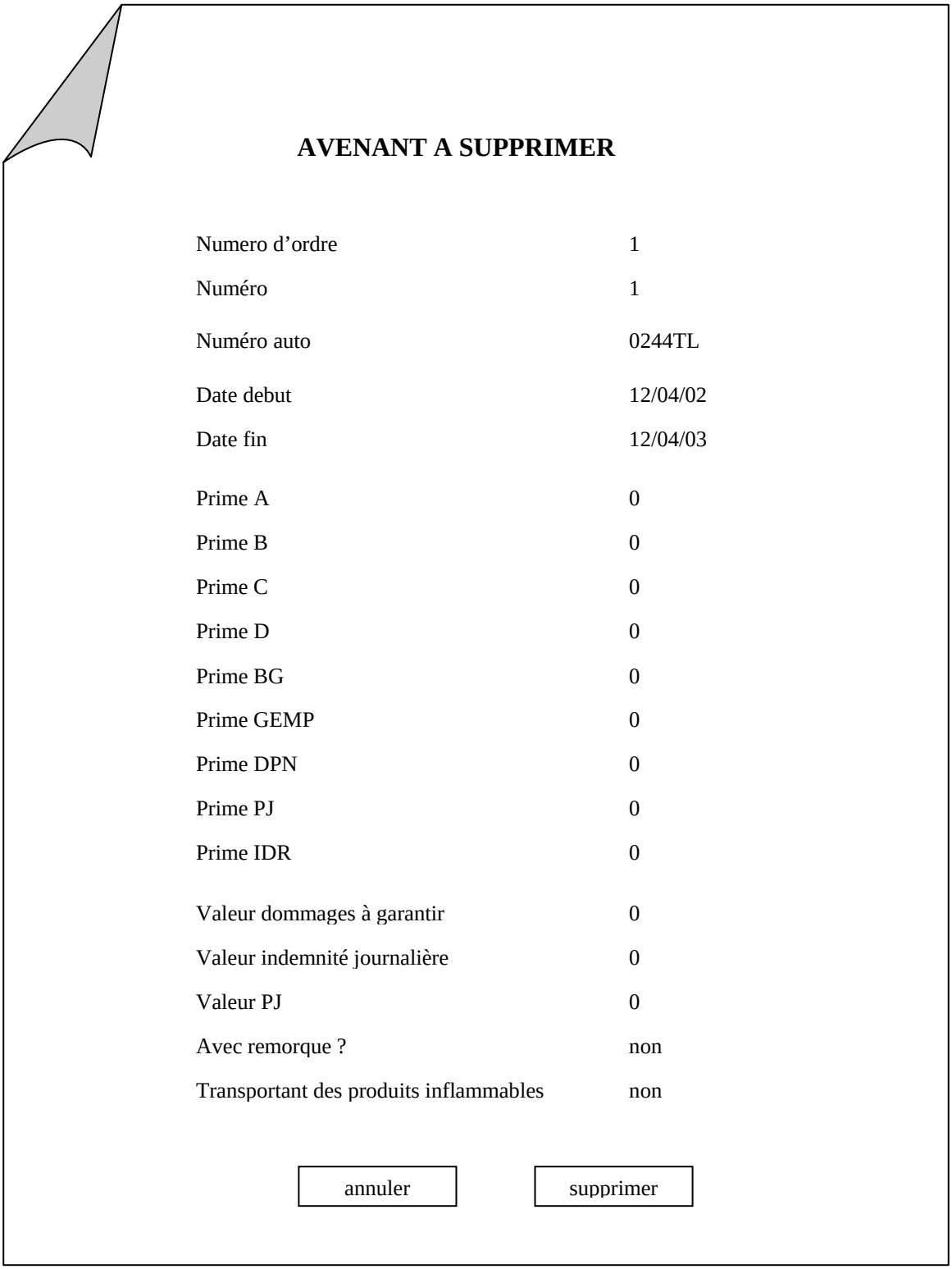
Avenantchoixauto.jsp



avenantremovelisteparauto.jsp



Avenantremoveaffiche.jsp & avenantremovepasse.jsp

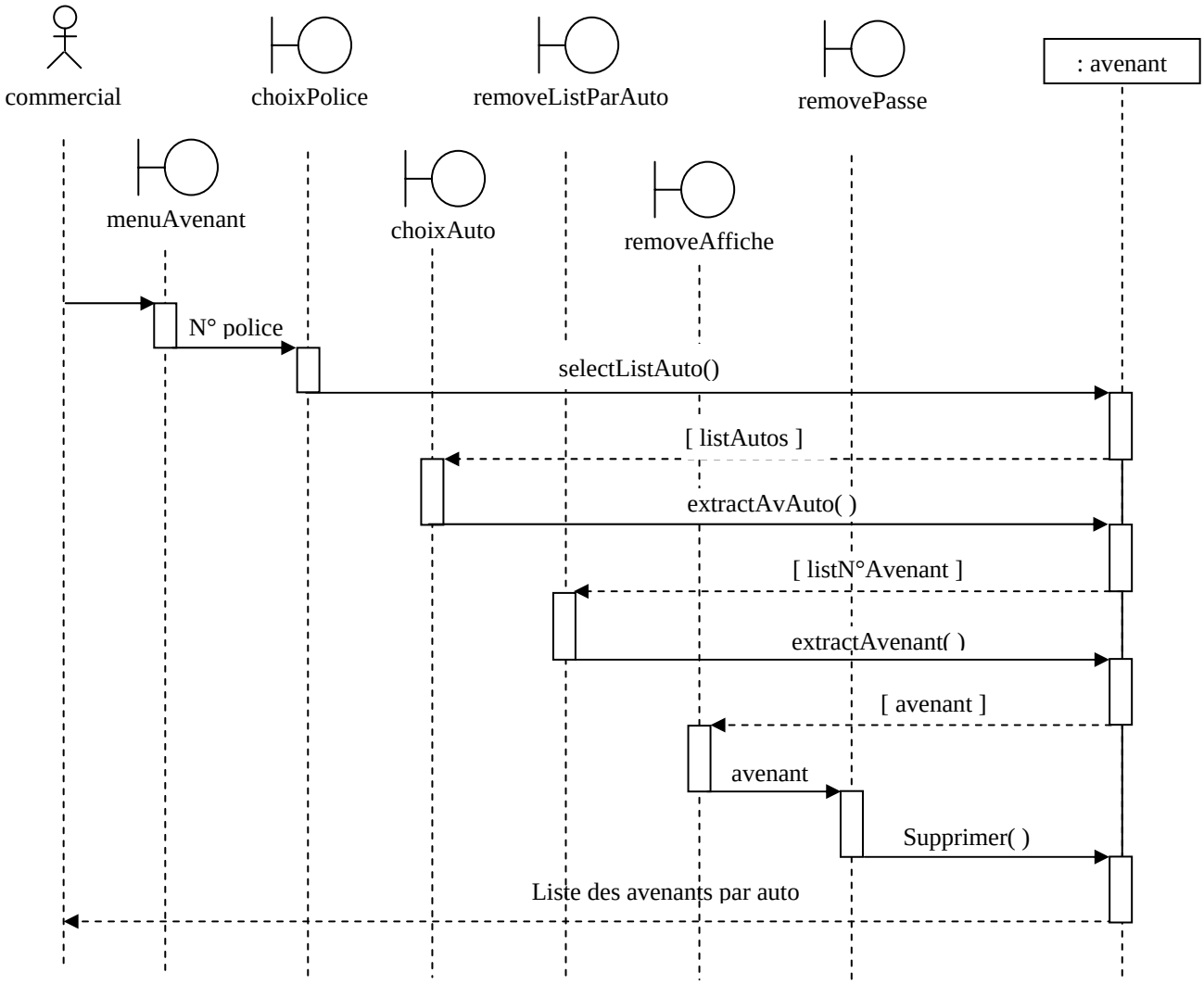


AVENANT A SUPPRIMER

Numero d'ordre	1
Numéro	1
Numéro auto	0244TL
Date debut	12/04/02
Date fin	12/04/03
Prime A	0
Prime B	0
Prime C	0
Prime D	0
Prime BG	0
Prime GEMP	0
Prime DPN	0
Prime PJ	0
Prime IDR	0
Valeur dommages à garantir	0
Valeur indemnité journalière	0
Valeur PJ	0
Avec remorque ?	non
Transportant des produits inflammables	non

annuler supprimer

III.5.5.3 - DIAGRAMME DE SEQUENCE



III.6 - CAS D'UTILISATION : CALCULER

Les calculs de prime sont effectués dans ce cas d'utilisation.

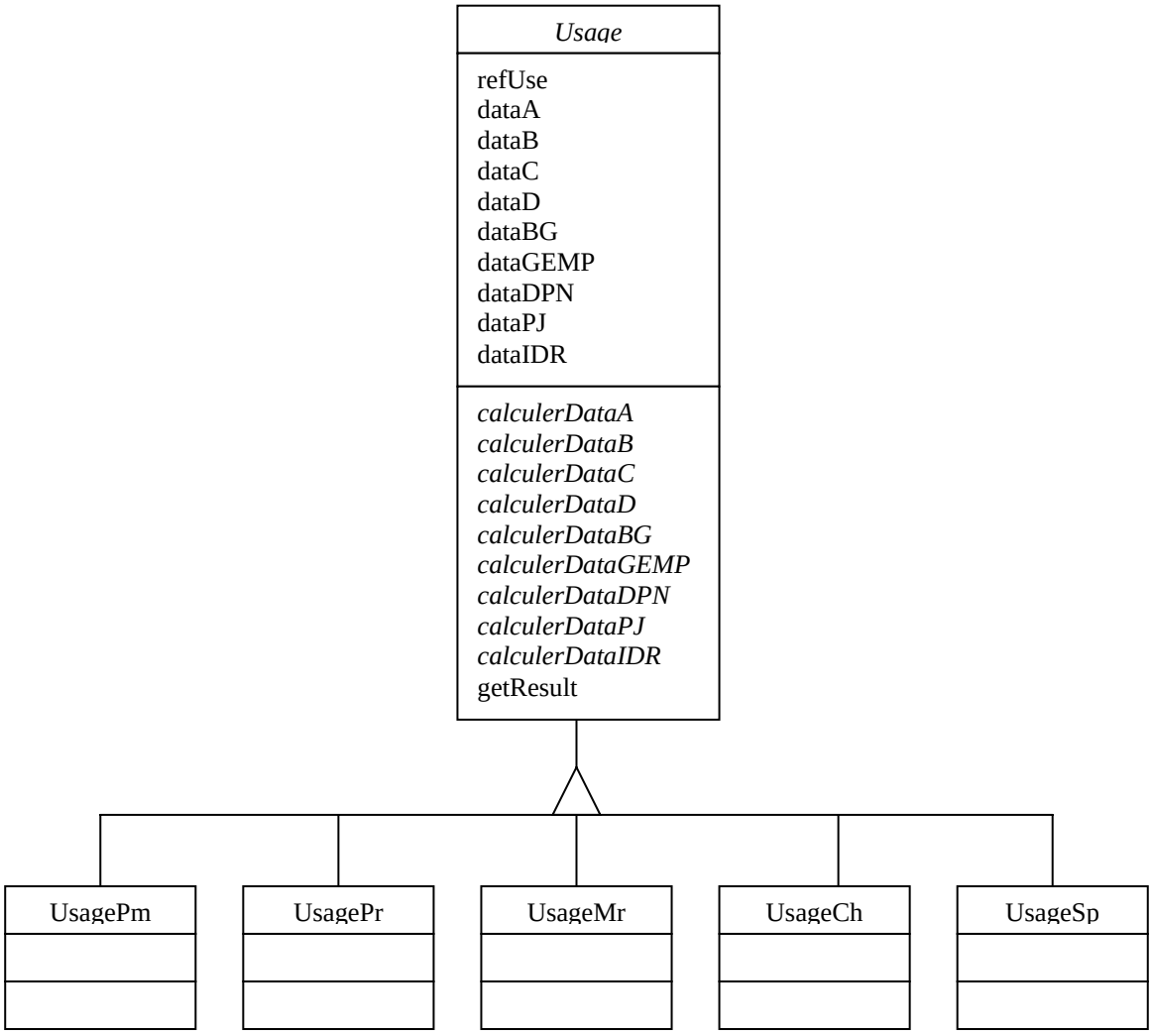
III.6.1 - SCENARIO

- La classe calcul instancie un objet calcul.
- L'objet calcul instancie à son tour un objet usage et lui passe les données relatives à l'auto.
- A l'appel de sa méthode calculer( ), il déclenche une itération sur tous les risques possibles et effectue des calculs.
- Lors de cet itération, des objets risque correspondant exactement aux informations passées à l'objet calcul sont instanciés pour permettre les calculs.

III.6.2 - DIAGRAMME DE CLASSE

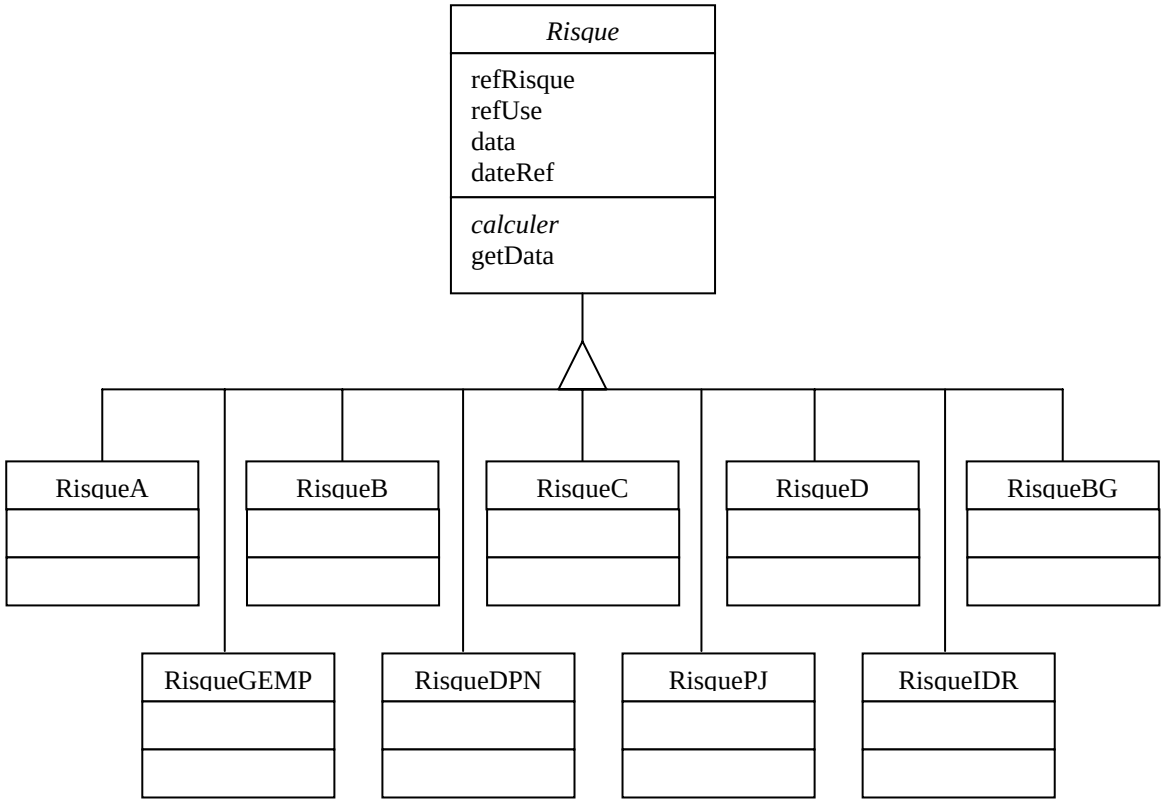
Le diagramme de classe de ce cas d'utilisation est visualisé en trois étape à cause de sa complexité. Cette complexité provient de la présence des héritages sur la classe usage et risque. Donc, pour mieux voir et ne pas surcharger le diagramme, il convient de séparer les diagrammes montrant seulement les héritages de ceux ne faisant intervenir que les classes abstraites (cas des classes usage et risque).

III.6.3 - DIAGRAMME DE CLASSE : USAGE



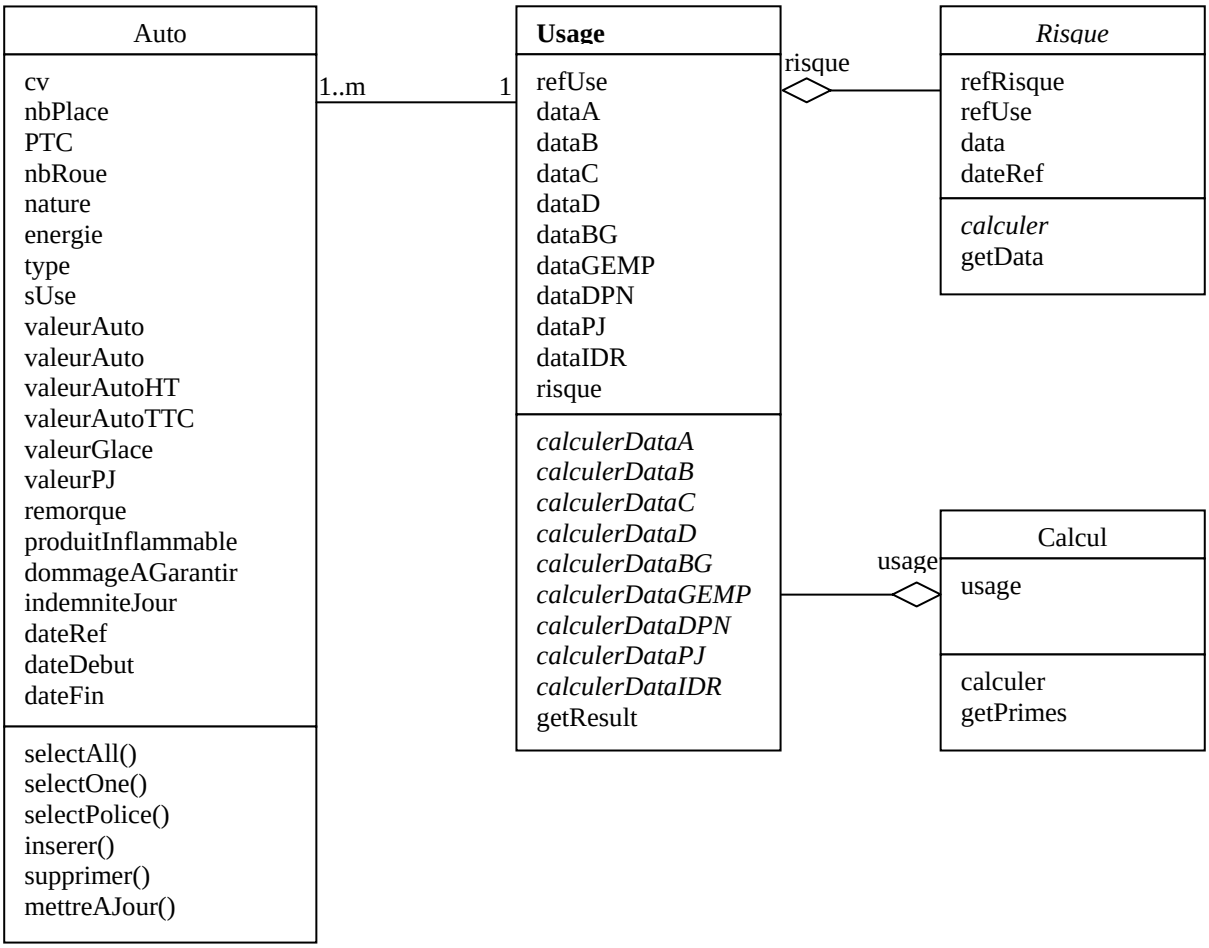
Ce diagramme nous montre une classe parent appelée usage. Elle est abstraite. Elle contient neuf méthodes abstraites pour calculer les primes relatives à chaque risque. A partir de cette classe abstraite, on obtient des classes dérivées représentant chacune la classe UsagePm pour le promenade, UsagePr pour le transport de personnel, UsageMr pour le transport de marchandise, UsageCh pour les véhicules de chantier et UsageSp pour les véhicules spéciaux.

III.6.4 - DIAGRAMME DE CLASSE : RISQUE



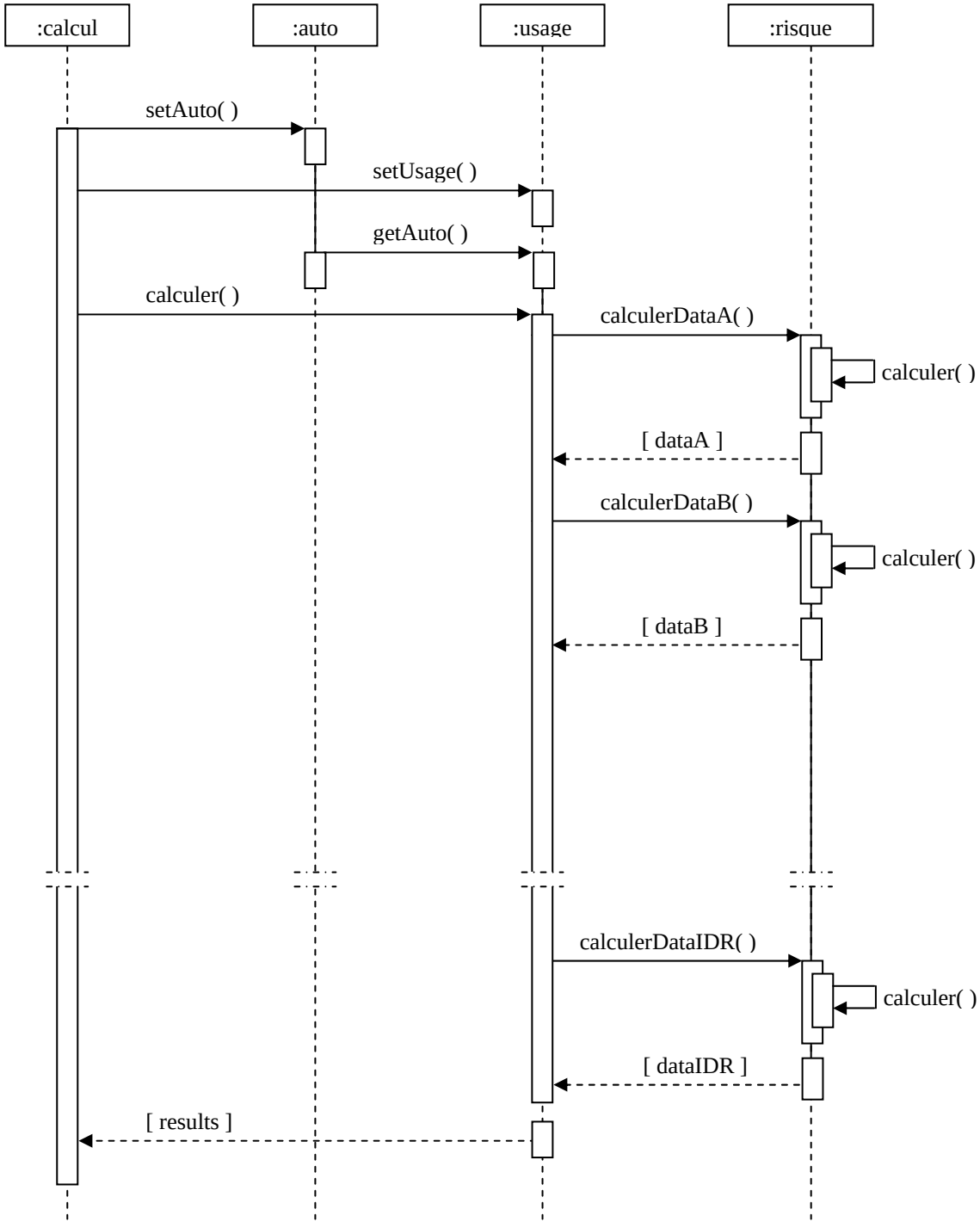
Ce diagramme nous montre une classe parent appelée risque. Elle est abstraite. Elle contient une seule méthode abstraite pour calculer les primes relatives à chaque risque. A partir de cette classe abstraite, on obtient des classes dérivées représentant chacune la classe RisqueA pour le risque A, RisqueB pour le risque B, RisqueC pour le risque C, RisqueD pour le risque D, RisqueBG pour le risque BG, RisqueGEMP pour le risque GEMP, RisqueDPN pour le risque DPN, RisquePJ pour le risque PJ et RisqueIDR pour le risque IDR.

III.6.5 - DIAGRAMME DE CLASSE D'ENSEMBLE



Ce diagramme de classe nous montre les différentes classes constituant le cas d'utilisation calculer. Nous n'avons donné que la classe abstraite pour la classe usage et risque pour des raisons citées supra.

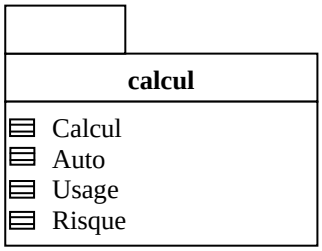
III.6.7 - DIAGRAMME DE SEQUENCE



III.7 - DIAGRAMME DE COMPOSANTS

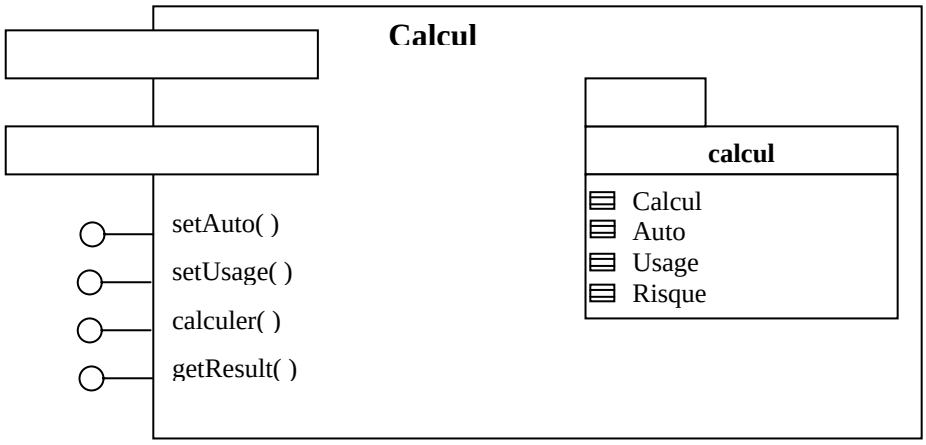
III.7.1 - EMPAQUETAGE

Les classes Calcul, Auto, Usage et Risque sont regroupées logiquement dans un paquet appelé calcul. Ce regroupement permet non seulement de simplifier la gestion des classes mais aussi de faciliter la compréhension.



III.7.2 - COMPOSANT

Les classes constituant le cas d'utilisation calculer sont regroupées dans un composant appelé calcul. Ce composant est un EJB et devra être localisé dans un EJB container.



III.8 - LES MODULES

III.8.1 - LE MODULE DE STOCKAGE

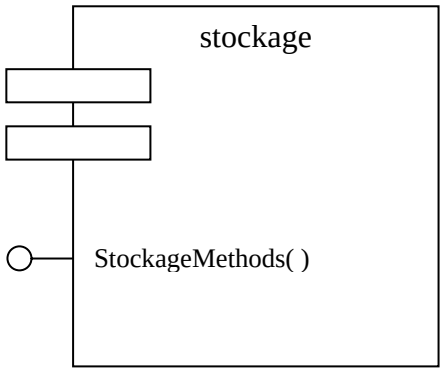
Le module de stockage doit être accessible par plusieurs composants de l'application. On veut isoler toutes les interactions avec les bases de données en une seule module. Ce module cache les méthodes d'accès qu'on utilise pour ajouter, supprimer, mettre à jour et consulter les informations sur les bases de données.

Le but est de donner un seul point d'accès à la base de données. En fait, les deux autres modules n'ont pas besoin de savoir s'il y a des bases de données, simplement, ils lancent des requêtes ou délivrent des données au module de stockage. Ainsi, le module de stockage est-il une application indépendante.

Cette conception a été faite avec des classes Java conçues pour s'occuper les requêtes d'accès à la base de données. Ses codes sont indépendants des deux autres modules mais son interface fournit toutes les méthodes nécessaires pour interagir avec la base de données.

Ce module est composé de deux classes :
- PrimeFac est dédié aux paramètres
- Assurauto est dédié aux autres données

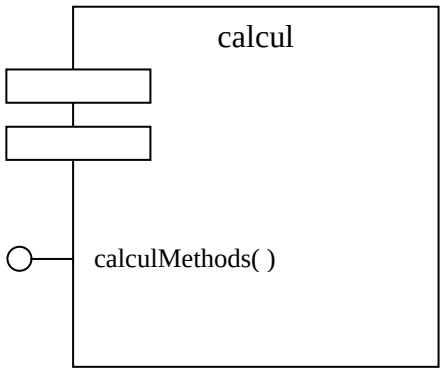
Ce module est représenté par le diagramme de composant ci dessous dont le diagramme de classe est donné plus loin. Les deux composants ci dessous sont équivalents. Ils ne représentent qu'un seul module à la différence près que l'un est détaillé et l'autre est simplifié pour être admis par équivalence.



III.8.2 - LE MODULE DE CALCUL

Le module de calcul effectue les calculs des primes demandés par l'utilisateur. Il prend en compte les données fournies par l'utilisateur et effectue automatiquement des extractions de données nécessaires au calcul. Après le calcul, il renvoie les résultats.

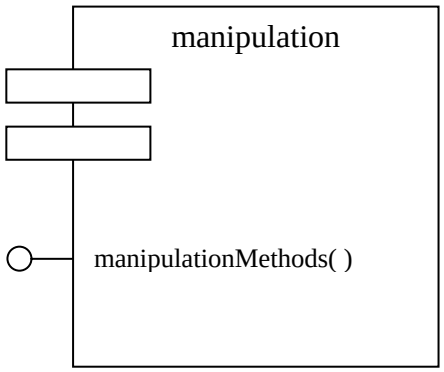
Ce module est déjà traité dans la partie UML calcul. Nous ne reprenons ici que le diagramme de composant pour pouvoir donner un aperçu de ce module et de son équivalence.



III.8.3 - MODULE MANIPULATION

Le module manipulation permet la sélection, l'insertion, la mise à jour, et la suppression des données de la base de données. Il interagit directement avec le module web et cache de ce module les accès aux bases de données via le module stockage et le calcul via le module calcul.

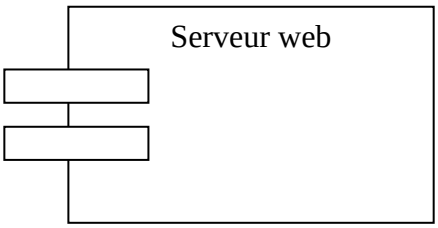
Le diagramme de composant suivant nous montre ce module.



III.8.4 - MODULE WEB

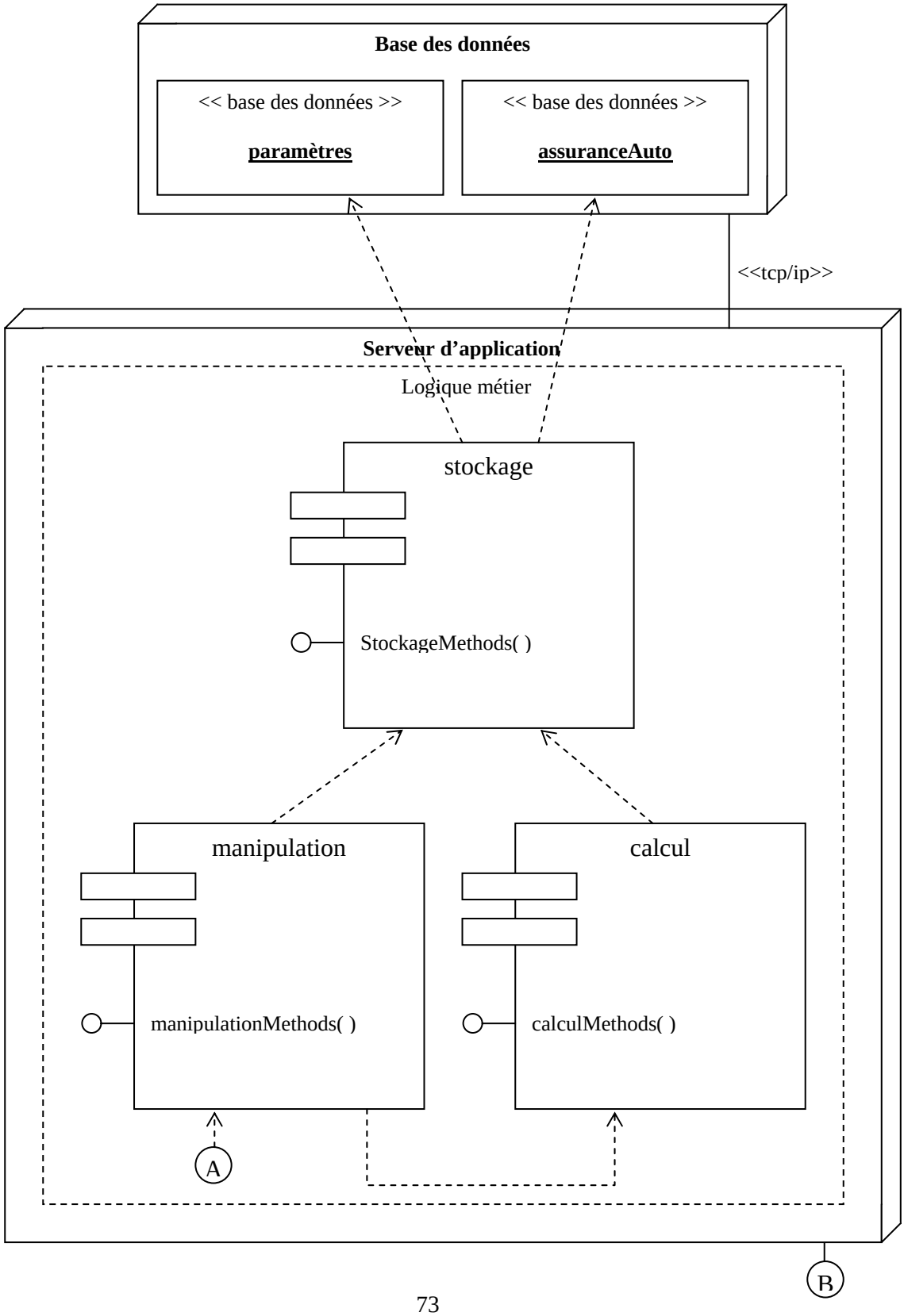
Le module web assure l'interface avec les utilisateurs. Il contient tous les pages JSP, HTML et les classes utilitaires. Il est la raison du projet. Donc c'est ici que sont centralisés les interfaces clients.

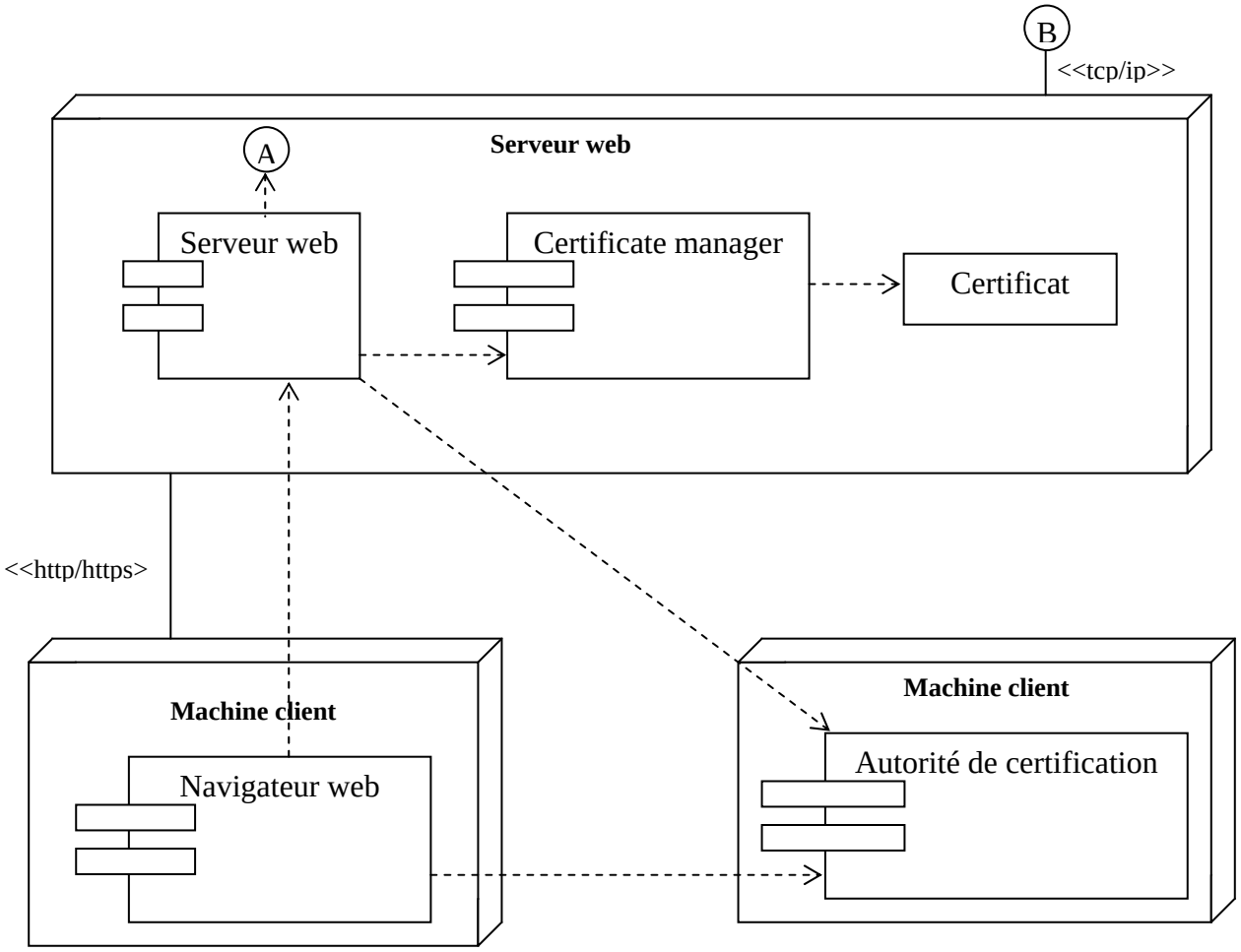
Le diagramme de composant suivant nous montre ce module.



III.9 - LE DEPLOIEMENT

Le diagramme de déploiement suivant nous montre la relation entre l'application et les composants matériels impliqués dans une situation réelle.




B
<<tcp/ip>>
Serveur web
A
Serveur web
Certificate manager
Certificat
<<http/https>
Machine client
Navigateur web
Machine client
Autorité de certification

• CHAPITRE IV - BASE DES DONNEES

IV.1 - INTRODUCTION

Une base de données est une entité dans laquelle il est possible de stocker des données de façon structurée. Ces données doivent pouvoir être utilisées par des programmes, par des utilisateurs différents. Ainsi, la notion de base de données est généralement couplée à celle de réseau, afin de pouvoir mettre en commun ces informations, d'où le nom de **base**. On parle généralement de système d'information pour désigner toute la structure regroupant les moyens mis en place pour pouvoir partager des données.

Une base de données permet de mettre des données à la disposition d'utilisateurs pour une consultation, une saisie ou bien une mise à jour, tout en s'assurant des droits accordés à ces derniers. Cela est d'autant plus utile que les données informatiques sont de plus en plus nombreuses.

Une base de données peut être locale, c'est-à-dire utilisable sur une machine par un utilisateur, ou bien répartie, c'est-à-dire que les informations sont stockées sur des machines distantes et accessibles par réseau.

L'avantage majeur des bases de données est la possibilité de pouvoir être accédées par plusieurs utilisateurs simultanément.

La section Base de données première partie se termine à l'obtention du modèle logique de données et à la mise en place du choix du SGBD utilisé.

IV.2 - CONCEPTION

IV.2.1 - METHODE UTILISEE

La conception nécessite une réflexion sur l'ensemble de l'organisation que l'on doit mettre en place. La phase de conception nécessite des méthodes permettant de mettre en place un modèle sur lequel on va s'appuyer. La modélisation consiste à créer une représentation virtuelle d'une réalité de telle façon à faire ressortir les points auxquels on s'intéresse.

Dans la conception de cette base, on utilise la méthode MERISE qui facilite beaucoup la représentation de la modélisation.

La méthode MERISE est basée sur la séparation des données et des traitements à effectuer en plusieurs modèles conceptuels et physiques. La séparation des données et des traitements assure une longévité du modèle.

IV.2.2 - DEMARCHE A SUIVRE

- Une première réunion détermine la ligne directive de chaque tâche surtout la mise en place des entités impliquées dans la création de la base.
- Elaboration d'une esquisse du modèle conceptuel de donnée(MCD) à partir de l'organigramme obtenu à partir de l'analyse des besoins

IV.3 - REALISATION

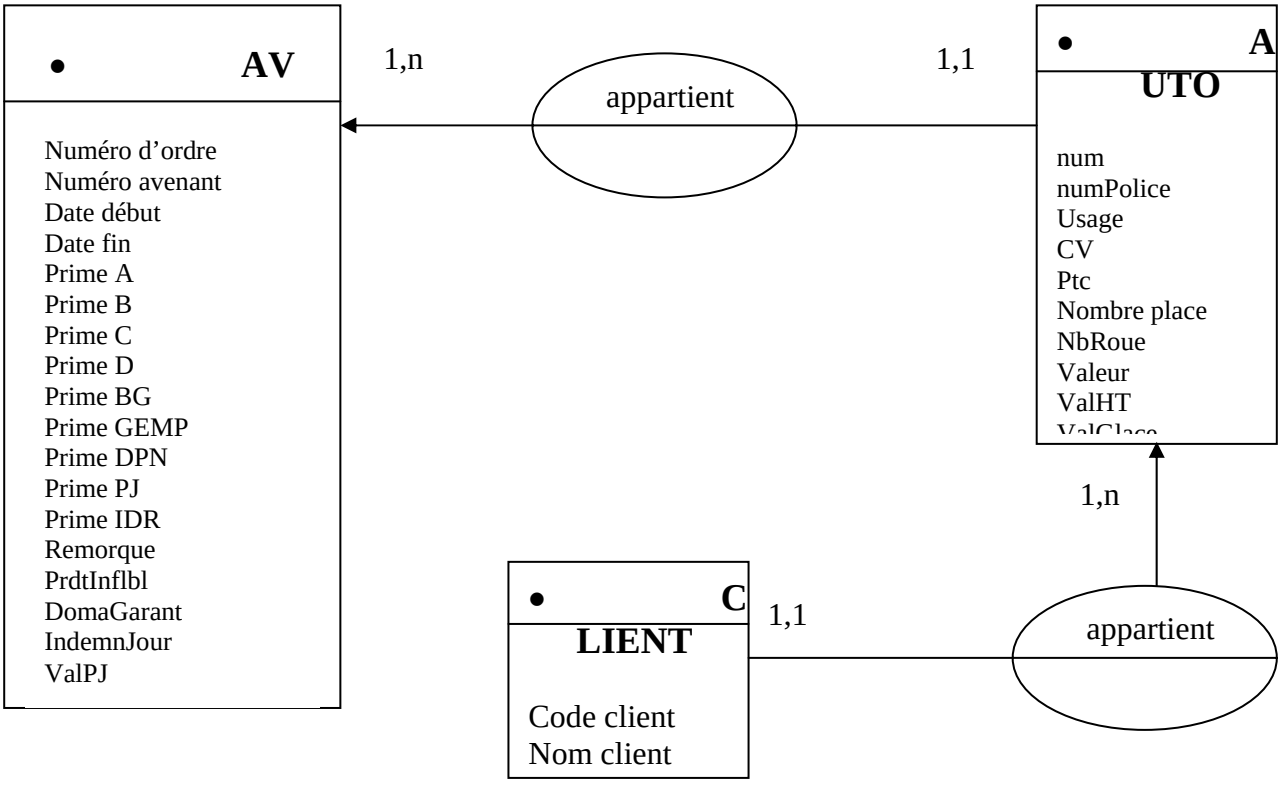
IV.3.1 - Mise en place de l'environnement technique :

- Un micro-ordinateur type Personal Computer travaillant sur un système d'exploitation Windows XP avec les logiciels nécessaires.
- Microsoft Office (Pour le document écrit)
- Win'Design 4.0 (Pour l'élaboration du MCD)

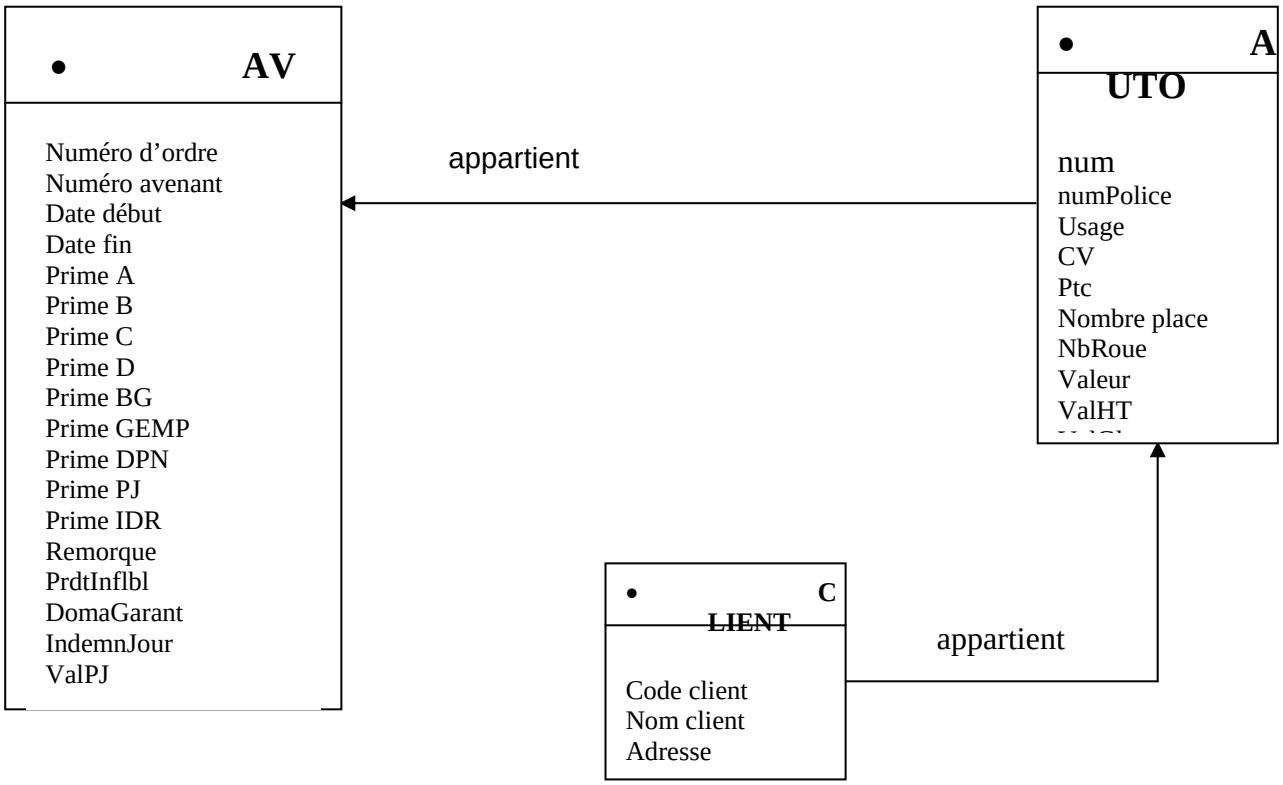
IV.3.2 - Elaboration du MCD et MLD sous Win'Design :

- Mise en place de tout les entités ainsi que leurs différentes propriétés.
- Création de la relation entre les entités, avec ses cardinalités.
- Génération du MLD (Modèle Logique de Donnée)
- Choix du SGBD utilisé pour gérer la base. Dans ce projet on utilise Microsoft SQL server 2000
- Génération du script nécessaire à l'analyse de requête sous SQL server.

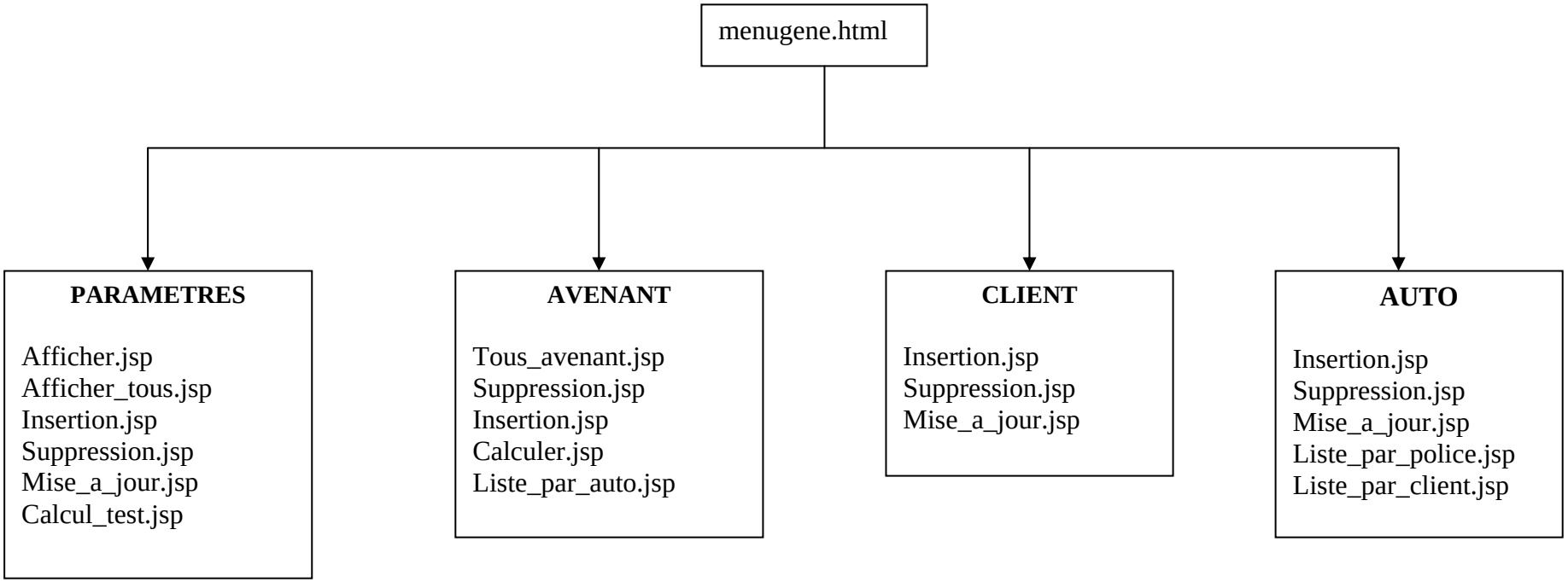
IV.3.3 - MODELES CONCEPTUELS DES DONNEES



IV.3.4 - MODELES LOGIQUE DES DONNEES



- **CHAPITRE V - PLAN DU SITE**



PARTIE III - PHASE DE CLOTURE DU PROJET

Cette phase regroupe le bilan de projet et les livrables.

• CHAPITRE I - BILAN DE PROJET

C'est dans le bilan de projet qu'on capitalise l'expérience acquise sur le projet pour les futures évolutions du système ou les nouveaux systèmes.

Le bilan de projet a pour objet l'analyse objective des succès ou des difficultés rencontrés et l'expression de toute disposition susceptible d'être retenue dans le futur pour l'amélioration de la qualité des produits et le déroulement du projet.

I.1 - BILAN TECHNIQUE

Le bilan technique consiste à :

- Evaluer l'adéquation des moyens mis en œuvre(choix techniques, efficacité de l'équipe projet)
- Faire la synthèse des problèmes rencontrés et des solutions mises en œuvre
- Recenser les suggestions et améliorations à apporter

I.1.1 - ADEQUATION DES MOYENS UTILISES

Le choix des moyens mis en œuvre reste adéquate au projet.

L'équipe de projet n'est constituée que d'une seule personne dont l'évaluation de l'efficacité reste très relative selon l'appréciateur mais à en juger par les résultats, l'équipe est très efficace en terme de performance et de rapidité.

I.1.2 - SYNTHESE DES PROBLEMES RENCONTRES ET SOLUTIONS MISES EN ŒUVRE

Lors de la réalisation du projet, aucun problème majeur n'a été rencontré.

Toutefois, des axes d'amélioration sont présentées ci-dessous quant à la performance de l'application :

- Modification de la relation entre une auto et une police permettant à une auto d'avoir différentes polices
- Ajout de possibilité de cession de véhicule
- Utilisation du cryptage des données avec SSL 128 bits pour la sécurité des transferts de données
- Contrôle de saisie
- Authentification du serveur et des clients par certificat
- Modification de l'algorithme de calcul de prime
- Contrôle d'intégrité des séries de pages demandées par les utilisateurs
- Personnalisation des pages en fonction du catégorie de l'utilisateur
- Amélioration de la structure du programme pour pouvoir recevoir des mises à jour et des extensions
- Amélioration d'accès à la base de données
- Mise en place des contrôles d'intégrité de la base de données
- Mise en place d'un programme de mise en cohérence des données
- Mise en place d'une base de rôle des utilisateurs
- Mise en place de l'historique des mises à jour effectuées

I.2 - BILAN DE L'ORGANISATION

Le problème d'organisation ne se pose pas puisqu'il n'y a qu'une seule personne qui assure toutes les fonctions. Il en va de même des circuits d'information. Sauf pour la non disponibilité des équipes issues de la société.

Pour un projet réel, les améliorations suivantes sont nécessaires :

- La constitution des équipes sont à officialiser par une lettre officielle,
- La société doit s'efforcer de rendre disponibles ses représentants,
- Les documents nécessaires (plan d'assurance et contrôle qualité, compte rendu des réunions, charte de projet, tableau de bord, …) devront être mis en place et exploités

I.3 - BILAN ECONOMIQUE

Le bilan économique consiste à :

- Dresser un bilan économique du projet et expliciter les causes de dépassement éventuel
- Evaluer le reste à faire éventuel

Le projet n'a pas dépassé les prévisions.

- Les outils utilisés sont de version évaluation pour l'éditeur d'application et le serveur d'application.
- La base de données relationnelle utilisée est la propriété de la société
- Le temps imparti est respecté

Pour concrétiser le projet et le rendre accessible de l'extérieur, les tâches suivantes sont encore à faire. La réalisation est évaluée à trois mois environ:

- Acquisition d'un nom de domaine
- Acquisition des certificats pour les utilisateurs et pour le serveur
- Extension de la structure de la base de données
- Amélioration de l'application (algorithme et structure)
- Déploiement sur plusieurs machines

I.4 - BILAN QUALITE

Le bilan de la qualité consiste à :

- Identifier les risques de non qualité ou les difficultés prévisibles en exploitation et maintenance (performances, sécurité, probabilité...).
- Proposer des orientations pour l'amélioration du référentiel qualité/méthodes.

I.4.1 - RISQUE DE NON QUALITE ET DIFFICULTES PREVISIBLES

Vu la puissance et la performance de la technologie, aucune risque de non qualité peut être rencontrée à l'exploitation.

Les difficultés prévisibles sont liées à la connaissance de l'outil. Mais l'exploitation seulement ne requiert pas beaucoup de compétences particulières. C'est un outil relativement simple et facile à utiliser même pour un profane.

I.4.2 - ORIENTATIONS POUR L'AMELIORATION DU REFERENTIEL QUALITE/METHODE

La liste des améliorations suivantes peut être avancée:

- Mettre en place les procédures de maintenance
- Les interventions doivent faire l'objet d'une analyse détaillée et archivée sous format électronique
- Des procédures écrites doivent être mises en place pour éviter de dépendre de la compétence d'une seule personne

I.5 - LIVRABLES

- Identifier les documents et les éléments logiciels qui doivent être archivés
- Procéder à l'archivage sous format électronique sur CD –ROM

Dans le cas du mémoire, on a les livrables suivants

- 01 lettre avec avis favorable du directeur du mémoire corrigé.
- 04 exemplaires de mémoire corrigé (pour les diverses bibliothèques)
- 01 CD comprenant le traitement de texte sous Word et la simulation (D.M.)
- 01 CD comprenant le traitement de texte sous Word et la simulation (Labo NTIC)

CONCLUSION

Ce travail montre que la technologie J2EE est la solution Java de l'implémentation d'une architecture multitiers. Dans ce projet, elle a permis [1]:

- De rendre l'application multi plateforme
- De simplifier la maintenance soft
- De s'affranchir des problèmes posés par les licences
- D'éliminer la dépendance aux éditeurs
- De mettre au point un système fiable et disponible à 99,99%
- De supporter un nombre de connexions relativement illimité
- De donner la possibilité de récupération de session en cas d'une panne d'un serveur
- De repartir les charges au niveau des serveurs
- De centraliser les données et les applications dans un endroit sécurisé et protégé
- De ne nécessiter qu'une faible bande passante (64 k est largement suffisant)
- De réduire le budget d'extension à un niveau très bas

Le projet est réalisé à des fins de démonstrations et de test. Les tests ont permis de démontrer sa facilité d'emploi, sa souplesse et sa performance. La société est tellement intéressée qu'elle a fini par former ses informaticiens à Java.

Si performante soit-elle, cette technologie a des inconvénients comme tant d'autres effets néfastes de l'automatisation, les conséquences sociales :

- La compression de personnel
- L'élimination des avantages des personnels de maintenance

Il y a aussi l'insuffisance des infrastructures de la télécommunication de chez nous. Les coûts de la communication restent encore très élevés.

Notons toutefois que ces inconvénients ne veulent pas dire que l'architecture multitiers est pire qu'un client serveur, mais elle ne nécessite pas beaucoup de maintenance ni de personnel technique et d'exploitation.

Pour les futures extensions, avec cette technologie, la société d'assurance Aro peut atteindre n'importe quel coin du monde sans aucun coût d'extension. Et avec l 'émergence du commerce électronique, elle permet aussi de laisser à chaque client le soin de gérer son propre compte, et le payement peut se faire avec n'importe quel moyen de paiement électronique.

Avec cette technologie, beaucoup de matériels et d'infrastructures utilisés aujourd'hui s'avéreront inutiles. Non seulement elle épargne beaucoup d'argent, mais les bénéfices seront énormes sans parler des avantages indirects.

ANNEXES

ANNEXE1 : COMPARATIFS DE PRIX ENTRE LOGICIELS LIBRES ET COMMERCIAUX

LOGICIELS COMMERCIAUX

	DEVELOPPEUR	**SERVEUR PAR PROCESSEUR**
BASE DE DONNEES : ORACLE	$US 1.000	$US 40.000
SERVEUR D'APPLICATION : IBM WEBSPHERE	$US 2.800	$US 23.000
UTILITAIRE ACCES BASE : COCOBASE	$US 6.000	$US 0
IDE : BORLAND JBUILDER	$US 3.500	$US 0
TOTAL	**$US 13.300**	**$US 63.000**

LOGICIELS LIBRES

	DEVELOPPEUR	**SERVEUR PAR PROCESSEUR**
BASE DE DONNEES : POSTGRESQL	$US 0	$US 0
SERVEUR D'APPLICATION : JBOSS	$US 0	$US 0
UTILITAIRE ACCES BASE : HIBERNATE	$US 0	$US 0
IDE : ECLIPSE	$US 0	$US 0
TOTAL	**$US 0**	**$US 0**

RECAPITULATIF

	LIBRES	**COMMERCIAUX**
DEVELOPPEUR	$US 0	$US 13.300
SERVEUR PAR PROCESSEUR (nb = 02)	$US 0	$US 126.000
TOTAL	**$US 0**	**$US 139.000**

1 $ US = 10.000 FMG (2.000 Ariary)

ANNEXE2 : COMPARATIFS DE COUT ENTRE SYSTEME CLIENT-SERVEUR ET MULTITIERS EN LOCAL

MATERIELS NECESSAIRES

	CLIENT SERVEUR	MULTITIERS
SERVEUR	PIV 2.8 (x 2)	PVI 2.8 (x 2)
CLIENT	PIII 800	PII 250

ORDRE DE PRIX DES MATERIELS ET SERVICE EN FMG

	UNITE	PU
PIV 2.8	unité	6.000.000
PIII 800	unité	3.500.000
PII 250	unité	1.500.000
Intervention	Heure	15.000

INSTALLATION LOCALE MONO CLIENT EN FMG

	CLIENT SERVEUR	MULTITIERS
Serveur	12.000.000	12.000.000
Client	3.500.000	1.500.000
TOTAL	15.500.000	13.500.000

INSTALLATION LOCALE 10 CLIENTS EN FMG

	QUANTITE	**CLIENT SERVEUR**	**QUANTITE**	**MULTITIERS**
Serveur	02 PIV 2.8	12.000.000	02 PIV 2.8	12.000.000
Client	10 PIII 800	35.000.000	10 PII 250	15.000.000
Réseau	Forfait	1.000.000	Forfait	1.000.000
Deployement serveur	01 j (08 h)	120.000	01 j (08 h)	120.000
Déployement client	01 h x 10	150.000	01 h x 0	0
Mise à jour client	01 h x 10	150.000	01 h x 0	0
TOTAL		**48.420.000**		**28.120.000**

1 Ariary = 5 FMG

ANNEXE3 : COMPARATIFS DE COUT ENTRE SYSTEME CLIENT-SERVEUR ET MULTITIERS EN RESEAU ETENDU

SERVICES UTILISES

On utilise la ligne spécialisée de TELMA.

- Prix pour 64 k vaut 1.500.000 Fmg par mois (300.000 Ariary)

INSTALLATION POUR UNE ZONE SERVEUR ET UNE ZONE CLIENT EN FMG

	QUANTITE	CLIENT SERVEUR	QUANTITE	MULTITIERS
Ligne spécilaisée serveur	16 x (x 64 k)	24.000.000	01 (x 64 k)	1.500.000
Ligne spécialisée client	16 x (x 64 k)	24.000.000	01 (x 64 k)	1.500.000
TOTAL		**48.000.000**		**3.000.000**

INSTALLATION POUR UNE ZONE SERVEUR ET DIX ZONES CLIENTS EN FMG

	QUANTITE	CLIENT SERVEUR	QUANTITE	MULTITIERS
Ligne spécilaisée serveur	10 x 16 x (x 64 k)	240.000.000	02 (x 64 k)	3.000.000
Ligne spécialisée client zone 1	10 x 16 x (x 64 k)	240.000.000	10 (x 64 k)	15.000.000
TOTAL		**96.000.000**		**18.000.000**

1 Ariary = 5 FMG

ANNEXE4 : CADRE LOGIQUE

1 - INTRODUCTION

La méthode du cadre logique fut tout d'abord élaborée en 1969 par Practical Concepts Inc., pour le compte de la U.S. Agency for International Development (USAID). Par la suite, la communauté internationale donatrice l'a adoptée et adaptée partout dans le monde. Elle est en effet maintenant utilisée sur une base participative pour planifier des projets et comme outil d'analyse aux fins d'approbation de projets ou comme cadre de suivi et d'évaluation. Au fil des ans, la méthode du cadre logique a été utilisée à des fins diverses, ce qui constitue un témoignage de sa valeur en tant qu'outil de gestion [4, 5].

2 - PROCESSUS DU CADRE LOGIQUE

La participation des intervenants est essentielle quand on utilise cette méthodologie aux fins de la conception et de la planification des projets de la méthode du cadre logique parce qu'il permet d'établir le niveau de compréhension et, si possible, celui du consensus. Pour les intervenants, la meilleure utilisation de la méthode du cadre logique consiste à :

- établir des objectifs stratégiques;
- définir une chaîne de résultats escomptés;
- identifier les hypothèses et les risques sous-jacents;
- choisir les indicateurs de rendement pour mesurer le progrès vers les résultats escomptés.

3 - STRUCTURE

La MCL nécessite la préparation d'une matrice à 12 cellules comportant trois rangées et quatre colonnes le « cadre logique ». Un cadre logique orienté vers les résultats décrit le rapport logique qui existe entre les composantes stratégiques d'un projet, les résultats prévus, les indicateurs de rendement, les hypothèses et les risques au niveau conceptuel. Le cadre logique ne doit pas être utilisé pour illustrer la façon dont le projet est mis en oeuvre.

4 - REMARQUE

Bien qu'un cadre logique doit être aussi complet que possible, il ne doit pas être trop détaillé. Il doit demeurer un moyen de faciliter la communication et une compréhension commune du projet par les intervenants. Il ne doit pas comporter d'explication exhaustive comprenant tous les détails techniques. Ils doivent être présentés séparément (p.ex., le diagramme de gant). Ils représenteront un renvoi aux aspects correspondants du cadre logique. Pour que ce dernier demeure utile tout au long du projet, il doit constituer une description sommaire du projet et, en conséquence, doit être révisé à mesure que les intervenants conviennent des changements à apporter à la conception du projet.

ANNEXE5 : SECURITE, CRYPTAGE ET MOS-SSL

Le module mod_ssl fournit une puissante cryptographie pour le serveur web Apache via les protocoles Secure Socket Layer (SSL v2/v3) et le Transport Layer Security (TLS v1).

Le package mod_ssl était créé en avril 1998 par Ralf S. Engelschall et était originalement dérivé du package Apache-SSL développé par Ben Laurie. Il reste sous licence BSD-style qui est équivalent à la licence utilisée par The Apache Group pour le serveur web Apache lui-même. Ce qui veut dire qu'il peut être utilisé librement à des fins commerciaux ou non commerciaux du moment qu'on retient les notices copyright de l'auteur [17, 18, 21, 22].

1 - ALGORITHMES CRYPTOGRAPHIQUES

C'est une technique permettant de transformer un message clair en message crypté. Une fois sous cette forme, le message peut être reconstitué par l'utilisation d'une clé secrète. Sans cette clé le message est inutile. Un bon algorithme cryptographique fait le décodage du message original très difficile.

Il y a deux catégories d'algorithmes cryptographiques : symétrique et asymétrique.

- La cryptographie symétrique requiert le destinateur et le destinataire de partager la clé, une information secrète qu'on utilise pour crypter et décrypter le message.
- La cryptographie asymétrique résout le problème d'échange de clé en définissant un algorithme utilisant deux clés, et chacune d'elle peut être utilisée pour crypter le message. Si l'une est utilisée pour crypter, l'autre est utilisée pour décrypter le message. Ceci rend possible de recevoir des messages sécurisés en publiant simplement l'une des clé (la clé publique) et tenir secret l'autre clé (la clé privée)

N'importe qui peut crypter le message en utilisant la clé publique, mais seulement le propriétaire de la clé privée peut lire le message.

2 - MESSAGE DIGESTS

Les message digests sont utilisés pour créer des représentations courtes, de longueur fixe des messages longs, de longueur variable. Les algorithmes digest sont conçus pour produire un unique digest pour différents messages. Les messages digest sont conçus pour rendre difficile de déterminer le message à partir du digest, et impossible de trouver deux messages différents pour créer le même digest, éliminant ainsi la substitution d'un message en un autre en maintenant le même digest.

3 - SIGNATURES NUMERIQUES

Les signatures numériques sont créées pour crypter le digest d'un message, et d'autres informations avec la clé privée du destinateur. Ainsi, n'importe qui peut décrypter la signature en utilisant la clé publique, seulement le destinateur connaît la clé privée. Ce qui veut dire qu'il l'avait signé. Incluant le digest dans la signature veut aussi dire que la signature est pour ce message, elle assure aussi l'intégrité du message pour qu'aucune personne ne puisse changer le digest et le signer encore.

Pour préserver de l'interception et de la réutilisation de la signature par un intercepteur après, le signature contient un unique nombre de séquences.

4 - CERTIFICATS

Si chaque partie possède un certificat validant l'identité de l'autre correspondant, confirme la clé publique, et est signé par un agence de confiance, alors les deux sont assurés qu'ils communiquent avec la bonne personne. Un tel agence de confiance est appelé Certificate Authority, et les certificats sont utilisés pour authentification.

5 - SECURE SOCKETS LAYER (SSL)

Le protocole Secure Sockets Layer est une couche de protocole qui peut être placée entre la couche protocole réseau orientée connexion (exemple TCP/IP) et la couche protocole application (exemple http). SSL fournit une communication sécurisée entre le client et le serveur en permettant une authentification mutuelle, l'utilisation des signatures numériques pour l'intégrité, et le cryptage pour la confidentialité.

Le protocole est conçu pour supporter un large choix d'algorithmes spécifiques pour la cryptographie, digest, et la signature. Les choix sont négociés entre le client et le serveur au début de session du protocole.

6 - ETABLISSEMENT DE SESSION

La session SSL est établie en respectant le handshake sequence entre le client et le serveur.

Les éléments du handshake sequence utilisés par le client et le serveur sont :

- Négociation du Cipher Suite à utiliser pendant le transfert de données
- Etablissement et partage de clé de session entre le client et le serveur
- Optionnellement, authentification du serveur par le client
- Optionnellement, authentification du client par le serveur

7 - DIGEST FUNCTION

Le choix de digest function détermine comment un digest est créé à partir d'une unité d'enregistrement. SSL supporte les suivants :

- No digest (choix null)
- MD5, un hash 128 bits
- Secure Hash Algorithm (SHA-1), un hash 160 bits

Le message digest est utilisé pour créer un Message Authentication Code (MAC) qui est crypté avec le message pour fournir intégrité et pour prévenir contre des attaques.

8 - COMMUNICATION HTTP SECURISEE

Une des utilisations de SSL est de sécuriser la communication http web entre le navigateur et le serveur web. Ce cas n'exclue pas l'utilisation de http non sécurisé. La version sécurisée est http sur SSL (nommé HTTPS), mais avec une différence principale : il utilise l'url https plutôt que http et un différent port du serveur (par défaut 443).

ANNEXE6 : UML

L'UML est un langage graphique de modélisation objet permettant de spécifier, de construire, de visualiser et de décrire les détails d'un système logiciel. Il est issu de la fusion de plusieurs méthodes dont Booch et OMT et est adapté à la modélisation de tous types de systèmes [29, 30, 31].

UML est un langage : il comprend un vocabulaire et un ensemble de règles centrés sur la représentation conceptuelle et physique d'un système logiciel.

Ses domaines d'utilisation sont :
- Visualisation d'un système
- Spécification d'un système
- Construction d'un système
- Documentation d'un système

1 - LE MODELE CONCEPTUEL D'UML

Le modèle conceptuel d'UML comprend les notions de base génériques du langage. Il définit trois sortes de « briques » de base :
- Les éléments, qui sont les abstractions essentielles à un modèle,
- Les relations, qui constituent les liens entre ces éléments,
- Les diagrammes, qui regroupe des éléments et des liens au sein de divers ensembles.

2 - LES ELEMENTS

Il existe 4 types d'éléments dans UML :
- Les éléments structurels (classe, interface, collaboration,…)
- Les éléments comportementaux (interaction, automate à états finis)
- Les éléments de regroupement (package)
- Les éléments d 'annotation (note)

3 - LES RELATIONS

Il existe 4 types de relations dans UML :
- La dépendance
- L'association
- La généralisation
- La réalisation

4 - LES DIAGRAMMES

C'est la représentation graphique d'un ensemble d'éléments et de relations qui constituent un système.

UML définit 9 types de diagrammes divisés en 2 catégories :
- Diagrammes statiques (appelés aussi diagrammes structurels) : diagrammes de classes, d'objets, de composants, de déploiements et de cas d'utilisation.
- Diagrammes dynamiques (appelés aussi diagrammes comportementaux) : diagrammes d'activités, de séquences, d'états-transitions et de collaborations.

4.1 - DIAGRAMMES DE CLASSES (statique)

Les diagrammes de classes expriment de manière générale la structure statique d'un système, en terme de classes et de relations entre ces classes.

4.1.1 - LES PACKAGES

Le package est un mécanisme d'ordre général qui permet d'organiser les éléments en groupes.

Le package permet de définir des sous-systèmes formés d'éléments ayant entre eux une certaine logique.

Leurs caractéristiques sont les suivantes :

- Ils regroupent des éléments de modélisation selon des critères purement logiques
- Ils permettent d'encapsuler des éléments de modélisation par l'intermédiaire d'interfaces
- Ils permettent de structurer un système en catégories ou sous-systèmes
- Ils servent de briques réutilisables dans la conception d'un logiciel

Chaque package doit avoir un nom différent de celui des autres packages et peut être composé d'autres éléments, y compris d'autres packages.

4.1.2 - LES CLASSES

Une classe est la représentation d'un ensemble d 'éléments partageant les mêmes attributs, les mêmes opérations, les mêmes opérations , les mêmes relations et les mêmes sémantiques.

En programmation orientée objet, une classe définit une abstraction, un type abstrait qui permettra d'instancier des objets.

4.1.3 - LES CLASSES ABSTRAITES

Une classe abstraite est une classe ne pouvant être instanciée directement. Une telle classe sert de spécification pour des objets instances de ses sous-classes.

4.1.4 - LES INTERFACES

Une interface décrit un contrat d'une classe ou d'un composant sans en imposer l'implémentation.

Une interface ne décrit aucune structure ni aucune implémentation. Elle ne peut donc pas contenir d'attributs, ni de méthodes fournies sous la forme d'une implémentation.

4.2 - DIAGRAMMES D'OBJETS (statique)

Les diagrammes d'objets permettent de représenter les relations existant entre les différentes instances des classes à un instant donné de la vie du système.

Dans les diagrammes d'objets, les instances peuvent être anonymes ou nommées.

4.3 - DIAGRAMMES DES COMPOSANTS (statique)

Le composant est un élément physique qui représente une partie implémentée d'un système. Il peut être du code, un script, etc. Ils représentent un ensemble d'interfaces.

Les interfaces d'un composant sont des éléments définissant le comportement offert à d'autres composants.

Le diagramme de composants permet de décrire les composants et leurs dépendances dans leur environnement d'implémentation.

Le nœud est une ressource matérielle du système étudié.

4.4 - DIAGRAMMES DE DEPLOIEMENTS (statique)

Les diagrammes de déploiement permettent de montrer la disposition physique des matériels qui composent le système, ainsi que la répartition des composants sur ces matériels représentés par des nœuds.

Ce type de diagramme est utilisé principalement pour la modélisation de trois types de systèmes : les systèmes embarqués, les systèmes client/serveur et les systèmes totalement répartis.

Un diagramme de déploiement permet également de représenter les relations entre différents nœuds.

4.5 - DIAGRAMMES DE CAS D'UTILISATION

Les diagrammes de cas d'utilisation représentent les cas d'utilisation du système, les acteurs et les relations existant entre eux.

Les diagrammes de cas d'utilisation décrivent sous la forme d'actions et de réactions, le comportement d'un système du point de vue d'un utilisateur. Ils permettent de définir les limites du système et les relations entre le système et l'environnement. Ce type de diagramme intervient tout au long du cycle de développement, depuis le cahier des charges jusqu'à la fin de la réalisation.

L'acteur représente un rôle joué par une personne ou une chose qui interagit avec un système.

Le cas d'utilisation est une modélisation d'une fonctionnalité ou d'une classe.

Les cas d'utilisations se déterminent en observant et en précisant, acteur par acteur, les séquences d'interaction du point de vue de l'utilisateur. Ils se décrivent en terme d'informations inchangées et d'étapes dans la manière d'utiliser le système. Un cas d'utilisation regroupe une famille de scénarios d'utilisation selon un critère fonctionnel. Il décrit les interactions potentielles, sans entrer dans les détails de l'implémentation.

4.6 - DIAGRAMMES DE SEQUENCE (dynamique)

Le diagramme de séquence montre des interaction entre objets selon un point de vue temporel. Ce type de diagramme sert à modéliser les aspects dynamiques des systèmes temps et des scénarios complexes mettant en œuvre peu d'objets.

Dans ce type de diagramme, l'accent est mis sur la chronologie des envois des messages. La représentation se concentre sur l'expression des interactions et non pas sur l'état ou le contexte des objets. Ce type de diagramme est usuellement utilisé pour illustrer les diagrammes de cas d'utilisation.

Dans un diagramme de séquence, les objets sont associés à une ligne de vie. La dimension verticale de celle ci représente l'écoulement du temps. Notons que la disposition des objets sur l'axe horizontal n'est pas importante.

4.7 - DIAGRAMMES DE COLLABORATIONS (dynamique)

Les diagrammes de collaborations montrent des interactions entre des objets qui peuvent être des instances des classes ou des acteurs. Ils permettent de représenter le contexte d'une interaction, car on peut y préciser les états des objets qui interagissent.

UML permet de spécifier de manière très précise l'ordre et les conditions d'envoi des messages sur un diagramme dynamique. Pour chaque message, il est possible d'indiquer :

- Les clauses qui conditionnent l'envoi
- Son rang, c'est à dire son numéro d'ordre par rapport aux autres messages
- Sa récurrence
- Ses arguments

4.8 - DIAGRAMMES D'ETATS-TRANSITIONS (dynamique)

Les diagrammes d'états-transitions servent à représenter des automates à états finis sous forme de graphes d 'états, reliés par des arcs orientés qui décrivent les transitions.

Les diagrammes d 'états-transtions permettent de décrire les changements d'états d'un objet ou d'un composant, en réponse aux interactions avec d'autres objets, composants ou acteurs.

ANNEXE7 : SECURITE DES RESSOURCES

1 – SECURITE WEB TIER

On peut protéger les ressources web en spécifiant les contraintes de sécurité. Les contraintes de sécurité déterminent qui sont autorisés à accéder à une collection de ressources web pouvant être une liste de URL ou des méthodes http décrivant les ressources à protéger[17, 18, 19, 21, 22].

Toute tentative d'accès aux ressources protégées amène le conteneur web à authentifier cet utilisateur. Le conteneur n'accepte les requêtes qu'après avoir prouvé l'identité de l'utilisateur et donné la permission d'accès à cette ressource.

En accédant à une ressource web protégée, le conteneur web active le mécanisme d'authentification configuré pour cette ressource. Trois types de configurations peuvent être appliquées :

- Authentification basique http
- Authentification basée sur formulaire
- Authentification par certificat client

1.1 – AUTHENTIFICATION BASIQUE

Avec cette configuration, le serveur authentifie l'utilisateur par un nom et un mot de passe obtenu du client web. L'authentification utilise le codeur Base64.

1.2 – AUTHENTIFICATION BASEE SUR FORMULAIRE

Cette configuration permet la personnalisation d'un écran de login et une page d'erreur à présenter aux utilisateurs par un browser http.

Ni l'authentification basique, ni l'authentification basée formulaire n'est sécurisée. L'authentification basée formulaire envoie les contenus de la boîte de dialogue en tant que texte et le serveur n'est pas authentifié. En authentification basique, les noms d'utilisateur et les mots de passe sont en texte codé, mais non crypté. Donc si quelqu'un intercepte la transmission, le mot de passe et le nom d'utilisateur peuvent être décodés facilement.

1.3 – AUTHENTIFICATION PAR CERTIFICAT CLIENT

Cette dernière authentification est plus sécurisée que les deux autres. Elle utilise HTTPS, dans lequel le serveur et, optionnellement le client authentifie l'un de l'autre avec les clés des certificats publiques.

Le SSL fournit le cryptage des données, l'authentification du serveur, l'intégrité des messages, et optionnellement l'authentification du client sur une connection TCP/IP.

La clé publique du certificat est comme un passeport. Elle est fournie par une organisation de confiance appelée CA qui fournit l'identification du porteur.

En spécifiant l'authentification par certificat du client, le serveur web authentifie le client en utilisant le certificat X.509.

ANNEXE8 : LES EJB

1 - LES SESSIONS BEANS

Le Session Bean représente un seul client à l'intérieur du serveur J2EE. Pour accéder une application déployée sur le serveur, le client invoque les méthodes du session bean. Le session bean exécute des travaux pour les clients en cachant les complexités des tâches à l'intérieur du serveur [1].

Il y a deux types de session bean : stateful et stateless

1.1 - STATEFUL SESSION BEAN

L'état d'un objet consiste en des valeurs de ses variables instanciés. Dans un stateful session bean, l'instance des variables représente l'état d'un unique client du session bean. L'état se maintient durant la session du client. Si le client termine ou enlève le bean, la session et son état disparaît.

1.2 - STATELESS SESSION BEAN

Le stateless session bean ne maintient pas des états de conversation pour un client particulier. Quand un client invoque une méthode d'un stateless bean, le variable d'instance d'un bean peut contenir un état, mais seulement pendant la durée de l'invocation seulement. A la fin de la méthode, l'état n'est plus retenu.

2 - ENTITY BEAN

Un entity bean représente un objet d'une mécanisme de stockage persistent. Dans le J2EE SDK, le mécanisme de stockage persistent est une base de donnée relationnelle. Typiquement, chaque entity bean correspond à une table d'une base de données relationnelle, et chaque instance correspond à une ligne de cette table.

3 - MESSAGE DRIVEN BEAN

Un message-driven bean est une enterprise bean permettant aux applications J2EE de traiter des messages asynchronement. Il utilise un écouteur de message JMS similaire aux écouteurs d'événement sauf qu'il recoit des messages au lieu des événements. Les messages peuvent être envoyés par un quelconque composant J2EE, une application client ou une application JMS ou même d'autres systèmes n'utilisant pas de la technologie J2EE.

BIBLIOGRAPHIE

[1] http://java.sun.com

[2] http://www.borland.com

[3] www.**comment**ca**marche**.net/

[4] http://www.ifad.org/evaluation/guide_f/annexb/b.htm

[5] www.iucn.org/brao/uicn_brao/vacancies/TDR_etude5.pdf

[6] www.**java**fr.com/code.aspx?ID=29581

[7] tvilda.stilius.net/**java**/**java_ssl**.php

[8] www.javaolympus.com/J2SE/Database/JDBC/JDBC.jsp

[9] www.complex-ite.net/**tutorial**s/JDBC/

[10] docs.roxen.com/roxen/2.1/**tutorial**/**database**/

[11] www.geekgirls.com/menu_**database**s.htm

[12] www.w3apps.com/**tutorial**.jsp

[13] www.comptechdoc.org/independent/ database/basicdb/dataintroduction.html

[14] http://httpd.apache.org/

[15] www.**sql**course.com/

[16] www.w3schools.com/**sql**/default.asp

[17] www.**cryptography**-**tutorial**.com/

[18] www.**cryptography**world.com/

[19] www.coreservlets.com/Apache-Tomcat-Tutorial/

[20] jakarta.**apache**.org/**tomcat**/

[21] www.**verisign**.com

[22] www.**thawte**.com/

[23] *Tarif automobile*
ARO, Octobre 2000

[24] www.**java**coffeebreak.com

[25] my.execpc.com/~gopalan/**java**

[26] www.developpez.net

[27] www.developpez.net/**forum**s/

[28] www.**java**fr.com

[29] www.sparxsystems.com.au/UML_Tutorial.htm

[30] www.parlez**uml**.com/

[31] www.cragsystems.co.uk/ITMUML/

[32] www.**jsp**tut.com/Index.html

Nom : RANDRIANAVALOTIANA

Prénom : Rivo Jean Michel

Adresse : ITV 45 TER Itaosy Andranonahoatra
101 Antananarivo
MADAGASIKARA

Email : rrjmichfriend@yahoo.fr

Titre du mémoire : Mise en place d'une architecture J2EE appliquée au calcul de prime d'assurance automobile

Nombre de pages : 98

Nombre de tableaux : 09

Nombre de figures : 75

Mots clés : BES, BMP, CMP, EAR, EJB, J2EE, JAR, JDBC, JDK, JNDI, JSP, MULTITIERS, OQL, SERVLET, UML, XML

Directeur de mémoire : RAMANANTSIZEHENA Pascal

RESUME

Si au début des années 90, le système client serveur s'était montré performant et plein d'avenir, il a montré ses limites avec la délocalisation des entreprises et les intégrations du public.

Et le système multitiers est apparu. Non seulement il permet de s'affranchir des limites de l'ancien système client serveur, et, sans parler des performances qu'il peut apporter, permet aussi de ne nécessiter que des investissements et des coûts d'exploitation relativement bas.

Le J2EE, un produit de Sun Microsystems, est la solution Java d'une architecture multitiers. Comme Java, elle est multiplateforme, et c'est sa force. Donc elle permet de faire tourner une application sur n'importe quel système d'exploitation sans nécessiter de recompilation.

Un autre avantage important de la technologie Jest qu'elle permet d'atteindre le public sans frais supplémentaire par rapport au client serveur. Ces frais incluent les coûts J2EE es installations, les coûts directs des personnels et des infrastructures et les coûts générés par l'organisation.

Et quid de la sécurité. D'abord du côté des matériels. Les serveurs peuvent être centralisés dans un lieu sûr et protégés physiquement. L'entreprise ne risque pas ainsi de voir ses données détruites ou volées. Et ceci facilite grandement la maintenance car les interventions se limitent à un lieu.

Pendant le transfert de données sur des réseaux publics comme l'Internet où les données risquent d'être interceptées et modifiées à tout moment par un tiers, le cryptage par SSL 128 bits est une solution très fiable et incassable de nos jours.

Comme nous l'avons vu, cette technologie est une solution accessible et à la portée des entreprises malagasy. Elle est très performante et très fiable et son coût d'exploitation est incroyablement bas.

SUMMARY

Today, more and more developpers write distributed transactional applications for the enterprise and thereby leverage the speed, security, and reliability of the server-side technology.

To reduce costs and fast-track application design and development, the J2EE provides a component-based approach to the design, development, assembly, and deployment of enterprise applications. The J2EE platform offers a multitiered distributed application model, reusable components, a unified security model, flexible transaction control, and web services support through integrated data interchange on XML-based open standards and protocols.

Not only can you deliver innovative business solutions to market faster than ever, but also your platform-independent component-based solutions are not tied to the products and APIs of any one vendor. Vendors and customers enjoy the freedom to choose the products and components that best meet their business and technological requirement

Nom : RANDRIANAVALOTIANA

Prénom : Rivo Jean Michel

Adresse : ITV 45 TER Itaosy Andranonahoatra
101 Antananarivo
MADAGASIKARA

Email : rrjmichfriend@yahoo.fr

Titre du mémoire : Mise en place d'une architecture J2EE appliquée au calcul de prime d'assurance automobile

Nombre de pages : 99

Nombre de tableaux : 09

Nombre de figures : 75

Mots clés : BES, BMP, CMP, EAR, EJB, J2EE, JAR, JDBC, JDK, JNDI, JSP, MULTITIERS, OQL, SERVLET, UML, XML

Directeur de mémoire : RAMANANTSIZEHENA Pascal