République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique.

Université Saad Dahlab, Blida
USDB.

Faculté des sciences.
Département informatique.

Mémoire pour l'obtention
d'un diplôme d'ingénieur d'état en informatique.
Option : system d'information

Sujet :

Conception et réalisation d'une base de données prestations sociales accessible via le web en architecture trois-tiers

Présenté par : Mr. Zeraoui Sofiane
Mr. Benhalima Mohamed

Promoteur : Mr. A. Cherif Zahar

Organisme d'accueil : SONATRACH (Direction Générale)

Soutenance le : , devant le jury composé de :

Nom. Président du jury, grade, organisme	Président
Nom. Examinateur 1, grade, organisme	Examinateur
Nom. Examinateur 2, grade, organisme	Examinateur

2005/2006

REMERCIEMENT

Mes remerciements vont à tout ce qui ont contribué, de près ou de loin à mener à terme ce travail, notamment les enseignants qui m'ont instruit tout au long de mon long parcours.

Sans oublier notre promoteur Mr CHERIF ZAHAR pour ces orientations et conseils fructueux.

Je remercie chaleureusement tout le personnel de la SONATRACH et plus particulièrement
Mr TAYEBI responsable du service prestations sociales et
Mr A.DAIMALLAH responsable de la formation et de l'infographie.

DÉDICACES

J'ai le grand plaisir de dédier le fruit de mes années d'études aux êtres qui me sont les plus chers, mes parents.

A ma mère qui s'est données tant de mal pour moi et qui ma offert amour et soutien depuis mon premier souffle.

A mon père pour avoir mis tous les moyens a ma disposition et qui ma encouragé et soutenu pendant mes études

A mes chère sœurs et mon frère qui m'ont données chaque un une aide très précieuse a leurs façons.

Et toutes les personnes que j'aime et qui m'aime.

A mon binôme Mohamed.

A mes chères amis, Mr Goumari, youcef, tarik, azouaou, ibrahim, yacine, chaoukette, hamza noureddine, lyes, souad, keireddine, abdelhak, zaki, ali, mahfoud, et tout mes amis de la promo 2005/2006

Zeraoui Sofiane

DÉDICACES

Je dédie ce mémoire :

A mes très chers parents qui mon beaucoup soutenu et encouragé durant mes années d'études

A ma grand mère et mon grand père

A ma Sœur Djamila et ses fils Wissam et Nabila

A mes frères Zoubir et Zakaria

A mes amis

A tous mes amis de la promo 2005/2006

Benhalima Mohamed

Résumé :

Ce document et les annexes qui l'accompagnent constituent le résultat de notre travail qui a consisté en la conception et la réalisation d'une base de données accessible via le web

Après analyse des informations recueillies auprès du service prestations sociales de la direction générale de Sonatrach, nous avons utilisé OMT&UML comme méthodologie pour la conception de notre système dont l'objectif est de faire le suivi des dossiers social du personnel de la direction générale de SONATRACH en se servant de la technologie Internet.

Dans cette perspective nous avons utilisé l'architecture trois-tiers, que nous avons détaillée au chapitre 3 du présent document.

En effet, à partir d'un serveur d'applications on accède à la base de données par l'approche composant avec les entités JavaBeans.

Mots Clés : OMT, UML, Architecture trois-tiers, l'approche composant, JavaBeans.

Sum up:

This document and the enclosures which go with constitute the result of our work which consisted in the conception and the realisation of a data base accessible via the Web.

After the analyse of the collected information from the social insurance benefits service of the general management of SONATRACH, we have used the OMT and UML as a methodology for the conception of our system which objective is to trace the social files of the general management of SONATRACH using the Internet technology.

For this outlook we used the three-third architecture that we have detailed in chapter3 of the present document.

In deed, from an application sever we accede to the data base by the component approach with the JavaBeans entities

Key Words : OMT, UML, Architecture trois-tiers, l'approche composant, JavaBeans.

SOMMAIRE

Chapitre 3 : L'approche composant

Chapitre 4 : Analyse et conception du système

Chapitre 5 : Implémentation et résultats

Liste des figures :

Liste des tableaux

INTRODUCTION GENERALE

SONATRACH en tant que l'une des principales sociétés pétrolières dans le monde n'a pas voulu rester en marge des avancées technologiques dans le domaine de l'Internet. Compte tenu que l'entreprise soit éparpillée un peu partout sur le territoire algérien et du fait que ce n'est pas toujours facile de faire le suivi des dossiers du personnel surtout si on est toujours en déplacement. Le besoin d'une application web pour le service prestations sociales se fait ressentir. La mission est de mettre en place cette application dans intranet de la société reliant tout d'abord les sites de l'entreprise à Alger, puis la rendre présente sur le web.

Le web s'est développé comme un hypertexte sur le réseau Internet pour permettre facilement l'accès à des fichiers chaînés, rapidement le besoin de couplage avec les bases de données est apparu. Trois raisons au moins motivent ce besoin :

- ✓ L'introduction du clients-serveur à présentation universelle (architecture trois-tiers)
- ✓ La génération des sites web dynamique composés à partir de template HTML et de données extraites de bases.
- ✓ Le commerce électronique, qui nécessite la gestion de catalogues et de transactions en base de données.

Notre projet d'étude consiste en la conception et la réalisation d'une base de données prestations sociales, accessible via le web en architecture 3-tiers pour le siége de la SONATRACH.

Pour faire ce travaille, il existe plusieurs techniques d'accès aux bases de données à partir d'un serveur d'applications ,notamment , l'approche par page comme le PHP, l'approche par scriptlette avec les page JSP ou encore l'approche par composant utilisant les JavaBeans dans les pages JSP.

Dans cette perspective nous avons retenu pour notre projet cette dernière approche pour les raisons suivantes :

- profit des avantages de la portabilité et la puissance de java.
- Utilisation d'une technologie récente de conception de sites dynamiques à savoir le JSP (java Server page).

Notre mémoire s'articule essentiellement autour de cinq chapitres :

- Dans le premier chapitre sera présenté l'étude du domaine où la société souhaite améliorer son fonctionnement
- dans le deuxième chapitre, sera mis l'accent sur les system d'informations en architecture trois-tiers et les bases de données&web
- dans le troisième chapitre, sera présenté l'approche composant, et la persistance des objets java avec les entités Beans du J2EE.
- Dans le quatrième chapitre, sera présenté le langage de modélisation objet UML. Et une méthode de conception OMT, puis l'analyse et la conception du system.
- Dans le dernier chapitre Nous présenterons les langages de programmations utilisés ainsi que la description détaillée de notre application

Enfin, une conclusion général et des perspectives clôturant notre travail.

Chapitre 1

Etude de l'existant

1. INTRODUCTION

Toute l'imagination et tout le savoir-faire de l'analyste seront sans emploi, s'ils ne peuvent s'exercer sur une base réaliste.

Pour que l'application soit mise à la disposition de ceux qui ont besoin et à qui elle est destinée, l'étude de l'existant est le point de passage obligé qui matérialise le premier contact avec un domaine ignoré, c'est la première étape dans notre étude, elle nous permet de découvrir en détail le domaine où la société souhaite améliorer son fonctionnement.

Cette étude a pour but de nous permettre de:

1/ connaître en détail l'architecture Internet du siège SONATRACH.

2/ connaître les fonctions des services « prestation sociale » ainsi que les flux d'informations externe et interne

3/ voir en détail les postes de travails et les fonctions de service prestations sociales.

Avant d'être un réseau ou un ensemble de technologies, Internet est un standard mondial de communication

Il offre la possibilité

- de gagner en productivité dans la gestion interne et dans les relations interentreprises, et
- d'ouvrir aux entreprises des marchés nouveaux.

SONATRACH, groupe pétrolier International, étant la plus importante entreprise nationale, doit utiliser cette technologie pour ne pas être à la marge du progrès.

En effet, Les technologies Internet concernent toutes les fonctions de l'entreprise :

- ✓ vendre
- ✓ se faire connaître
- ✓ trouver des partenaires
- ✓ faire de la veille technologique et de l'intelligence économique
- ✓ transmettre des documents écrits, sonores ou vidéo

✓ conduire des projets, faxer, téléphoner, participer à des bourses, travailler en réseau....

2. ARCHITECTURE INTERNET

- **DMZ**

  Contenant les serveurs devant être perçus de l'extérieur.

- **Cisco PIX F**

  • Système (hard & soft) dédié sécurisé en temps réel,

  • Nombreuses options de connexion LAN :

  – Ethernet

  – Fast Ethernet

  – Token Ring ET FDDI

  • Support de 250.000 connexions simultanées.

- **Serveur proxy**

Avec l'utilisation du serveur proxy Les performances peuvent être encore améliorées ; L'enregistrement des sites WEB dans sa mémoire cache rendent la connexion plus rapide.

PROXY serveur tournant sur un port précis en attente de connexion cliente, capable d'accepter et de traiter plusieurs demandes de la part de plusieurs clients simultanément, Son rôle est de récupérer les informations sur les requêtes de ses clients et de les transmettre à PROXYREQUETE pour les traitements et le renvoie de la réponse.

- **routeur**

Internet résulte de l'interconnexion de différents réseaux physiques par des machines appelées routeurs. Chaque réseau, contenant un ensemble d'hôtes, est connecté, éventuellement, à un routeur. Le routeur permet de déterminer l'adresse physique de destinataire et de faire circuler les messages (paquets).

- **Switch**

Les ponts permettent d'étendre les possibilités du LAN : nombre de stations, distance, confidentialité, taux de défaillance. Un pont multiport est appelé commutateur (switch). Ils permettent également d'optimiser les débits. Les ponts sont des ordinateurs complets travaillant au niveau de la couche 3 et sont souvent multi protocoles. Un pont reçoit les trames circulant sur les segments raccordés, les analyses. Il récupère ainsi les numéros Ethernet des stations actives sur un segment. En fonction du destinataire, il émet ou rejette la trame. Un pont compte comme une station sur chaque segment. On mesure la qualité d'un pont par son taux de filtrage et son taux de transfert. Le maximum théorique de ces deux quantités est14 880 paquets de 64 octets par seconde.

La figure 1.1 illustre l'architecture Internet de siége Sonagraphe ; chaque Vlan correspond à une direction de siége SONATRACH c'est-à-dire architecturé en Vlan fonctionnels (Virtual Local Area Network) correspondants aux différentes structures qui y sont hébergées,

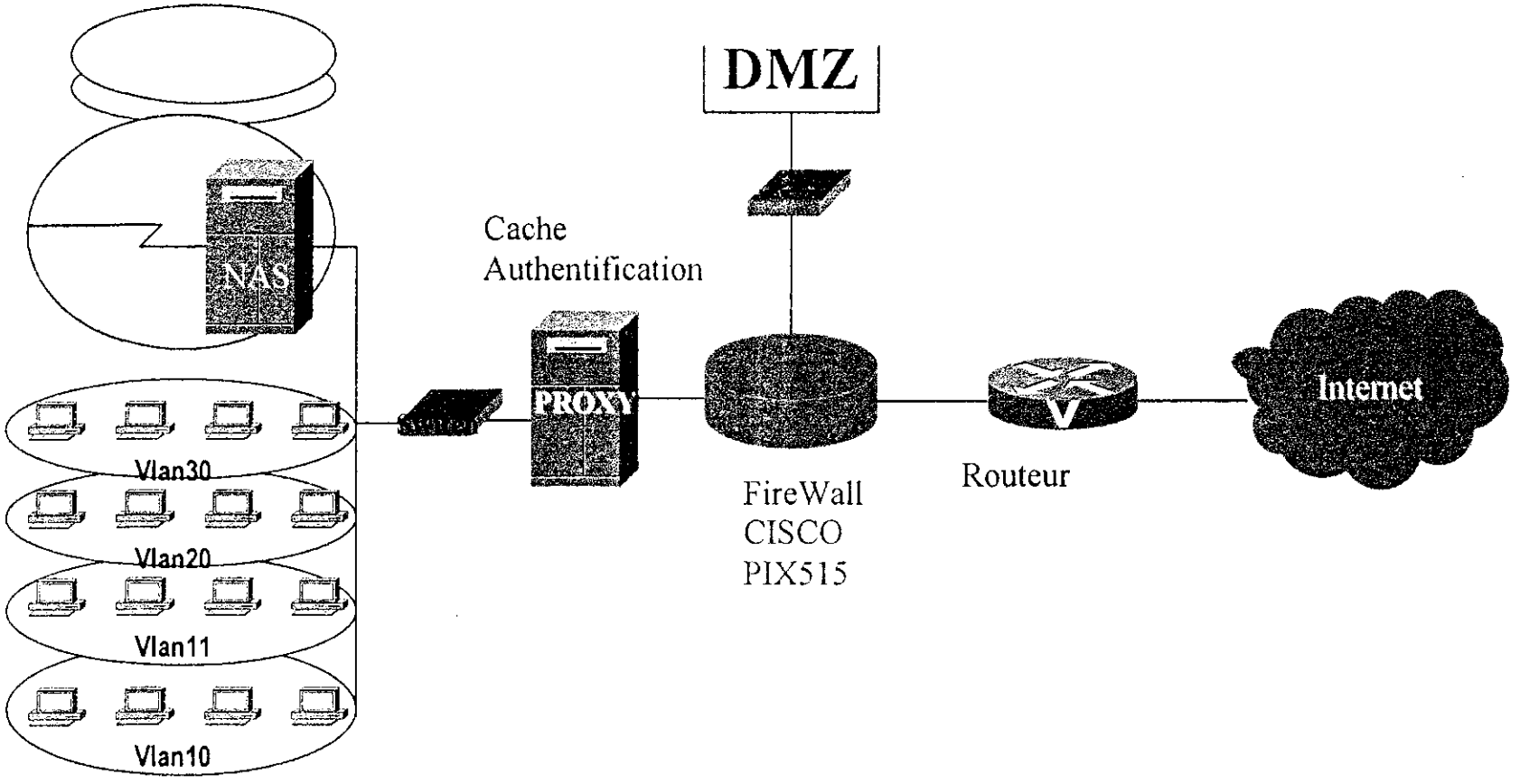


Figure 1.1. Architecture Internet de siége SONATRACH

3. Présentation de service prestation sociale :

Notre but est de mettre en place une application web en vue de la relation d'un system d'information pour le suivi des dossiers social du personnel de la direction générale de SONATRACH.

De ce fait l'application doit rependre aux besoins du service prestations sociales qui accompli entre autre les tâches suivantes :

- Les remboursements médicaux.
- Les allocations familiales
- La mutuelle
- La retraite.

3.1. Flux D'information externe :

3.1.1 Le Schéma de Flux D'information externes

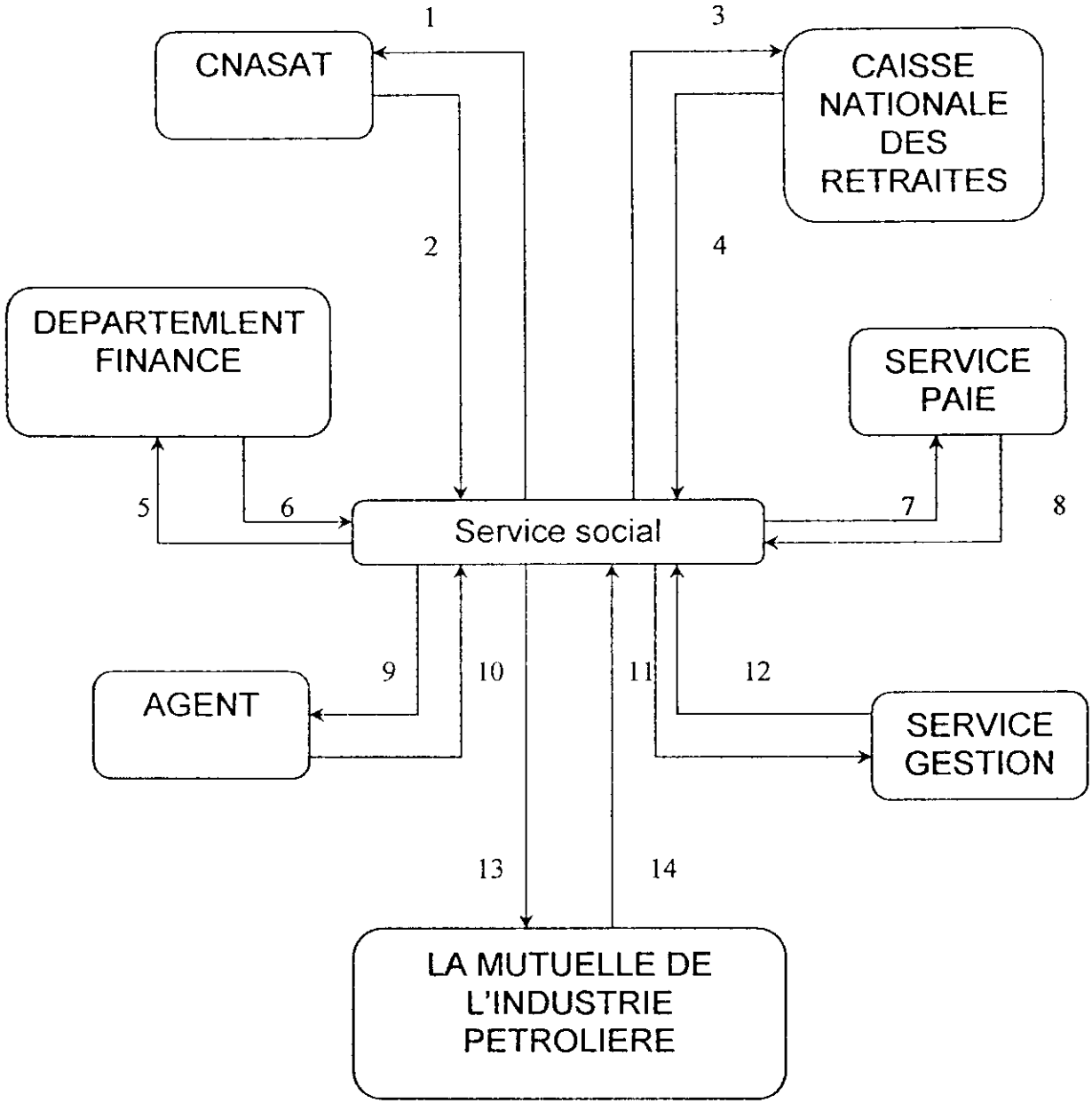


Figure 1.2 flux d'informations externe

3.1.2. Description du Flux d'informations Externes

N° DE LIAISON	DESIGNATION
01	- Envoi de la déclaration d'emploi à la CNAS. - Envoi des bordereaux d'exploitation à la CNAS. - Envoi des dossiers frais médicaux et arrêts de travail à la CNAS. - Envoi des bordereaux de paiement des allocations familiales, états des impayés et les bordereaux de réclamation à la CNAS. - Envoi des pièces de renouvellement et des situations nouvelles à la CNAS. - Envoi des dossiers indemnités journalières à la CNAS. - Envoi des lettres types à la CNAS.
02	- Réception des bordereaux d'exploitation. - Réception des cartes d'assurance. - Réception des chèques barrés. - Réception des décomptes de la CNAS. - Réception des récapitulations de la CNAS. - Réception des bordereaux de paiement des allocations familiales et des bordereaux d'exploitation (indemnités journalières ou remboursement des frais médicaux) avec accusés de réception.
03	- Envoi des dossiers de mise en retraite à la CNR ainsi que la lettre type.
04	- Réception de la notification de la CNR.

05	- Envoi des bordereaux de paiement internes au département finance. - Envoi des chèques barrés au département finance. - Envoi des récapitulations au département finance. - Envoi des bordereaux de versement des cotisations au département finance.
06	- Réception des notes internes du département finance ainsi que les lettres types.
07	- Envoi des bordereaux de paiement internes au service paie.
08	Réception des journaux de paie du service paie.
09	- Envoi des cartes d'assurance aux intéressés. - Envoi des décomptes à l'agent. - Envoi des convocations à l'agent.
10	- réception des dossiers frais médicaux et arrêts de travail. - réception des dossiers allocations familiales. - Réception des fiches familiales, certificats de scolarité, bulletin de naissance, extrait de décès et une copie de jugement. - Réception de mise en retraite des agents. - Réception des pièce : facture cliniques, acte des mariage, prescription médicale pour la circoncision.
11	- envoi de la photocopie de la notification au service de gestion.
12	- Réception d'une fiche de renseignement du service de gestion. - Réception des décisions du service de gestion.
13	- envoi des document MIP + décomptes CNAS + les pièces + bordereaux nominatif de prestation a la MIP. - Envoi de chèque barré et du bordereau de versement de

	cotisation a la MIP.
14	- réception de bordereau de versements de cotisation avec accusé de réception.

Tableau 1.1 Flux d'informations Externes

3.2. Flux d'information interne :

3.2.1 Schéma de flux d'informations internes

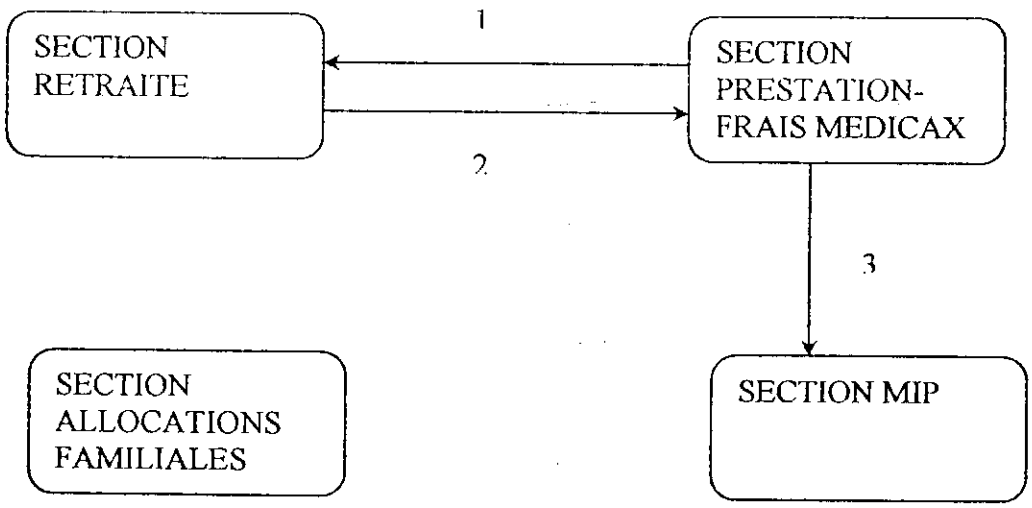


Figure 1.3 flux d'information interne

3.2.2. DESCRIPTION DU FLUX D'INFORMATIONS INTERNES

N^0 DE LIAISON	DESIGNATION
1	- envoi des bordereaux d'exploitation des arrêts de travail a la section retraite. - Envoi des déclarations d'accidents de travail et des bordereaux d'exploitation avec accusé de réception.
2	- Envoi des arrêts de travail a la section prestation frais médicaux. - Envoi des déclarations d'accident de travail, des infirmités et des questionnaires a la section prestations frais médicaux.
3	- Envoi des décomptes frais médicaux a la section *MIP*

Tableau 1.2 Flux d'informations internes

3.2.3. Fiche de Fonction des postes de travail :

Fiche de Fonction du Poste 01 :

Désignation : chef section prestations / assurance sociale

Section : retraite

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribuées

- indemnités journalières
- retraite
- capitale décès

b- tache à accomplir :

N°	Tâches	fréquences	remarque
1	- consultation des butins de paie des agents	-1fois/mois	-envoyer par le service de paie
2			
3	- exploitation des arrêts de travail	-1fois/mois	
4	- établissement des dossiers des indemnités journalières	-1fois/mois	
5	- Remplissage des fiches médicales	-1fois/mois	
6	- Etablissement des bordereau e paiement des prestation	-1fois/mois	-en 3exemplaires
7	- envoi des dossiers des indemnités journalières	-1fois/mois	
8	- mise à jour des fiches médicales	-1fois/mois	
9	- Envoi de la récapitulation et du chèque et des décomptes au service finance	-1fois/mois	
10	- établissement des bordereaux d'envoi pour les envoyer au service paie	-1fois/mois	
11	- recensement des agents partant en retraite	- 1fois/an	
12	- Envoi des convocations aux agents partant en retraite	- 1fois/an	
13	- envoi des dossiers de la mise en retraite à la CNR	- 1fois/an	
14	- Envoi d'une note interne et d'une notification au service GESTION	-irrégulier	

15	- aviser la CNR de la cessation de travail des retraités - Positionnement des retraités ainsi que tous les renseignements les concernant sur le registre du suivi personnel mis en retraite	-irrégulier -irrégulier	

Tableau 1.3 Fonction du Poste 01

Fiche de Fonction du Poste 2 :

Désignation : charger de gestion

Section : MIP

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribuées :

- remboursement MIP (20%)
- Cotisation

b- tâches accomplir :

N°	Tâches	fréquences	remarque
1	- établissement des décomptes de prestation	-1fois/semaine	
2	- établissements des bordereaux nominatifs de prestation	-1fois/semaine	
3	- envoi des décomptes et des bordereaux a la MIP	-1fois/semaine	
4	- établissement des bordereaux d'envoi	-1fois/mois	
5	- classement des décomptes	-1fois/mois	-par direction et par ordre alphabétique
6	- remplissage des fiches d'adhésion	-aléatoire	
7	- consultation des journaux de paie	-1fois/mois	
8	- établissement des bordereaux de versement de cotisation	-1fois/mois	
9	- envoi des chèques, bordereaux de cotisation et des bordereaux d'exploitation a la MIP	-1fois/mois	

Tableau 1.4 Fonction du Poste 02

Fiche de Fonction du Poste 3

Désignation : cadre de gestion administration

Section : prestation – frais médicaux

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.S)

– Fonction :

a- Tâches attribuées

- remboursements des frais médicaux
- immatriculation

b- tâches à accomplir :

N^0	Tâches	Fréquences	Remarques
1	- positionnements des dossiers sur le fichier du suivi des remboursements des frais médicaux	1/semaine	-Par N^0 SS
2	- établissement des bordereaux 'exploitation	1/semaine	-Par N^0 SS
3	- envoi des dossiers et des bordereaux a la CNAS	1/semaine	-Pour la collectivité
4	- classements des décomptes	1/semaine	-Pour la
5	- mise à jour du fichier du suivi des remboursements des frais médicaux	1/semaine	collectivité
6	- classements des décomptes par direction et par ordre alphabétique	1/mois	
7	- calcule du montant globale de chaque décompte	1/mois	
8	- établissement des bordereaux de paiements	1/mois	
9	- envoi des bordereaux de paiements aux services PAIE et FINANCE	1/mois	
10	- envoi des décomptes aux intéressés	1/mois	
11	- remplissage des fiches signalétique	Aléatoire	
12	- remplissage des déclarations d'emploi du travailleur	Aléatoire	
13	- envoi des déclarations d'emploi du travailleur a la CANS	1/semaine	
14	- positionnement des N^0 de sécurité sociale sur les fiches signalétiques	Aléatoire	
15	- envoi des cartes d'assurance aux intéressés	Aléatoire	

Fiche de fonction du poste 4

Désignation : cadre de gestion allocation familiale

Section : allocation familiale

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribués :

- s'occupe des allocations familiales

b- tâches à accomplir :

N^0	Tâches	Fréquences	Remarque
1	- envoi des convocations a l'intéressé	Aléatoire	
2	- établissements des attestations de salaire et de travail	Aléatoire	
3	- établissements des fiches d'allocations	Aléatoire	
4	- établissements des bordereaux d'exploitations	1/mois	
5	- établissements de bordereaux de réclamations ou additif	1/mois	
6	- établissement des notes internes	1/mois	
7	- envoi des bordereaux	1/mois	
8	- envoi des notes internes	Aléatoire	
9	- envoi des pièces de renouvellement et des situations nouvelles a la CNAS	1/mois	

Tableau 1.5 Fonction du Poste 04

4. Conclusion :

On a pu découvrir en détail les moyens et l'infrastructure où la société souhaite fonctionner notre application. Ensuite, on a mis en évidence les fonctions du service « prestations sociales » en définissant les postes de travail, les tâches qui lui sont attribuées, et les flux d'informations qui circulent entre eux.

Notre but dans le chapitre suivant est de détailler les systèmes d'information en architecture trois-tiers et les bases de données &web, car l'approche composant avec les JavaBeans s'applique dans le domaine des bases de données&web

Chapitre 2

Système d'information
&
Architecture trois-tiers

1. Introduction :

Dans ce chapitre nous allons exposer la notion des systèmes d'informations distribuées et l'architecture trois-tiers ainsi que les bases de données&web.

L'objectif premier d'un système d'information quel qu'il soit est de permettre à plusieurs utilisateurs d'accéder aux mêmes informations. Pour cela il faut donc regrouper les informations utilisées par l'entreprise. En terme technique, cela se traduit par la centralisation des données au sein d'une base de données. L'évolution des systèmes d'information s'est donc basée sur une meilleure subdivision entre les tâches à réaliser pour permettre l'exploitation de ces données par les utilisateurs finaux. Ceci permet de structurer plus efficacement les informations ce qui entraîne à la fois une meilleure organisation de l'entreprise et une meilleure efficacité technique. Cette subdivision a été facilitée par l'avènement des technologies orientées objets qui s'appliquent aussi bien au modèle client-serveur qu'au modèle Internet. Ces technologies permettent une séparation entre les différents composants du système. Il devient alors possible de réaliser de nouvelles architectures permettant la mise à disposition des informations sous différentes formes tout en diminuant les temps de développement. Ces technologies permettent également de faire collaborer une grande diversité de systèmes. On parle alors d'architecture distribuée. Il est ainsi possible de présenter des données en provenance d'un mainframe mélangées à des données en provenance d'un SGBDR, le tout étant affiché dans un browser sur la même page HTML. [LEF97]

2. Système d'information et l'architecture trois-tiers :

Tout système d'information nécessite la réalisation de trois groupes de fonctions: le stockage des données, la logique applicative et la présentation. Ces trois parties sont indépendantes les unes des autres: on peut ainsi vouloir modifier la présentation sans modifier la logique applicative. La conception de chaque partie doit également être indépendante, toutefois la conception de la couche la plus basse est utilisée dans la couche d'au dessus. Ainsi la conception de la logique applicative se base sur le modèle de données, alors que la conception de la présentation dépend de la logique applicative [LEF97]

2.1. Trois tâches importantes

- **Stockage et accès aux données**

Le système de stockage des données a pour but de conserver une quantité plus ou moins importantes de données de façon structurée. On peut utiliser pour cette partie des systèmes très variés qui peuvent être des systèmes de fichiers, des mainframes, des systèmes de bases de données relationnelles, etc. Le point commun entre tous ces systèmes est qu'ils permettent le partage des données qu'ils contiennent via un réseau. La méthode d'accès à ces données dépendra du type d'organisation de ces données. Dans le cas d'une base de données relationnelle, l'accès peut se faire par des API qui dépendent du langage et de l'environnement. Ainsi en JAVA l'accès se fait via JDBC, alors qu'en C++ il se fera à l'aide d'ODBC. Quel que soit l'API, le langage SQL est utilisé.

- **Logique applicative**

La logique applicative est la réalisation informatique du mode de fonctionnement de l'entreprise. Cette logique constitue les traitements nécessaires sur l'information afin de la rendre exploitable par chaque utilisateur. Les utilisateurs peuvent avoir des besoins très variés et évolutifs. Il devient alors nécessaire de permettre l'évolution du système sans pour autant devoir tout reconstruire. Cette partie utilise les données pour les présenter de façon exploitable par l'utilisateur.

Il convient donc de bien identifier les besoins des utilisateurs afin de réaliser une logique applicative utile tout en structurant les données utilisées.

- **Présentation**

La présentation est la partie la plus immédiatement visible pour l'utilisateur. Elle a donc une importance primordiale pour rendre attrayante l'utilisation de l'informatique. Son évolution a été très importante depuis les débuts de l'informatique. Depuis les terminaux en mode texte connectés à des mainframes jusqu'au HTML de nos jours en passant par les applications graphiques développées en client serveur, il y a eu beaucoup de chemin parcouru. Différents types d'interfaces demeurent intéressantes. En effet l'ergonomie d'un site Intranet HTML n'est pas forcément idéale pour tous les types d'applications. Il peut être intéressant de proposer plusieurs types d'interface pour une seule logique applicative. Par exemple une entreprise disposant d'un site de commerce électronique peut proposer un accès à la liste de ses produits sur Internet en HTML mais disposer d'une interface d'administration réalisée à l'aide d'une applet graphique [LEF97]

2.2. Principe de l'architecture trois-tiers

Le principe d'une architecture trois-tiers est relativement simple: il consiste à séparer la réalisation des trois parties vues précédemment (stockage des données, logique applicative, présentation). Nous avons déjà pu entrevoir la possibilité de séparer la conception de ces trois subdivisions, ici il s'agit de séparer leur implantation. Tout comme dans le client-serveur cette séparation signifie qu'il est possible de déployer chaque partie sur un serveur indépendant, toutefois cela n'est pas obligatoire. La mise en place de ce type d'architecture permet dans tous les cas une plus grande évolutivité du système. Il est ainsi possible de commencer par déployer les deux serveurs sur la même machine, puis de déplacer le serveur applicatif sur une autre machine lorsque la charge devient excessive. Les éléments permettant la réalisation classique d'un système en architecture trois tiers sont les suivants:

- système de base de donnée relationnel (SGBDR) pour le stockage des données
- serveur applicatif pour la logique applicative
- navigateur web pour la présentation

Il est important de remarquer que l'essentiel du travail de développement sera implanté au niveau du serveur applicatif. Le SGBDR nécessitera un travail d'administration surtout dans le cas d'une quantité de données importante. Le travail de conception de la base de donnée sera la pierre angulaire du système. En effet l'ensemble du développement s'appuiera sur cette conception. Le navigateur web nécessitera la programmation de code spécifique permettant de gérer l'affichage par ce navigateur. Ce code sera placé sur le serveur applicatif pour permettre une mise à jour sans nécessiter de nouveaux déploiements.

2.3. Avantages de cette architecture

Cette architecture se développe actuellement au sein des entreprises grâce aux nombreux avantages qu'elle présente. Malgré la différence évidente entre une architecture trois tiers et un système client-serveur (l'apparition d'un serveur pour la logique applicative), le système reste basé sur les technologies éprouvées détaillées précédemment (aspect relationnel et transaction). La logique applicative est déplacée au niveau du serveur d'application mais reste programmée à l'aide des mêmes technologies liées aux bases de données relationnelles. En particulier l'utilisation du langage SQL reste jusqu'à présent la solution la plus intéressante au niveau de la qualité logicielle. Elle présente à la fois une grande fiabilité, une bonne disponibilité, une excellente évolutivité,... Toutefois il faut prendre en compte deux facteurs importants: d'une part le choix du SGBDR (ils n'ont pas tous les même qualités), d'autre part la qualité des programmes utilisant la base de données (aussi bien au niveau de la conception que de la programmation). L'avantage principal d'une architecture multi-tiers est la facilité de déploiement. L'application en elle même n'est déployée que sur la partie serveur (serveur applicatif et serveur de base de données). Le client ne nécessite qu'une installation et une configuration minime. En effet il suffit d'installer un navigateur web compatible avec l'application pour que le client puisse accéder à l'application, ce navigateur étant par ailleurs souvent installé par défaut sur toutes les machines. Cette facilité de déploiement aura pour conséquence non seulement de réduire le coût de déploiement mais aussi de permettre une évolution régulière du système. Cette

évolution ne nécessitera que la mise à jour de l'application sur le serveur applicatif. Ceci est très important car cette évolutivité est un des problèmes majeurs de l'informatique. Le troisième avantage est l'amélioration de la sécurité. Dans un système client-serveur tous les clients accédaient à la base de données ce qui la rendait vulnérable. Avec une architecture multi-tiers l'accès à la base n'est effectué que par le serveur applicatif. Ce serveur est le seul à connaitre la façon de se connecter à cette base. Il ne partage aucune des informations permettant l'accès aux données, en particulier le login et le password de la base. Il est alors possible de gérer la sécurité au niveau de ce serveur applicatif, par exemple en maintenant la liste des utilisateurs avec leurs mots de passe ainsi que leurs droits d'accès aux fonctions du système. On peut même améliorer encore la sécurité par la mise en place d'une architecture réseau interdisant totalement l'accès au serveur de base de données pour les utilisateurs finaux. La mise en place de firewall correctement configuré permettra ceci. Dans le cas de la mise en place d'un workflow au sein de l'entreprise Kallisto, l'avantage le plus intéressant est sans aucun doute l'évolutivité du système. En effet les besoins actuels sont relativement vastes, toutefois la réalisation totale semble relativement longue et difficile à réaliser d'un seul coup. De plus des problèmes risquent de nécessiter une mise à jour répétée du système. Un autre aspect intéressant est la possibilité d'utiliser le système en Extranet, ce qui chez Kallisto permettra de faciliter la communication entre le siège social situé à Lyon et l'agence de Paris [LEF97]

3. Les bases de données et le web

3.1. La révolution Internet :

S'il est un phénomène qui a marqué le monde de l'informatique ces dernières années, c'est bien celui d'Internet.
Ce réseau mondial, créé en 1969 par l'armée américaine, puis utilisé par les chercheurs et autres scientifiques, a connu une croissance phénoménale auprès du grand public avec l'introduction du *World Wide Web*[3.1] en 1989. Ce dernier permet

de publier simplement des informations richement mises en forme et pouvant même, par la suite, contenir des données multimédia.

La véritable révolution du WWW réside dans son caractère universel, rendu possible par l'utilisation de standards reconnus.

3.1.1. Les standards d'Internet

L'universalité du Web repose sur des standards simples et admis par tous :

- HTML, pour la description des pages disponibles sur le Web,
- HTTP, pour la communication entre navigateur et serveur Web,
- TCP/IP, le protocole réseau largement utilisé par les systèmes Unix,
- CGI[3.2], l'interface qui permet de déclencher à distance des traitements sur les serveurs Web.

- HTML (HyperText Markup Langage)

HTML est le langage de description de pages hypertexte utilisé par le World Wide Web, il est issu de SGML[3.3].

Une page HTML est composée de son contenu propre (du texte plus ou moins richement mis en forme) et de références statiques vers d'autres sources d'informations (images, liens vers d'autres documents...).

La consultation d'une page HTML n'implique donc que très rarement le chargement du seul fichier décrivant la page, il s'accompagne en général de celui de nombreux fichiers annexes. Ces chargements mettent en oeuvre le protocole HTTP.

- HTTP

HTTP est un protocole réseau applicatif (dernier niveau du modèle OSI) sans connexion utilisé pour l'échange des données sur le Web. En fait, une connexion HTTP est créée pour chaque requête et ne dure que pendant l'exécution de cette dernière.

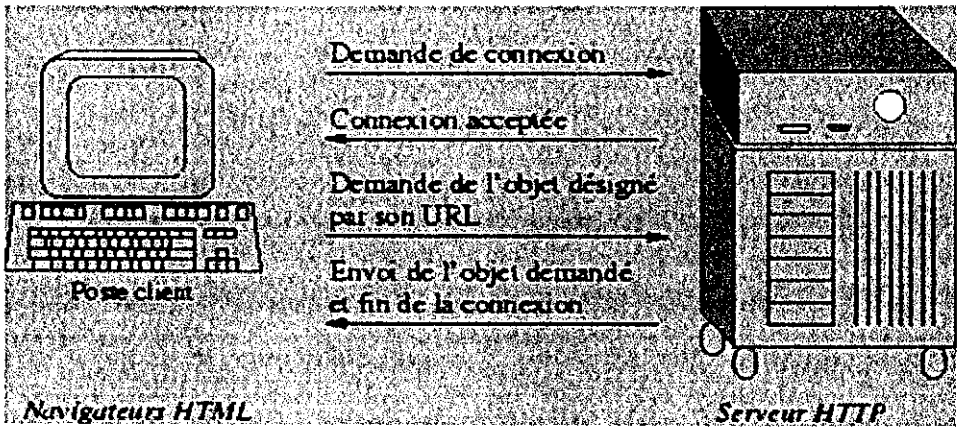


Figure 2.1: Le fonctionnement de base de HTTP[LEF97]

Le protocole HTTP, en temps que protocole réseau applicatif, s'appuie sur un protocole de transport indépendant qui, dans le cadre d'Internet, est TCP/IP[3,4].

- TCP/IP (Transmission Control Protocol / Internet Protocol)

TCP/IP est un protocole réseau de niveau trois et quatre (réseau et transport) qui s'est largement imposé sur les systèmes Unix, puis sur Internet, et fait aujourd'hui figure de standard universel.

Si le fonctionnement de TCP/IP s'adapte bien à la topologie et à la qualité de service du réseau Internet, on ne peut en dire autant du mariage avec le protocole HTTP. En effet, TCP/IP utilise un mécanisme de ``démarrage lent'' afin d'éviter les engorgements du réseau. Ce mécanisme permet à la machine émettrice d'ouvrir progressivement une fenêtre de congestion en doublant le nombre de paquets émis à chaque aller-retour. En général, la courte durée des échanges HTTP ne permet pas à la fenêtre de congestion d'atteindre la largeur de bande fournie par le réseau local [LEF97].

- **CGI (Common Gateway Interface)**

CGI est un standard permettant d'écrire des extensions compatibles avec la grande majorité des serveurs HTTP. Ces extensions permettent l'exécution d'une action par le serveur à la demande d'un client.

Ce mécanisme relativement simple, voire même rustique, entraîne l'exécution d'un processus propre à chaque invocation, ce qui est très consommateur de ressources.

De ce fait, des extensions comme ISAPI[3.5], NSAPI[3.6] ou les servlets Java sont souvent préférés au standard CGI.

3.1.2. Adaptation à l'entreprise : Intranet

Aucun des mécanismes mis en oeuvre par Internet n'est exempt de défaut et il est relativement simple de trouver plus performant. En fait, la force de l'ensemble repose essentiellement dans son universalité.

La notion d'Intranet est née de l'intégration des principes d'Internet et des technologies déployées dans l'entreprise :

- on utilise le réseau local de l'entreprise,
- les données sont toujours gérées par un SGBD,
- les mécanismes utilisés pour interroger le SGBD sont toujours les mêmes.

3.2. Répartition des traitements

L'architecture trois tiers, encore appelée client-serveur de deuxième génération ou client-serveur distribué, sépare l'application en trois niveaux de service distincts :

- **premier niveau :** l'affichage et les traitements locaux (contrôles de saisie, mise en forme de données...) sont pris en charge par le poste client,
- **deuxième niveau :** les traitements applicatifs globaux sont pris en charge par le service applicatif,

- **troisième niveau** : les services de base de données sont pris en charge par un SGBD.

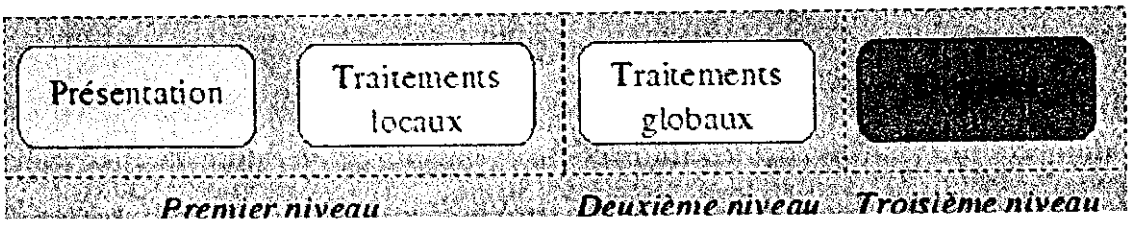


Figure 2.2: Le découpage d'une application en pavés fonctionnels indépendants

Tous ces niveaux étant indépendants, ils peuvent être implantés sur des machines différentes, de ce fait :

- le poste client ne supporte plus l'ensemble des traitements, il est moins sollicité et peut être moins évolué, donc moins coûteux,
- les ressources présentes sur le réseau sont mieux exploitées, puisque les traitements applicatifs peuvent être partagés ou regroupés (le serveur d'application peut s'exécuter sur la même machine que le SGBD),
- la fiabilité et les performances de certains traitements se trouvent améliorées par leur centralisation,
- il est relativement simple de faire face à une forte montée en charge, en renforçant le service applicatif.

Dans le cadre d'un Intranet, le poste client prend la forme d'un simple navigateur Web, le service applicatif est assuré par un serveur HTTP et la communication avec le SGBD met en oeuvre les mécanismes bien connus des applications client-serveur de la première génération.

Ce type d'architecture fait une distinction nette entre deux tronçons de communication indépendants et délimités par le serveur HTTP :

Le premier tronçon relie le poste client au serveur Web pour permettre l'interaction avec l'utilisateur et la visualisation des résultats. On l'appelle **circuit froid** et n'est composé que de standards (principalement HTML et HTTP). Le serveur Web tient le rôle de ``façade HTTP'',

- le deuxième tronçon permet la collecte des données, il est aussi appelé **circuit chaud**. Les mécanismes utilisés sont comparables à ceux mis en oeuvre pour une application deux tiers. Ils ne franchissent jamais la façade HTTP et, de ce fait, peuvent évoluer sans impacter la configuration des postes clients.

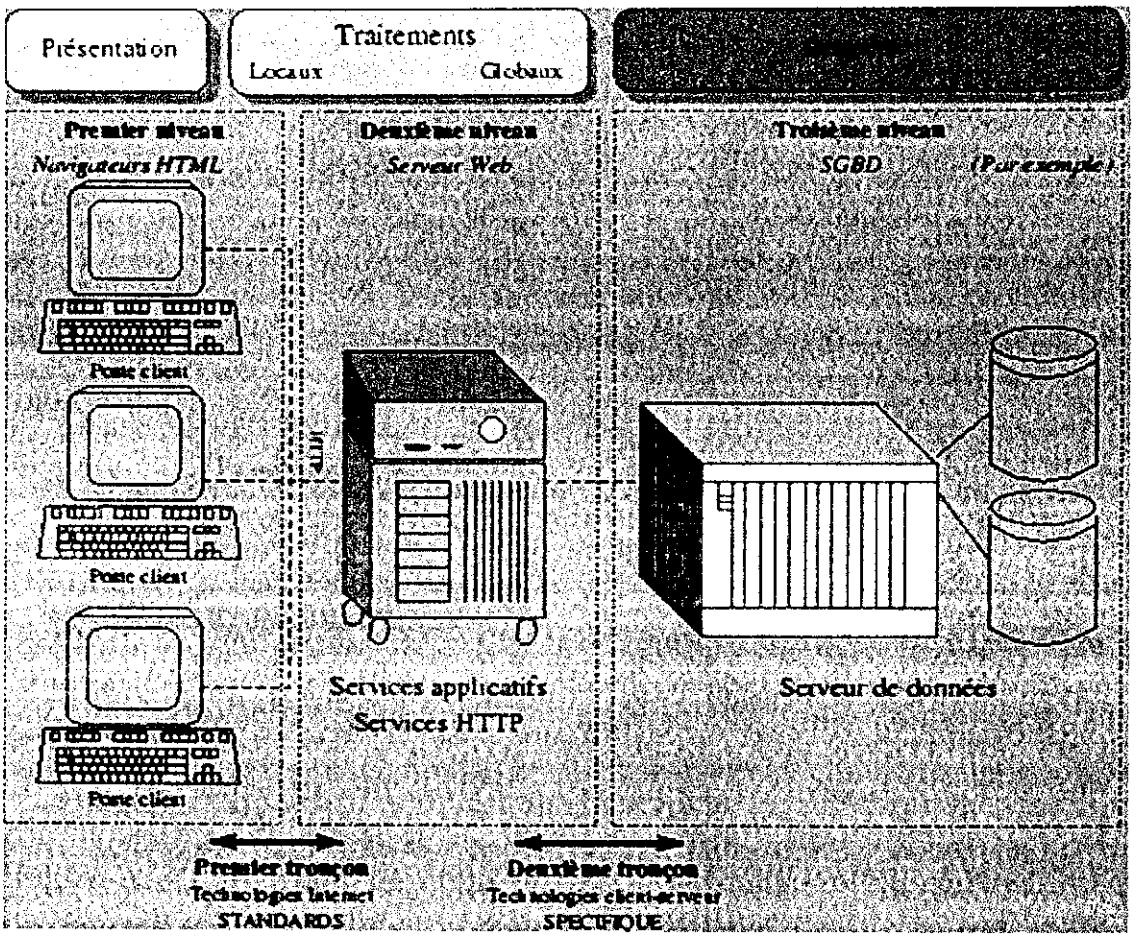


Figure 2.3: Répartition des couches applicatives dans une architecture trois tiers

3.3. Le client léger

3.3.1. Présentation

Dans l'architecture trois tiers, le poste client est communément appelé client léger ou *Thin Client*, par opposition au client lourd des architectures deux tiers. Il ne prend en charge que la présentation de l'application avec, éventuellement, une partie de logique applicative permettant une vérification immédiate de la saisie et la mise en forme des données. Il est souvent constitué d'un simple navigateur Internet.

Le poste client ne communique qu'avec la façade HTTP de l'application et ne dispose d'aucune connaissance des traitements applicatifs ou de la structure des données exploitées. Les évolutions de l'application sont donc possibles sans nécessiter de modification de la partie cliente.

Par exemple, un internaute se connectant à *www.yahoo.fr* pour effectuer une recherche provoque, sans même le savoir, l'exécution de traitements sur le serveur. Si ces traitements évoluent, ce qui doit arriver relativement souvent, le client continuera à utiliser le service sans se rendre compte des changements (sauf s'ils lui apportent de nouveaux services).

De plus, ce même internaute peut se connecter au serveur en utilisant tout type de poste client disposant d'un navigateur compatible HTML (PC sous Windows, Macintosh, Station Unix, WebPhone...).

On voit donc ici la force des architectures trois tiers par rapport au client-serveur de première génération. Le déploiement est immédiat, les évolutions peuvent être transparentes pour l'utilisateur et les caractéristiques du poste client sont libres.

3.3.2. Ergonomie

- Utilisation d'HTML

Les pages HTML, même avec l'aide de langages de script, sont loin d'atteindre les possibilités offertes par l'environnement Windows :

- le multifenêtrage n'est pas facile à mettre en oeuvre,
- le déroulement de l'application doit se faire séquentiellement,
- les pages affichées sont relativement statiques,
- le développement multiplateforme peut être contraignant[3.7].
- l'ergonomie de l'application est limitée aux possibilités du navigateur.

Pour ces raisons, certaines applications ne sont pas réalisables dans une architecture de type Intranet (infocentres, applications bureautiques...).

Dans les autres cas, l'appauvrissement de l'interface utilisateur est souvent le gage d'une plus grande facilité de prise en main de l'application.

Il est rare en effet de devoir suivre une formation pour apprendre à se servir d'un site Web particulier. La plupart du temps, une formation générale à l'ergonomie des sites Web suffit.

Cette perte de richesse peut donc se transformer en avantage, à condition de respecter une charte graphique et ergonomique cohérente pour toutes les applications Intranet d'une entreprise.

Il est possible d'aller au delà des possibilités offertes par le langage HTML en y introduisant des applets Java ou des contrôles ActiveX. Nous ne parlerons pas ici des modules d'extension du navigateur, encore appelés *plug-in*, qui étaient en vogue avant l'arrivée des solutions Java et ActiveX, car cette solution est trop contraignante à déployer.

- **Utilisation de Java**

Java est un langage de développement orienté objet et multiplateforme introduit par Sun en 1995. Il s'agit de l'adaptation à Internet du langage OAK[3.8], initialement étudié pour les environnements de petite taille. Il permet, entre autre, d'écrire de petites applications, appelées *applet*, pouvant être intégrées à des pages HTML pour en enrichir le contenu.

Le caractère multiplateforme de Java se prête bien à une utilisation sur Internet, où les caractéristiques des postes clients ne sont pas maîtrisées. Il repose sur l'utilisation d'un interpréteur de pseudo-code Java, pompeusement appelé ``machine virtuelle".

Les programmes Java ne sont pas compilés en code machine, mais en pseudo-code Java uniquement compréhensible par la machine virtuelle. Cette dernière interprète le code et se charge de lier les modules au moment de l'exécution, en les téléchargeant si nécessaire. La liaison dynamique des modules au moment de l'exécution permet d'optimiser le trafic réseau, puisqu'on ne charge que le strict nécessaire.

Pour ces raisons, un programme Java s'exécute plus lentement que son équivalent compilé. La compilation à la volée des programmes Java et plus encore, la technologie d'optimisation dynamique de code *HotSpot* de Sun, permettent de réduire l'écart de performance avec des programmes compilés.

Java propose aujourd'hui une large palette de composants graphiques[3.9] et multimédia qui permettent d'atteindre la richesse fonctionnelle des applications Windows.

Une applet Java est aussi capable d'exploiter directement un serveur de données en utilisant JDBC[3.10] ou de faire appel à des procédures distantes en utilisant RMI[3.11] ou CORBA[3.12]. Nous verrons ces mécanismes par la suite.

4. Conclusion

Etant donné que notre travail est basé sur les bases de données et le Web nous avons mis l'accent sur l'architecture trois-tiers et ses trois niveaux ainsi que l'environnement logiciel qui permettent a cette architecture de fonctionné sur le Web, pour la conception d'applications web et des pages dynamiques.
Notre but dans le chapitre suivant est d'étudier et de détailler l'approche composante avec les JavaBeans.

Chapitre 3

L'approche composant

1. Introduction :

Notre objectif dans ce chapitre est de présenter la réalisation technique de l'architecture trois-tires, et l'approche que nous avons adoptés pour la réalisation de notre application ; puis on verra en détail les composants JavaBeans et leurs fonctionnements ainsi que l'architecture orientée servlets.

2. Outils pour la réalisation de system d'information en architecture trois-tiers :

Nous avons donc vu ce qu'est une architecture trois-tiers ainsi que ses avantages. Nous allons maintenant présenter les choix techniques effectués pour la réalisation de cette architecture. En fait, il existe actuellement plusieurs solutions permettant ce type d'architecture. Ces technologies possèdent chacune leurs avantages et leurs inconvénients. On peut ainsi construire un système multi-tiers basé uniquement sur des technologies Microsoft comme ODBC, ASP, SQL Server... Ces technologies sont relativement efficaces mais ont pour principal inconvénient de n'être pas du tout portables. Kallisto utilise également des technologies spécifiques à Oracle basées sur Oracle Application Server, le serveur applicatif d'Oracle. Cette technologie est particulièrement efficace car elle permet de n'utiliser que des API natives à Oracle. Toutefois les systèmes développés avec ces outils ne pourront pas fonctionner avec autre chose. Cet inconvénient n'est pas obligatoirement important puisqu' Oracle fonctionne tout de même sur un grand nombre de systèmes (la plupart des UNIX et Windows). La solution choisie ici est basée sur le langage JAVA. Elle utilise une base de données Oracle, un serveur applicatif compatible J2EE nommé WebLogic, et un navigateur web comme client. Le JAVA étant un langage 100% portable, le développement du système ne dépendra absolument pas de la machine sur laquelle il fonctionnera. De plus WebLogic suivant les dernières spécifications de J2EE, les développements réalisés pourront fonctionner sur de nombreux autres serveurs applicatifs.

Oracle

Afin de stocker les données il fallait également une base de données. La société Kallisto étant spécialisée dans les bases Oracle, ce choix s'imposait. J'ai d'ailleurs pu profiter d'une formation interne dispensée la semaine de mon arrivée par un expert Oracle (10 ans d'expérience). Oracle est un système de gestion de base de données relationnelle très connu. Il est réputé pour être performant, fiable,... Oracle est en fait la référence en matière de base de données relationnelle. Les possibilités d'administration sont importantes et permettent de gérer des bases de tailles importantes (ce qui n'est pas primordial dans notre cas).

Java 2 Enterprise Edition (J2EE)

Le langage JAVA est un langage objet qui présente de nombreux avantages. Le premier d'entre eux est de supporter les notions de la programmation orienté-objet (encapsulation, héritage, classe, objet). Il permet une programmation relativement simple sans se préoccuper de notions complexes présentes en C++ (indirection de pointeurs, fonctions virtuelles,...). Mais ces intérêts sont relativement mineurs comparés à l'apport que représente les interfaces de programmation standardisés proposées par Sun. Ces interfaces, couplées à la compilation en un ByteCode exécutable sur une machine virtuelle, donnent aux applications un niveau de portabilité maximale. Dans notre cas nous sommes intéressés par le développement d'une application Intranet utilisant des accès à une base de données. Il existe en JAVA un grand nombre d'API destinées à réaliser ce type de programmation et regroupées sous le terme J2EE (Java 2 Enterprise Edition). Ces API forment ce qu'on nomme un Framework. Il s'agit de proposer des interfaces guidant le développeur vers une architecture prédéterminée. On profite donc à la fois d'un ensemble d'outils indispensables au développement mais aussi d'un guide pour l'élaboration d'une méthode de développement. Ce guide reste suffisamment flexible pour nous permettre d'adapter la structure de notre application à nos besoins. L'objectif majeur de J2EE est la réalisation d'applications en architecture distribuée. Les technologies JAVA intégrées dans cette plate-forme sont:

- Enterprise JavaBeans (EJB)
- Common Object Request Broker Architecture (CORBA[3.12])
- Java Servlets 2.1
- Java Server Pages 1.1 (JSP)
- Java Message Service (JMS)
- Java Transaction API (JTA)
- JavaMail 1.1
- Java Database Connectivity 2.0 (JDBC[3.10])
- Java Naming and Directory Interface 1.2 (JNDI)
- eXtensible Markup Language (XML)

3. l'approche composant :

Les composants sont des éléments indépendants et réutilisables qui encapsulent un comportement applicatif ou des données en un paquetage distincte. Tels des dispositifs de type <boites noire>, ils effectuent des opérations sans révéler leurs mécanismes internes. Comme une interface abstraite fait le lien entre le comportement applicatif et l'implémentation, il est ainsi épargné aux utilisateurs les détails complexes d'un code touffu. Les fonctionnalités apportées n'augmentent pas la complexité globale de l'application. En outre, les composants ne sont pas exclusifs à une application ni a une utilisation unique .ils utilisées comme des briques élémentaires dans des projets qui parfois,n'ont aucun rapport entre eux.
L'abstraction et la réutilisation sont les deux principes essentiels de la conception orientée composant. La figure 3-1 illustre cette approche en montrant comment on peut combiner des composants logiciels indépendants. [DKK01]

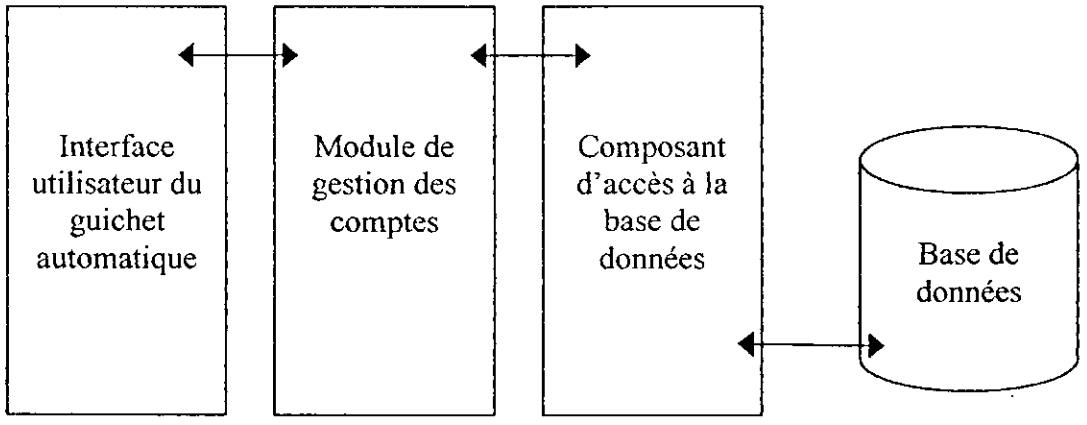


Figure 3-1 une application basée sur les composants

Les composants sont donc des éléments logiciels réutilisables qui peuvent être reliés pour construire une application. Un bon modèle de composants minimise le travail de codage des relations entre les différents éléments de l'application. Le principe des architectures de composants consiste à utiliser une interface commune pour manipuler des objets avec des outils de développement, la seule nécessité qui incombe à l'outille étant de prendre encharge l'interface.

3.1. Avantages des architectures à base de composants

Lorsqu'un architecte entreprend la conception d'une nouvelle maison, il se base sur des composants qui vent lui permettre tout à la fois de gagner du temps et de réduire le coûts engager, plutôt que de concevoir l'installation électrique à zéro, il va par exemple utiliser des composants déjà existants. Ainsi cet architecte ne va-t-il pas concevoir des systèmes de climatisations personnalisés, mais plutôt va-t-il les sélectionner à partir de modèles déjà disponibles sur le marché. Selon toute probabilité, cet architecte ne possédera pas le savoir-faire ni les ressource nécessaires qui lui permettraient de concevoir de tels systèmes. Inversement, le constructeur d'appareils de climatisation n'a aucun savoir-faire en construction de bâtiments. L'approche orienté composants permet à l'architecte et à l'entrepreneur de concentrer leurs efforts sur ce que, chacun de leur coté, savent le mieux faire. Il en va de même en informatique. Les architectures par composants offrent la possibilité de masquer la complexité des composants donnés par l'adjonction d'une interface, à travers laquelle le composant réagit avec son environnement ou avec d'autres composants.

On peut utiliser la même argumentation pour illustrer le concept de réutilisation et d'interchangeabilité. L'entreprise de construction a opté pour un climatiseur fixe par des écrous standard et fonctionnent sur une tension électrique standard. Le propriétaire de la maison pourra ainsi cet appareil par un nouveau modèle plus performant, et sans qu'il doive reconstruire toute la maison. La normalisation des environnements et des méthodes de conception a permis d'obtenir des systèmes souples, faciles à maintenir. Les composants logiciels ont été conçus pour opérer dans ces environnements spécifiques. L'existence de règles prédéfinies qui

régissent leur interaction avec les autres composants augmente d'autant leur interchangeabilité. [DKK01]

3.2. Fonctionnement des composants JavaBeans

Le composants JavaBeans sont des composants logiciels écrits en java, conformes aux spécifications exposées dans l'API des java beans, cette API qui a été crée par Sun en collaboration avec d'autres de l'industrie énonce les regèles que doivent suivre les développeurs de logiciels pour crées des composants logiciels indépendants et réutilisables, comme beaucoup de composants logiciels, les JavaBeans renferment un état et un comportement. Par l'utilisateur de balise de JSP conçues pour les JavaBeans dans leurs pages web, les développeurs de contenus peuvent tirer profit de la puissance de java en ajoutant des éléments dynamiques à leurs pages, sans pour autant écrire la moindre ligne de code java. Rappelons tout d'abord les principales caractéristiques des composants JavaBeans avant de détailler leur utilisation dans les JSP.

- Les conteneurs de JavaBeans :

Un conteneur JavaBeans est une application, un environnement ou un langage de programmation qui permet aux développeurs de faire appel à des composants, de les configurer et d'accéder a leurs données et a leurs méthodes, les applications qui se servent directement des composants JavaBeans sont exclusivement écrites en java mais les conteneurs permettent de les utilisés a des niveaux conceptuels plus élevés. Pour cela, les composants java beans exposent leurs fonctionnalités et leur comportement au conteneur peut ainsi les manipuler d'une façon plus intuitive. Le conteneur java beans définit ses propres critères de présentation et d'interaction avec le composant et écrit lui-même le code java qui en résulte.

Si vous avez déjà utilisé les outils visuels beans box de Sun, Visual age pour java d'IBM, visuel café de webgain ou d'autres outils de développement java, vous avez

déjà manipulé des composants. Ces ateliers de développements fonctionnent avec des conteneurs JavaBeans qui permettent de manipuler visuellement les composants. Ainsi peut-on par le simple déplacement d'icônes définir les caractéristiques des composants java, leur comportement, leur lien avec d'autres objets, et bâtir une application entière. L'application ainsi définie générer tout le code java nécessaire. Les conteneurs de JSP, d'une façon similaire, permettent aux développeurs créer des applications java basées sur le web sans récrire de code java. L'interaction avec les composants dans les JSP se fait par l'intermédiaire de balises qui peuvent se mêler a du code HTML classique. [DKK01]

- **Les propriétés de composants :**

Les conteneurs permettent de manipuler les composants en termes de propriétés (ce sont des attributs de composants JavaBeans identifiés par un nom, qui maintiennent son état de contrôlent son comportement). Un composant est défini par ces propriétés sans lesquelles il serait pratiquement inutilisable. Les propriétés d'un composant JavaBeans peuvent être modifiées pendant l'exécution par le conteneur pour contrôler les spécificités de son comportement. Les valeurs de propriétés constituent le seul mécanisme que le conteneur utilise pour mettre les JavaBeans à la disposition du développeur.
A titre d'exemple, supposons que l'on possède un JavaBeans watherBean qui détient des informations sur les conditions atmosphériques du moment et des prévisions météorologiques. Le JavaBeans peut récupérer ces informations en accèdent aux serveurs du National weather service ou les extraire d'une base de données, sachant que tout comme l'utilisateur de JavaBeans, il nous est d'aucune utilité de savoir comment ce composant obtient ses informations. Tout ce qui nous intéresse en tant que développeurs, c'est que weatherBean soit capable de nous fournir des informations telles que les températures actuelles, les maximales attendues ou les risques de précipitations. Chacun de ces éléments d'information est proposé au conteneur de JavaBeans sous forme d'une propriété du composant dont le »s valeurs sont accessibles par la page ou l'application web.

Chaque composant possédera un ensemble de propriétés en fonction des informations qu'il contient .on peut le personnaliser en initialisant soi-même les

valeurs des ses propriétés. Le créateur du JavaBeans va imposer des restrictions sur chacune de ses propriétés, ce qui lui permet de contrôler l'accès qui en fait. Une propriété peut être accessible en lecture seulement, en écriture seulement ou en mise à jour (lecture/écriture).c'est cette notion d'accessibilité qui permet au concepteur du JavaBeans d'imposer des restrictions sur la façon de l'utiliser. Ainsi, dans l'exemple weatherBean, cela n'aura aucun sens de permettre aux développeurs de modifier la valeur de la propriété du composant qui indique la température maximale de la journée. Cette information est gérée par le JavaBeans et ne devrait être accessible qu'en lecture. D'autre part, si le java beans est doté d'une propriété qui contient le code postal d'une région dont on connaître la météo, il serait normal de permettre aux développeurs de le spécifier. Une telle propriété serait accessible en, lecture/écriture.

- **Les propriétés liées et les propriétés de déclenchement :**

Certaines propriétés sont utilisées comme des déclencheurs, pour déclencher un comportement ou renvoyer des comptes rendus. La lecture de telles propriétés ou leur modification a pour conséquence immédiate la réalisation par un composant JavaBeans d'une certaine action sur le serveur : il peut s'agir mettre à jour les valeurs d'autres propriétés ou de lancer une autre tâche sur le serveur. La mise à jour de la valeur de la propriété de code postal par exemple pourrait conduire le JavaBeans à consulter les services du *National Weather Service* pour s'informer sur les conditions atmosphériques qui correspondent au nouveau code postal. Il mettra ensuite à jour ses autres propriétés relatives à la météo en conséquence. Dans ce cas, les propriétés qui ont trait à la météo et au code postal sont considérées comme des propriétés liées parce que la modification de la valeur de l'une d'entre elles met à jour les valeurs des autres.

- **Les propriétés indexées**

Il est également possible pour une propriété unique de mémoriser plusieurs valeurs. Ces propriétés sont dites *indexées* car chaque valeur enregistrée dans la propriété est accessible via un numéro d'indice qui pointe sur

la valeur souhaitée. On peut par exemple réclamer la première valeur de la liste, la troisième ou la vingtième. Ainsi, notre WeatherBean pourrait avoir une propriété qui prenne en charge les prévisions météorologiques pour les cinq jours à venir. Cependant, tous les conteneurs de JavaBeans ne fournissent pas un mécanisme simple pour manipuler directement ces propriétés multivaluées. Par exemple, les balises de Java-Beans des JSP ne reconnaissent pas les propriétés indexées. Il faut utiliser en lieu et place des scriptlets, des expressions JSP ou les balises JSP personnalisées (détaillées aux chapitres 13 et 14) pour pouvoir accéder à de telles propriétés.

- **Les types de données des propriétés :**

Les propriétés des JavaBeans peuvent être utilisées pour prendre en charge une quantité importante d'informations. Par exemple, les propriétés de WeatherBean peuvent mémoriser des informations très diverses comme les températures, les risques de précipitations, les prévisions météo, les codes postaux, etc. Chaque propriété d'un composant JavaBeans ne peut contenir qu'un type particulier de données. Ses valeurs ont un type Java, qui est utilisé en interne par le composant et dans le code Java généré par le conteneur des JavaBeans. Les propriétés peuvent bien entendu supporter n'importe quel type primitif de Java tel que int (entier simple précision) ou double (entier double précision), ainsi que des objets Java tels que String et Date. Elles peuvent également mémoriser des objets définis par les utilisateurs, voire même d'autres JavaBeans. Les propriétés indexées mémorisent généralement un tableau de saleurs qui ont le même type de données.

Le conteneur de JavaBeans détermine la façon dont il convient de manipuler les valeurs de Propriétés d'un composant. On va référencer les valeurs des propriétés par leur type de données en utilisant des scriptlets et des expressions Ainsi, si une propriété contient des valeurs de type «entier on ne peut obtenir et déposer que des Valeurs entières. Cependant, avec les balises Beans, on traite chaque propriété comme si elle ne contenait que du texte String). Lorsqu'on initialise la saleur d'une propriété d'un JavaBeans, on lui transmet du texte. De même, lorsqu'on effectue une

lecture du contenu d'une propriété, on va récupérer du texte indépendamment du type de données interne utilisé dans le JavaBeans. Cette stratégie orientée texte permet de manipuler d'une façon qui soit simple les balises Bean de JSP et se marie bien avec HTML.

Le conteneur de JSP effectue toutes les conversions de type nécessaires. Par exemple, lorsqu'on initialise une propriété de type entier, le conteneur de JSP fait les appels Java nécessaires pour 'convertir les suites de caractères numériques qu'on lui a fournies en une valeur entière. Bien entendu, ce processus de conversion nous oblige à transmettre les valeurs textuelles appropriées de façon que Java puisse les convertir correctement dans le type de données natif. Si une propriété prend en charge des valeurs à virgule flottante, par exemple. Une erreur est générée si on essaie de lui affecter des valeurs comme banane, pain, cent ou (3,9).

De. Concepteurs astucieux de JavaBeans peuvent ainsi contrôler les valeurs des propriétés en acceptant des valeurs de type chaîne de caractères pour des propriétés qui ne sont pas de ce type et en effectuant eux-mêmes les conversions. Cette technique doit être utilisée pour toute valeur qui n'est ni une chaîne de caractères ni un type primitif de Java. Par conséquent, il pourrait être parfaitement autorisé d'attribuer à une propriété de type entier la valeur cent si le concepteur du JavaBeans l'a préparée à une telle affectation.

4. Les différentes fonctions des composants JavaBeans :

Les JavaBeans peuvent être répartis selon trois grandes catégories fonctionnelles : les composants graphiques utilisés dans les interfaces utilisateur graphiques. Les composants de données qui permettent l'accès aux informations, et les composants de services (appelés aussi *worker beans)* qui peuvent effectuer des tâches ou des calculs spécifiques. Bien entendu. Un JavaBeans *peut* appartenir *à* plusieurs de *ces catégories à la fois.*

- **Les composants graphiques**

En raison du développement des composants graphiques. Cette utilisation des composants JavaBeans est une des plus répandues. Ce sont des éléments tels que les champs de textes, les listes de sélection ou tout objet graphique utile pour la conception d'interface utilisateur. En regroupant des composants graphiques dans des JavaBeans, les environnements de développement Jasa peuvent profiter des avantages offerts par le support de la programmation graphique de ces composants. Les développeurs peuvent créer leurs interfaces simplement en assemblant les éléments souhaités. Toutefois, comme les composants graphiques ont été conçus pour s'exécuter avec les applications Java graphiques, ils ne sont pas compatibles avec les JSP qui sont plus orientées vers la conception d'interfaces HTML.

- **Les composants de données**

On peut avec ces JavaBeans accéder aux données que le composant lui-même n'a pas forcément la capacité de collecter ou de générer. Le calcul et la collecte des données mémorisées dans les JavaBeans de données sont de la responsabilité d'un autre composant ou service plus contple.se. Ces JavaBeans ne sont accessibles qu'en lecture. Ils permettent donc de récupérer des données mais pas de les modifier.

Certains JavaBeans de données permettent d'initialiser des propriétés afin de contrôler le format et le filtre des données avant de les retourner via d'autres propriétés. Par exemple, un **AccountStatusBean** pourrait être doté d'une propriété currentType qui contrôlerait si la propriété associée au solde du compte retourne les données en dollars, en livres ou en francs suisses. Les JavaBeans de données, en raison de leur simplicité, sont utiles pour normaliser l'accès aux informations en fournissant une interface stable.

- **Les composants de services**

Comme leur nom l'indique, ces JavaBeans permettent d'accéder à un comportement ou un service particulier. C'est la raison pour laquelle ils sont quelquefois appelés les JavaBeans travailleurs. Ils peuvent extraire des informations des hases de données, effectuer des calculs ou formater des informations. Comme il n'y a pas d'autre possibilité pour interagir avec un JavaBeans que de le faire par ses propriétés, on sa accéder à ses services de la même façon. Dans une conception classique, on va initialiser les valeurs de certaines propriétés qui contrôlent

5. Architecture des applications web :

Quelle que soit la complexité d'une application web, il est toujours conseillé d'analyser son architecture selon trois logiques (figure 3-2) :

- ✓ La couche de présentation, qui présente des résultats à l'utilisateur et en détermine l'apparence ;
- ✓ La couche de contrôle, qui contrôle le flot d'exécution de l'application ; et
- ✓ La couche de logique applicative « modèle», qui gère les données de l'application, exécute les traitements et communique directement avec les ressources sous-jacentes.

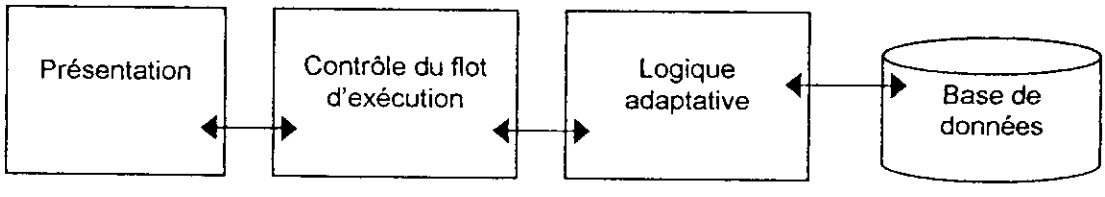


Figure 3-2 architecture des applications web par niveaux logique

5.1 Architecture orientée servlets :

C'est une approche qui consiste en effet à ne confier aux page JSP que les aspects de présentation, en délèguent les aspects « back-end» de flot d'exécution et de logique applicative à une ou plusieurs servlets. Les requêtes sont alors directement dirigées aux pages de présentation JSP via une servlet, qui effectue toutes les action nécessaire au fonctionnement de l'application. Une servlet peut jouer l'un ou plusieurs des trois rôles suivants :

- Exécuter des actions pour le compte d'une page JSP, par exemple soumettre une commande ;
- Livrer à une page JSP des données pour quelles soient affichées, par exemple un enregistrement de base de données ;
-Contrôler l'enchaînement des pages JSP d'une même application.

Après avoir effectué son traitement, une servlet transfère la requête vers la page JSP appropriée, voire vers une page HTML statique (voire figure 3-3)

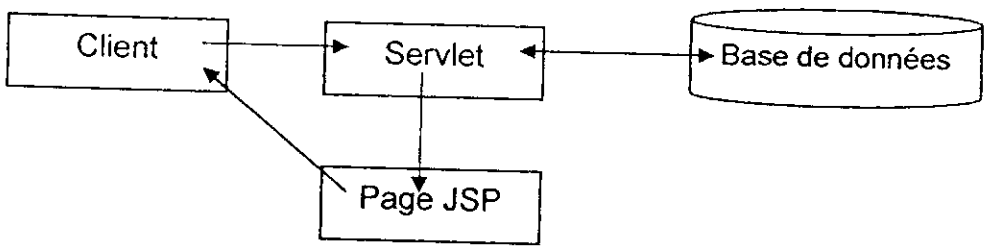


Figure 3-3 Enchaînement d'exécution d'une application basée sur des servlet

Ceux qui connaissent le pattern de médiateur (gestion de comportement) s'apercevront qu'il s'agit d'appliquer le même principe, mais à des pages et composants JSP plutôt qu'a des objet java. Le principe de ce pattern est de créer un composants de l'application interagissent, entre eux et avec les sources de données, cette approche permet de découpler la relation qui existait entre les pages jsp :on peut les coder indépendamment sans forcément inclure les détails de fonctionnement des autres pages. En outre, la séparation entre présentation et logique applicative est encor plus nette. Plus concrètement, il s'agit de minimiser la

quantité de traitements effectués par les pages elles-mêmes, pour s'appuyer sur des servlets dédiées. Le code JSP frontal est moins complexe car il se limite désormais au simple affichage de données ou à la collecte de données utilisateur.
De même, les servlets n'ont à contenir aucun détail de présentation des informations ; elle gère que le contrôle de l'exécution et la production des données à passer aux pages jsp à des fins de présentation à l'utilisateur. [DKK01]

6. Conclusion :

Un composant JavaBeans est une classe java qui obéit à des conventions de nommage et de conception simples qui précisées dans les spécifications JavaBeans Ces composants ne doivent pas nécessairement étendre une classe de base spécifique ni implémenter une interface particulière. Une classe qui applique les conventions JavaBeans et qui est manipulée en respectant ces conventions est un composant JavaBeans.

Chapitre 4

Analyse et conception du système

I. INTRODUCTION :

La modélisation et la spécification orienté objet est une méthode pour penser et écrire des problèmes organiser autour de concepts du monde réel.

- ✓ Comprendre des problèmes.
- ✓ Communiquer avec des experts
- ✓ Modéliser avec des organisations
- ✓ Préparer la documentation
- ✓ Concevoir des programmes et des bases de données.

Le processus comporte les étapes suivantes :

- on réalise un modèle analytique des aspects essentiels de l'application, sans aucun priori d'implémentation. Ce modèle comporte les objets essentiels du domaine applicatif avec leur relation.
- ensuite en prend des décision de conception et on affine le modèle initial pour préciser et optimiser l'implémentation.
- Les objets applicatifs ainsi écrits forment le cadre de la programmation à venir.
- Enfin, on traduit le modèle dans un langage de programmation quelconque

1.1. Qu'est ce que l'orienter objet ?

On déclare orienté objet un logiciel organiser autour d'entités groupant des données arbitraire (comme les struct c, et les record pascal) et des procédures.

- on appelle encapsulation le processus d'agrégation des ces données et les processus d'autres caractéristiques sont habituellement requises des objets au sens propre.
  - l'identité
  - la classification
  - le polymorphisme
  - l'héritage

1.1.1 l'identité

Les objets sont des entités qui peuvent être distinguées. Notamment, deux objets distincts peuvent avoir exactement les mêmes données membres.

1.1.2 la classification

Les objets ayant des attributs (données et comportements) similaire sont groupés dans une classe. La classe décrit tous les caractéristique des objets de sont type, une classe qui abstrait un objet du mondes réel intègre les données qui semblent pertinentes en fonction du degré de détail voulu.

1.1.3 le polymorphisme

La même fonction peut avoir des effets différents sur des objets de classes distinctes.

1.1.4 l'héritage

Des caractéristiques partagées par différentes classes peuvent être décrites dans une classe intermédiaire dont les autres héritent, ce qui permet de les décrire par les seules modifications
Qu'elle apporte au schéma de base .l'héritage apporte une grande puissance d'expression et d'abstraction au modèle objet.

II. La methode de concetion OMT etl e langage de modelisation UML:

1. Object Modeling Technique (OMT)

C'est une méthode d'analyse orientée objet pour le développement des système d'information. OMT apporte deux choses [CEE00] :

- une méthodologie pour le développement orienté objet.
- une notation graphique pour la communication de concepts orientés objet. Dans cette partie nous utiliserons le langage de modélisation UML.

1.1. Méthodologie orientée objet

La méthodologie consiste en la réalisation d'un modèle du domaine applicatif, complété par des détails d'implémentation au cours de la phase de spécification et conception du système le processus comporte les étapes suivantes : analyse, modélisation du système, modélisation des objets, programmation.

1.1.1. Analyse

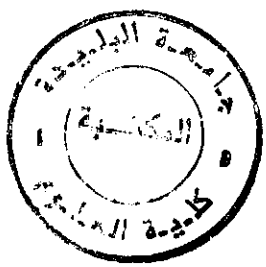
- à partir d'un énoncé du besoin, l'analyste construit (avec le demandeur) un modèle du monde réel qui fasse apparaître ses propriétés pertinentes.
- le modèle issu de l'analyse est une description précise, concise, de ce que le client demande, non de comment on peut le faire.
- les objets du modèle sont des objets du domaine, jamais des objets issus de l'informatique.
- une bonne analyse peut être comprise et critiquée par des experts du domaine qui ne sont pas des informaticiens.
- une bonne analyse ne comporte aucune décision d'implantation [LAU96].

1.1.2. Modélisation du système

- l'ingénieur (system designer en anglais) prend des décisions de haut niveau sur l'architecture du système.
- pendant cette phase, le système cible est organisé en sous systèmes pertinents relativement au modèle initial et aux choix d'architecture.
- l'ingénieur décide des caractéristiques à optimiser, définit une stratégie pour aborder le problème, et fait une première évaluation des besoins en ressources humaines [LAU96].

1.1.3. Modélisation des objets

- l'ingénieur décrit un modèle détaillé basé sur l'analyse et le modèle du système, qui respecte les choix stratégiques.
- l'accent est ici porté sur les structures de données et les algorithmes nécessaires pour réaliser chaque classe.
- les classes initiales restent pertinentes, mais elles sont complétées par les classes d'implémentation, choisies en fonction de besoins concrets en performance exprimés pour le système final.
- les objets initiaux et les objets informatiques sont décrits dans le même langage, bien qu'ils figurent dans des plans conceptuels distincts [LAU96].



1.1.4. Programmation

- dans cette phase bien connue, les classes et relations décrites précédemment sont traduites dans un langage de programmation.
- la programmation devrait être une part mineure de l'ensemble du processus, les décisions importantes ayant déjà été prises.

- le langage cible influence toujours la conception dans une certaine mesure, mais celle ci ne devrait jamais dépendre de détails trop pointus d'un langage donné.

- la programmation doit obéir à des règles de contrôle qualité suffisantes pour :
  - garantir la traçabilité des programmes par rapport à la spécification (vérification).
  - aboutir à un résultat suffisamment réutilisable, extensible, portable ... [LAU96].

2. Unfied modeling language (UML)

2.1 Introduction a l'UML:

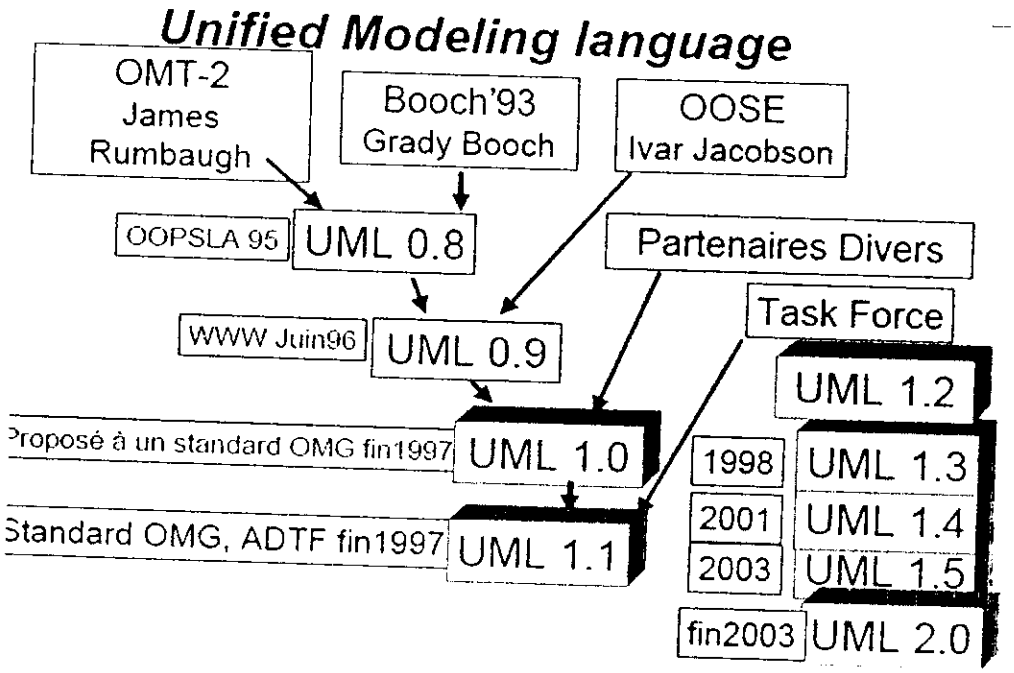


Figure 4-1 origines de uml [ROO]

Pourquoi modéliser

. Un modèle est une simplification de la réalité qui permet de mieux comprendre le système à développer.

. Il permet

- De visualiser le système comme il est ou comme il devrait l'être.
- De valider le modèle vis à vis des clients
- De spécifier les structures de données et le comportement du système.
- De fournir un guide pour la construction du système.
- De documenter le système et les décisions prises.

Qu'apporte la modélisation objet

. Plus grande indépendance du modèle par rapport aux fonctionnalités demandées.

. Des fonctionnalités peuvent être rajoutées ou modifiées, le modèle objet ne change pas.

. Plus proche du monde réel.

Les objectifs d'UML

• Représenter des systèmes entiers

• Etablir un couplage explicite entre les concepts et les artefacts exécutables

• Prendre en compte les facteurs d'échelle

• Créer un langage de modélisation utilisable à la fois par les humains et les machines

Recherche d'un langage commun unique

– Utilisable par toutes les méthodes

– Adapté à toutes les phases du développement

– Compatible avec toutes les techniques de réalisation

UML un langage

. UML n'est pas une méthode

. UML est un langage de modélisation objet

. UML a été adopté par toutes les méthodes objet

. UML est dans le domaine public, c'est une norme

UML un langage pour

. Visualiser

. Chaque symbole graphique a une sémantique

. Spécifier De manière précise et complète, sans ambiguïté,

. Construire Les classes, les relations SQL peuvent être générées automatiquement

. Documenter Les différents diagrammes, notes, contraintes,

Exigences seront présentés dans un document.

UML et les domaines d'utilisation

. Systèmes d'information des entreprises

. Les Banques et les services financiers

. Télécommunications

. Transport

. Défense et aérospatiale

. Scientifique

. Applications distribuées par le WEB

Les 9 diagrammes en UML

UML est un langage graphique et repose sur neuf types de diagrammes. Chacun de ces diagrammes utilise le même principe : les concepts sont représentés par des symboles, et les relations entre les concepts sont représentées par des lignes qui relient les symboles. Le vocabulaire et la grammaire d'UML sont très réduits.

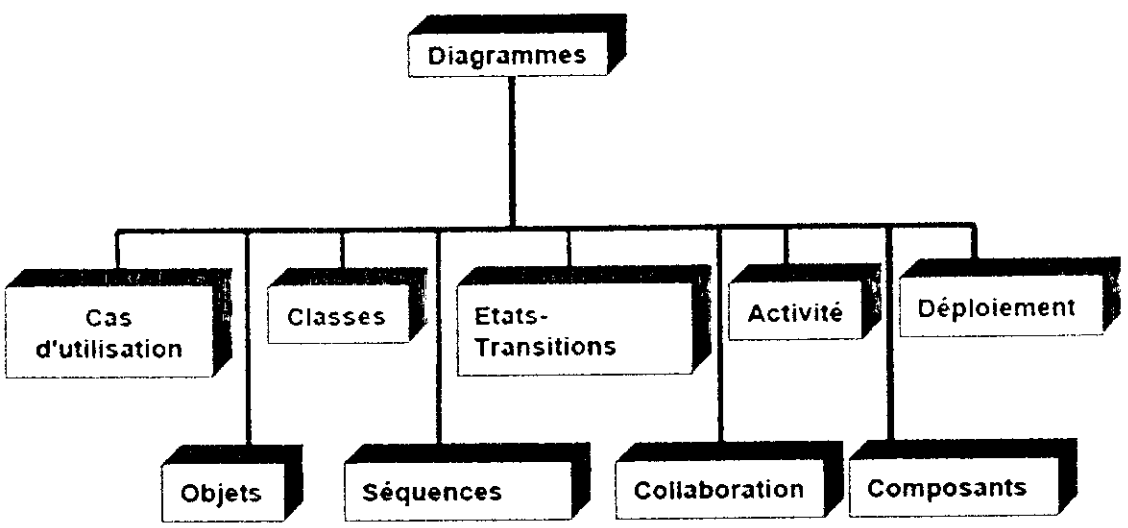
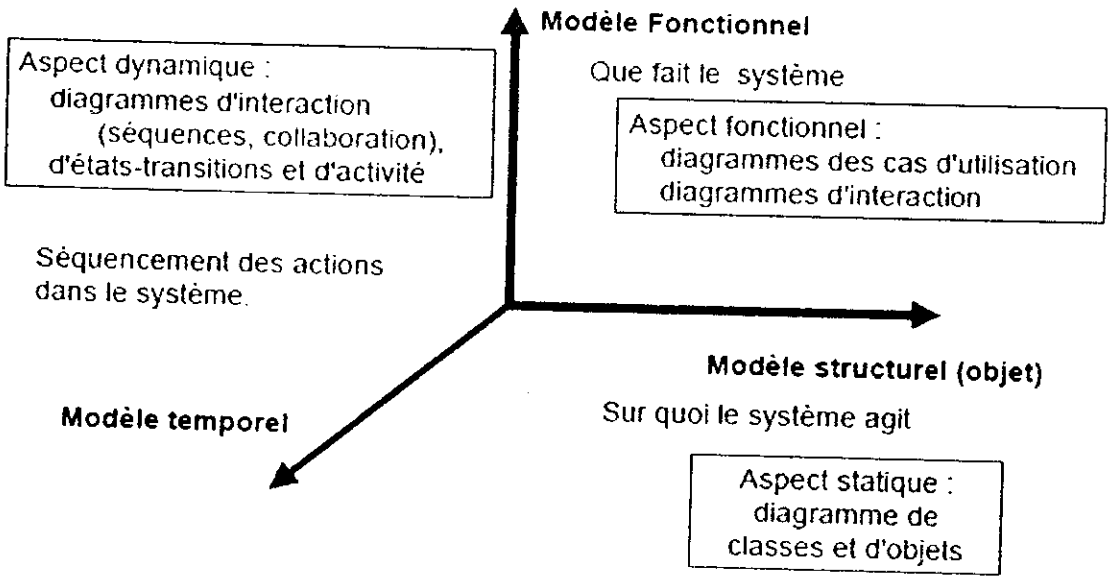


Figure 4-2 les 9 diagrammes de l'uml [ROO]

Les trois composantes d'une modélisation



Phases de la modélisation

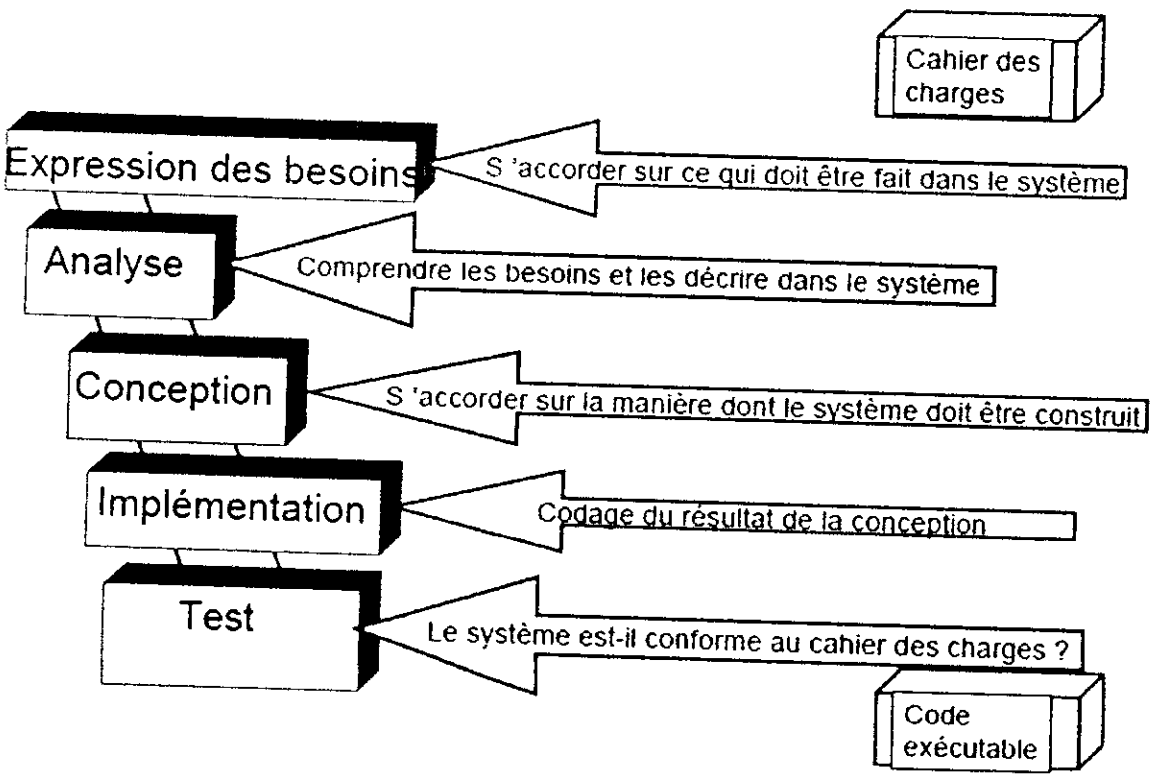


Figure 4-3 Phases de la modélisation [ROO]

2.2 Les diagramme de l'UML

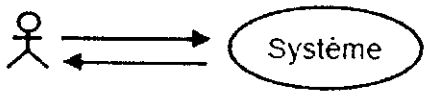
2.2.1 Le diagramme des Use Cases ou des cas d'utilisation

-> Ce que doit faire le système

Sans spécifier comment il le fait

But des Use Cases

Les cas d'utilisation représentent les Fonctionnalités que le système doit savoir faire. Chaque cas d'utilisation décrit un ensemble d'interactions successives d'une entité en dehors du système (utilisateur) avec le système lui même pour réaliser une fonctionnalité.



Les Uses Cases permettent :

• De connaître le comportement du système sans spécifier comment ce comportement sera réalisé.

• De définir les limites précises du système

• Au développeur de bien comprendre l'attente des utilisateurs et les experts du domaine.

De plus les Use Cases sont :

• Des instruments de validation et de test du système en cours et en fin de construction.

Modèle des cas d'utilisations

. Un diagramme de cas d'utilisation définit :

. le système

. les acteurs

. les cas d'utilisations

. les liens entre acteurs et cas d'utilisations

. Un modèle de cas d'utilisation se définit par

. des diagrammes de cas d'utilisation

. une description textuelle des scénarios d'utilisation

. une description de ces scénarios par

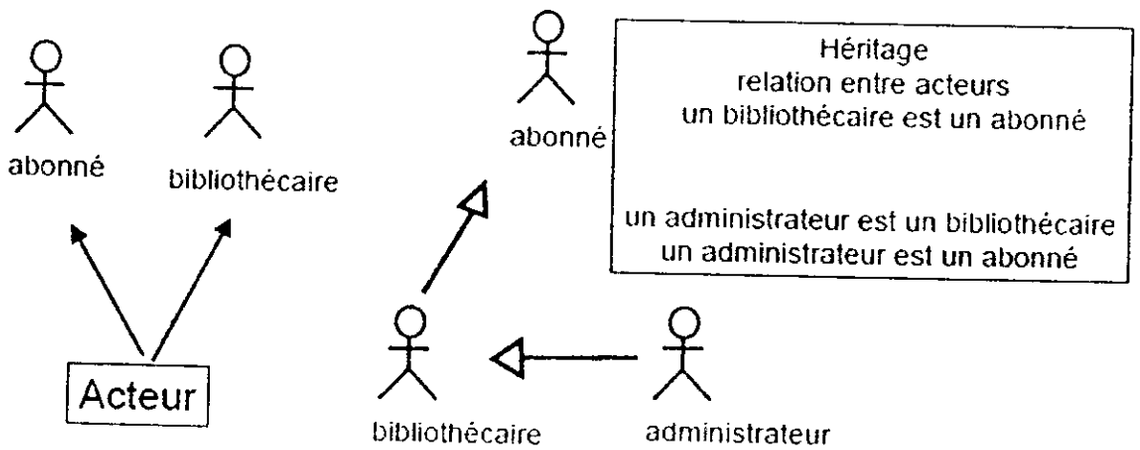
. les diagrammes de séquences

. les diagrammes de collaboration

Les Acteurs

. Un acteur représente une personne ou un périphérique qui joue un rôle (interagit) avec le système.

. Relation entre acteurs : généralisation (héritage)



Les cas d'utilisation (use-case)

. Un cas d'utilisation est un moyen de représenter les différentes possibilités d'utiliser un système.

. Il exprime toujours une suite d'interactions entre un acteur et l'application.

. Il définit une fonctionnalité utilisable par un acteur.

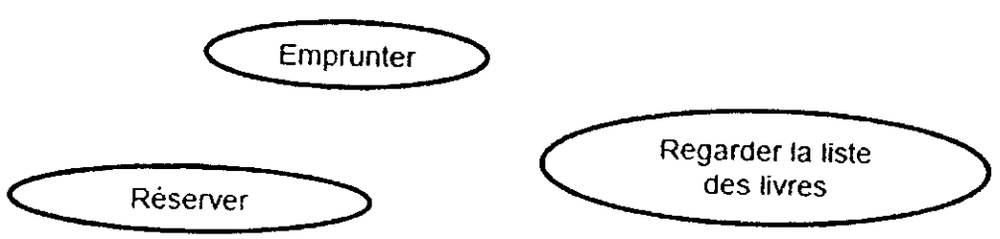
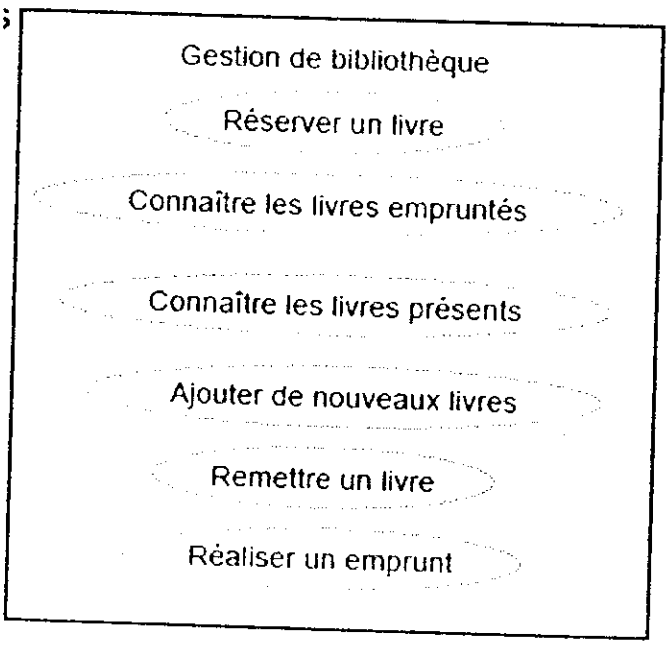


Figure 4-4 les cas d'utilisation

Le système

. Le système définit l'application informatique, il ne contient donc pas les acteurs, mais les cas d'utilisation et leurs associations



Exemple

Les diagrammes de cas d'utilisation

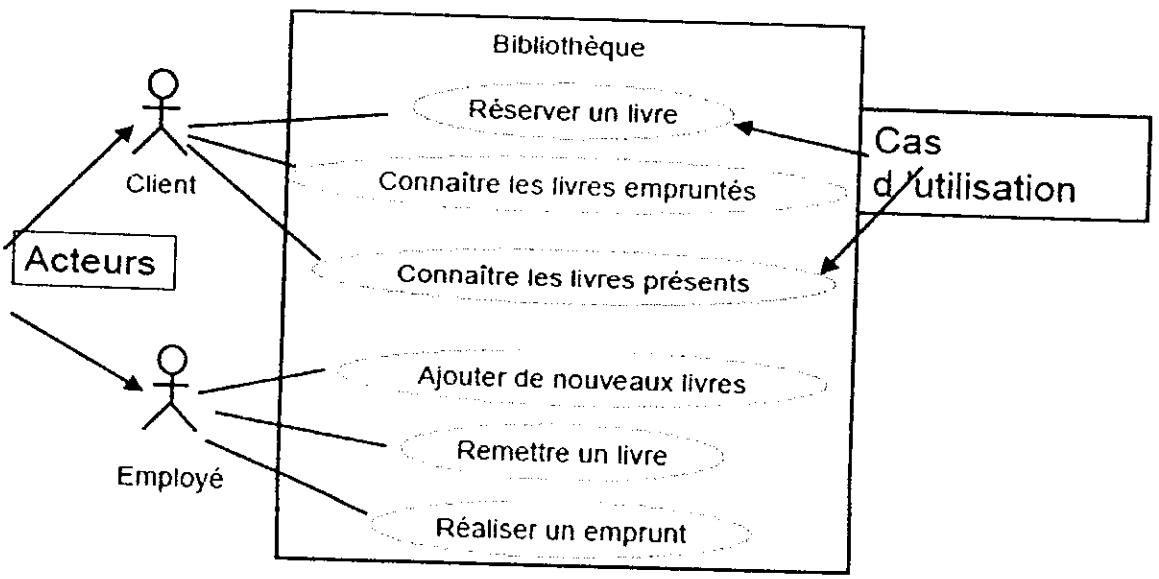


Figure 4-5 diagrammes de cas d'utilisation

Scénarios d'un cas d'utilisation

. La description d'un cas d'utilisation se fait par des scénarios qui définissent la suite logique des interactions qui constituent ce cas.

. On peut définir des scénarios simples ou des scénarios plus détaillés faisant intervenir les variantes, les cas d'erreurs, etc.

. Cette description se fait de manière simple, par un texte compréhensible par les personnes du domaine de l'application.

. Elle précise ce que fait l'acteur et ce que fait le système

. La description détaillée pourra préciser les contraintes de l'acteur et celles du système.

Exemple Scénarios d'un cas d'utilisation

description simplifiée

Le client se présente devant un terminal:

- (1) Le système affiche un message d'accueil.
- (2) Le client choisit l'opération réservation parmi les différentes opérations proposées.
- (3) Le système lui demande de s'authentifier.
- (4) Le client donne son identification (nom, mot de passe).
- (5) Le système lui demande de choisir un livre.
- (6) Le client précise le livre qu'il désire.
- (7) Le système lui précise si un exemplaire du livre lui est réservé.

2.2.2 Les diagrammes de séquence

Ces diagrammes représentent également la dynamique de fonctionnement du système également. En revanche, la représentation temporelle des événements est mise en avant plutôt que la représentation spatiale. [FAP00]

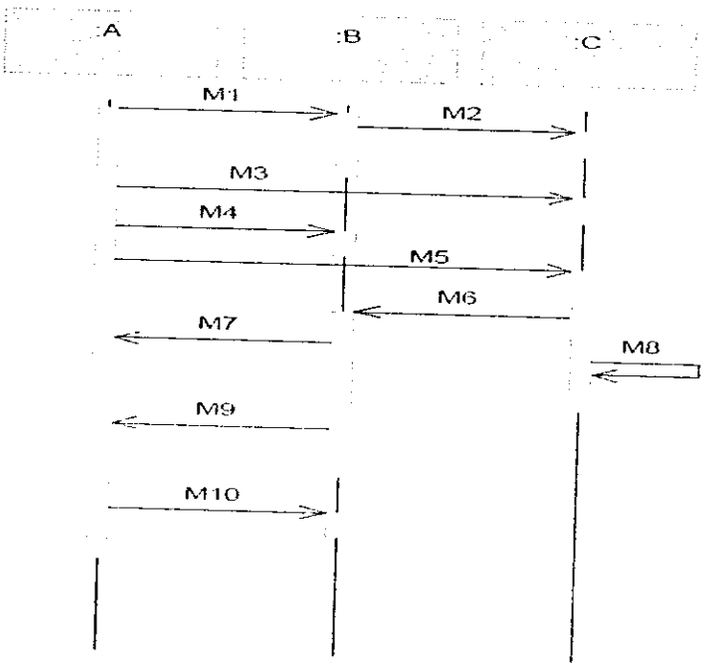


Figure 4-6 diagramme de séquence

Scénario par diagramme de séquences

. Suite aux descriptions textuelles, le scénario peut être représenté en utilisant un diagramme de séquences.

Le diagramme de séquences permet:

.de visualiser l'aspect temporel des interactions

.de connaître le sens des interactions (acteur vers système ou contraire)

Exemple

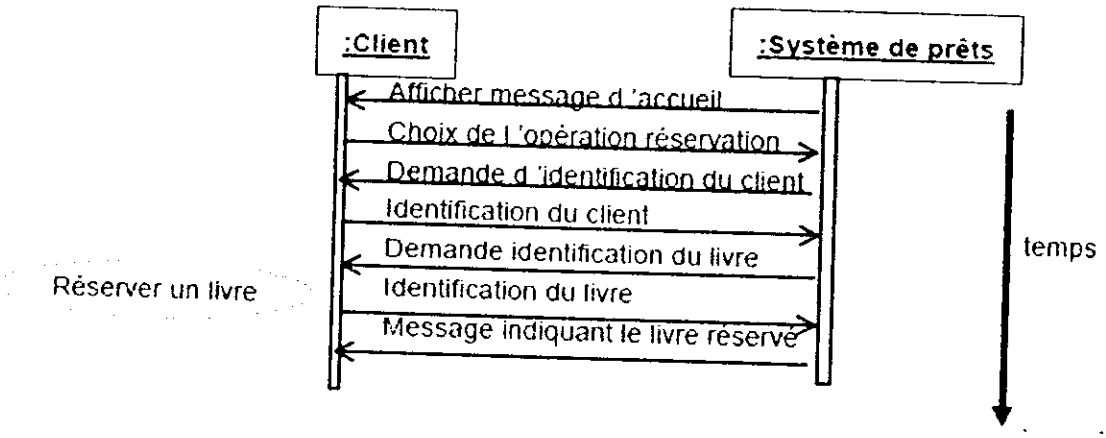


Figure 4-7 exemples de diagramme de séquence [ROO]

2.2.3 Les diagrammes de collaboration

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait

figurer les envois de messages, annotés par leur ordre d'apparition. [FAP00]

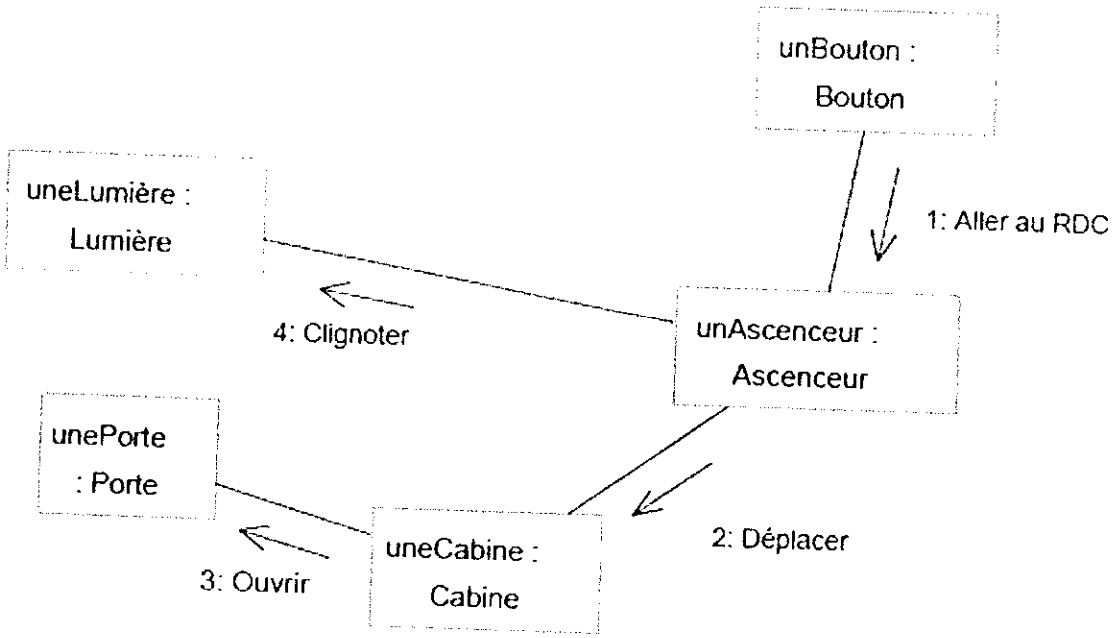


Figure 4-8 diagramme de collaboration

2.2.4 Les diagrammes de classes

Ces diagrammes décrivent l'architecture du système; on y représente les classes et les relations entre classes, qu'elles soient d'héritage, d'agrégation ...

La classe

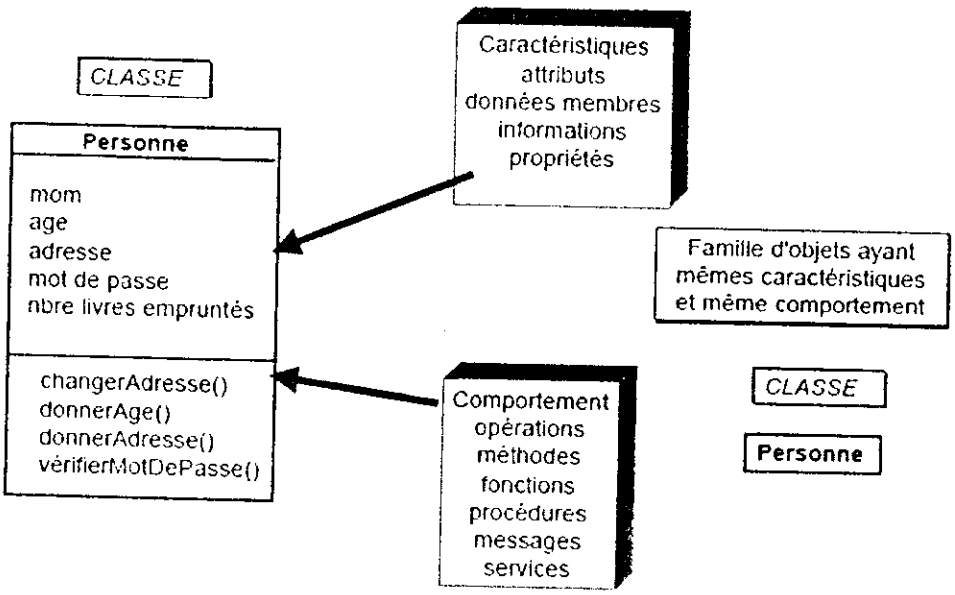
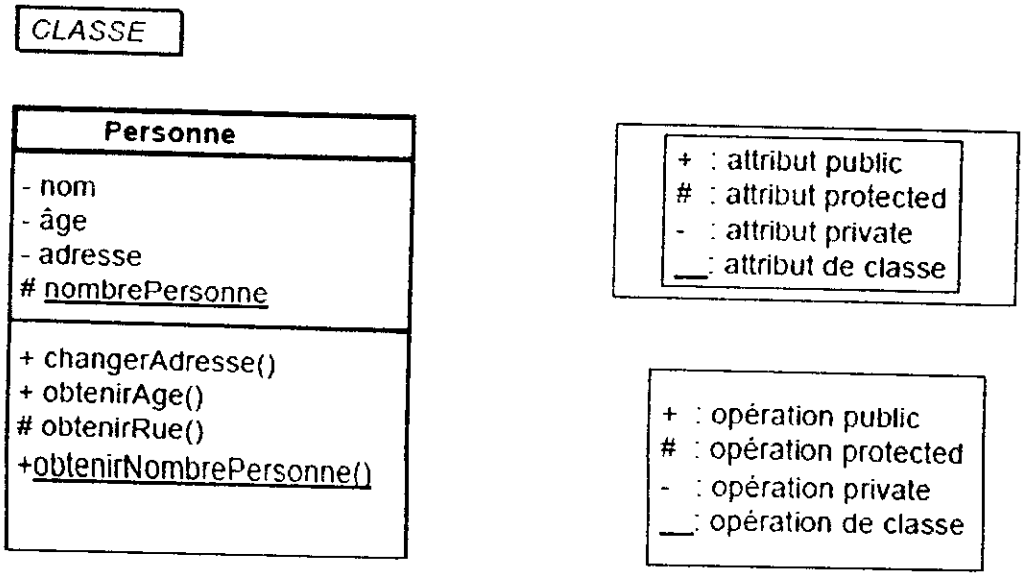


Figure 4-9 les classes

Protection des attributs et des Opérations



Exemple

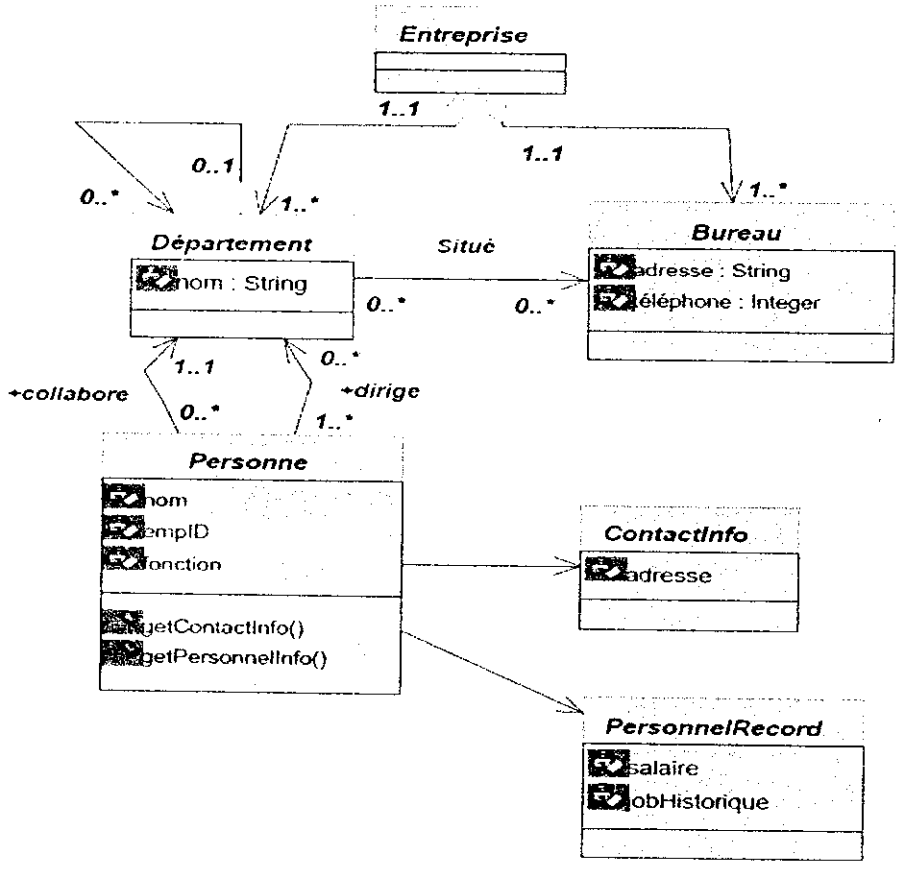


Figure 4-10 diagramme de classes [FAP00]

Résumé

On a présenté une méthode de conception OMT et un langage de modélisation d'un system d'information par objets. Le langage de modélisation UML respecte les trois étapes de l'étude : l'étape statique, dynamique, fonctionnelle. Statique pour aboutir a un modèle objet, dynamique pour décrire la réalisation des objets du system aux événements et les itération entre les objets eux-mêmes, et enfin, l'étape fonctionnelle qui comprend des contraintes entre les valeurs, les information de contrôles et de structure de d'objet appartenant respectivement au modèle dynamique et au modèle objet.
Nous allons utiliser cette méthode pour la conception de notre system

3. Analyse et conception :

Pour concevoir notre système, nous avons choisis la méthode de conception « OMT » qui sera appliquée en adoptant le langage de modélisation objet UML.
La technique de modélisation par objet (uml) associe trois modèles liés mais distinct : le modèle objet, le modèle dynamique et le modèle fonctionnel.
Le modèle objet décrit la structure des données sur lesquelles le modèle dynamique et le modèle fonctionnel opèrent.
Le modèle dynamique décrit la structure de contrôle des objets, met en évidence les décisions qui dépendent de la valeur des objets et invoque des fonctions.
Le modèle fonctionnel décrit les fonctions invoquées par les opérations du modèle objet et du modèle dynamique.

3.1. Analyse

3.1.1. ANALYSE DES BESOINS :

La base de données doit être accessible via les réseaux intranet et Internet, donc elle assure un accès non limité dans le temps et le lieu avec une gestion de ressources.
Le système doit assurer aux agents des services tel que : consultation, recherche..., et ceci selon les privilèges qu'on leur donne lors de leurs inscription.

- Le système a le contrôle des ressources : ajout, modification, suppression de données.
- Contrôle des abonnés : inscription d'un utilisateur, lister les utilisateurs, modifier les coordonnées d'un utilisateur,...
- Attribution des privilèges aux abonnées : simple consultation, consultation et recherche....

Les besoins selon l'utilisateur du système seront :

Agent :

- Consulter un son répertoire,
- Rechercher des informations selon des critères bien spécifiques.
- Contacter un agent service,

Administrateur:

- Lister les utilisateurs,
- Supprimer un privilège à un utilisateur,
- Affecter un privilège a un utilisateur.

Agents du service :

- Consulter la base de données
- Contacter un utilisateur,
- Faire des Recherches.
- Modifier les informations propres à un utilisateur.

Un administrateur est considéré aussi comme utilisateur avec privilèges donc il a accès à l'ensemble des fonctionnalistes du système

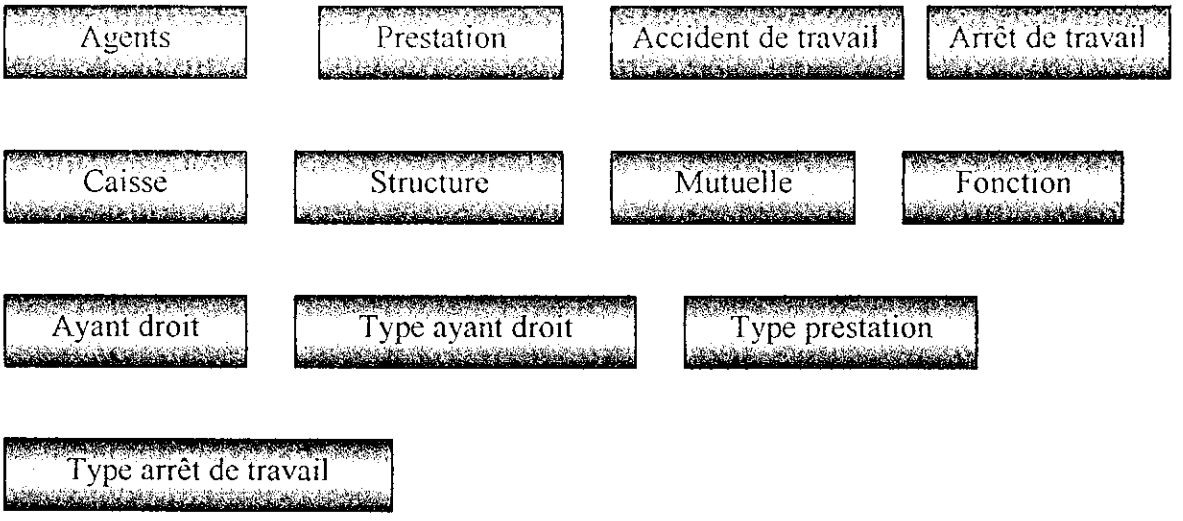
La phase d'analyse, première phase de la modélisation OMT, a pour objectif de décrie de manière précise, concise, correcte et compréhensible un modèle du monde réel.
L'analyse ne se préoccupe pas des solutions mais des questions, elle identifie le quoi faire et l'environnement d'un système sans décrire le comment qui est le propre de la conception.

3.1.2. Modèle objet

Un modèle objet saisit la structure statique d'un système. On montre les objets, les relations entre les objets et les attributs qui caractérisent chaque classe. Les étapes nécessaires à la construction, de ce modèle sont :

- identifications des classes.
- Préparation du dictionnaire de données.
- Identification des attributs de chaque classe.
- Identification des associations.

3.1.2.1 Identification des classes



3.1.2.2. Dictionnaire de données

- **Identification des classes**

Agents : ensemble du personnels de la direction générale de SONATRACH

Prestation : ensemble des prestations déposés par les agents

Accident de travail : les accidents de travail enregistrer

Arrêt de travail : les arrêt de travail qui on été déposer au service

Caisse : c'est les caisses où sont affiliés les agents de la direction générale SONATRACH

Structure : ensemble de structure de la direction ou appartient les agents

Mutuelle : la mutuelle où sont affiliés les agents de la direction

Fonction : les fonction qu'exerce l'agent

Ayant droit : les personnes qui on le droit a la prestation.

Type ayant droit : type d'ayant droit a la prestation.

Type de prestation : type de prestations sociales.

Type arrêt de travail : types d'arrêts de travail répertorie.

- **Identification des attributs :**

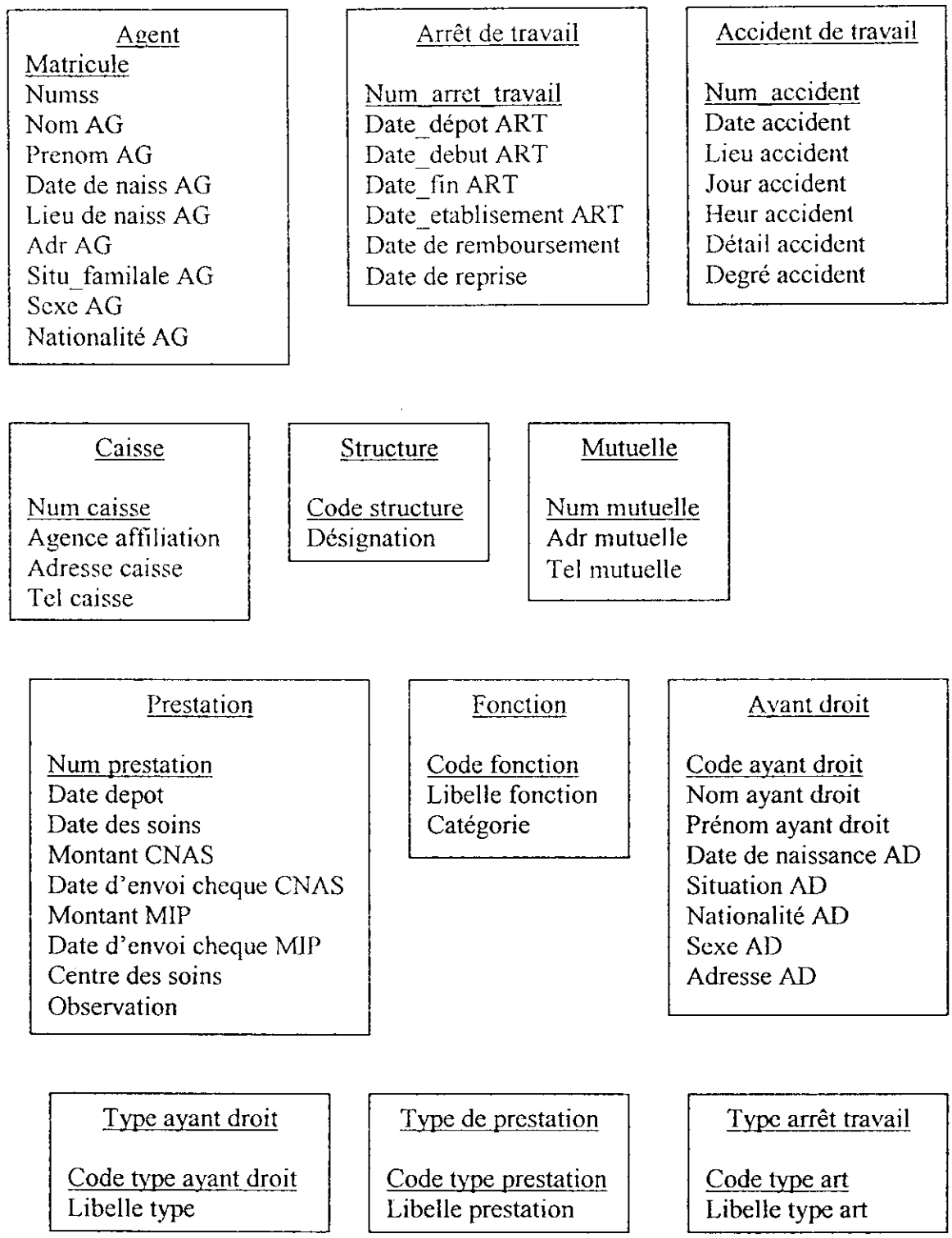


Figure 4.11 Identification des attributs

- **Identification des associations :**

Tableau 4.1. Identification des associations

association	Dimension	cardinalité	Classes-intervenant
exerce	2	1..*- 1	Agent - fonction
A pour benif	2	* - 1	Prestation - ayant droit
Est de type	2	1.* - 1	Ayant droit - type ayant droit
Est de type1	2	1.* - 1	Prestation - type prestation
Est de type2	2	1.* - 1	Arrêt de travail - type arrêt de travail
Présente	2	* - 1	Agent - prestation
Appartient	2	1.* - 1	Agent - structure
Affilie	2	1.* - 1	Agent - caisse
Affilie1	2	* - *	Agent - mutuel
Peut avoir	2	1.* - *	Agent - accident de travail
Dépôt arrêt de travail	2	1 - *	Agent - arrêt de travail

- **exerce :** chaque agent exerce une fonction
- **a pour benif :** pour chaque prestation a pour bénéficiaire des ayant droit
- **est de type :** chaque ayant droit appartiens a un type
- **est de type1 :** chaque prestation présenté appartiens a un type de prestation
- **est de type2 :** chaque arrêt de travail déposé appartiens a un type de travail
- **présente :** l'agent présente une prestation
- **appartiens :** un agent appartiens a une structure
- **affilie :** un agent affilié a une caisse
- **Affilie1 :** agent est affilié a une mutuelle
- **Peu avoir :** un agent peut avoir un accident de travail
- **Dépôt arrêt de travail :** agent déposé un arrêt de travail

3.1.2.3 Diagramme de classes :

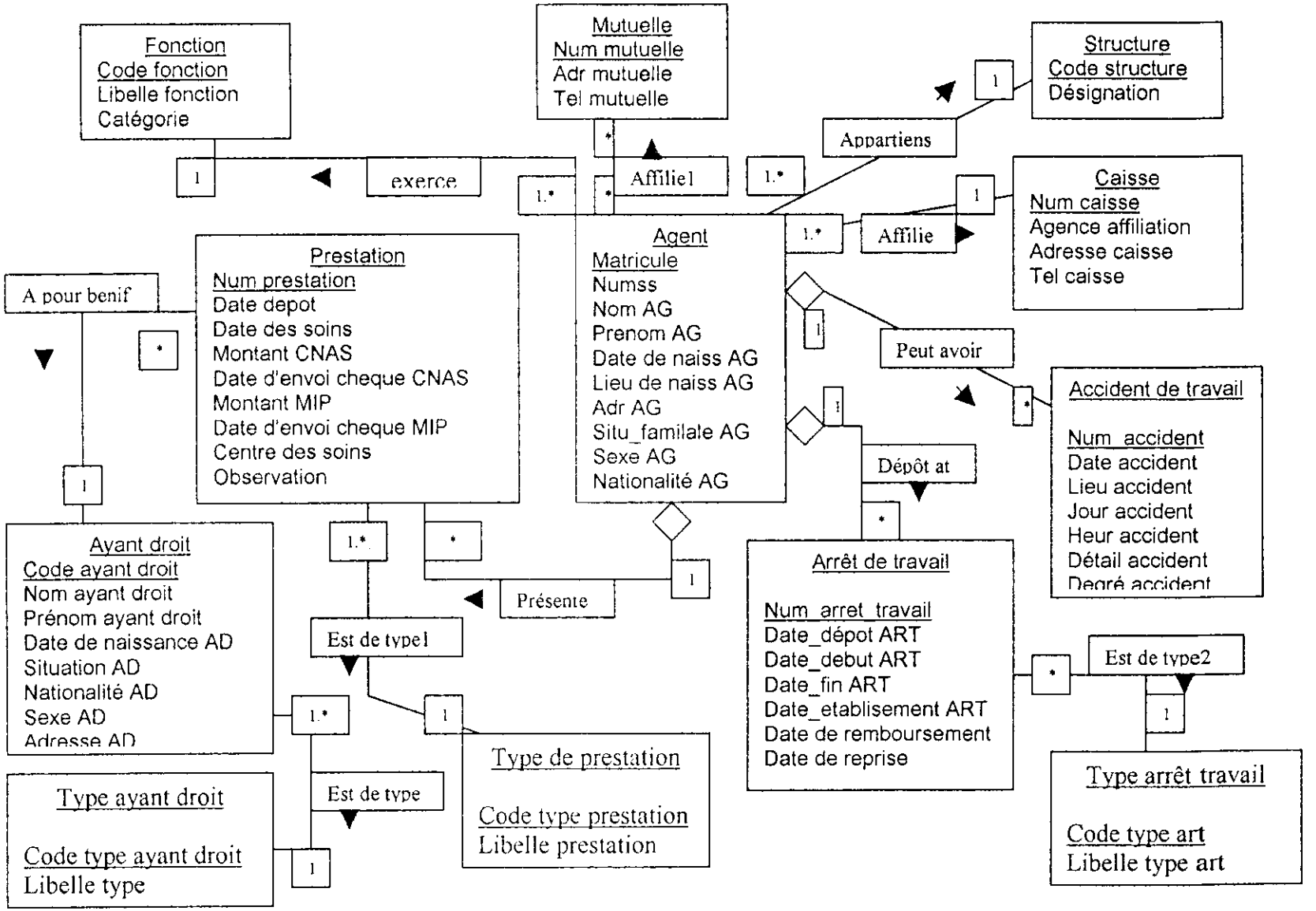


Figure 4.12. Diagramme des classes

3.1.3. Model fonctionnelle :

Le model fonctionnelle décrit l'aspect fonctionnelle du system cette phase consiste a rependre a la question suivante "que fait le system " et pour cela on utilise le diagramme "use case". Les étapes nécessaires à la construction, de ce modèle sont :

- le système
. les acteurs
. les cas d'utilisations
. les liens entre acteurs et cas d'utilisations

- Les acteurs :

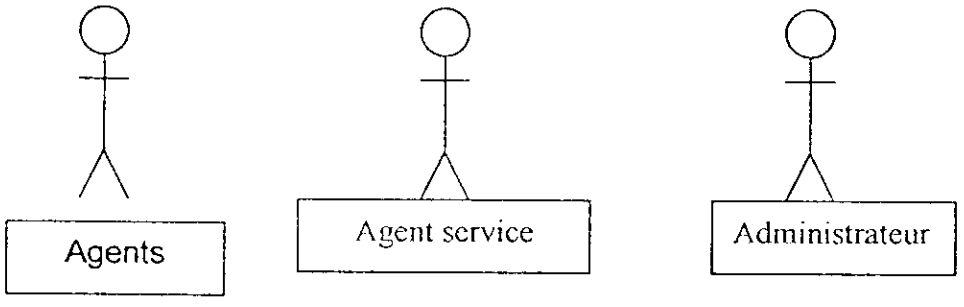


Figure 4-13 les acteurs

- Les cas d'utilisation (use case) :

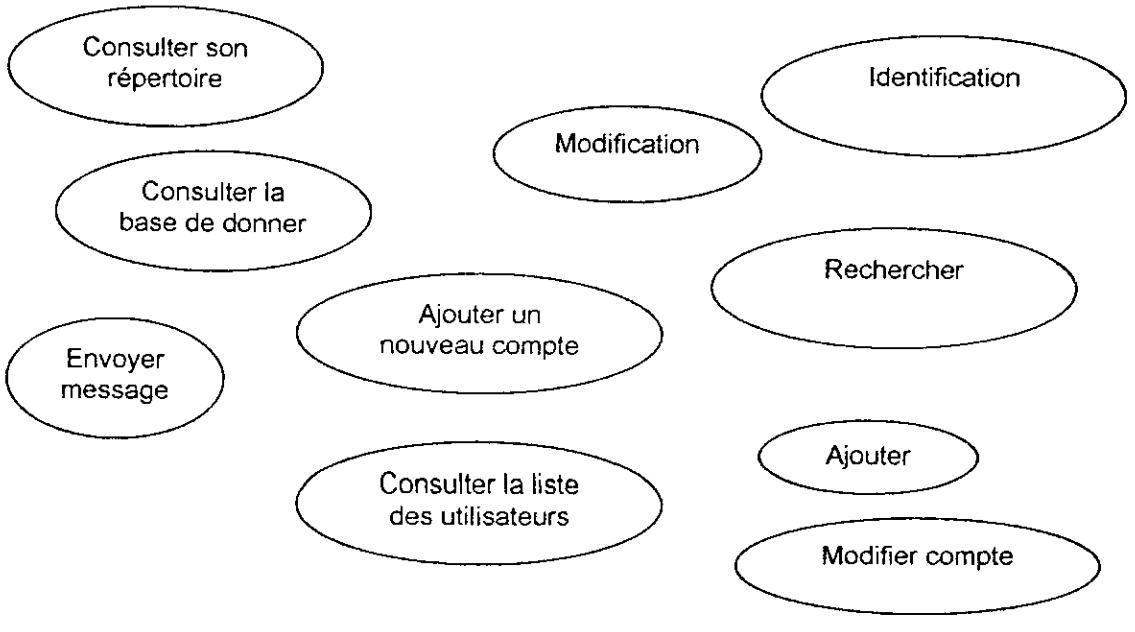
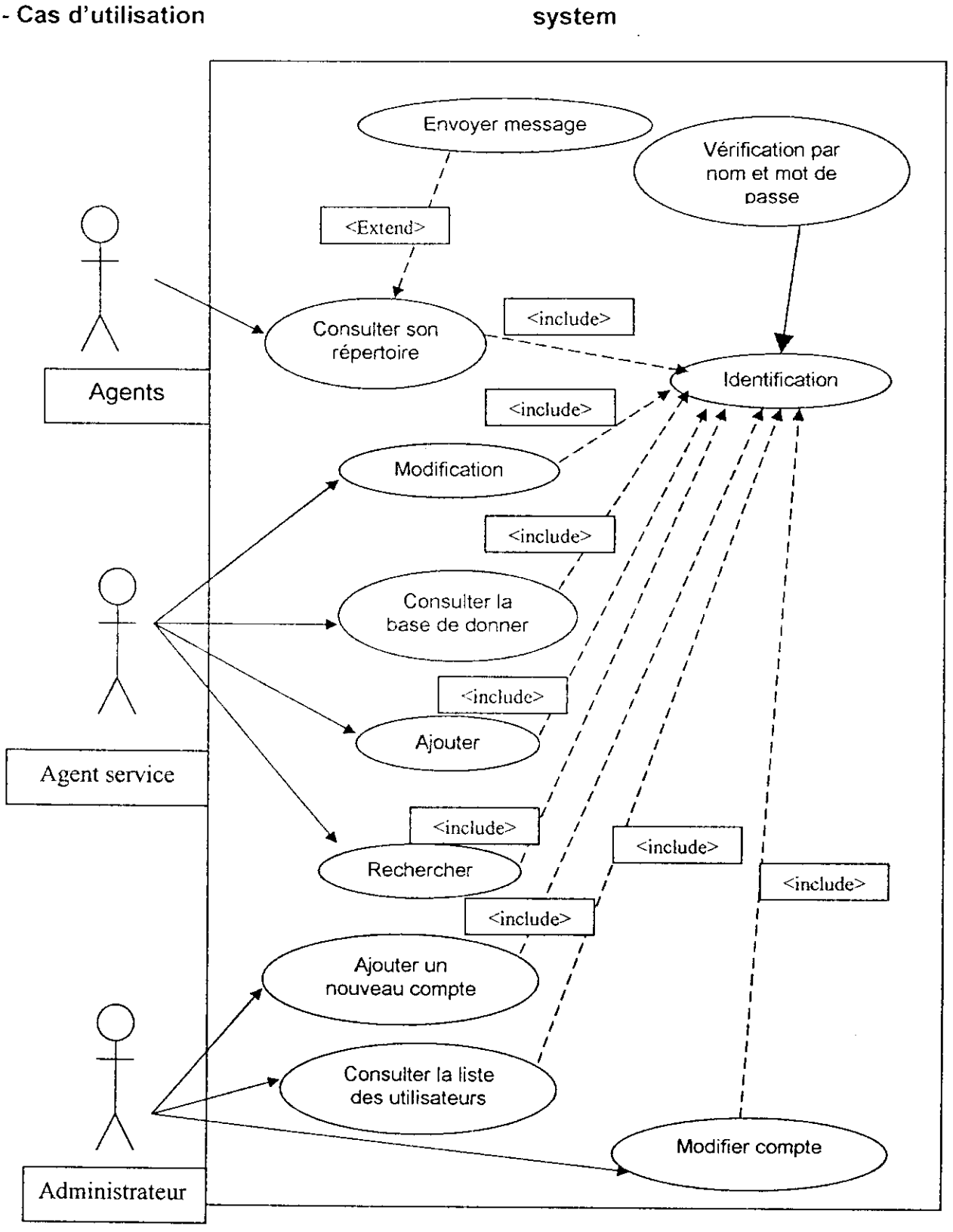


Figure 4-14 les cas d'utilisation

- Cas d'utilisation



Les acteurs

Figure 4-15 diagramme de cas d'utilisation

3.1.4. Modèle dynamique :

le modèle dynamique décrit des concepts traitant du flot de contrôle des interactions et des séquences d'opérations dans un système d'objet actif (concurrente). [CEE00]

- **Les scénarios par diagramme de séquence**

Le diagramme de séquences permet:

.de visualiser l'aspect temporel des interactions

.de connaître le sens des interactions (acteur vers système ou contraire)

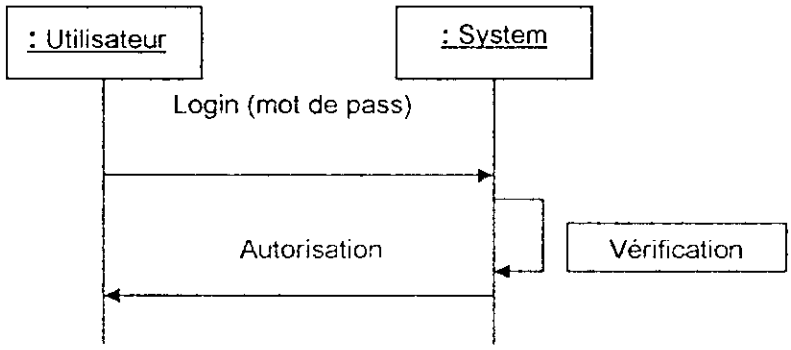


Figure 4-16 Identifications des utilisateurs

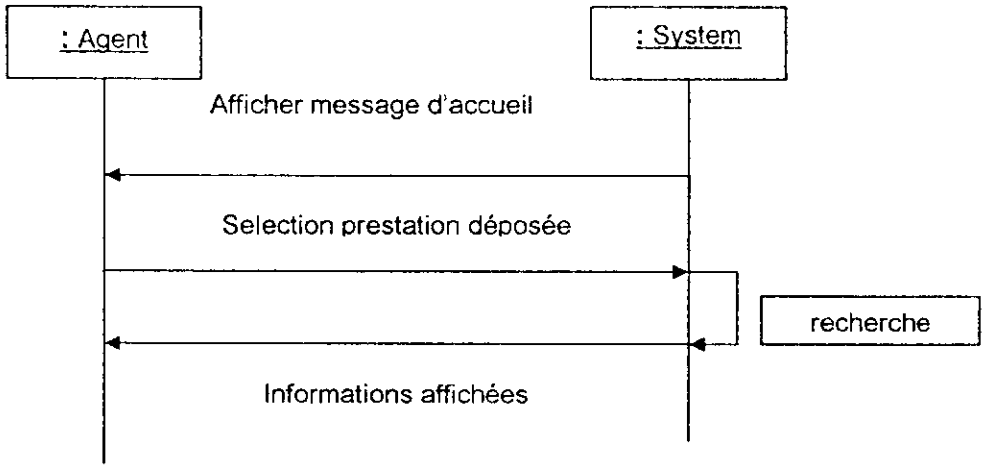


Figure 4-17 Consulter les prestations relatives a une personne

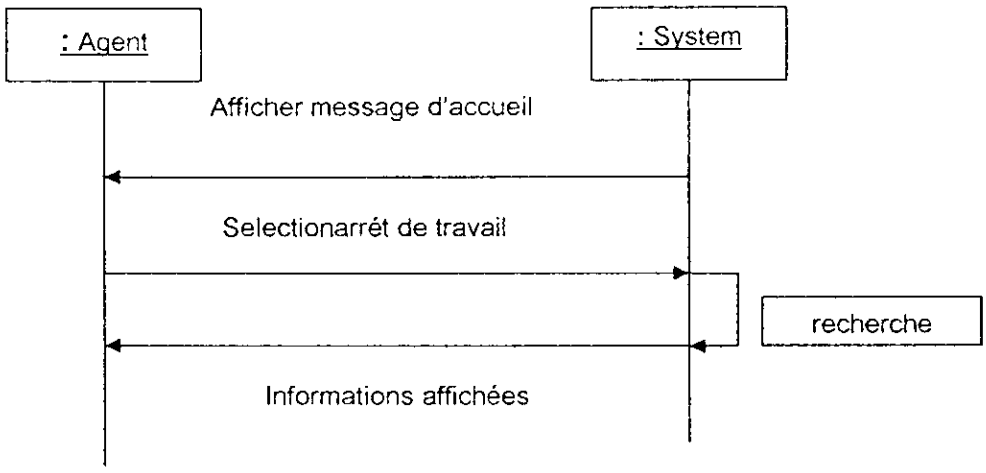


Figure 4-18 Consulter arrêt de travail relatif a une personne

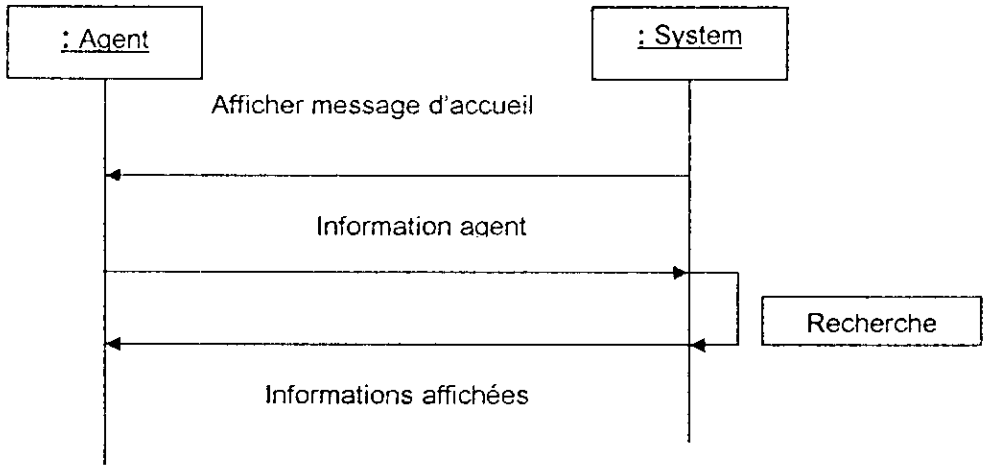


Figure 4-19 Consulter les informations sur l'agent

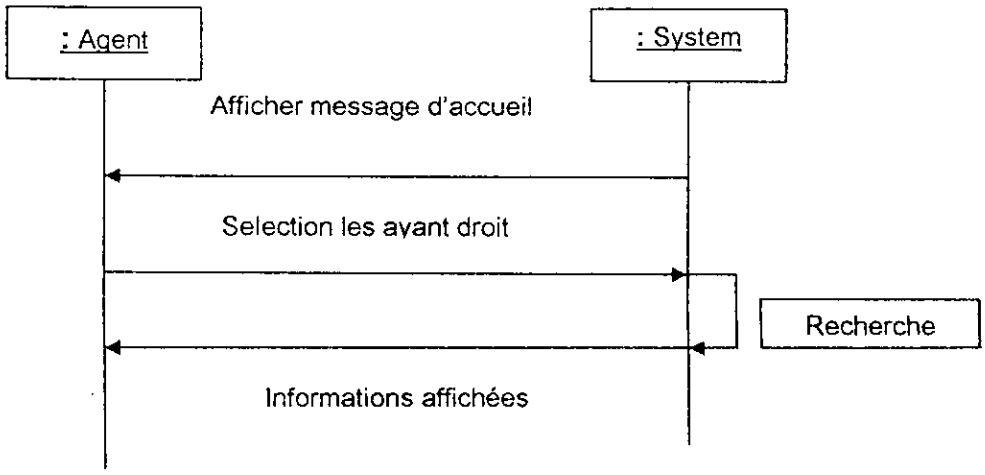


Figure 4-20 Consulter les ayant droit relatif a une personne

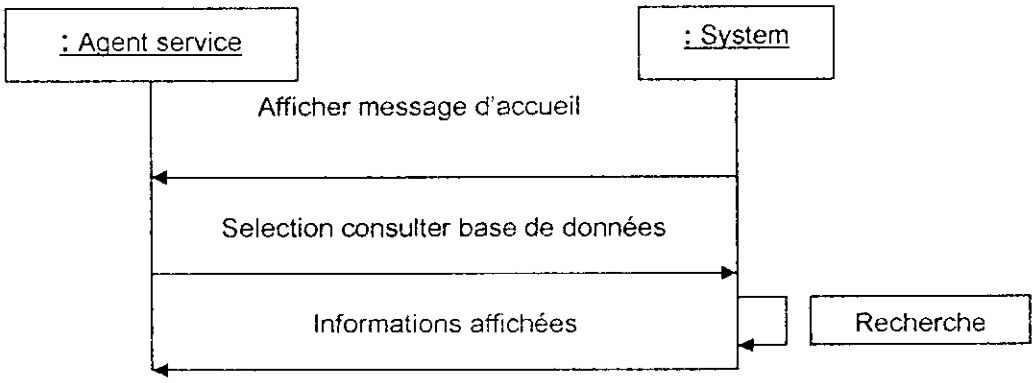


Figure 4-21 Consulter la base de données

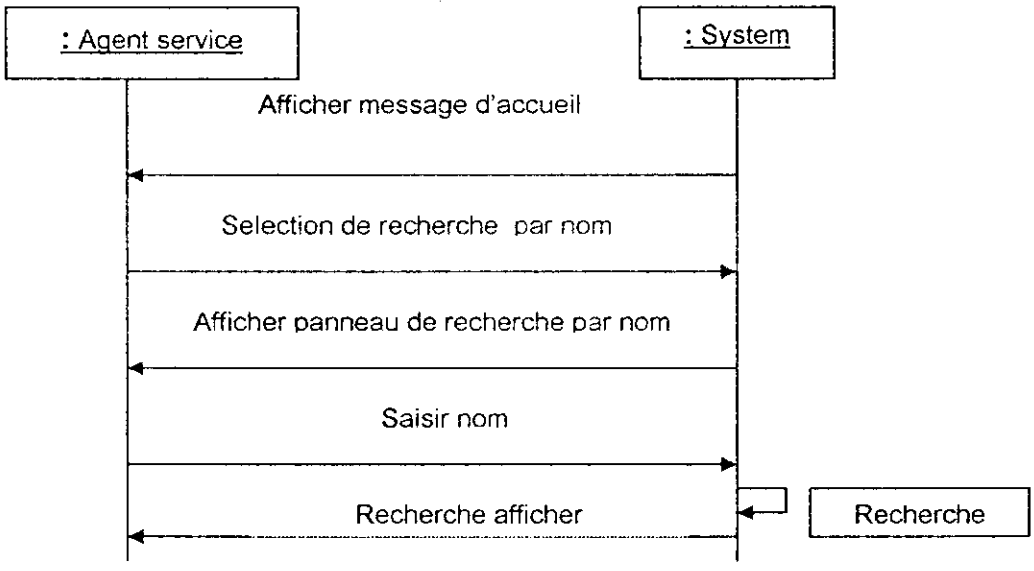


Figure 4-22 Recherche d'un agent par nom

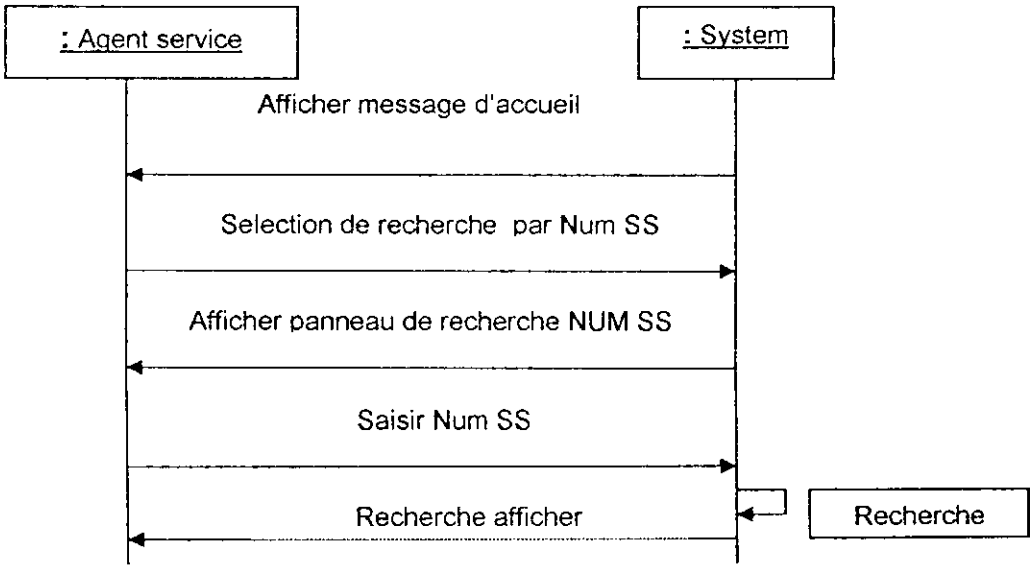


Figure 4-23 Recherche pare Numéro de sécurité social

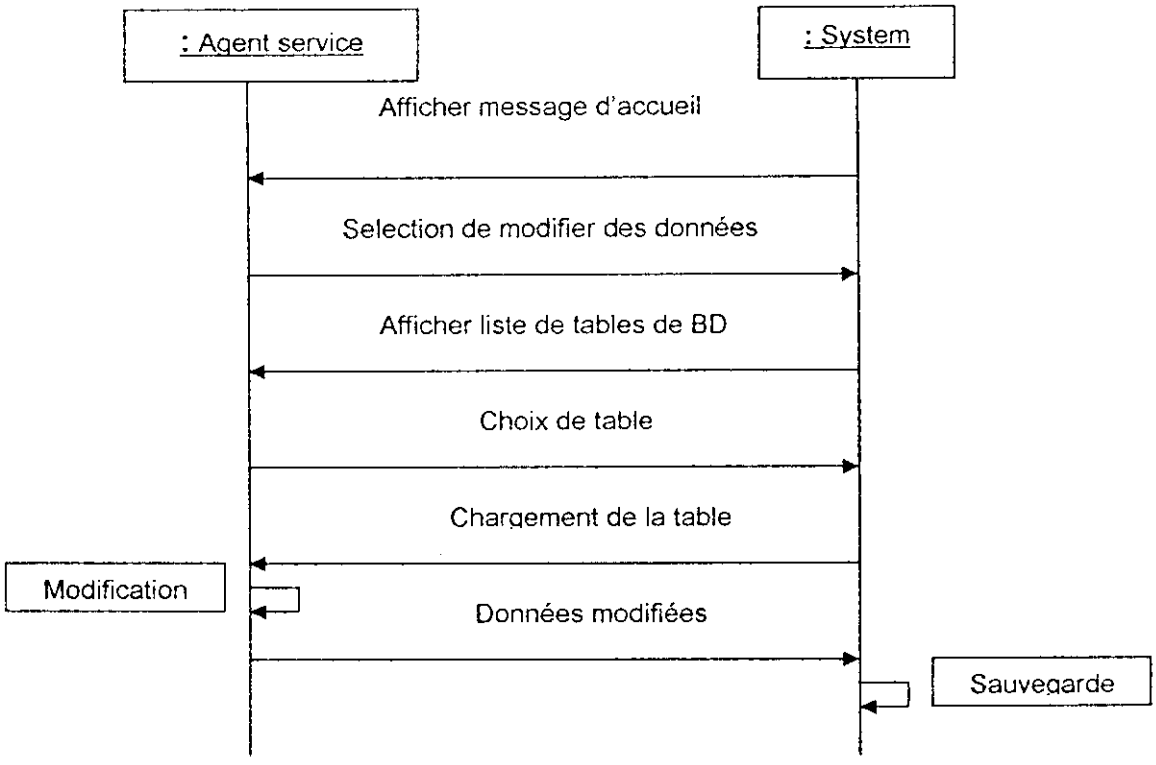


Figure 4-24 Modifier des données de la base

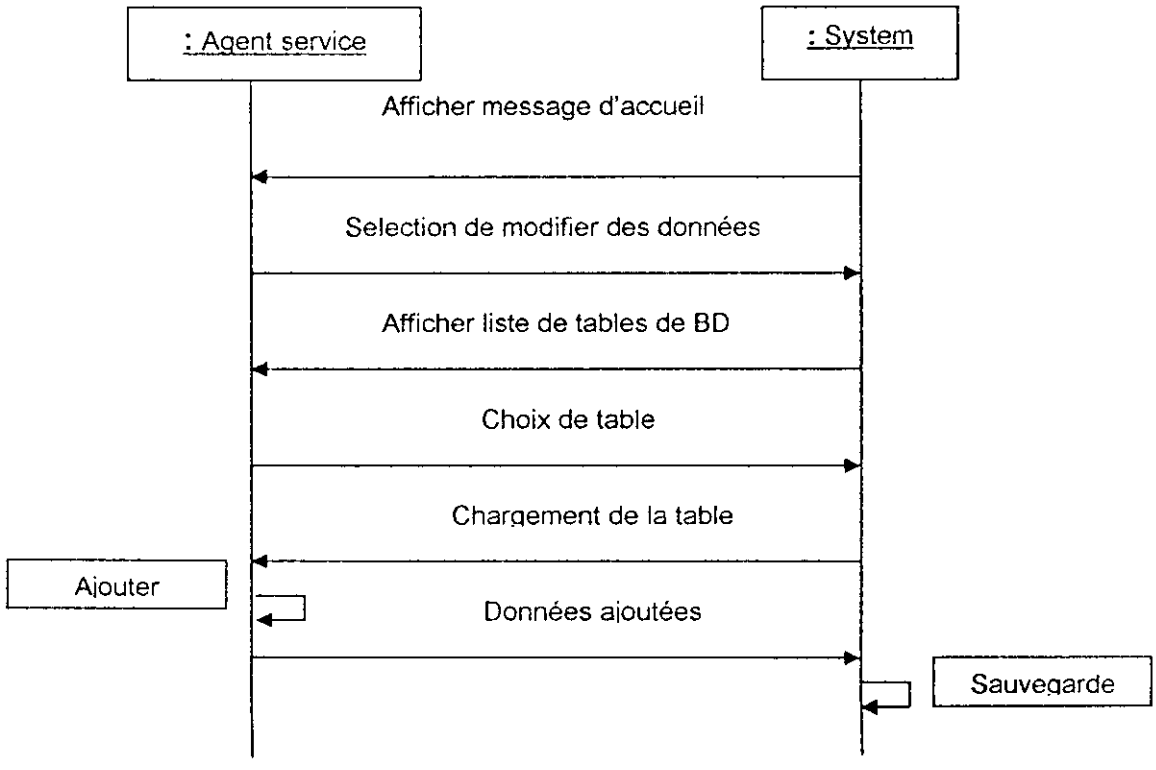


Figure 4-25 Ajouter des données a la base

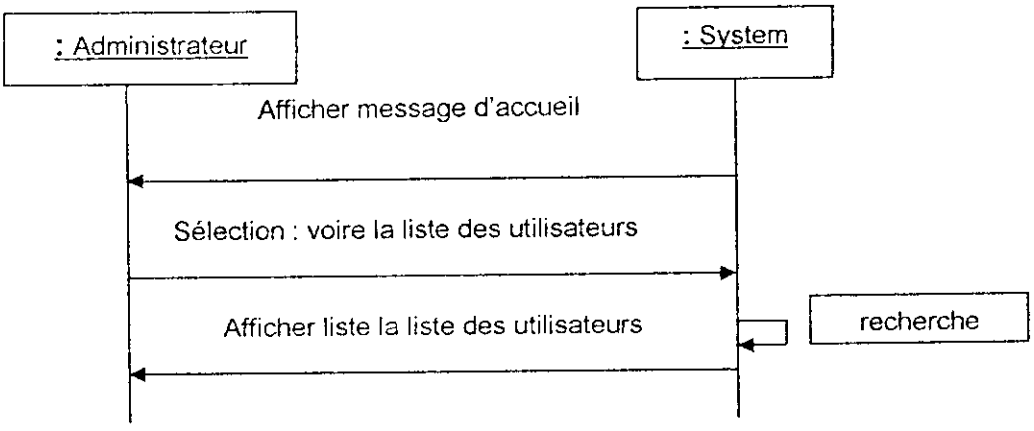


Figure 4-26 Consulter la liste des utilisateurs

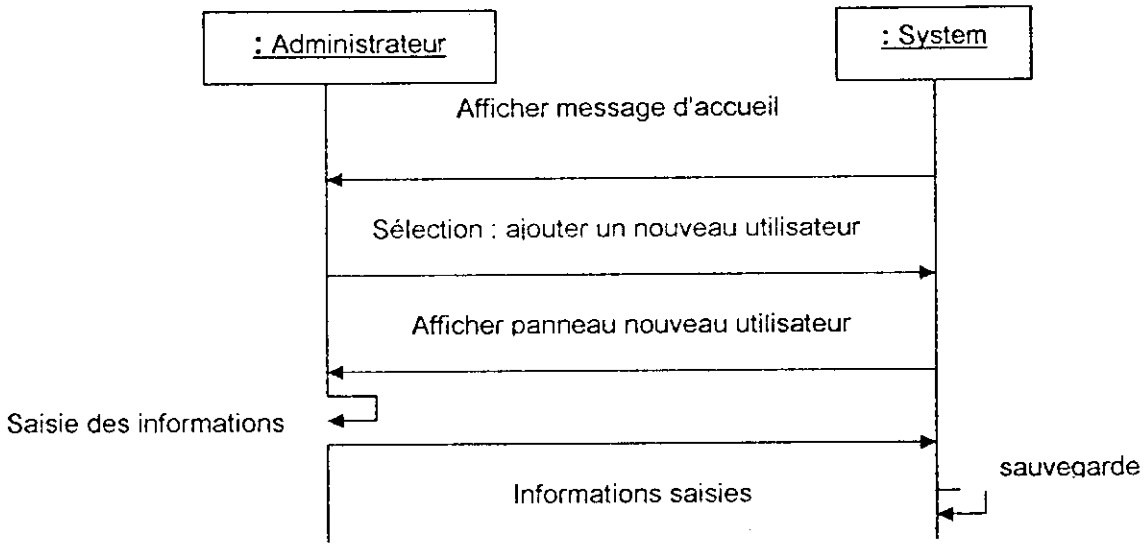


Figure 4-27 Ajouter un utilisateur

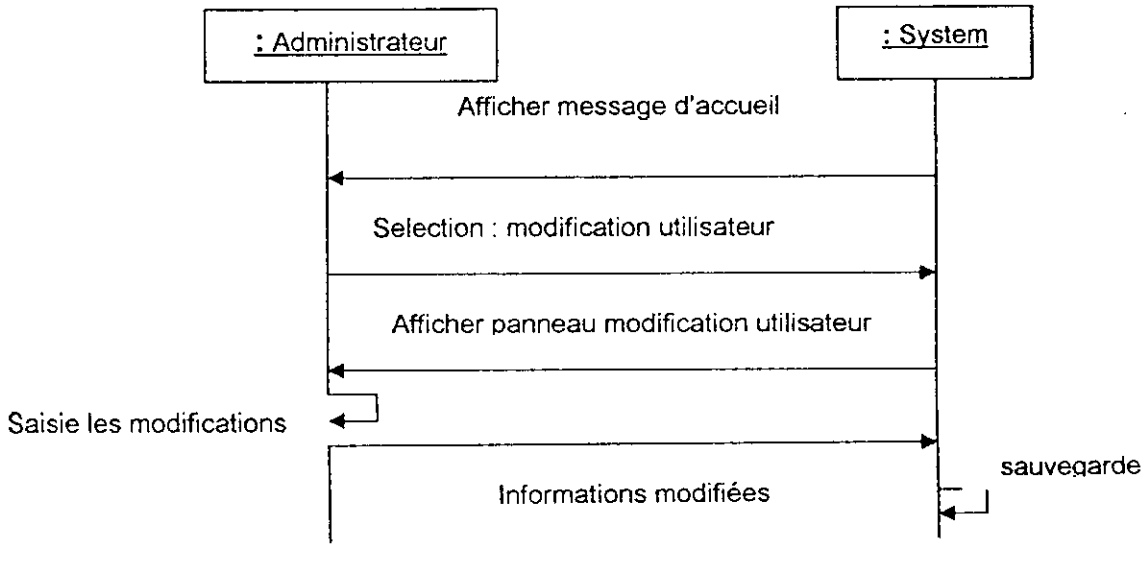


Figure 4-28 Modification utilisateur

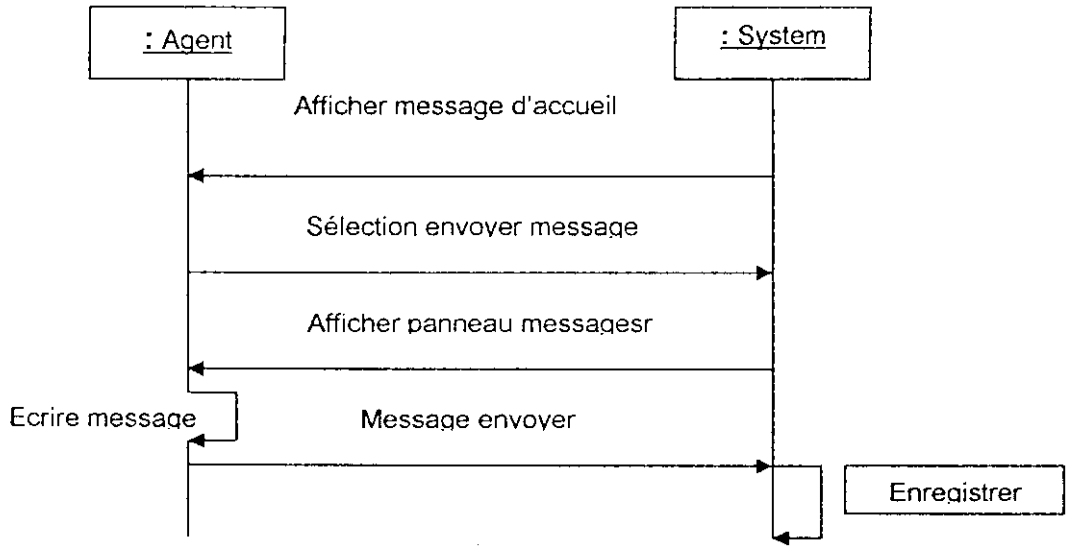


Figure 4-29 Envoyer message

- **Diagrammes de collaborations**

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait Figurer les envois de messages.

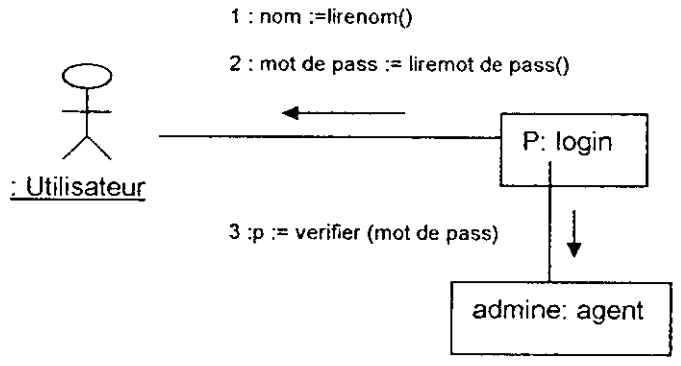


Figure 4-30 Diagramme de collaboration –indentification-

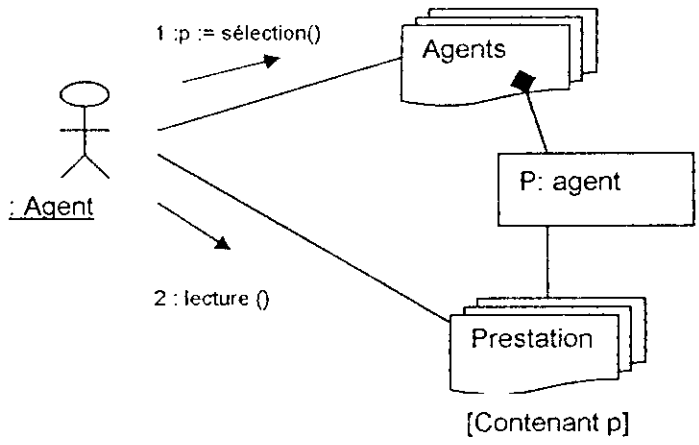


Figure 4-31 Digramme de collaboration de consultation prestation relatif a une personne

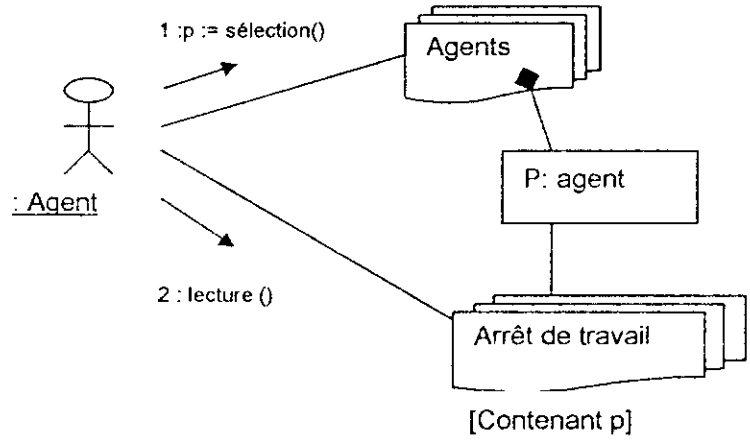


Figure 4-32 Diagramme de collaboration consulter arrêt de travail relatif à une personne

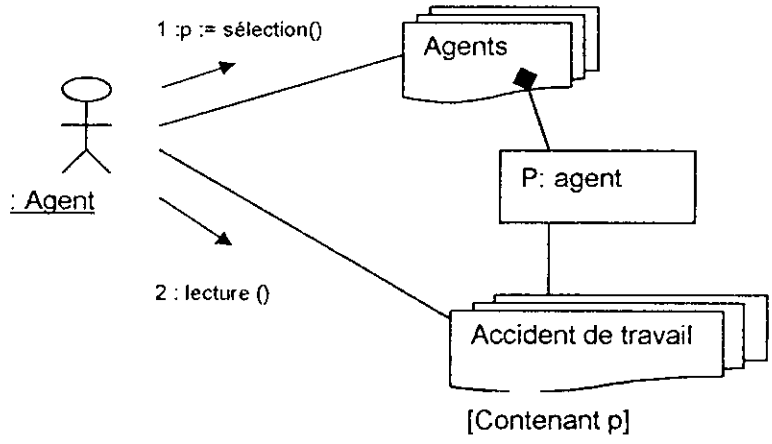


Figure 4-33 Diagramme de collaboration consulter accident de travail relatif à une personne

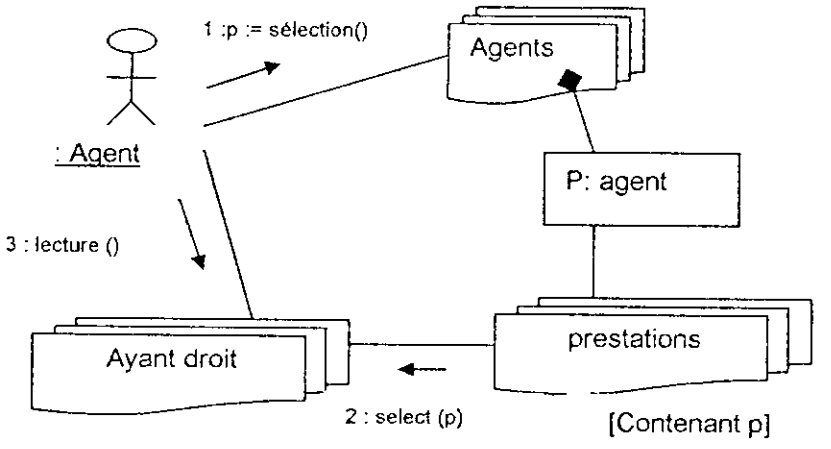


Figure 4-34 Diagramme de collaboration consulter ayant droit relatif à une personne

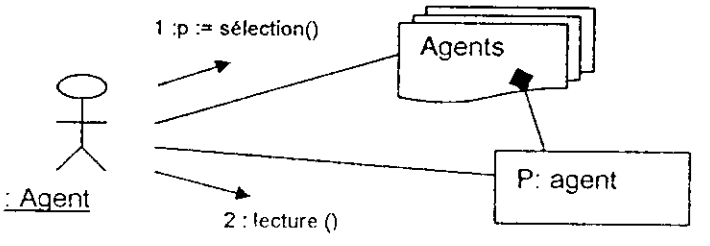


Figure 4-35 Diagramme de collaboration consulter information sur un agent

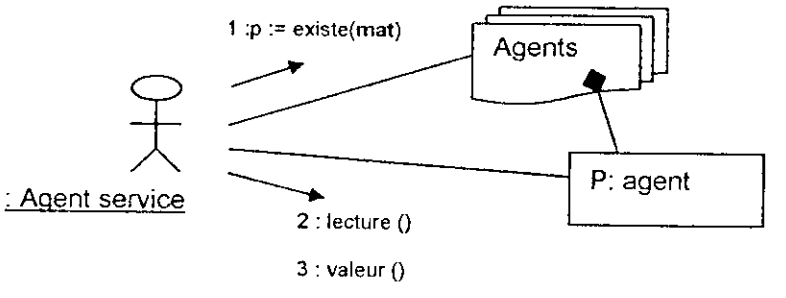


Figure 4-36 Recherche information sur un agent par matricule

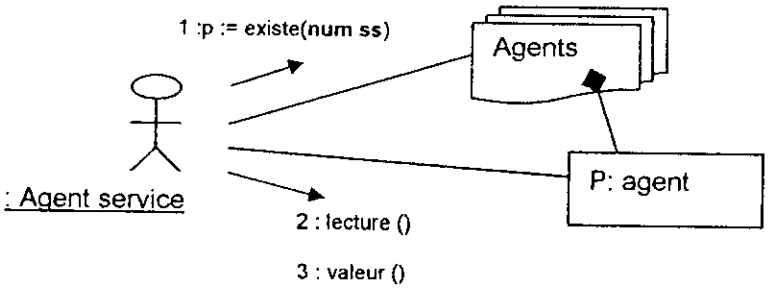


Figure 4-37 Recherche information sur un agent par numéro de sécurité sociale

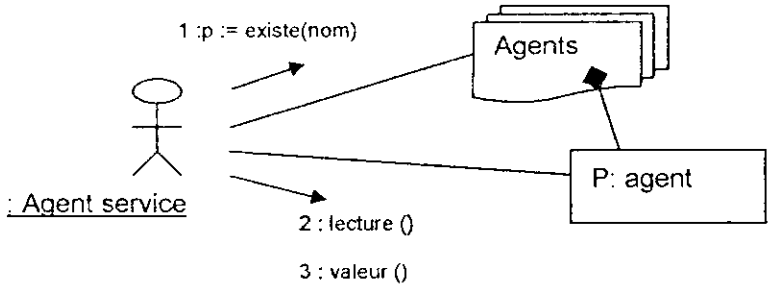


Figure 4-38 Recherche information sur un agent par nom

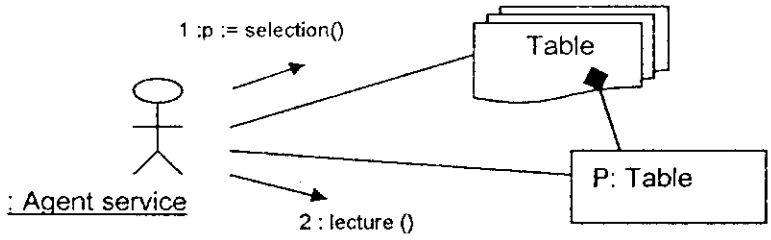


Figure 4-39 Consulter la base de données

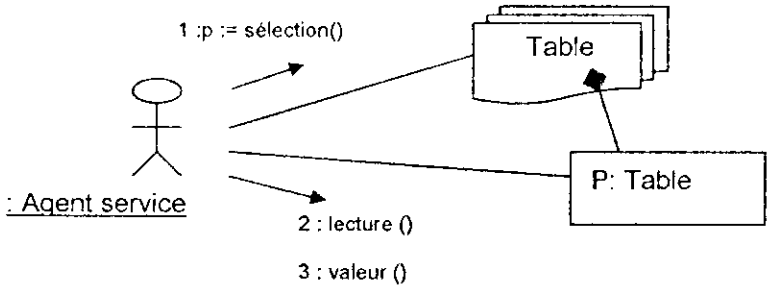


Figure 4-40 Modification de données de la base

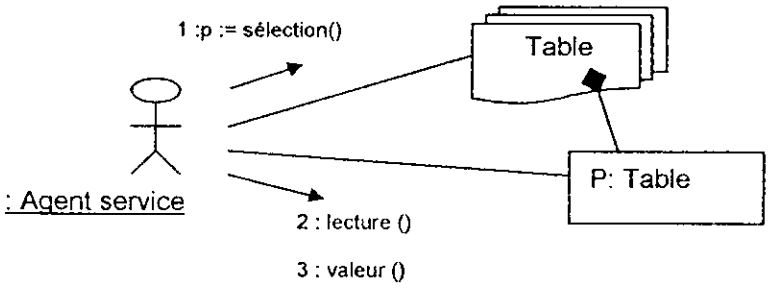


Figure 4-41 Ajouter des données à la base

4. CONCEPTION DE SYSTEME

4.1. Règles de passage entre modèle objet et modèle tables relationnelles

➢ Représentation des classes d'objets en tables :

- chaque classe est représentée par un ou plusieurs tables.

➢ Représentation des associations d'objets en tables :

- Chaque association plusieurs à plusieurs est représentée par une table distincte.
- Une association un à plusieurs est représentée par une table distincte ou peut être enfouie comme clé étrangère dans la table pour l'une ou l'autre des classes. Pour les associations un à plusieurs ou un à un, il n'y a pas de cycle, on dispose de l'option supplémentaire qui consiste à ranger l'association et les objets liés dans une seule table.

Ayant conscience que cela peut introduire une redondance et violer des formes normales.

- Les noms de rôles sont incorporés en tant que partie du nom de l'attribut de la clé étrangère.
- Les associations n-aires (n>2) se représentent par des tables distinctes. Ce qui aide parfois à promouvoir une association n-aires en une classe.
- Une association qualifiée se représente en une table distincte avec au moins trois attributs, la clé primaire de chaque classe liée est le qualificatif.
- Les agrégations suivent les mêmes règles que les associations.

➢ **Représentation de la généralisation de l'héritage simple en tables :**

- On représente chaque superclasse et chaque sous-classe par une table.
- S'il n'y a de table de superclasse, les attributs sont dupliqués dans chaque table de sous-classe.
- S'il n'y a de table de sous-classe, on apporte tous les attributs de sous-classe dans la superclasse.

4.2. Traduction du modèle objet en base de données relationnelles

En appliquant les règles de passage au modèle objet, on obtient la représentation logique de la base de données qui se représente comme suit :

Nom fichier : AGENT		
Nom champ	**type**	**longueur**
Matricule	AN	06
Num sécurité sociale	N	12
Nom AG	A	15
Prénom AG	A	15
Date de naiss AG	D	08
Lieu de naiss AG	AN	15
Adr AG	A	01
Situation_familale AG	AN	25
Sexe AG	A	01
Nationalité AG	A	15
Code fonction	AN	10
Num caisse	AN	05
N° affiliation	N	06
Date affilmiation	N	12
	D	08

Tableau 4-2- Représentations logique de la classe agents

Nom fichier : TYPE PRESTATION		
Nom champ	**type**	**longueur**
Code type prestation	A	03
libelle	AN	15

Tableau 4-3 Représentations logique de la classe type prestation

Nom fichier : ARRET DE TRAVAIL		
Nom champ	**type**	**longueur**
Num_arret_travail	N	06
Date_dépot ART	D	08
Date_debut ART	D	08
Date_fin ART	D	08
Date_etablisement ART	D	08
Date de remboursement	D	08
Date de reprise	D	08
Code type arrêt de travail	A	03
matricule	AN	06

Tableau 4-4 Représentations logique de la classe arrêt de travail

Nom fichier : ACCIDENT DE TRAVAIL		
Nom champ	**type**	**longueur**
Num_accident	N	03
Date accident	D	08
Lieu accident	AN	25
Jour accident	A	07
Heur accident	N	05
Détail accident	AN	15
Degré accident	AN	20
matricule	AN	06

Tableau 4-5 Représentations logique de la classe accident de travail

Nom fichier : STRUCTURE		
Nom champ	**type**	**longueur**
Code structure	AN	05
Désignation	A	25

Tableau 4-6 Représentations logique de la classe structure

Nom fichier : PRESTATION		
Nom champ	**type**	**Longueur**
Num prestation	N	08
Date depot	D	08
Date des soins	D	08
Montant CNAS	N	07
Date d'envoi cheque CNAS	D	08
Montant MIP	N	07
Date d'envoi cheque MIP	D	08
Centre des soins	AN	15
Observation	A	20
Code ayant droit	N	02
Code type prestation	A	03
matricule	AN	06

Tableau 4-7 Représentations logique de la classe prestation

Nom fichier : FONCTION		
Nom champ	**type**	**longueur**
Code fonction	AN	10
Libelle	AN	20
catégorie	A	02

Tableau 4-8 Représentations logique de la classe fonction

Nom fichier : AYANT DROIT		
Nom champ	**type**	**longueur**
Code ayant droit	N	02
Nom ayant droit	A	15
Prénom ayant droit	A	15
Date de naissance AD	D	08
Situation AD	A	01
Nationalité AD	A	15
Sexe AD	A	01
Adresse AD	AN	25
Code ayant droit	A	03

Tableau 4-9 Représentations logique de la classe ayant droit

Nom fichier : TYPE ARRET DE TRAVAIL		
Nom champ	**type**	**longueur**
Code type arrêt de travail	A	03
Libelle type arrêt de travail	AN	20

Tableau 4-10 Représentations logique de la classe type arrêt de travail

Nom fichier : TYPE AYANT DROIT		
Nom champ	**type**	**longueur**
Code type ayant droit	A	03
libelle	AN	20

Tableau 4-11 Représentations logique de la classe type ayant droit

Nom fichier : CAISSE		
Nom champ	**type**	**longueur**
Num caisse	N	06
Agence affiliation	N	06
Adresse affiliation	AN	30
tel	N	09

Tableau 4- 12 Représentations logique de la classe caisse

4.3. CONCEPTION DES OBJETS

Chaque table de schéma conceptuel est représentée par la définition d'un objet au niveau du modèle objet externe, en les raffinant avec l'ajout des méthodes qui sont directement déduites des actions du modèle dynamique ou des traitements du modèle fonctionnel et en respectant les règles de passage énoncées précédemment. La conception des objets du système est définie par les prototypes des classes suivantes :

classe : AGENT	
Attribut	**méthode**
Matricule	Sélection ()
Num sécurité sociale	Image ()
Nom AG	Valeur ()
Prénom AG	Existe (string)
Date de naiss AG	Existe (double)
Lieu de naiss AG	
Adr AG	
Situation_familale AG	
Sexe AG	
Nationalité AG	
Code fonction	
Num caisse	
N° affiliation	
Date affilmiation	

Tableau 4-13 prototype de la classe agents

classe : prestation	
Attribut	**méthode**
Num prestation	Sélection ()
Date dépôt	Image ()
Date des soins	Valeur ()
Montant CNAS	Existe (double)
Date d'envoi cheque CNAS	
Montant MIP	
Date d'envoi cheque MIP	
Centre des soins	
Observation	
Code ayant droit	
Code type prestation	
matricule	

Tableau 4-14 prototype de la classe prestation

classe : arrêt de travail	
Attribut	**méthode**
Num_arret_travail	Sélection ()
Date_dépot ART	Image ()
Date_debut ART	Valeur ()
Date_fin ART	Existe (double)
Date_etablisement ART	
Date de remboursement	
Date de reprise	
Code type arrêt de travail	
matricule	

Tableau 4-15 prototype de la classe arrêt de travail

classe : accident de travail	
Attribut	**méthode**
Num_accident	Sélection ()
Date accident	Image ()
Lieu accident	Valeur ()
Jour accident	Existe (double)
Heur accident	
Détail accident	
Degré accident	
matricule	

Tableau 4-16 prototype de la classe accident de travail

5. CONCLUSION

L'objectif de la modélisation est de comprendre le problème, son domaine d'application et de construire une conception correcte avant de passer à l'implémentation.

Les trois étapes de l'étude ont été étudiées, l'étape statique, dynamique et fonctionnelle. Reste à entamer l'implémentation.

Chapitre 5

Implémentations et résultats

1. Introduction :

Au cours de notre étude, plusieurs résultats ont été obtenu, notamment le modèle dynamique, le modèle fonctionnel, la conception des objets du système.
L'ensemble des ces résultats a été soumis à l'approbation de la direction SONATRACH.
Les résultats de cette étude feront l'objet de projection sur machine, selon l'outil choisi pour le niveau physique et permettront de rendre la solution opérationnelle.

2. Environnement technique de développement

2.1 Présentation des langages de programmation utilisés

Pour l'implémentation et la gestion de la base de données, et le développement de l'application on a utilisé les langages suivants :

2.1.1. Le langage SQL

Pour la mise en place et la gestion de la base de données, le langage utilisé est le SQL, donc parfaitement compatible avec la plate forme choisie, de plus il est entièrement conçu pour le web.
Le SQL est u langage de requête structuré (**S**tructured **Q**uery **L**angage) destiné à communiquer avec des bases de données relationnelles implémentées dans des SGBDR
Le langage SQL permet d'effectuer divers opération comme la création, l'extraction, la modification, la suppression, la fusion, etc..., sur des collections de données, par l'intermédiaire d'instructions particulière appelées des commande, souvent assistées d'ailleurs par des clauses ou des options.

2.1.2. Les JSP :

Les JSP (Java Server Pages) sont un standard permettant de développer des applications Web interactives, c'est-à-dire dont le contenu est dynamique. C'est-à-dire qu'une page web JSP (repérable par l'extension .jsp) aura un contenu pouvant être différent selon certains paramètres (des informations stockées dans une base de données,

Les JSP sont intégrables au sein d'une page Web en HTML à l'aide de balises spéciales permettant au serveur Web de savoir que le code compris à l'intérieur de ces balises doit être interprété afin de renvoyer du code HTML au navigateur du client.
Les JSP permettent donc d'écrire facilement des servlets, en incluant dans des balises spécifiques le code JSP au sein du fichier HTML. De cette façon, elles fournissent une technologie rapide afin de créer des pages dynamiques.
De plus, les JSP étant basées sur Java côté serveur, elles possèdent toutes les caractéristiques faisant la force de Java :

les JSP sont multithreadées,
les JSP sont portables,
les JSP sont orientées objet,
les JSP sont sûres,

2.2. Implémentation

La base de données a été implémentée avec oracle 9i :

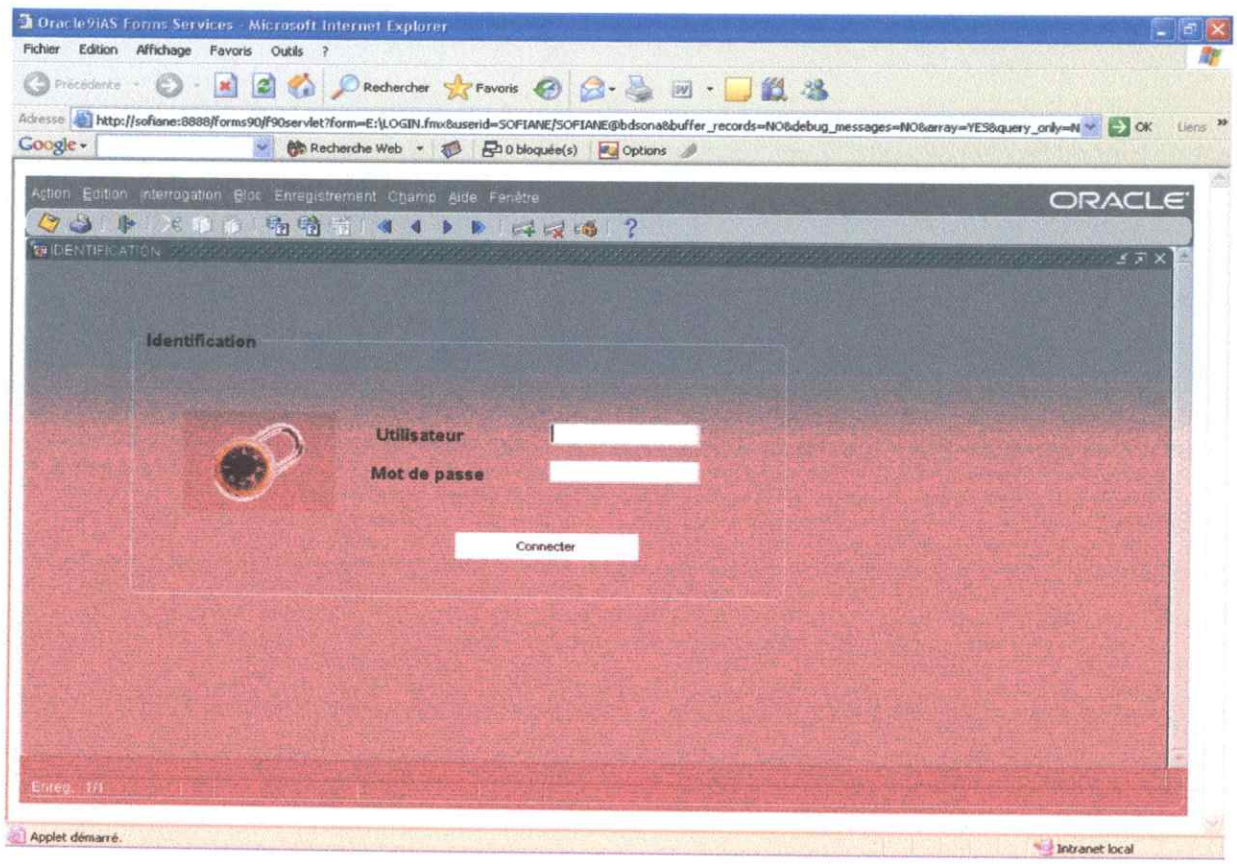
2.2.1 Oracle 9i :

C'est un system relationnel de gestion de base de données Multi-Utilisateurs à haute performance tournant sous le système d'exploitation Windows (XP/NT/2000 serveur.) ou linux, il est conçu pour être exploité sur un ordinateur serveur et être le composant serveur dans un environnement trois-tiers

2.3. Test de l'application :

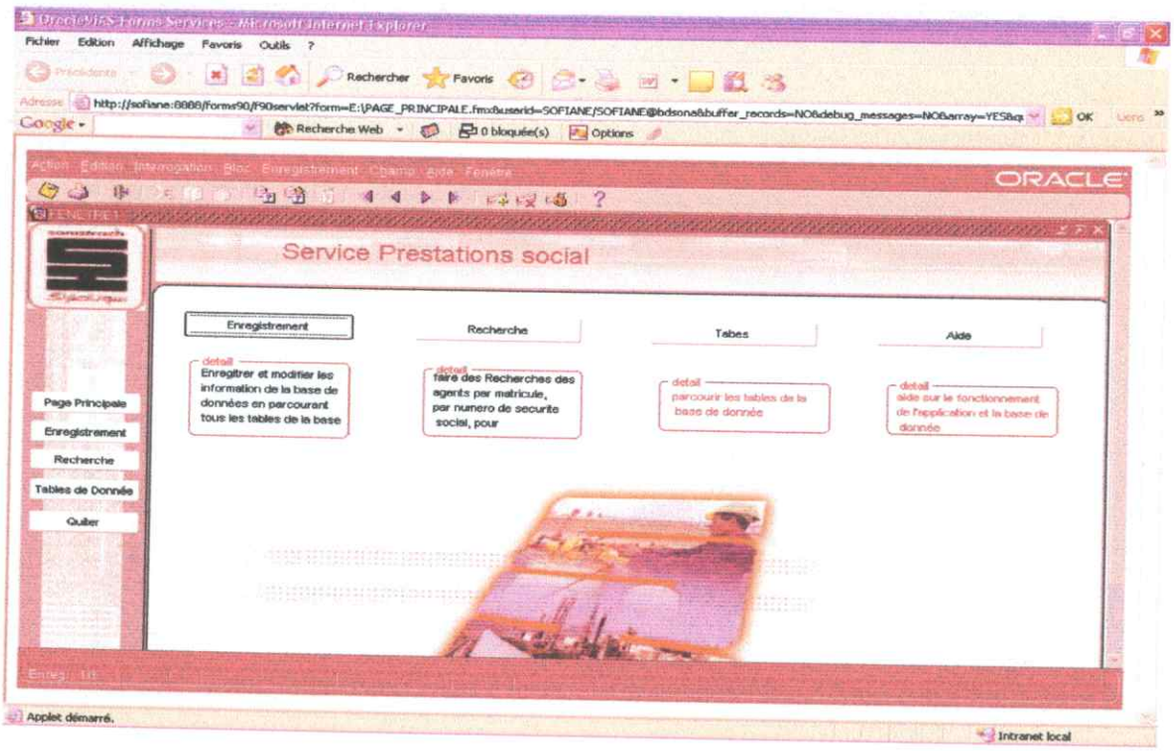
L'utilisateur de notre application web peu faire le suivi des dossiers des prestations sociales en utilisant l'intranet de la société ou l'Internet.

L'utilisateur doit saisir son nom et son mot de passe.

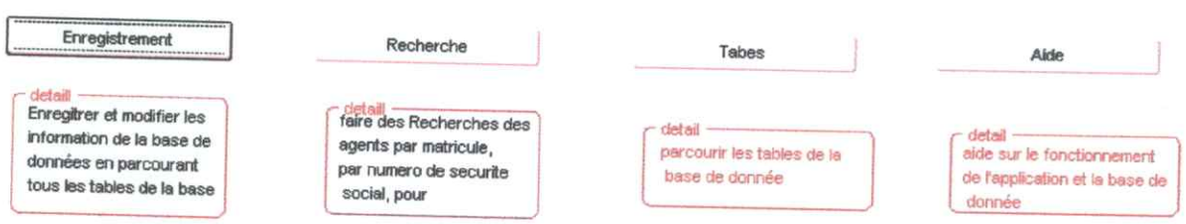


Apres la saisie un composant s'occupe de vérifier la validation des informations en entrée, en cas d'erreur, un message s'affiche pour l'indiquer

Si les données sont valides, l'utilisateur aura une page contenant toutes les opérations qu'il peut effectuer, dans se cas c'est l'agent du service qui est identifiée.

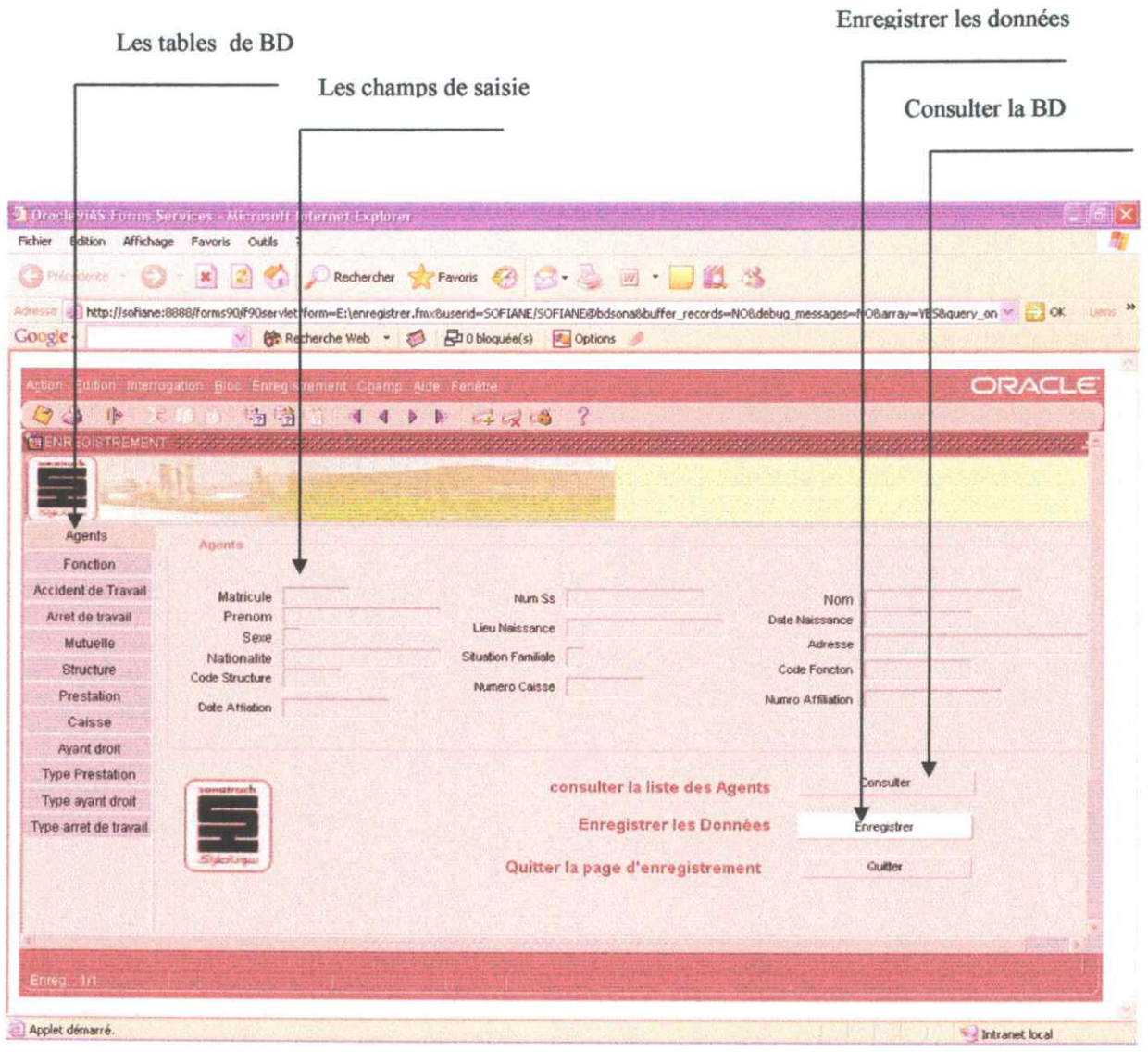


La page principale permet d'accéder aux différentes fonctions de l'application, dans ce qui suit nous allons détailler chaque sous partie.

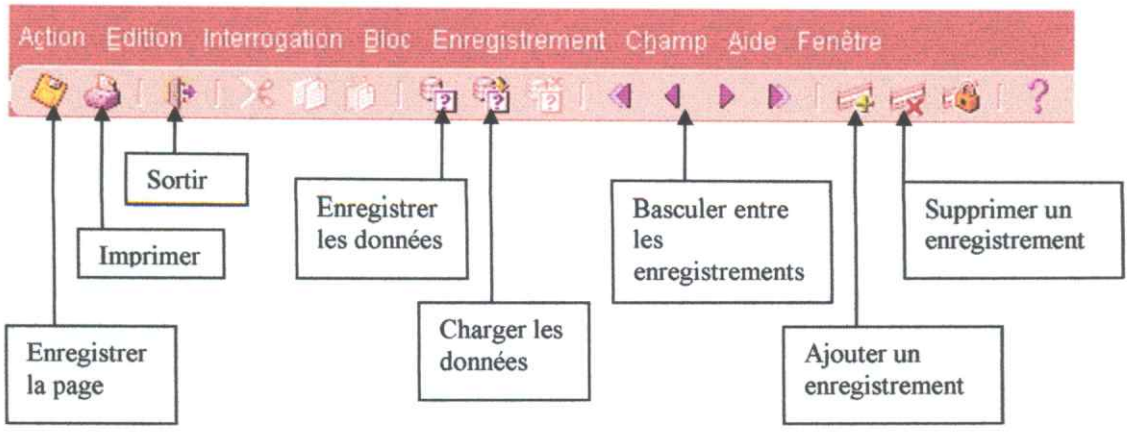


2.3.1. Page Enregistrement :

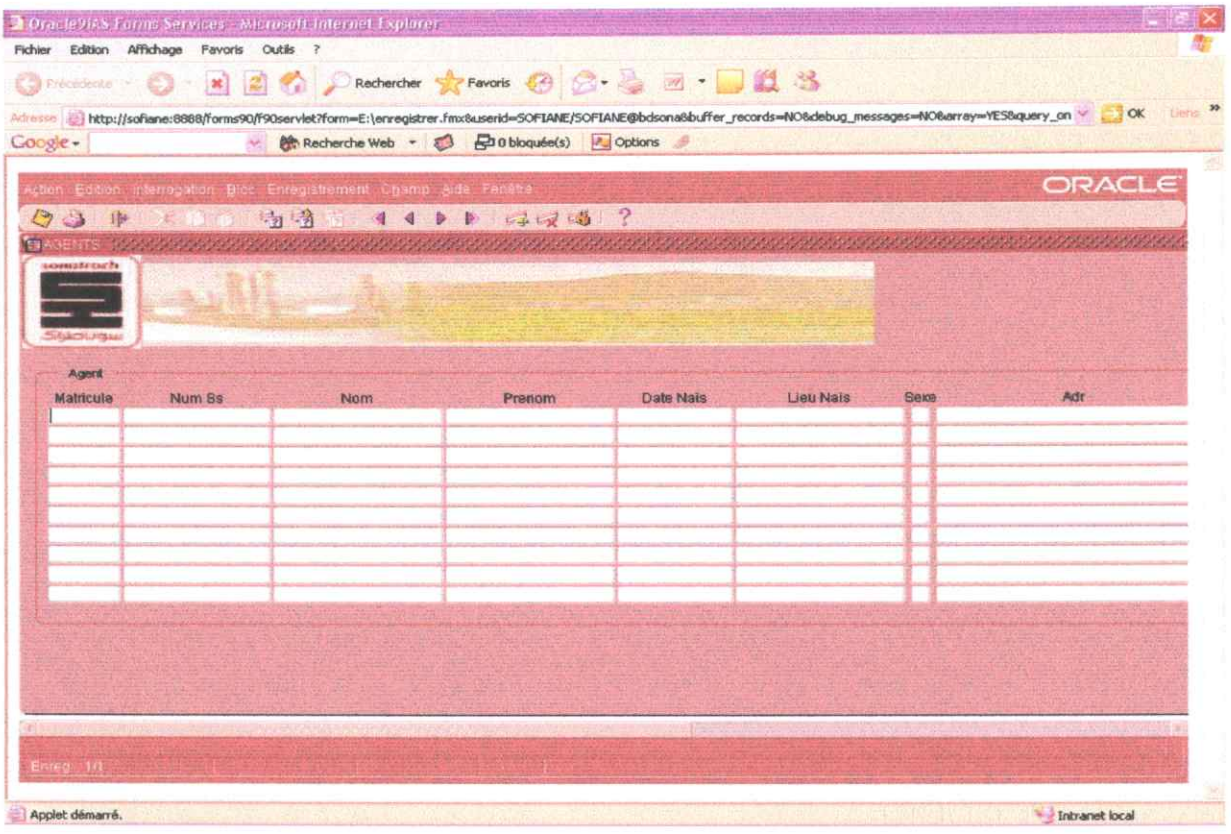
La page enregistrement permet de saisir les données dans la base et de consulter, en va détailler cette partie :



- zone de contrôle : permet de charger, supprimer ou modifier les données

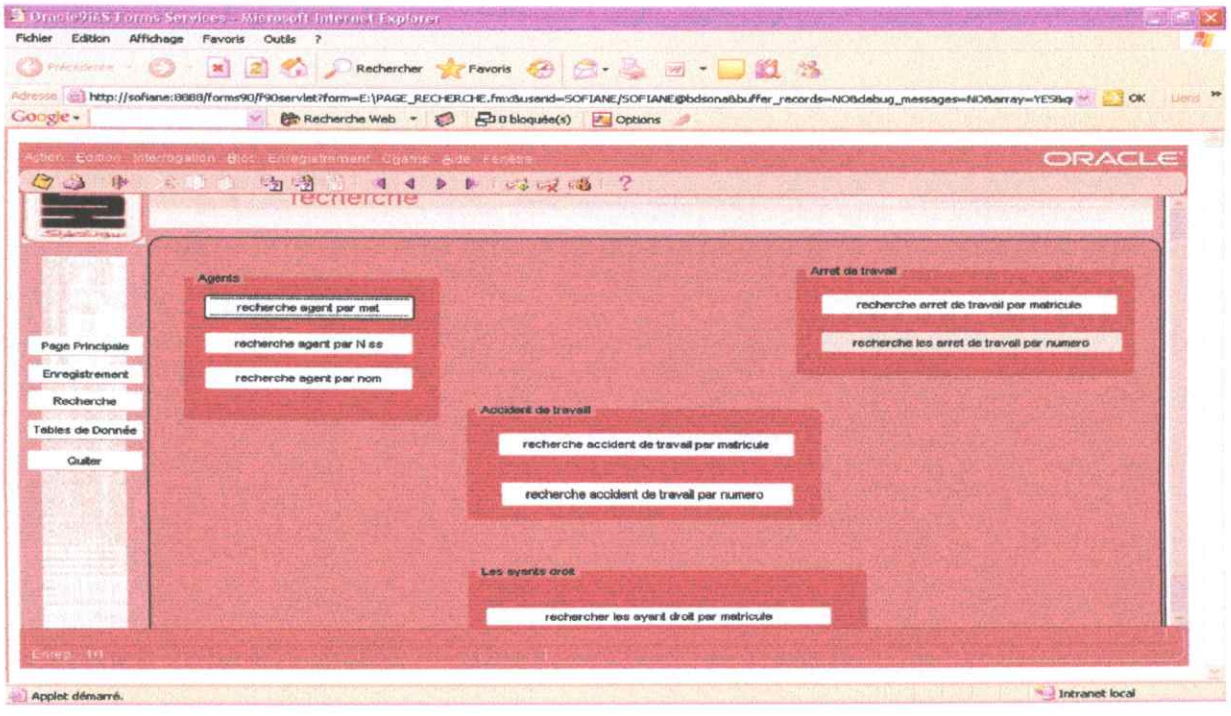


En choisissant de consulter le BD on aura la page suivante :

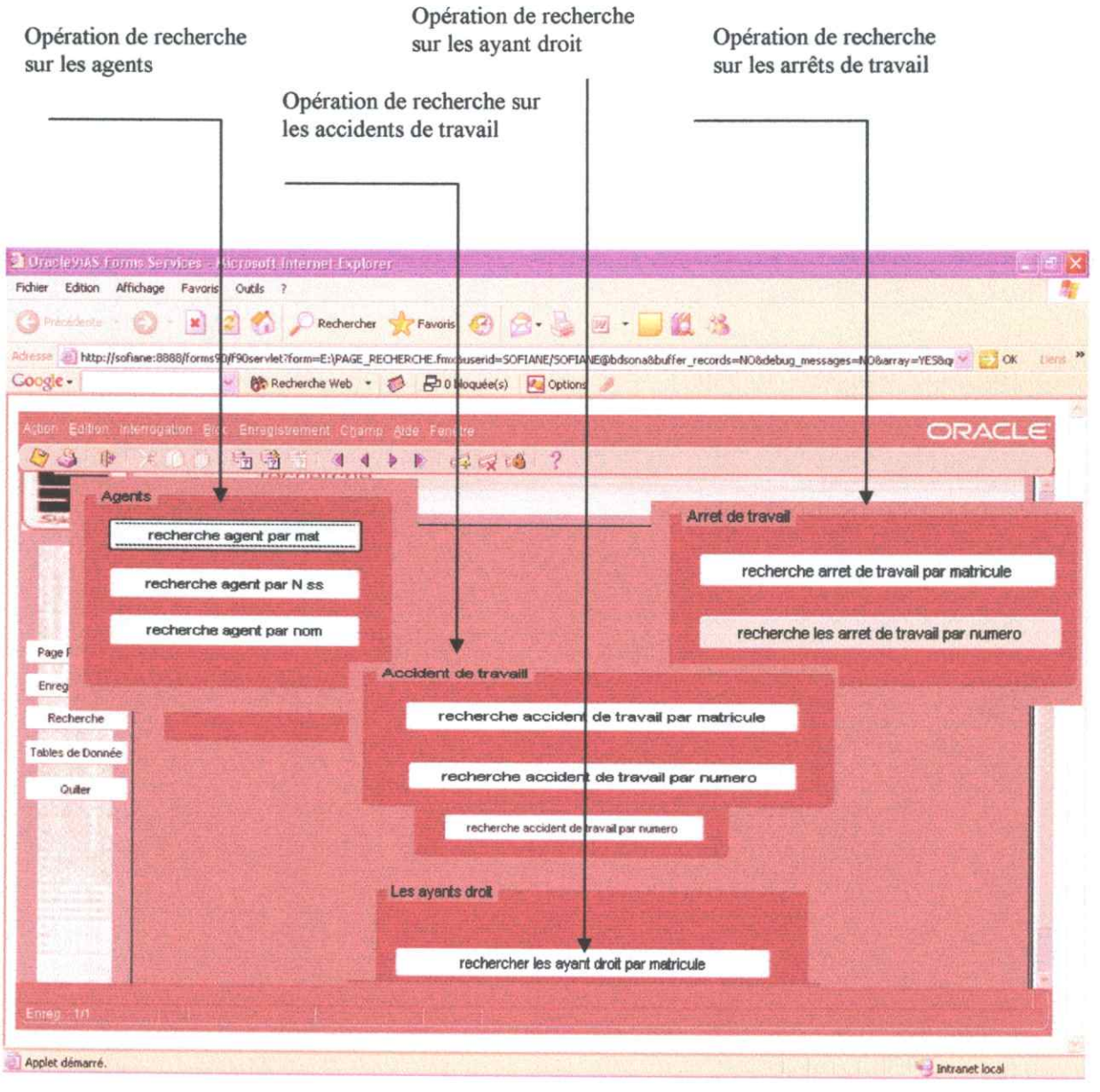


2.3.2. Page de recherche :

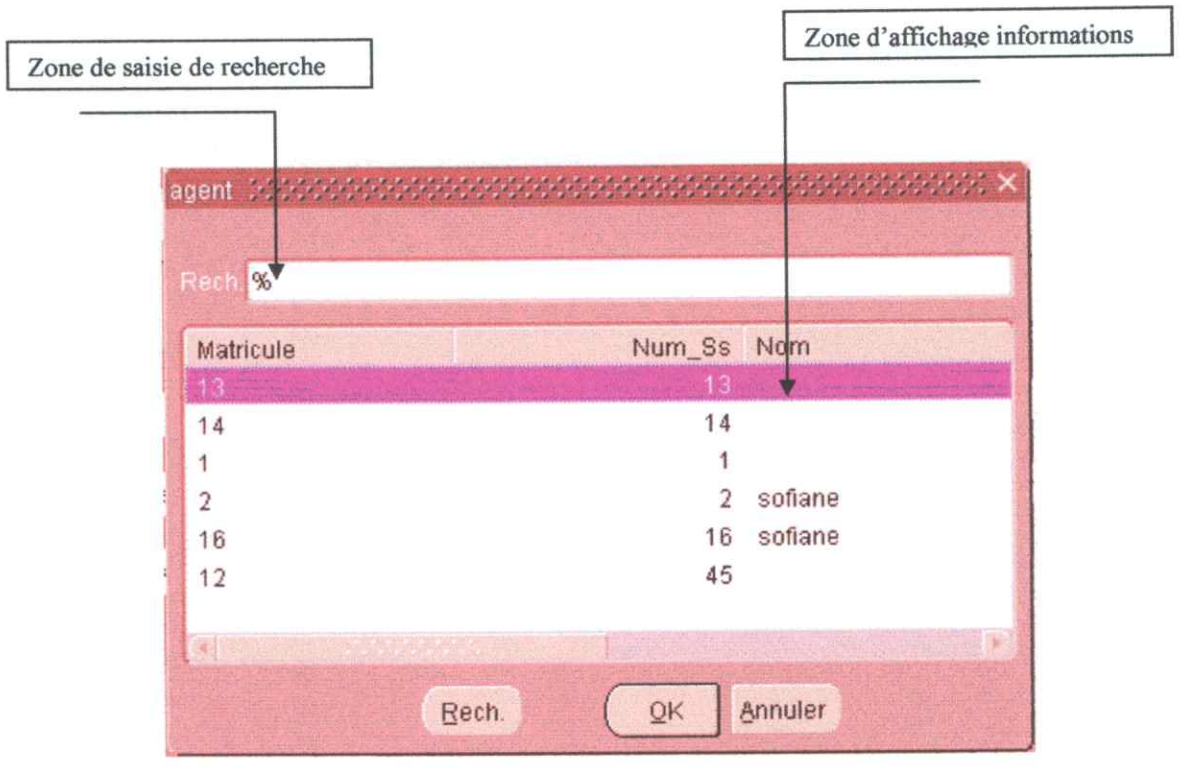
La page de recherche permet de faire des recherches ciblées notamment sur les agents, les arrêts de travail, les accidents de travail, etc....



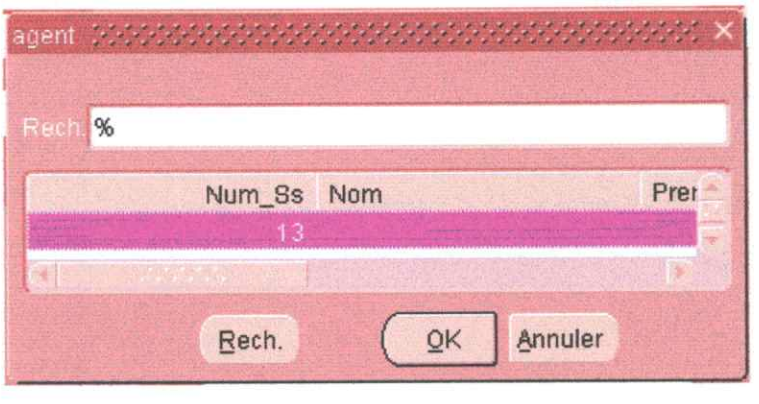
En détaillera cette page et son fonctionnement dans ce qui va suivre :



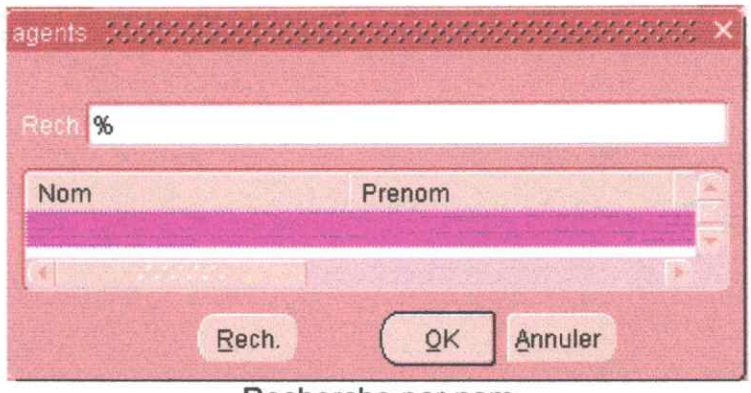
On choisissant l'anglet agent la figure suivante apparaît, et on pourra faire des recherche concernant les agents, comme recherche par matricule ou par numéro de sécurité sociale, etc..



Recherche par matricule.



Recherche par numéro de sécurité sociale.



Recherche par nom

3. conclusion :

Dans cette phase nous sommes arrivés à faire une application web pour le service prestations sociales qui fait le suivi et la gestion du personnelles de direction en appliquant les techniques d'accès aux bases de données ,et on utilisant les outils de développement tel que Oracle9i, Oracle IDS, JSP,etc....

Conclusion générale :

Notre travail ayant pour objectif la conception et la réalisation d'une base de données pour le service prestations sociales fonctionnant sur le réseau Internet et intranet, nous avons réalisé l'application en utilisant une architecture dite architecture trois-tiers, et une approche composant comme technique d'accès a la base de données a partir d'un serveur d'applications

A partir de ce serveur d'applications nous avons élaboré des composants JavaBeans qui effectuent des tâches bien spécifiques telle que charger, enregistrer des informations dans un serveur de données, faire une connection persistante a la base de données, etc.... puis nous avons entamé l'étape d'analyse qui nous a permis d'aboutir aux résultats attendus.

De l'approche méthodologique et du langage de modélisation adopté nous avons réalisés une base données au moyens de la méthode OMT et du langage UML avec le SGBD oracle 9i sous l'environnement Windows. Concernant la partie présentation nous avons utilisés les JSP et Oracle IDS

Grâce à la technique d'accès aux données a partir d'un serveur d'applications par l'utilisation des entités JavaBeans nous avons obtenu les résultats escomptés. Ces résultats ont été soumis à l'appréciation des responsables concernés de la direction générale de SONATRACH et nous espérons qu'ils constitueront des arguments valables pour qu'il soit intégré dans leur système.

Annexes

A. Internet

B. Le SQL

C. Présentation de Sonatrach

1 - INTRODUCTION

Contrairement aux idées reçues, Internet n'est pas un phénomène récent. C'est en 1969 que le gouvernement américain (plus précisément le DOD : Department Of Defense, soit le ministère de la défense) décide de mettre en place un système (Arpanet) pour relier les ordinateurs entre eux à des fins de défense nationale. En 1986, la NSF (National Science Foundation) et la NASA décident d'interconnecter les universités américaines. Très rapidement, ce système sera étendu à de nombreuses universités étrangères. Depuis 1992, Internet est devenu accessible au grand public.

Internet repose sur trois principaux types de services :

- La messagerie (e-mail ou mél en français) qui permet l'échange de documents et courriers électroniques. Un aspect intéressant de la messagerie est la possibilité de s'inscrire à une mailing list, c'est à dire émettre et recevoir des messages à une communauté de personnes ayant des centres d'intérêt communs.
- Les forums de discussions (newsgroups) qui permettent à plusieurs individus de se regrouper autour d'un même thème.
- Le Web permet d'accéder à des pages via un navigateur (browser). Ces pages contiennent du texte, des données, des fichiers, des images, du son, ...

Depuis son début, Internet connaît une croissance très forte. Le nombre de serveurs a presque doublé sur un an (source : http://www.nw.com/zone/host-count-history), passant de 19 à 38 millions en juillet 1998. Selon IDC, les dépenses liées au développement de sites passeront de 211 milliards de dollars cette année à un montant estimé de 954 Milliards de dollars en 2002. Les dépenses spécifiques au commerce en ligne passeront, toujours sur la même période, de 17 à 167 Milliards de dollars. Enfin, selon le groupe Forrester (http://www.forrester.com), le nombre de sites doublerait tous les 53 jours.

Toutes les activités informationnelles de la vie courante peuvent être réalisées sur Internet :

Consulter des bibliothèques, des rapports, des études, voir des expositions, visiter des écoles, contacter des experts, écouter la radio, voir la télévision, lire un journal ou un livre, "rencontrer" des gens et échanger des idées, ... L'information sur Internet est très, voire trop abondante.

2 - OUVERTURE A INTERNET

Internet, et son petit frère à l'échelle de l'entreprise intranet, deviennent aujourd'hui des éléments incontournables du système d'information. Le succès récent de cette technologie relativement ancienne est du à l'apparition du World Wide Web (WWW) en 1989, permettant à un utilisateur d'accéder au réseau Internet à travers un navigateur (browser). Ce dernier lui permet de visualiser les informations sous une forme ergonomique, sans avoir besoin de connaissances en informatique.
Les pages visualisées ne sont pas stockées sur le poste client mais sont envoyées, à la demande, par un serveur Web. Elles sont écrites en langage HTML (Hyper Text Markup Langage) et contiennent du texte formaté, des liens vers d'autres documents ou d'autres parties de la page présentée et des images.

2.1 - ACCES A DES PAGES STATIQUES

Affichage de rapports statiques. L'outil d'aide à la décision doit permettre de générer les rapports sous une forme HTML, afin de les visualiser depuis un navigateur et de les mettre à disposition des autres utilisateurs, par exemple sur un serveur d'entreprise, Intranet. Les pages définies sont alors des pages statiques, contenant à la fois la présentation et les données et non mises à jour dynamiquement. Afin de présenter aux utilisateurs des informations récentes, l'outil doit permettre de régénérer automatiquement les rapports à intervalles réguliers (par exemple tous les jours) ou après chaque chargement de données dans le Data Warehouse.
Automatisme du lien entre l'outil d'aide à la décision et le serveur Web. Le serveur Web devient ici un serveur d'entreprise, permettant simplement de stocker des documents et de les mettre à la disposition de toute personne possédant un navigateur Internet et ayant le droit d'accéder à ces pages. Pour que cette solution soit viable, il faut que l'outil soit capable de mettre automatiquement les pages HTML à disposition sur le serveur Web.
Même si elle apparaît sommaire, cette fonctionnalité peut être assez intéressante. Par exemple, dans le cadre d'une entreprise et d'un réseau intranet, les rapports élémentaires peuvent être mis à la disposition de l'ensemble des utilisateurs qui n'ont

ainsi pas à maîtriser et à employer un outil d'aide à la décision pour accéder aux informations qui les intéressent.
Mais, au delà de ces fonctionnalités statiques, il est également nécessaire d'accéder directement aux données à travers le navigateur et de générer des pages HTML dynamiquement, à la demande de l'utilisateur.

2.2 - ACCES A DES PAGES DYNAMIQUES

Au delà de l'envoi de pages statiques, le serveur Web est aujourd'hui capable de créer dynamiquement des pages, à la demande de l'utilisateur. Ceci peut se faire à travers des scripts CGI (Common Gateway Interface). Ils vont alors se charger d'interroger la base de données. Des interfaces plus évoluées sont proposées par Netscape avec NSAPI et par Microsoft avec ISAPI. Elles sont plus performantes et, contrairement à CGI, ne nécessitent pas la création d'un processus séparé à chaque exécution de script. Par exemple, ISAPI fait appel à des DLLs et non à des exécutables. Plus performantes, elles sont cependant moins fiables car un problème survenant lors de l'exécution d'un script risque de provoquer l'arrêt du processus qui est alors également celui du serveur Web (on parle de démon HTTP).
D'autre part, le langage Javascript permet de joindre des programmes à des pages HTML, afin de soulager le serveur et d'exécuter certains programmes au niveau client. Il peut s'agir, par exemple, de programmes permettant de ontrôler si l'utilisateur a bien renseigné tous les champs obligatoires dans un formulaire, avant de l'envoyer vers le serveur, afin d'éviter des aller et retours inutiles.
Enfin, le langage Java permet de créer de petites applications (appelées des applets) qui pourront être chargées directement sur le poste client et exécutées à partir du navigateur, il faut pour cela que ce dernier soit compatible Java.
Au delà du simple partage de rapports, la génération de pages dynamiques est une caractéristique indispensable. L'utilisateur doit pouvoir formuler ses requêtes et récupérer les résultats à travers son navigateur Internet.
D'autre part, il est nécessaire de lui laisser manipuler les données, par exemple dans le cas d'outils permettant d'effectuer de l'analyse multidimensionnelle, de naviguer dans les données.

Au delà d'applications " clé en main " mises à la disposition de l'utilisateur et lui permettant de manipuler les données dans le cadre qui lui a été imparti, ces outils devraient permettre à l'utilisateur de définir ses requêtes aussi librement qu'il le fait avec l'outil, de même pour la valorisation des résultats. Ceci permet alors de mettre à la disposition de l'ensemble des utilisateurs les données de l'entreprise, évite les coûts et les efforts d'installation et de mise à niveau des produits, l'application étant alors basée sur le serveur.

1. Présentation de SQL

SQL signifie Structured Query Language c'est-à-dire Langage d'interrogation structuré.

En fait SQL est un langage complet de gestion de bases de données relationnelles. Il a été conçu par IBM dans les années 70. Il est devenu le langage standard des systèmes de gestion de bases de données (SGBD) relationnelles (SGBDR).

C'est à la fois :

_ un langage d'interrogation de la base (ordre SELECT)

_ un langage de manipulation des données (LMD; ordres UPDATE, INSERT, DELETE)

_ un langage de définition des données (LDD ; ordres CREATE, ALTER, DROP),

_ un langage de contrôle de l'accès aux données (LCD ; ordres GRANT, REVOKE).

Le langage SQL est utilisé par les principaux SGBDR: DB2, Oracle, Informix, Ingres, RDB,... Chacun de ces SGBDR a cependant sa propre variante du langage. Ce support de cours présente un noyau de commandes disponibles sur l'ensemble de ces SGBDR, et leur implantation dans Oracle Version 7.

2. Normes SQL

SQL a éte normalisé dès 1986 mais les premières normes, trop incomplètes, ont été ignorées par les éditeurs de SGBD.

La norme actuelle SQL-2 (appelée aussi SQL-92) date de 1992. Elle est acceptée par tous mais les SGBD relationnels qui dominent actuellement le marché (en particulier Oracle) ne sont toujours pas totalement adaptés à cette norme.

SQL-2 définit trois niveaux :

_ Full SQL (ensemble de la norme)

_ Intermediate SQL

_ Entry Level (ensemble minimum à respecter pour se dire à la norme SQL-2)

3. Utilitaires associés

Comme tous les autres SGBD, Oracle comprend plusieurs utilitaires qui facilitent l'emploi du langage SQL et le développement d'applications de gestion s'appuyant sur une base de données relationnelle. En particulier SQL-FORMS facilite grandement la réalisation des traitements effectués pendant la saisie ou la modification des données en interactif par l'utilisateur. Il permet de dessiner les écrans de saisie et d'indiquer les traitements associés à cette saisie. D'autres utilitaires permettent de décrire les états de sorties imprimés, de sauvegarder les données, d'échanger des données avec d'autres logiciels, de travailler en réseau ou de constituer des bases de données réparties entre plusieurs sites.

Ce cours se limitant strictement à l'étude du langage SQL, nous n'étudierons pas tous ces utilitaires. Nous verrons les commandes essentielles d'un seul utilitaire, SQLPLUS, qui facilite l'utilisation interactive du langage SQL par un utilisateur. Ces commandes permettent de modifier les ordres SQL et de constituer des fichiers de commandes SQL que l'on peut réutiliser ensuite.

Les commandes du langage SQL peuvent être tapées directement au clavier par l'utilisateur ou elles peuvent être incluses dans un programme écrit dans un langage de troisième génération (Cobol, Langage C, Fortran, Ada,...) grâce à un précompilateur fourni par Oracle.

4. SGBD Oracle

Oracle est un SGBD (système de gestion de bases de données) édité par la société du même nom (Oracle Corporation - http://www.oracle.com/), leader mondial des bases de données.

La société *Oracle Corporation* a été créée en 1977 par Lawrence Ellison, Bob Miner, et Ed Oates. Elle s'appelle alors *Relational Software Incorporated (RSI)* et commercialise un Système de Gestion de Bases de données relationnelles (SGBDR ou RDBMS pour *Relational Database Management System*) nommé *Oracle.*

En 1979, le premier prototype (RDBMS - RSI1) intégrant la séparation des espaces d'adressage entre les programmes utilisateurs et le noyau Oracle est commercialisé. Cette version est entièrement développée en langage assembleur. La seconde version (RDBMS - RSI2) est un portage de l'application sur d'autres plates-formes.

En 1983 la troisième version apporte des améliorations au niveau des performances et une meilleure prise en charge du SQL. Cette version est entièrement codée en langage C. A la même époque RSI change de raison sociale et devient *Oracle*.

En 1984 la première version d'Oracle (Oracle 4) est commercialisée sur les machines IBM.

En 1985 Oracle 5 permet une utilisation client-serveur grâce au middleware *SQL*Net*.

En 1986 Oracle a été porté sur la plateforme 8086.

En 1988 Oracle 6 est disponible sur un grand nombre de plates-formes et apporte de nombreuses nouvelles fonctionnalités ainsi qu'une amélioration notable des performances.

En 1991, Oracle 6.1 propose une option *Parallel Server* (dans un premier temps sur la DEC VAX, puis rapidement sur de nombreuses autres plates-formes).

En 1992, Oracle 7 sort sur les plates-formes UNIX (elle ne sortira sur les plates-formes Windows qu'à partir de 1995). Cette version permet une meilleure gestion de la mémoire, du CPU et des entrées-sorties. La base de données est accompagnée d'outils d'administration (SQL*DBA) permettant une exploitation plus aisée de la base. En 1997, la version Oracle 7.3 (baptisée *Oracle Universal Server*) apparaît, suivie de la version 8 offrant des capacités objet à la base de données

Oracle est écrit en langage C et est disponible sur de nombreuses plates-formes matérielles (plus d'une centaine) dont :

- AIX (IBM)
- Solaris (Sun)
- HP/UX (Hewlett Packard)

- Windows NT (Microsoft)

Oracle depuis la version 8.0.5 est disponible sous Linux

Les versions d'Oracle

Oracle se décline en plusieurs versions

- Oracle Server **Standard**, une version comprenant les outils les plus courants de la solution Oracle. Il ne s'agit pas pour autant d'une version bridée...
- Oracle Server **Enterprise Edition**

Les fonctionnalités d'Oracle

Oracle est un SGBD permettant d'assurer :

- La définition et la manipulation des données
- La cohérence des données
- La confidentialité des données
- L'intégrité des données
- La sauvegarde et la restauration des données
- La gestion des accès concurrents
-

Les composants d'Oracle

Outre la base de données, la solution Oracle est un véritable environnement de travail constitué de nombreux logiciels permettant notamment une administration graphique d'Oracle, de s'interfacer avec des produits divers et d'assistants de création de bases de données et de configuration de celles-ci.

On peut classer les outils d'Oracle selon diverses catégories :

- Les outils d'administration
- Les outils de développement
- Les outils de communication
- Les outils de génie logiciel
- Les outils d'aide à la décision

Les outils d'administration d'Oracle

Oracle est fourni avec de nombreux outils permettant de simplifier l'administration de la base de données. Parmi ces outils, les plus connus sont :

- Oracle Manager (SQL*DBA)
- NetWork Manager
- Oracle Enterprise Manager
- Import/Export : un outil permettant d'échanger des données entre deux bases Oracle

Outils de développement d'Oracle

Oracle propose également de nombreux outils de développement permettant d'automatiser la création d'applications s'interfaçant avec la base de données. Ces outils de développement sont :

- Oracle Designer
- Oracle Developer
- SQL*Plus : une interface interactive permettant d'envoyer des requêtes SQL et PL/SQL à la base de données. SQL*Plus permet notamment de paramétrer l'environnement de travail (formatage des résultats, longueur d'une ligne, nombre de lignes par page, ...)
- Oracle Developper : il s'agit d'une suite de produits destinés à la conception et à la création d'applications client-serveur. Il est composé de 4 applications :
    - Oracle Forms (anciennement SQL*Forms) : un outil permettant d'interroger la base de données de façon graphique sans connaissances préalables du

langage SQL. SQL*Forms permet ainsi de développer des applications graphiques (fenêtres, formulaires, ...) permettant de sélectionner, modifier et supprimer des données dans la base.

- Oracle Reports (SQL*ReportWriter) : un outil permettant de réaliser des états
- Oracle Graphics : un outil de génération automatique de graphiques dynamiques pour présenter graphiquement des statistiques réalisées à partir des données de la base
- Procedure Builder : un outil permettant de développer des procédures, des fonctions et des packages

Outils de programmation

Oracle dispose d'un grand nombre d'interfaces (API) permettant à des programmes écrits dans divers langages de s'interfacer avec la base de données en envoyant des requêtes SQL. Ces interfaces (appelées précompilateurs) forment une famille dont le nom commence par *PRO** :

- Pro*C
- Pro*Cobol
- Pro*Fortran
- Pro*Pascal
- Pro*PLI
- ...

3. Les commandes SQL

En SQL, les commandes sont dites de format libre. Cela signifie qu'il n'existe aucune règle indiquant qu'un mot donné doit commencer à une position particulière de la ligne. Cette manière d'écriture est plus lisible.

Quelques commandes SQL :

CREATE Table : permet de créer une table,
DROP Table : pour supprimer une table,
INSERT : en ajoutant des lignes aux colonnes, on encadre les valeurs par des apostrophes,
SELECT : permet d'afficher les données sélectionnées,
UPDATE : pour modifier la valeur d'une donnée,
DELETE : pour supprimer une enregistrement,

Des conditions sont associées à ces commandes grâce à la clause **WHERE**, elle est utilisée pour retrouver les enregistrements qui répondent à une certaine condition. Cette condition est appelée simple. En connectant plusieurs conditions simples par l'utilisation des opérateurs : **AND**, **OR** et **NOT**, on aura les conditions combinées.
La clause **IN** est une manière concise de formuler certaines conditions.

L'opérateur **BETWEEN** n'est pas une fonctionnalité essentielle de SQL, mais il rend toutefois certaines commandes SELECT plus simples.
Dans la plupart des cas, les conditions nécessitent une concordance exacte. Pour cela, l'opérateur **LIKE** avec un caractère joker est utilisé.

L'ordre des lignes dans une table est sans importance dans un SGBD. D'un point de vue pratique, lors de l'interrogation d'une base de données relationnelle, le résultat s'affiche sans ordre prédéfini. Les lignes peuvent s'afficher dans l'ordre choisis spécifié à l'aide de la clause **ORDER BY**.

SQL dispose de fonctions pour calculer :
la somme : en utilisant la fonction **SUM**,
la moyenne : grâce à la fonction **AVG**,
ainsi pour compter, on utilise **COUNT**,
et pour trouver la valeur du maximum et de minimum, en introduisant les fonctions **MAX** et **MIN** respectivement.
L'opérateur **DISTINCT** n'est pas une fonction, mais il peut néanmoins se révéler utile dans certaines situations, couplé avec la fonction **COUNT**.

La clause GROUP BY permet de grouper des données dans un ordre particulier puis de calculer des statistiques si nécessaire.
La clause HAVING est utilisée pour les groupes, en effet, elle fait pour les groupes ce que la commande WHERE fait pour lignes.

En SQL, on réalise la jointure entre tables en incluant une condition dans la clause WHERE de façon à s'assurer que les colonnes en concordance contiennent des valeurs égales. Pour réaliser la jointure, on utilise des sous requêtes avec **IN** et **EXISTS**.

I. PRESENTATION GENERALE

I.1. Présentation de l'organisme d'accueil (SONATRACH)

I.1.1. Dénomination :

Société Nationale pour la recherche, la production, le Transport, la transformation et la Commercialisation des Hydrocarbures.

I.1.2. Forme juridique :

Entreprise publique et économique a sa création, elle est transformée, sans création d'une nouvelle personne morale, en une société par actions (SPA) par le décret présidentiel N°98-48 du 11 Février 1998.

I.1.3. Siége social :

Le siège social de SONATRACH est à Hydra ALGER, sis Djenane El Malik. Il peut être transféré en tout autre lieu du territoire national par délibération de l'assemblée générale.

I.1.4. Capital social :

SONATRACH dispose d'un capital social de **245.000.000.000 (deux cents quarante cinq milliards de dinars)** réparti en deux cent quarante cinq mille actions d'un million de dinars chacune, entièrement et exclusivement souscrit et libellé par l'état.

I.2. Historique de la SONATRACH

L'entreprise nationale SONATRACH (Société nationale de transport de transformation et de commercialisation des hydrocarbures) a été créée le **31/12/1963 (décret 63-491)** pour assurer la responsabilité de la production du transport et la commercialisation des hydrocarbures.

Les missions et les prérogatives de l'entreprise nationale SONATRACH ont été élargies le **22 septembre 1966 (décret 66-626)** ; ainsi ses missions qui se limitaient à l'origine au transport, la transformation et la commercialisation des hydrocarbures ont été élargies à tous les domaines de l'industrie pétrolière, à savoir la prospection, la recherche, la production, le transport, la transformation et la commercialisation des hydrocarbures.

Depuis le **24 février 1971**, date de nationalisation des hydrocarbures, l'entreprise a pris en charge l'ensemble du domaine minier et s'est vue confier le développement de toutes les branches de l'industrie pétrolière.

La SONATRACH est passée de 33 agents en 1964 à 103.000 vers la fin des années 80. Pour assurer une meilleure gestion et améliorer les performances dans le cadre de la politique nationale pour la réorganisation de l'économie du pays, elle entreprend sa restructuration pour donner naissance à 17 entreprises industrielles.

Actuellement, la SONATRACH compte un effectif de 36.000 agents environ et conserve pour sa part la charge des opérations de recherche, de production, de transport par canalisation, de traitement, conditionnement et liquéfaction des hydrocarbures liquides et gazeux.

Dans le cadre de la restructuration décidé en 1982, la SONATRACH a fait l'objet d'un découpage qui a donner naissance à treize entreprises parmi lesquelles, NAFTAL, NAFTEC, ENTP, ASMIDAL, ENSP, ...etc.

I.3. Missions principales de la SONATRACH

Sous l'autorité d'un Directeur Général, la SONATRACH a notamment pour missions essentielles :

- Le développement, la conservation et la valorisation des réseaux énergétiques sur tout le territoire national.

- La reconstitution et l'augmentation des réserves d'hydrocarbures.

- L'intensification des efforts d'exploitation et capitalisation des études réalisées dans ce domaine, pour une meilleure connaissance du sous-sol et la mise en évidence des réserves d'hydrocarbures.

- La diversification des marchés et des produits destinés a l'exportation.

- L'approvisionnement énergétique national à moyen terme, compte-rendu des réserves nationales.

- L'adaptation de l'outil commercial aux exigences du marché énergétique pour une meilleure maîtrise de ses mécanismes et des performances commerciales accrus.

- Le développement la maîtrise et la maintenance des complexes de production, de transport et de conditionnement des hydrocarbures.

- Le développement des techniques modernes de gestion nationale par le biais de la formation continue.

Figure 1-1 Organigramme de l'ensemble de l'entreprise de SONATRACH

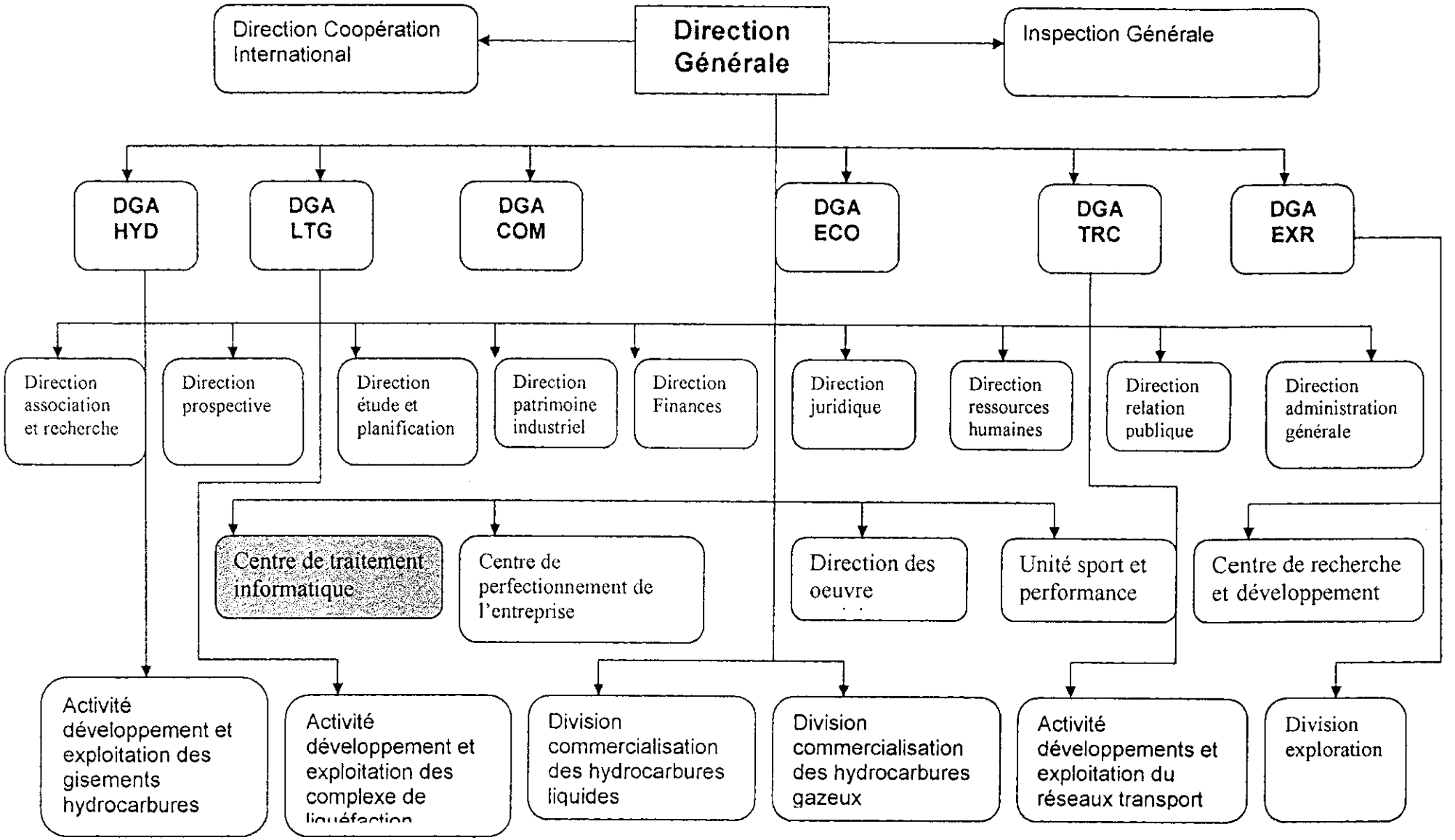
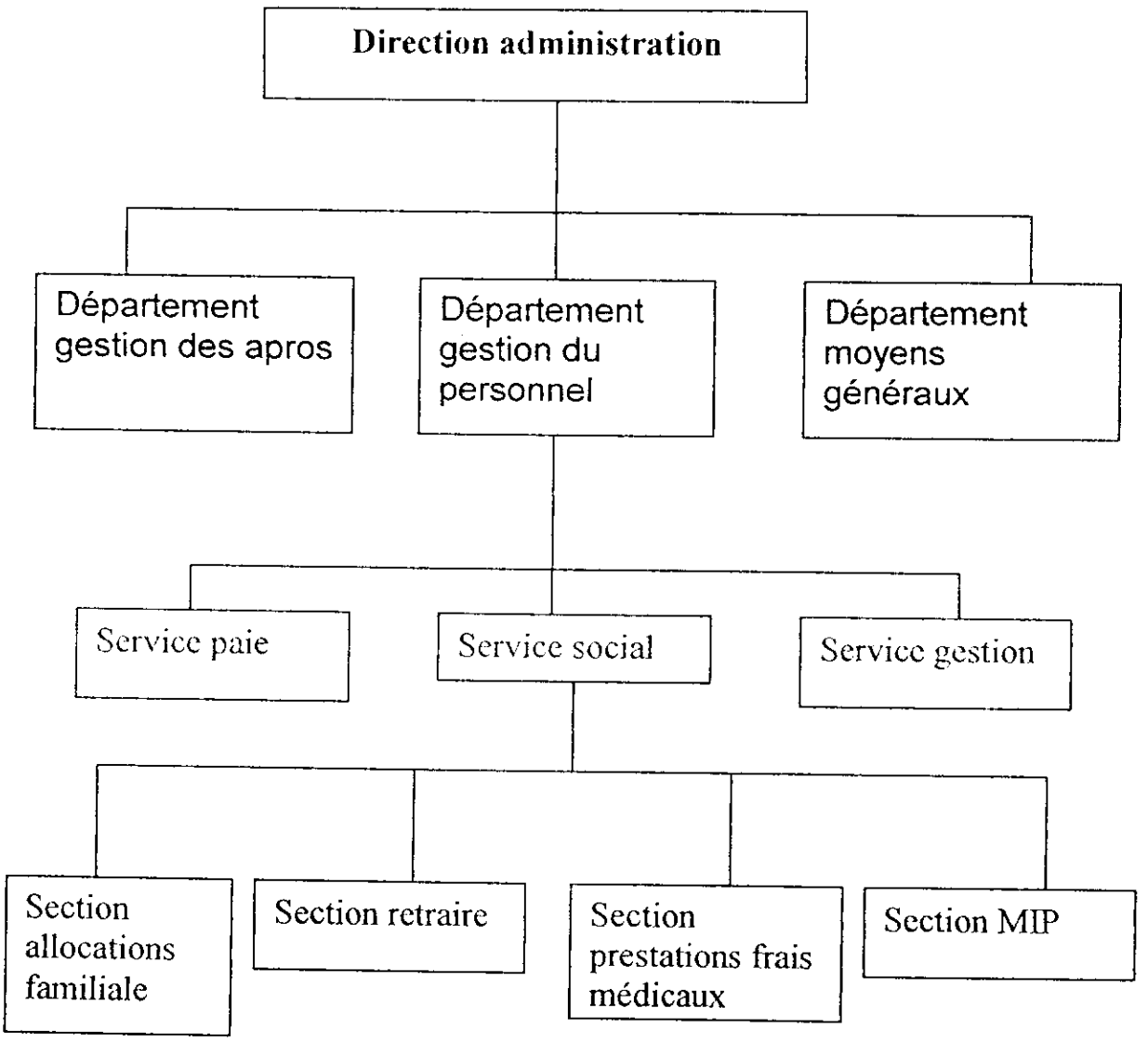


Figure 1-2 Organigramme de la structure d'étude



Note

Web[3.1]

WWW (World Wide Web): Encore appelé Web. En anglais, web signifie tissage ou toile d'araignée, en référence à la topologie du réseau Internet. Le CERN (Centre Européen de Recherches Nucléaires) fut l'acteur principal du WWW. Par la suite, le suivi du WWW fût pris en charge par le W3C (World Wide Web Consortium).

CGI[3.2]

CGI (Common Gateway Interface) : Nous verrons par la suite que cette interface qui présente l'avantage d'être standard et reconnue par tous, n'est pas très performante et tend à être progressivement remplacée.

SGML[3.3]

SGML (Standard General Markup Language) : Standard de description de document permettant de se concentrer sur le contenu plutôt que sur la forme.

TCP/IP[3.4]

HTTP n'est pas du tout lié au protocole TCP/IP et on pourrait envisager de faire fonctionner une version adaptée à un protocole comme IPX.

ISAPI[3.5]

ISAPI est l'API d'extension des serveurs Web de Microsoft.

NSAPI[3.6]

NSAPI est l'API d'extension des serveurs Web de Netscape.

contraignant[3.7]

Pas question, en effet, d'utiliser le click droit pour les applications devant fonctionner sur Macintosh, par exemple.

OAK[3.8]

OAK signifie chêne en anglais.

graphiques[3.9]

Les traditionnelles classes AWT, mais aussi, depuis la version 1.2 du JDK (Java Development Kit), les classes SWING permettant de réaliser des interfaces graphiques indépendantes du système hôte.

JDBC[3.10]

JDBC (Java Data Base Connectivity) : API fédérant la connexion des programmes Java à des bases de données.

RMI[3.11]

RMI (Remote Method Invocation) : Mécanisme d'appel de méthode distante utilisé par les programmes Java.

CORBA[3.12]

CORBA (Common Object Request Broker Architecture) : Architecture objet de l'OMG visant à l'intégration d'applications par la communication entre objets sur le réseau de l'entreprise.

Références bibliographiques :

[LEF97] Alain LEFEBURE. Eyrolles, 1997
"Intranet clent-serveur universel"

[ROO] Robert OGOR
"Modélisation avec UML".

[PRF03] Pascal Roques , Franck Vallée 2003
"UML en action, de l'analyse des besoins a la conception en java ".

[FAP00] Fabrice POPINEAU 2000
"Introduction a la conception orientée objet ".

[LAU96] Laurent HENOCQUE - Ecole supérieur d'ingénieur de luming, 1996
"Introduction a la méthode OMT "

[DKK01] Duane K. FIELDS, Mark A. KOLB 2001.
" JSP java server page "

[JMF97] Jean-Marie FAVRE 1997
"Java Beans"

[GER01] George Reese ,2001
" JDBC et JAVA Guide du programmeur "

[GIB03] Gilles Briard avec la collaboration de la société Digora, 2003
" Oracle 9i sous windows "

Les sites web:

www.sun.com

www.oracle.com

cedric.babault.free.fr

www.abcdoc.net

www.java-source.com

www.moteurprog.com

www.eunis.org

www.orsys.fr

www.eclipse.org

jakarta.apache.org

www.lesjsp.com

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique.

Université Saad Dahlab, Blida
USDB.

Faculté des sciences.
Département informatique.

Mémoire pour l'obtention
d'un diplôme d'ingénieur d'état en informatique.
Option : system d'information

Sujet :

Conception et réalisation d'une base de données prestations sociales accessible via le web en architecture trois-tiers

Présenté par : Mr. Zeraoui Sofiane
Mr. Benhalima Mohamed

Promoteur : Mr. A. Cherif Zahar

Organisme d'accueil : SONATRACH (Direction Générale)

Soutenance le : , devant le jury composé de :

Nom. Président du jury, grade, organisme	Président
Nom. Examinateur 1, grade, organisme	Examinateur
Nom. Examinateur 2, grade, organisme	Examinateur

2005/2006

REMERCIEMENT

Mes remerciements vont à tout ce qui ont contribué, de près ou de loin à mener à terme ce travail, notamment les enseignants qui m'ont instruit tout au long de mon long parcours.

Sans oublier notre promoteur Mr CHERIF ZAHAR pour ces orientations et conseils fructueux.

Je remercie chaleureusement tout le personnel de la SONATRACH et plus particulièrement
Mr TAYEBI responsable du service prestations sociales et
Mr A.DAIMALLAH responsable de la formation et de l'infographie.

DÉDICACES

J'ai le grand plaisir de dédier le fruit de mes années d'études aux êtres qui me sont les plus chers, mes parents.

A ma mère qui s'est données tant de mal pour moi et qui ma offert amour et soutien depuis mon premier souffle.

A mon père pour avoir mis tous les moyens a ma disposition et qui ma encouragé et soutenu pendant mes études

A mes chère sœurs et mon frère qui m'ont données chaque un une aide très précieuse a leurs façons.

Et toutes les personnes que j'aime et qui m'aime.

A mon binôme Mohamed.

A mes chères amis, Mr Goumari, youcef, tarik, azouaou, ibrahim, yacine, chaoukette, hamza noureddine, lyes, souad, keireddine, abdelhak, zaki, ali, mahfoud, et tout mes amis de la promo 2005/2006

Zeraoui Sofiane

DÉDICACES

Je dédie ce mémoire :

A mes très chers parents qui mon beaucoup soutenu et encouragé durant mes années d'études

A ma grand mère et mon grand père

A ma Sœur Djamila et ses fils Wissam et Nabila

A mes frères Zoubir et Zakaria

A mes amis

A tous mes amis de la promo 2005/2006

Benhalima Mohamed

Résumé :

Ce document et les annexes qui l'accompagnent constituent le résultat de notre travail qui a consisté en la conception et la réalisation d'une base de données accessible via le web

Après analyse des informations recueillies auprès du service prestations sociales de la direction générale de Sonatrach, nous avons utilisé OMT&UML comme méthodologie pour la conception de notre système dont l'objectif est de faire le suivi des dossiers social du personnel de la direction générale de SONATRACH en se servant de la technologie Internet.

Dans cette perspective nous avons utilisé l'architecture trois-tiers, que nous avons détaillée au chapitre 3 du présent document.

En effet, à partir d'un serveur d'applications on accède à la base de données par l'approche composant avec les entités JavaBeans.

Mots Clés : OMT, UML, Architecture trois-tiers, l'approche composant, JavaBeans.

Sum up:

This document and the enclosures which go with constitute the result of our work which consisted in the conception and the realisation of a data base accessible via the Web.

After the analyse of the collected information from the social insurance benefits service of the general management of SONATRACH, we have used the OMT and UML as a methodology for the conception of our system which objective is to trace the social files of the general management of SONATRACH using the Internet technology.

For this outlook we used the three-third architecture that we have detailed in chapter3 of the present document.

In deed, from an application sever we accede to the data base by the component approach with the JavaBeans entities

Key Words : OMT, UML, Architecture trois-tiers, l'approche composant, JavaBeans.

SOMMAIRE

Chapitre 3 : L'approche composant

Chapitre 4 : Analyse et conception du système

Chapitre 5 : Implémentation et résultats

Liste des figures :

Liste des tableaux

INTRODUCTION GENERALE

SONATRACH en tant que l'une des principales sociétés pétrolières dans le monde n'a pas voulu rester en marge des avancées technologiques dans le domaine de l'Internet. Compte tenu que l'entreprise soit éparpillée un peu partout sur le territoire algérien et du fait que ce n'est pas toujours facile de faire le suivi des dossiers du personnel surtout si on est toujours en déplacement. Le besoin d'une application web pour le service prestations sociales se fait ressentir. La mission est de mettre en place cette application dans intranet de la société reliant tout d'abord les sites de l'entreprise à Alger, puis la rendre présente sur le web.

Le web s'est développé comme un hypertexte sur le réseau Internet pour permettre facilement l'accès à des fichiers chaînés, rapidement le besoin de couplage avec les bases de données est apparu. Trois raisons au moins motivent ce besoin :

- ✓ L'introduction du clients-serveur à présentation universelle (architecture trois-tiers)
- ✓ La génération des sites web dynamique composés à partir de template HTML et de données extraites de bases.
- ✓ Le commerce électronique, qui nécessite la gestion de catalogues et de transactions en base de données.

Notre projet d'étude consiste en la conception et la réalisation d'une base de données prestations sociales, accessible via le web en architecture 3-tiers pour le siége de la SONATRACH.

Pour faire ce travaille, il existe plusieurs techniques d'accès aux bases de données à partir d'un serveur d'applications ,notamment , l'approche par page comme le PHP, l'approche par scriptlette avec les page JSP ou encore l'approche par composant utilisant les JavaBeans dans les pages JSP.

Dans cette perspective nous avons retenu pour notre projet cette dernière approche pour les raisons suivantes :

- profit des avantages de la portabilité et la puissance de java.
- Utilisation d'une technologie récente de conception de sites dynamiques à savoir le JSP (java Server page).

Notre mémoire s'articule essentiellement autour de cinq chapitres :

- Dans le premier chapitre sera présenté l'étude du domaine où la société souhaite améliorer son fonctionnement
- dans le deuxième chapitre, sera mis l'accent sur les system d'informations en architecture trois-tiers et les bases de données&web
- dans le troisième chapitre, sera présenté l'approche composant, et la persistance des objets java avec les entités Beans du J2EE.
- Dans le quatrième chapitre, sera présenté le langage de modélisation objet UML. Et une méthode de conception OMT, puis l'analyse et la conception du system.
- Dans le dernier chapitre Nous présenterons les langages de programmations utilisés ainsi que la description détaillée de notre application

Enfin, une conclusion général et des perspectives clôturant notre travail.

Chapitre 1

Etude de l'existant

1. INTRODUCTION

Toute l'imagination et tout le savoir-faire de l'analyste seront sans emploi, s'ils ne peuvent s'exercer sur une base réaliste.

Pour que l'application soit mise à la disposition de ceux qui ont besoin et à qui elle est destinée, l'étude de l'existant est le point de passage obligé qui matérialise le premier contact avec un domaine ignoré, c'est la première étape dans notre étude, elle nous permet de découvrir en détail le domaine où la société souhaite améliorer son fonctionnement.

Cette étude a pour but de nous permettre de:

1/ connaître en détail l'architecture Internet du siège SONATRACH.

2/ connaître les fonctions des services « prestation sociale » ainsi que les flux d'informations externe et interne

3/ voir en détail les postes de travails et les fonctions de service prestations sociales.

Avant d'être un réseau ou un ensemble de technologies, Internet est un standard mondial de communication

Il offre la possibilité

- de gagner en productivité dans la gestion interne et dans les relations interentreprises, et
- d'ouvrir aux entreprises des marchés nouveaux.

SONATRACH, groupe pétrolier International, étant la plus importante entreprise nationale, doit utiliser cette technologie pour ne pas être à la marge du progrès.

En effet, Les technologies Internet concernent toutes les fonctions de l'entreprise :

- ✓ vendre
- ✓ se faire connaître
- ✓ trouver des partenaires
- ✓ faire de la veille technologique et de l'intelligence économique
- ✓ transmettre des documents écrits, sonores ou vidéo

✓ conduire des projets, faxer, téléphoner, participer à des bourses, travailler en réseau....

2. ARCHITECTURE INTERNET

- **DMZ**

  Contenant les serveurs devant être perçus de l'extérieur.

- **Cisco PIX F**

  • Système (hard & soft) dédié sécurisé en temps réel,

  • Nombreuses options de connexion LAN :

  – Ethernet

  – Fast Ethernet

  – Token Ring ET FDDI

  • Support de 250.000 connexions simultanées.

- **Serveur proxy**

Avec l'utilisation du serveur proxy Les performances peuvent être encore améliorées ; L'enregistrement des sites WEB dans sa mémoire cache rendent la connexion plus rapide.

PROXY serveur tournant sur un port précis en attente de connexion cliente, capable d'accepter et de traiter plusieurs demandes de la part de plusieurs clients simultanément, Son rôle est de récupérer les informations sur les requêtes de ses clients et de les transmettre à PROXYREQUETE pour les traitements et le renvoie de la réponse.

- **routeur**

Internet résulte de l'interconnexion de différents réseaux physiques par des machines appelées routeurs. Chaque réseau, contenant un ensemble d'hôtes, est connecté, éventuellement, à un routeur. Le routeur permet de déterminer l'adresse physique de destinataire et de faire circuler les messages (paquets).

- **Switch**

Les ponts permettent d'étendre les possibilités du LAN : nombre de stations, distance, confidentialité, taux de défaillance. Un pont multiport est appelé commutateur (switch). Ils permettent également d'optimiser les débits. Les ponts sont des ordinateurs complets travaillant au niveau de la couche 3 et sont souvent multi protocoles. Un pont reçoit les trames circulant sur les segments raccordés, les analyses. Il récupère ainsi les numéros Ethernet des stations actives sur un segment. En fonction du destinataire, il émet ou rejette la trame. Un pont compte comme une station sur chaque segment. On mesure la qualité d'un pont par son taux de filtrage et son taux de transfert. Le maximum théorique de ces deux quantités est14 880 paquets de 64 octets par seconde.

La figure 1.1 illustre l'architecture Internet de siége Sonagraphe ; chaque Vlan correspond à une direction de siége SONATRACH c'est-à-dire architecturé en Vlan fonctionnels (Virtual Local Area Network) correspondants aux différentes structures qui y sont hébergées,

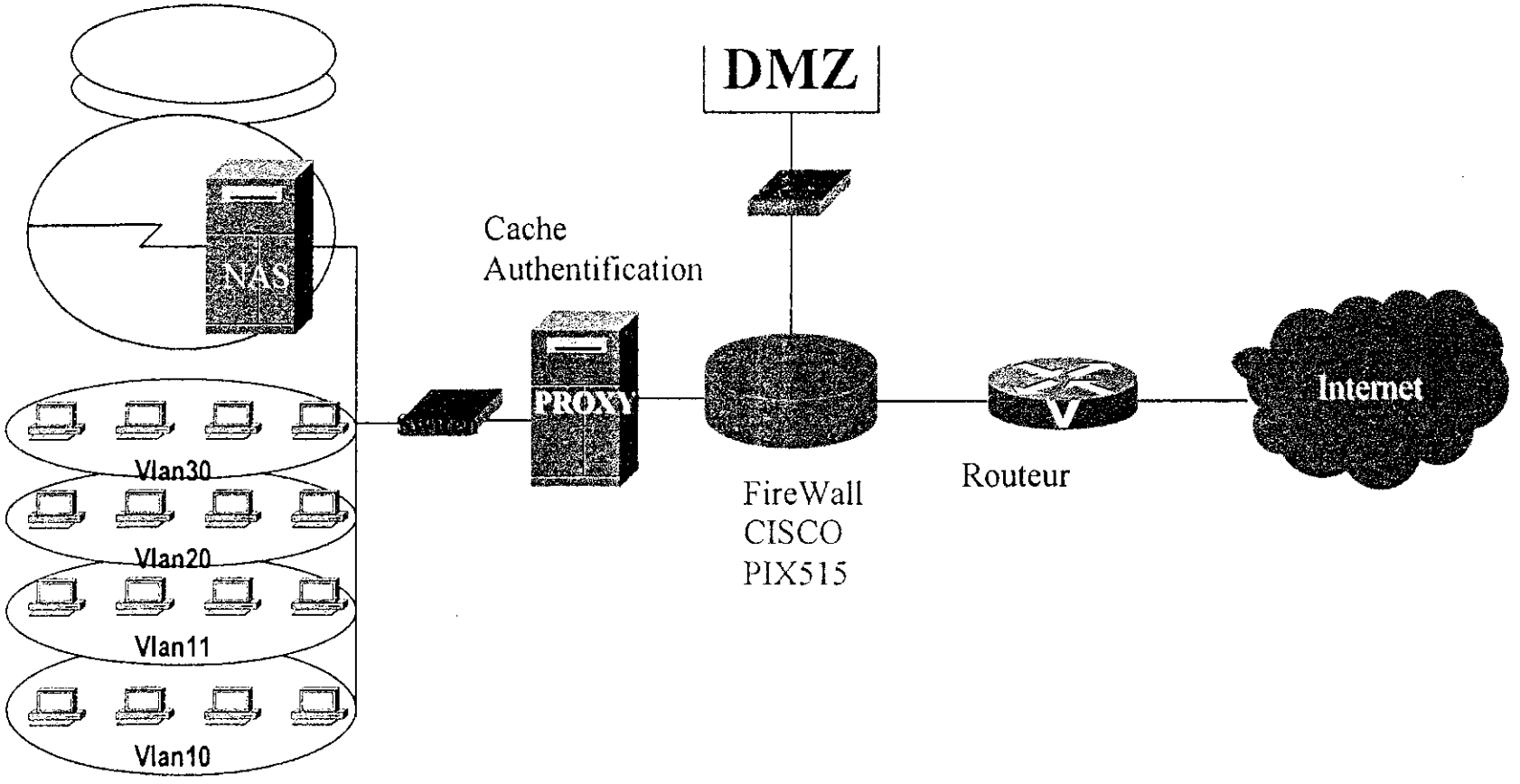


Figure 1.1. Architecture Internet de siége SONATRACH

3. Présentation de service prestation sociale :

Notre but est de mettre en place une application web en vue de la relation d'un system d'information pour le suivi des dossiers social du personnel de la direction générale de SONATRACH.

De ce fait l'application doit rependre aux besoins du service prestations sociales qui accompli entre autre les tâches suivantes :

- Les remboursements médicaux.
- Les allocations familiales
- La mutuelle
- La retraite.

3.1. Flux D'information externe :

3.1.1 Le Schéma de Flux D'information externes

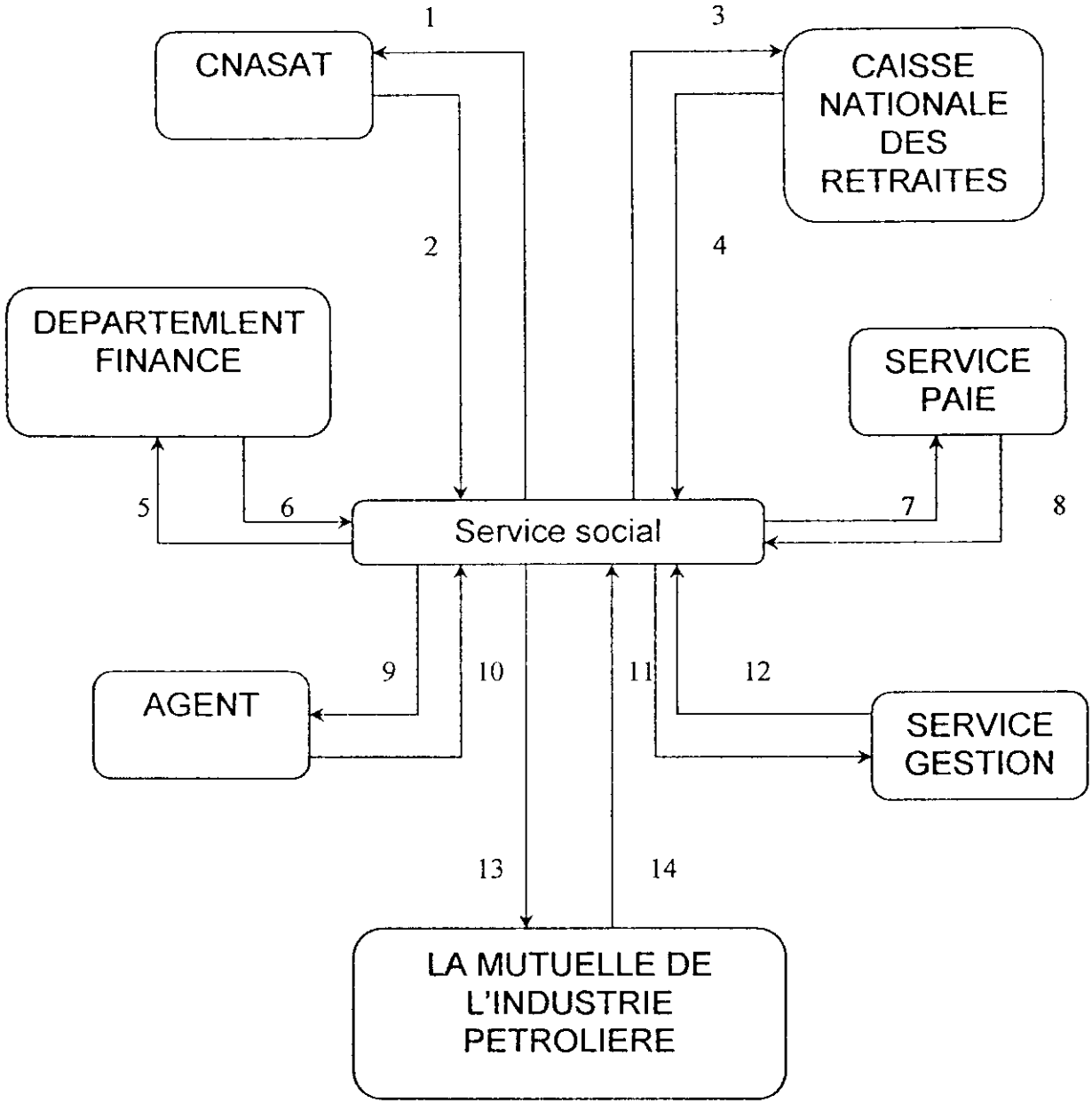


Figure 1.2 flux d'informations externe

3.1.2. Description du Flux d'informations Externes

N° DE LIAISON	DESIGNATION
01	- Envoi de la déclaration d'emploi à la CNAS. - Envoi des bordereaux d'exploitation à la CNAS. - Envoi des dossiers frais médicaux et arrêts de travail à la CNAS. - Envoi des bordereaux de paiement des allocations familiales, états des impayés et les bordereaux de réclamation à la CNAS. - Envoi des pièces de renouvellement et des situations nouvelles à la CNAS. - Envoi des dossiers indemnités journalières à la CNAS. - Envoi des lettres types à la CNAS.
02	- Réception des bordereaux d'exploitation. - Réception des cartes d'assurance. - Réception des chèques barrés. - Réception des décomptes de la CNAS. - Réception des récapitulations de la CNAS. - Réception des bordereaux de paiement des allocations familiales et des bordereaux d'exploitation (indemnités journalières ou remboursement des frais médicaux) avec accusés de réception.
03	- Envoi des dossiers de mise en retraite à la CNR ainsi que la lettre type.
04	- Réception de la notification de la CNR.

05	- Envoi des bordereaux de paiement internes au département finance. - Envoi des chèques barrés au département finance. - Envoi des récapitulations au département finance. - Envoi des bordereaux de versement des cotisations au département finance.
06	- Réception des notes internes du département finance ainsi que les lettres types.
07	- Envoi des bordereaux de paiement internes au service paie.
08	Réception des journaux de paie du service paie.
09	- Envoi des cartes d'assurance aux intéressés. - Envoi des décomptes à l'agent. - Envoi des convocations à l'agent.
10	- réception des dossiers frais médicaux et arrêts de travail. - réception des dossiers allocations familiales. - Réception des fiches familiales, certificats de scolarité, bulletin de naissance, extrait de décès et une copie de jugement. - Réception de mise en retraite des agents. - Réception des pièce : facture cliniques, acte des mariage, prescription médicale pour la circoncision.
11	- envoi de la photocopie de la notification au service de gestion.
12	- Réception d'une fiche de renseignement du service de gestion. - Réception des décisions du service de gestion.
13	- envoi des document MIP + décomptes CNAS + les pièces + bordereaux nominatif de prestation a la MIP. - Envoi de chèque barré et du bordereau de versement de

	cotisation a la MIP.
14	- réception de bordereau de versements de cotisation avec accusé de réception.

Tableau 1.1 Flux d'informations Externes

3.2. Flux d'information interne :

3.2.1 Schéma de flux d'informations internes

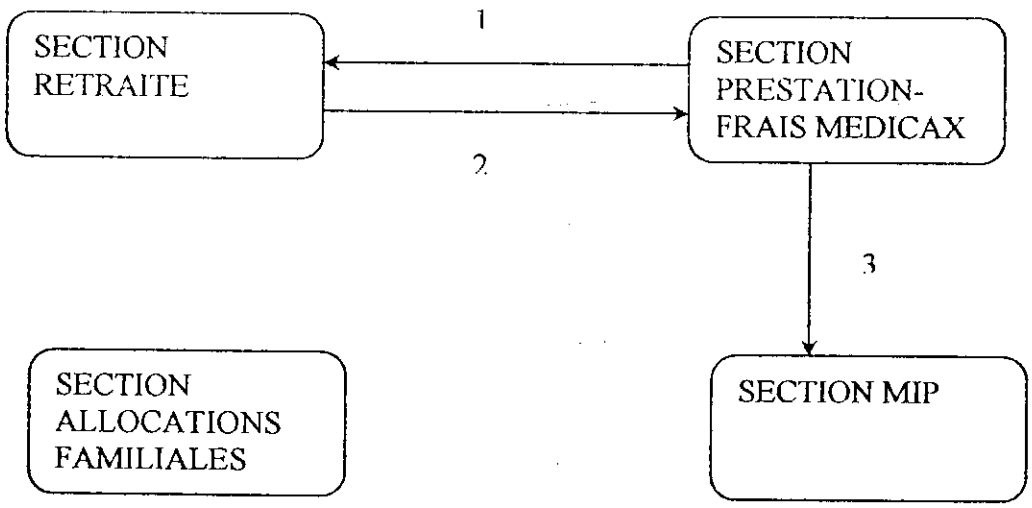


Figure 1.3 flux d'information interne

3.2.2. DESCRIPTION DU FLUX D'INFORMATIONS INTERNES

N^0 DE LIAISON	DESIGNATION
1	- envoi des bordereaux d'exploitation des arrêts de travail a la section retraite. - Envoi des déclarations d'accidents de travail et des bordereaux d'exploitation avec accusé de réception.
2	- Envoi des arrêts de travail a la section prestation frais médicaux. - Envoi des déclarations d'accident de travail, des infirmités et des questionnaires a la section prestations frais médicaux.
3	- Envoi des décomptes frais médicaux a la section *MIP*

Tableau 1.2 Flux d'informations internes

3.2.3. Fiche de Fonction des postes de travail :

Fiche de Fonction du Poste 01 :

Désignation : chef section prestations / assurance sociale

Section : retraite

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribuées

- indemnités journalières
- retraite
- capitale décès

b- tache à accomplir :

N°	Tâches	fréquences	remarque
1	- consultation des butins de paie des agents	-1fois/mois	-envoyer par le service de paie
2			
3	- exploitation des arrêts de travail	-1fois/mois	
4	- établissement des dossiers des indemnités journalières	-1fois/mois	
5	- Remplissage des fiches médicales	-1fois/mois	
6	- Etablissement des bordereau e paiement des prestation	-1fois/mois	-en 3exemplaires
7	- envoi des dossiers des indemnités journalières	-1fois/mois	
8	- mise à jour des fiches médicales	-1fois/mois	
9	- Envoi de la récapitulation et du chèque et des décomptes au service finance	-1fois/mois	
10	- établissement des bordereaux d'envoi pour les envoyer au service paie	-1fois/mois	
11	- recensement des agents partant en retraite	- 1fois/an	
12	- Envoi des convocations aux agents partant en retraite	- 1fois/an	
13	- envoi des dossiers de la mise en retraite à la CNR	- 1fois/an	
14	- Envoi d'une note interne et d'une notification au service GESTION	-irrégulier	

15	- aviser la CNR de la cessation de travail des retraités - Positionnement des retraités ainsi que tous les renseignements les concernant sur le registre du suivi personnel mis en retraite	-irrégulier -irrégulier	

Tableau 1.3 Fonction du Poste 01

Fiche de Fonction du Poste 2 :

Désignation : charger de gestion

Section : MIP

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribuées :

- remboursement MIP (20%)
- Cotisation

b- tâches accomplir :

N°	Tâches	fréquences	remarque
1	- établissement des décomptes de prestation	-1fois/semaine	
2	- établissements des bordereaux nominatifs de prestation	-1fois/semaine	
3	- envoi des décomptes et des bordereaux a la MIP	-1fois/semaine	
4	- établissement des bordereaux d'envoi	-1fois/mois	
5	- classement des décomptes	-1fois/mois	-par direction et par ordre alphabétique
6	- remplissage des fiches d'adhésion	-aléatoire	
7	- consultation des journaux de paie	-1fois/mois	
8	- établissement des bordereaux de versement de cotisation	-1fois/mois	
9	- envoi des chèques, bordereaux de cotisation et des bordereaux d'exploitation a la MIP	-1fois/mois	

Tableau 1.4 Fonction du Poste 02

Fiche de Fonction du Poste 3

Désignation : cadre de gestion administration

Section : prestation – frais médicaux

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.S)

– Fonction :

a- Tâches attribuées

- remboursements des frais médicaux
- immatriculation

b- tâches à accomplir :

N^0	Tâches	Fréquences	Remarques
1	- positionnements des dossiers sur le fichier du suivi des remboursements des frais médicaux	1/semaine	-Par N^0 SS
2	- établissement des bordereaux 'exploitation	1/semaine	-Par N^0 SS
3	- envoi des dossiers et des bordereaux a la CNAS	1/semaine	-Pour la collectivité
4	- classements des décomptes	1/semaine	-Pour la
5	- mise à jour du fichier du suivi des remboursements des frais médicaux	1/semaine	collectivité
6	- classements des décomptes par direction et par ordre alphabétique	1/mois	
7	- calcule du montant globale de chaque décompte	1/mois	
8	- établissement des bordereaux de paiements	1/mois	
9	- envoi des bordereaux de paiements aux services PAIE et FINANCE	1/mois	
10	- envoi des décomptes aux intéressés	1/mois	
11	- remplissage des fiches signalétique	Aléatoire	
12	- remplissage des déclarations d'emploi du travailleur	Aléatoire	
13	- envoi des déclarations d'emploi du travailleur a la CANS	1/semaine	
14	- positionnement des N^0 de sécurité sociale sur les fiches signalétiques	Aléatoire	
15	- envoi des cartes d'assurance aux intéressés	Aléatoire	

Fiche de fonction du poste 4

Désignation : cadre de gestion allocation familiale

Section : allocation familiale

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribués :

- s'occupe des allocations familiales

b- tâches à accomplir :

N^0	Tâches	Fréquences	Remarque
1	- envoi des convocations a l'intéressé	Aléatoire	
2	- établissements des attestations de salaire et de travail	Aléatoire	
3	- établissements des fiches d'allocations	Aléatoire	
4	- établissements des bordereaux d'exploitations	1/mois	
5	- établissements de bordereaux de réclamations ou additif	1/mois	
6	- établissement des notes internes	1/mois	
7	- envoi des bordereaux	1/mois	
8	- envoi des notes internes	Aléatoire	
9	- envoi des pièces de renouvellement et des situations nouvelles a la CNAS	1/mois	

Tableau 1.5 Fonction du Poste 04

4. Conclusion :

On a pu découvrir en détail les moyens et l'infrastructure où la société souhaite fonctionner notre application. Ensuite, on a mis en évidence les fonctions du service « prestations sociales » en définissant les postes de travail, les tâches qui lui sont attribuées, et les flux d'informations qui circulent entre eux.

Notre but dans le chapitre suivant est de détailler les systèmes d'information en architecture trois-tiers et les bases de données &web, car l'approche composant avec les JavaBeans s'applique dans le domaine des bases de données&web

Chapitre 2

Système d'information
&
Architecture trois-tiers

1. Introduction :

Dans ce chapitre nous allons exposer la notion des systèmes d'informations distribuées et l'architecture trois-tiers ainsi que les bases de données&web.

L'objectif premier d'un système d'information quel qu'il soit est de permettre à plusieurs utilisateurs d'accéder aux mêmes informations. Pour cela il faut donc regrouper les informations utilisées par l'entreprise. En terme technique, cela se traduit par la centralisation des données au sein d'une base de données. L'évolution des systèmes d'information s'est donc basée sur une meilleure subdivision entre les tâches à réaliser pour permettre l'exploitation de ces données par les utilisateurs finaux. Ceci permet de structurer plus efficacement les informations ce qui entraîne à la fois une meilleure organisation de l'entreprise et une meilleure efficacité technique. Cette subdivision a été facilitée par l'avènement des technologies orientées objets qui s'appliquent aussi bien au modèle client-serveur qu'au modèle Internet. Ces technologies permettent une séparation entre les différents composants du système. Il devient alors possible de réaliser de nouvelles architectures permettant la mise à disposition des informations sous différentes formes tout en diminuant les temps de développement. Ces technologies permettent également de faire collaborer une grande diversité de systèmes. On parle alors d'architecture distribuée. Il est ainsi possible de présenter des données en provenance d'un mainframe mélangées à des données en provenance d'un SGBDR, le tout étant affiché dans un browser sur la même page HTML. [LEF97]

2. Système d'information et l'architecture trois-tiers :

Tout système d'information nécessite la réalisation de trois groupes de fonctions: le stockage des données, la logique applicative et la présentation. Ces trois parties sont indépendantes les unes des autres: on peut ainsi vouloir modifier la présentation sans modifier la logique applicative. La conception de chaque partie doit également être indépendante, toutefois la conception de la couche la plus basse est utilisée dans la couche d'au dessus. Ainsi la conception de la logique applicative se base sur le modèle de données, alors que la conception de la présentation dépend de la logique applicative [LEF97]

2.1. Trois tâches importantes

- **Stockage et accès aux données**

Le système de stockage des données a pour but de conserver une quantité plus ou moins importantes de données de façon structurée. On peut utiliser pour cette partie des systèmes très variés qui peuvent être des systèmes de fichiers, des mainframes, des systèmes de bases de données relationnelles, etc. Le point commun entre tous ces systèmes est qu'ils permettent le partage des données qu'ils contiennent via un réseau. La méthode d'accès à ces données dépendra du type d'organisation de ces données. Dans le cas d'une base de données relationnelle, l'accès peut se faire par des API qui dépendent du langage et de l'environnement. Ainsi en JAVA l'accès se fait via JDBC, alors qu'en C++ il se fera à l'aide d'ODBC. Quel que soit l'API, le langage SQL est utilisé.

- **Logique applicative**

La logique applicative est la réalisation informatique du mode de fonctionnement de l'entreprise. Cette logique constitue les traitements nécessaires sur l'information afin de la rendre exploitable par chaque utilisateur. Les utilisateurs peuvent avoir des besoins très variés et évolutifs. Il devient alors nécessaire de permettre l'évolution du système sans pour autant devoir tout reconstruire. Cette partie utilise les données pour les présenter de façon exploitable par l'utilisateur.

Il convient donc de bien identifier les besoins des utilisateurs afin de réaliser une logique applicative utile tout en structurant les données utilisées.

- **Présentation**

La présentation est la partie la plus immédiatement visible pour l'utilisateur. Elle a donc une importance primordiale pour rendre attrayante l'utilisation de l'informatique. Son évolution a été très importante depuis les débuts de l'informatique. Depuis les terminaux en mode texte connectés à des mainframes jusqu'au HTML de nos jours en passant par les applications graphiques développées en client serveur, il y a eu beaucoup de chemin parcouru. Différents types d'interfaces demeurent intéressantes. En effet l'ergonomie d'un site Intranet HTML n'est pas forcément idéale pour tous les types d'applications. Il peut être intéressant de proposer plusieurs types d'interface pour une seule logique applicative. Par exemple une entreprise disposant d'un site de commerce électronique peut proposer un accès à la liste de ses produits sur Internet en HTML mais disposer d'une interface d'administration réalisée à l'aide d'une applet graphique [LEF97]

2.2. Principe de l'architecture trois-tiers

Le principe d'une architecture trois-tiers est relativement simple: il consiste à séparer la réalisation des trois parties vues précédemment (stockage des données, logique applicative, présentation). Nous avons déjà pu entrevoir la possibilité de séparer la conception de ces trois subdivisions, ici il s'agit de séparer leur implantation. Tout comme dans le client-serveur cette séparation signifie qu'il est possible de déployer chaque partie sur un serveur indépendant, toutefois cela n'est pas obligatoire. La mise en place de ce type d'architecture permet dans tous les cas une plus grande évolutivité du système. Il est ainsi possible de commencer par déployer les deux serveurs sur la même machine, puis de déplacer le serveur applicatif sur une autre machine lorsque la charge devient excessive. Les éléments permettant la réalisation classique d'un système en architecture trois tiers sont les suivants:

- système de base de donnée relationnel (SGBDR) pour le stockage des données
- serveur applicatif pour la logique applicative
- navigateur web pour la présentation

Il est important de remarquer que l'essentiel du travail de développement sera implanté au niveau du serveur applicatif. Le SGBDR nécessitera un travail d'administration surtout dans le cas d'une quantité de données importante. Le travail de conception de la base de donnée sera la pierre angulaire du système. En effet l'ensemble du développement s'appuiera sur cette conception. Le navigateur web nécessitera la programmation de code spécifique permettant de gérer l'affichage par ce navigateur. Ce code sera placé sur le serveur applicatif pour permettre une mise à jour sans nécessiter de nouveaux déploiements.

2.3. Avantages de cette architecture

Cette architecture se développe actuellement au sein des entreprises grâce aux nombreux avantages qu'elle présente. Malgré la différence évidente entre une architecture trois tiers et un système client-serveur (l'apparition d'un serveur pour la logique applicative), le système reste basé sur les technologies éprouvées détaillées précédemment (aspect relationnel et transaction). La logique applicative est déplacée au niveau du serveur d'application mais reste programmée à l'aide des mêmes technologies liées aux bases de données relationnelles. En particulier l'utilisation du langage SQL reste jusqu'à présent la solution la plus intéressante au niveau de la qualité logicielle. Elle présente à la fois une grande fiabilité, une bonne disponibilité, une excellente évolutivité,... Toutefois il faut prendre en compte deux facteurs importants: d'une part le choix du SGBDR (ils n'ont pas tous les même qualités), d'autre part la qualité des programmes utilisant la base de données (aussi bien au niveau de la conception que de la programmation). L'avantage principal d'une architecture multi-tiers est la facilité de déploiement. L'application en elle même n'est déployée que sur la partie serveur (serveur applicatif et serveur de base de données). Le client ne nécessite qu'une installation et une configuration minime. En effet il suffit d'installer un navigateur web compatible avec l'application pour que le client puisse accéder à l'application, ce navigateur étant par ailleurs souvent installé par défaut sur toutes les machines. Cette facilité de déploiement aura pour conséquence non seulement de réduire le coût de déploiement mais aussi de permettre une évolution régulière du système. Cette

évolution ne nécessitera que la mise à jour de l'application sur le serveur applicatif. Ceci est très important car cette évolutivité est un des problèmes majeurs de l'informatique. Le troisième avantage est l'amélioration de la sécurité. Dans un système client-serveur tous les clients accédaient à la base de données ce qui la rendait vulnérable. Avec une architecture multi-tiers l'accès à la base n'est effectué que par le serveur applicatif. Ce serveur est le seul à connaitre la façon de se connecter à cette base. Il ne partage aucune des informations permettant l'accès aux données, en particulier le login et le password de la base. Il est alors possible de gérer la sécurité au niveau de ce serveur applicatif, par exemple en maintenant la liste des utilisateurs avec leurs mots de passe ainsi que leurs droits d'accès aux fonctions du système. On peut même améliorer encore la sécurité par la mise en place d'une architecture réseau interdisant totalement l'accès au serveur de base de données pour les utilisateurs finaux. La mise en place de firewall correctement configuré permettra ceci. Dans le cas de la mise en place d'un workflow au sein de l'entreprise Kallisto, l'avantage le plus intéressant est sans aucun doute l'évolutivité du système. En effet les besoins actuels sont relativement vastes, toutefois la réalisation totale semble relativement longue et difficile à réaliser d'un seul coup. De plus des problèmes risquent de nécessiter une mise à jour répétée du système. Un autre aspect intéressant est la possibilité d'utiliser le système en Extranet, ce qui chez Kallisto permettra de faciliter la communication entre le siège social situé à Lyon et l'agence de Paris [LEF97]

3. Les bases de données et le web

3.1. La révolution Internet :

S'il est un phénomène qui a marqué le monde de l'informatique ces dernières années, c'est bien celui d'Internet.
Ce réseau mondial, créé en 1969 par l'armée américaine, puis utilisé par les chercheurs et autres scientifiques, a connu une croissance phénoménale auprès du grand public avec l'introduction du *World Wide Web*[3.1] en 1989. Ce dernier permet

de publier simplement des informations richement mises en forme et pouvant même, par la suite, contenir des données multimédia.

La véritable révolution du WWW réside dans son caractère universel, rendu possible par l'utilisation de standards reconnus.

3.1.1. Les standards d'Internet

L'universalité du Web repose sur des standards simples et admis par tous :

- HTML, pour la description des pages disponibles sur le Web,
- HTTP, pour la communication entre navigateur et serveur Web,
- TCP/IP, le protocole réseau largement utilisé par les systèmes Unix,
- CGI[3.2], l'interface qui permet de déclencher à distance des traitements sur les serveurs Web.

- HTML (HyperText Markup Langage)

HTML est le langage de description de pages hypertexte utilisé par le World Wide Web, il est issu de SGML[3.3].

Une page HTML est composée de son contenu propre (du texte plus ou moins richement mis en forme) et de références statiques vers d'autres sources d'informations (images, liens vers d'autres documents...).

La consultation d'une page HTML n'implique donc que très rarement le chargement du seul fichier décrivant la page, il s'accompagne en général de celui de nombreux fichiers annexes. Ces chargements mettent en oeuvre le protocole HTTP.

- HTTP

HTTP est un protocole réseau applicatif (dernier niveau du modèle OSI) sans connexion utilisé pour l'échange des données sur le Web. En fait, une connexion HTTP est créée pour chaque requête et ne dure que pendant l'exécution de cette dernière.

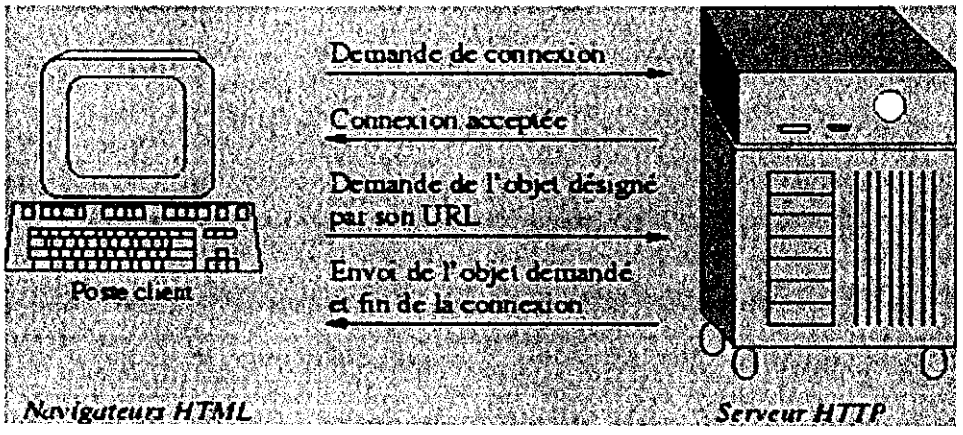


Figure 2.1: Le fonctionnement de base de HTTP[LEF97]

Le protocole HTTP, en temps que protocole réseau applicatif, s'appuie sur un protocole de transport indépendant qui, dans le cadre d'Internet, est TCP/IP[3,4].

- TCP/IP (Transmission Control Protocol / Internet Protocol)

TCP/IP est un protocole réseau de niveau trois et quatre (réseau et transport) qui s'est largement imposé sur les systèmes Unix, puis sur Internet, et fait aujourd'hui figure de standard universel.

Si le fonctionnement de TCP/IP s'adapte bien à la topologie et à la qualité de service du réseau Internet, on ne peut en dire autant du mariage avec le protocole HTTP. En effet, TCP/IP utilise un mécanisme de ``démarrage lent'' afin d'éviter les engorgements du réseau. Ce mécanisme permet à la machine émettrice d'ouvrir progressivement une fenêtre de congestion en doublant le nombre de paquets émis à chaque aller-retour. En général, la courte durée des échanges HTTP ne permet pas à la fenêtre de congestion d'atteindre la largeur de bande fournie par le réseau local [LEF97].

- **CGI (Common Gateway Interface)**

CGI est un standard permettant d'écrire des extensions compatibles avec la grande majorité des serveurs HTTP. Ces extensions permettent l'exécution d'une action par le serveur à la demande d'un client.

Ce mécanisme relativement simple, voire même rustique, entraîne l'exécution d'un processus propre à chaque invocation, ce qui est très consommateur de ressources.

De ce fait, des extensions comme ISAPI[3.5], NSAPI[3.6] ou les servlets Java sont souvent préférés au standard CGI.

3.1.2. Adaptation à l'entreprise : Intranet

Aucun des mécanismes mis en oeuvre par Internet n'est exempt de défaut et il est relativement simple de trouver plus performant. En fait, la force de l'ensemble repose essentiellement dans son universalité.

La notion d'Intranet est née de l'intégration des principes d'Internet et des technologies déployées dans l'entreprise :

- on utilise le réseau local de l'entreprise,
- les données sont toujours gérées par un SGBD,
- les mécanismes utilisés pour interroger le SGBD sont toujours les mêmes.

3.2. Répartition des traitements

L'architecture trois tiers, encore appelée client-serveur de deuxième génération ou client-serveur distribué, sépare l'application en trois niveaux de service distincts :

- **premier niveau :** l'affichage et les traitements locaux (contrôles de saisie, mise en forme de données...) sont pris en charge par le poste client,
- **deuxième niveau :** les traitements applicatifs globaux sont pris en charge par le service applicatif,

- **troisième niveau** : les services de base de données sont pris en charge par un SGBD.

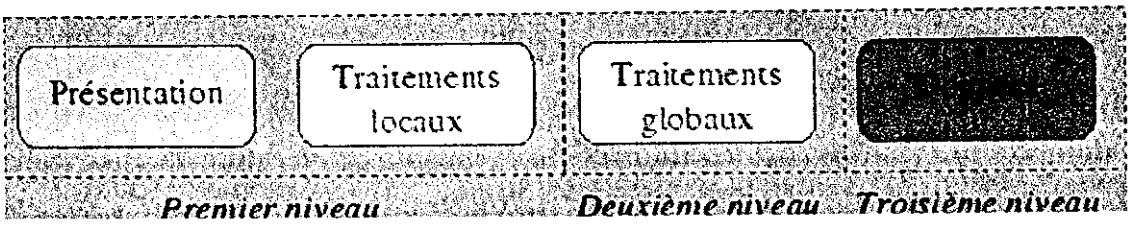


Figure 2.2: Le découpage d'une application en pavés fonctionnels indépendants

Tous ces niveaux étant indépendants, ils peuvent être implantés sur des machines différentes, de ce fait :

- le poste client ne supporte plus l'ensemble des traitements, il est moins sollicité et peut être moins évolué, donc moins coûteux,
- les ressources présentes sur le réseau sont mieux exploitées, puisque les traitements applicatifs peuvent être partagés ou regroupés (le serveur d'application peut s'exécuter sur la même machine que le SGBD),
- la fiabilité et les performances de certains traitements se trouvent améliorées par leur centralisation,
- il est relativement simple de faire face à une forte montée en charge, en renforçant le service applicatif.

Dans le cadre d'un Intranet, le poste client prend la forme d'un simple navigateur Web, le service applicatif est assuré par un serveur HTTP et la communication avec le SGBD met en oeuvre les mécanismes bien connus des applications client-serveur de la première génération.

Ce type d'architecture fait une distinction nette entre deux tronçons de communication indépendants et délimités par le serveur HTTP :

Le premier tronçon relie le poste client au serveur Web pour permettre l'interaction avec l'utilisateur et la visualisation des résultats. On l'appelle **circuit froid** et n'est composé que de standards (principalement HTML et HTTP). Le serveur Web tient le rôle de ``façade HTTP'',

- le deuxième tronçon permet la collecte des données, il est aussi appelé **circuit chaud**. Les mécanismes utilisés sont comparables à ceux mis en oeuvre pour une application deux tiers. Ils ne franchissent jamais la façade HTTP et, de ce fait, peuvent évoluer sans impacter la configuration des postes clients.

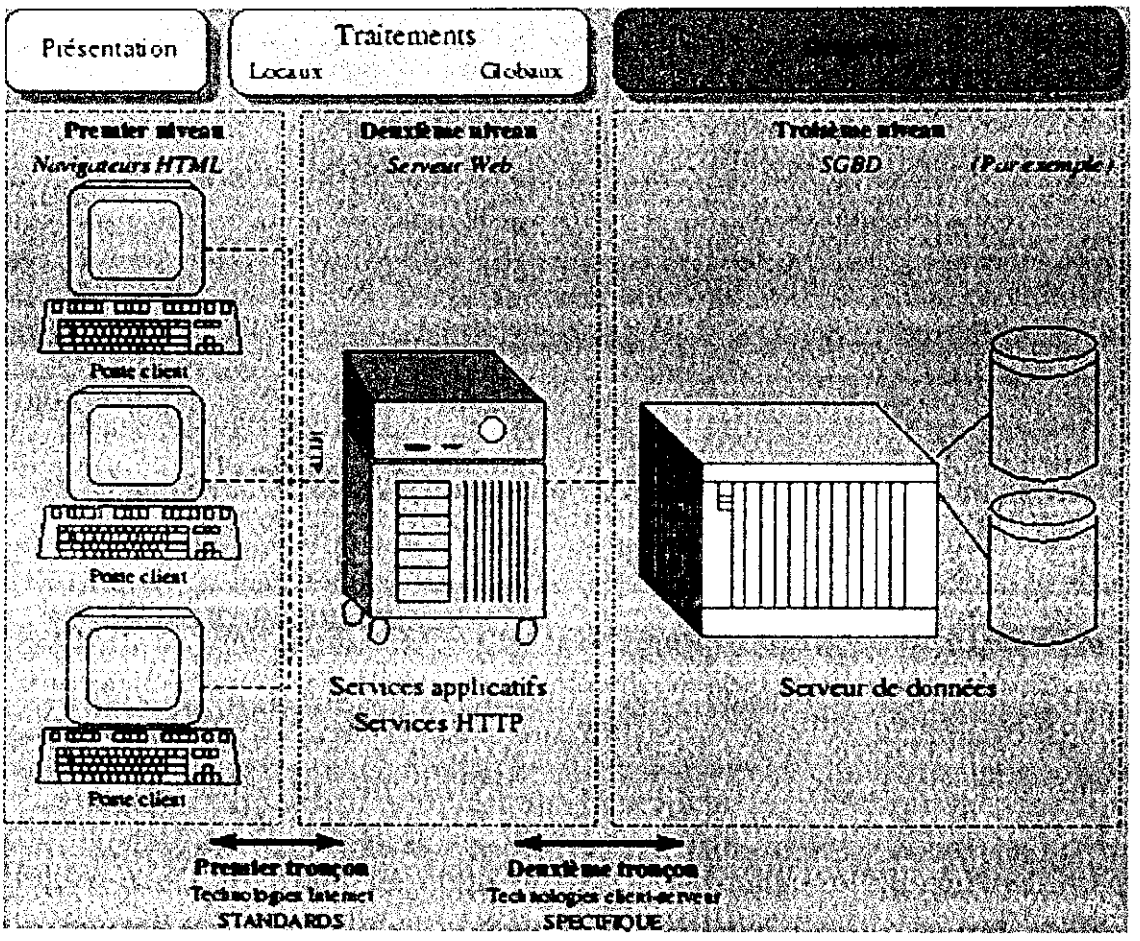


Figure 2.3: Répartition des couches applicatives dans une architecture trois tiers

3.3. Le client léger

3.3.1. Présentation

Dans l'architecture trois tiers, le poste client est communément appelé client léger ou *Thin Client*, par opposition au client lourd des architectures deux tiers. Il ne prend en charge que la présentation de l'application avec, éventuellement, une partie de logique applicative permettant une vérification immédiate de la saisie et la mise en forme des données. Il est souvent constitué d'un simple navigateur Internet.

Le poste client ne communique qu'avec la façade HTTP de l'application et ne dispose d'aucune connaissance des traitements applicatifs ou de la structure des données exploitées. Les évolutions de l'application sont donc possibles sans nécessiter de modification de la partie cliente.

Par exemple, un internaute se connectant à *www.yahoo.fr* pour effectuer une recherche provoque, sans même le savoir, l'exécution de traitements sur le serveur. Si ces traitements évoluent, ce qui doit arriver relativement souvent, le client continuera à utiliser le service sans se rendre compte des changements (sauf s'ils lui apportent de nouveaux services).

De plus, ce même internaute peut se connecter au serveur en utilisant tout type de poste client disposant d'un navigateur compatible HTML (PC sous Windows, Macintosh, Station Unix, WebPhone...).

On voit donc ici la force des architectures trois tiers par rapport au client-serveur de première génération. Le déploiement est immédiat, les évolutions peuvent être transparentes pour l'utilisateur et les caractéristiques du poste client sont libres.

3.3.2. Ergonomie

- Utilisation d'HTML

Les pages HTML, même avec l'aide de langages de script, sont loin d'atteindre les possibilités offertes par l'environnement Windows :

- le multifenêtrage n'est pas facile à mettre en oeuvre,
- le déroulement de l'application doit se faire séquentiellement,
- les pages affichées sont relativement statiques,
- le développement multiplateforme peut être contraignant[3.7].
- l'ergonomie de l'application est limitée aux possibilités du navigateur.

Pour ces raisons, certaines applications ne sont pas réalisables dans une architecture de type Intranet (infocentres, applications bureautiques...).

Dans les autres cas, l'appauvrissement de l'interface utilisateur est souvent le gage d'une plus grande facilité de prise en main de l'application.

Il est rare en effet de devoir suivre une formation pour apprendre à se servir d'un site Web particulier. La plupart du temps, une formation générale à l'ergonomie des sites Web suffit.

Cette perte de richesse peut donc se transformer en avantage, à condition de respecter une charte graphique et ergonomique cohérente pour toutes les applications Intranet d'une entreprise.

Il est possible d'aller au delà des possibilités offertes par le langage HTML en y introduisant des applets Java ou des contrôles ActiveX. Nous ne parlerons pas ici des modules d'extension du navigateur, encore appelés *plug-in*, qui étaient en vogue avant l'arrivée des solutions Java et ActiveX, car cette solution est trop contraignante à déployer.

- **Utilisation de Java**

Java est un langage de développement orienté objet et multiplateforme introduit par Sun en 1995. Il s'agit de l'adaptation à Internet du langage OAK[3.8], initialement étudié pour les environnements de petite taille. Il permet, entre autre, d'écrire de petites applications, appelées *applet*, pouvant être intégrées à des pages HTML pour en enrichir le contenu.

Le caractère multiplateforme de Java se prête bien à une utilisation sur Internet, où les caractéristiques des postes clients ne sont pas maîtrisées. Il repose sur l'utilisation d'un interpréteur de pseudo-code Java, pompeusement appelé ``machine virtuelle".

Les programmes Java ne sont pas compilés en code machine, mais en pseudo-code Java uniquement compréhensible par la machine virtuelle. Cette dernière interprète le code et se charge de lier les modules au moment de l'exécution, en les téléchargeant si nécessaire. La liaison dynamique des modules au moment de l'exécution permet d'optimiser le trafic réseau, puisqu'on ne charge que le strict nécessaire.

Pour ces raisons, un programme Java s'exécute plus lentement que son équivalent compilé. La compilation à la volée des programmes Java et plus encore, la technologie d'optimisation dynamique de code *HotSpot* de Sun, permettent de réduire l'écart de performance avec des programmes compilés.

Java propose aujourd'hui une large palette de composants graphiques[3.9] et multimédia qui permettent d'atteindre la richesse fonctionnelle des applications Windows.

Une applet Java est aussi capable d'exploiter directement un serveur de données en utilisant JDBC[3.10] ou de faire appel à des procédures distantes en utilisant RMI[3.11] ou CORBA[3.12]. Nous verrons ces mécanismes par la suite.

4. Conclusion

Etant donné que notre travail est basé sur les bases de données et le Web nous avons mis l'accent sur l'architecture trois-tiers et ses trois niveaux ainsi que l'environnement logiciel qui permettent a cette architecture de fonctionné sur le Web, pour la conception d'applications web et des pages dynamiques.
Notre but dans le chapitre suivant est d'étudier et de détailler l'approche composante avec les JavaBeans.

Chapitre 3

L'approche composant

1. Introduction :

Notre objectif dans ce chapitre est de présenter la réalisation technique de l'architecture trois-tires, et l'approche que nous avons adoptés pour la réalisation de notre application ; puis on verra en détail les composants JavaBeans et leurs fonctionnements ainsi que l'architecture orientée servlets.

2. Outils pour la réalisation de system d'information en architecture trois-tiers :

Nous avons donc vu ce qu'est une architecture trois-tiers ainsi que ses avantages. Nous allons maintenant présenter les choix techniques effectués pour la réalisation de cette architecture. En fait, il existe actuellement plusieurs solutions permettant ce type d'architecture. Ces technologies possèdent chacune leurs avantages et leurs inconvénients. On peut ainsi construire un système multi-tiers basé uniquement sur des technologies Microsoft comme ODBC, ASP, SQL Server... Ces technologies sont relativement efficaces mais ont pour principal inconvénient de n'être pas du tout portables. Kallisto utilise également des technologies spécifiques à Oracle basées sur Oracle Application Server, le serveur applicatif d'Oracle. Cette technologie est particulièrement efficace car elle permet de n'utiliser que des API natives à Oracle. Toutefois les systèmes développés avec ces outils ne pourront pas fonctionner avec autre chose. Cet inconvénient n'est pas obligatoirement important puisqu' Oracle fonctionne tout de même sur un grand nombre de systèmes (la plupart des UNIX et Windows). La solution choisie ici est basée sur le langage JAVA. Elle utilise une base de données Oracle, un serveur applicatif compatible J2EE nommé WebLogic, et un navigateur web comme client. Le JAVA étant un langage 100% portable, le développement du système ne dépendra absolument pas de la machine sur laquelle il fonctionnera. De plus WebLogic suivant les dernières spécifications de J2EE, les développements réalisés pourront fonctionner sur de nombreux autres serveurs applicatifs.

Oracle

Afin de stocker les données il fallait également une base de données. La société Kallisto étant spécialisée dans les bases Oracle, ce choix s'imposait. J'ai d'ailleurs pu profiter d'une formation interne dispensée la semaine de mon arrivée par un expert Oracle (10 ans d'expérience). Oracle est un système de gestion de base de données relationnelle très connu. Il est réputé pour être performant, fiable,... Oracle est en fait la référence en matière de base de données relationnelle. Les possibilités d'administration sont importantes et permettent de gérer des bases de tailles importantes (ce qui n'est pas primordial dans notre cas).

Java 2 Enterprise Edition (J2EE)

Le langage JAVA est un langage objet qui présente de nombreux avantages. Le premier d'entre eux est de supporter les notions de la programmation orienté-objet (encapsulation, héritage, classe, objet). Il permet une programmation relativement simple sans se préoccuper de notions complexes présentes en C++ (indirection de pointeurs, fonctions virtuelles,...). Mais ces intérêts sont relativement mineurs comparés à l'apport que représente les interfaces de programmation standardisés proposées par Sun. Ces interfaces, couplées à la compilation en un ByteCode exécutable sur une machine virtuelle, donnent aux applications un niveau de portabilité maximale. Dans notre cas nous sommes intéressés par le développement d'une application Intranet utilisant des accès à une base de données. Il existe en JAVA un grand nombre d'API destinées à réaliser ce type de programmation et regroupées sous le terme J2EE (Java 2 Enterprise Edition). Ces API forment ce qu'on nomme un Framework. Il s'agit de proposer des interfaces guidant le développeur vers une architecture prédéterminée. On profite donc à la fois d'un ensemble d'outils indispensables au développement mais aussi d'un guide pour l'élaboration d'une méthode de développement. Ce guide reste suffisamment flexible pour nous permettre d'adapter la structure de notre application à nos besoins. L'objectif majeur de J2EE est la réalisation d'applications en architecture distribuée. Les technologies JAVA intégrées dans cette plate-forme sont:

- Enterprise JavaBeans (EJB)
- Common Object Request Broker Architecture (CORBA[3.12])
- Java Servlets 2.1
- Java Server Pages 1.1 (JSP)
- Java Message Service (JMS)
- Java Transaction API (JTA)
- JavaMail 1.1
- Java Database Connectivity 2.0 (JDBC[3.10])
- Java Naming and Directory Interface 1.2 (JNDI)
- eXtensible Markup Language (XML)

3. l'approche composant :

Les composants sont des éléments indépendants et réutilisables qui encapsulent un comportement applicatif ou des données en un paquetage distincte. Tels des dispositifs de type <boites noire>, ils effectuent des opérations sans révéler leurs mécanismes internes. Comme une interface abstraite fait le lien entre le comportement applicatif et l'implémentation, il est ainsi épargné aux utilisateurs les détails complexes d'un code touffu. Les fonctionnalités apportées n'augmentent pas la complexité globale de l'application. En outre, les composants ne sont pas exclusifs à une application ni a une utilisation unique .ils utilisées comme des briques élémentaires dans des projets qui parfois,n'ont aucun rapport entre eux.
L'abstraction et la réutilisation sont les deux principes essentiels de la conception orientée composant. La figure 3-1 illustre cette approche en montrant comment on peut combiner des composants logiciels indépendants. [DKK01]

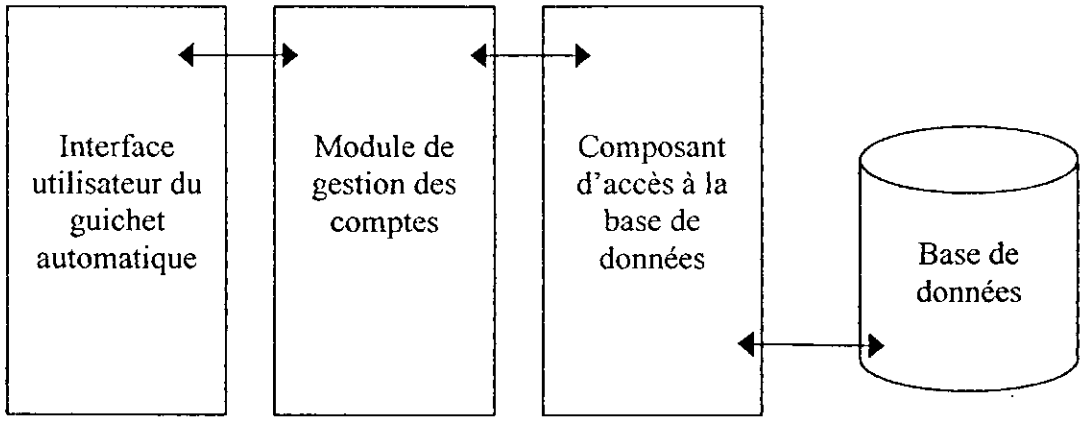


Figure 3-1 une application basée sur les composants

Les composants sont donc des éléments logiciels réutilisables qui peuvent être reliés pour construire une application. Un bon modèle de composants minimise le travail de codage des relations entre les différents éléments de l'application. Le principe des architectures de composants consiste à utiliser une interface commune pour manipuler des objets avec des outils de développement, la seule nécessité qui incombe à l'outille étant de prendre encharge l'interface.

3.1. Avantages des architectures à base de composants

Lorsqu'un architecte entreprend la conception d'une nouvelle maison, il se base sur des composants qui vent lui permettre tout à la fois de gagner du temps et de réduire le coûts engager, plutôt que de concevoir l'installation électrique à zéro, il va par exemple utiliser des composants déjà existants. Ainsi cet architecte ne va-t-il pas concevoir des systèmes de climatisations personnalisés, mais plutôt va-t-il les sélectionner à partir de modèles déjà disponibles sur le marché. Selon toute probabilité, cet architecte ne possédera pas le savoir-faire ni les ressource nécessaires qui lui permettraient de concevoir de tels systèmes. Inversement, le constructeur d'appareils de climatisation n'a aucun savoir-faire en construction de bâtiments. L'approche orienté composants permet à l'architecte et à l'entrepreneur de concentrer leurs efforts sur ce que, chacun de leur coté, savent le mieux faire. Il en va de même en informatique. Les architectures par composants offrent la possibilité de masquer la complexité des composants donnés par l'adjonction d'une interface, à travers laquelle le composant réagit avec son environnement ou avec d'autres composants.

On peut utiliser la même argumentation pour illustrer le concept de réutilisation et d'interchangeabilité. L'entreprise de construction a opté pour un climatiseur fixe par des écrous standard et fonctionnent sur une tension électrique standard. Le propriétaire de la maison pourra ainsi cet appareil par un nouveau modèle plus performant, et sans qu'il doive reconstruire toute la maison. La normalisation des environnements et des méthodes de conception a permis d'obtenir des systèmes souples, faciles à maintenir. Les composants logiciels ont été conçus pour opérer dans ces environnements spécifiques. L'existence de règles prédéfinies qui

régissent leur interaction avec les autres composants augmente d'autant leur interchangeabilité. [DKK01]

3.2. Fonctionnement des composants JavaBeans

Le composants JavaBeans sont des composants logiciels écrits en java, conformes aux spécifications exposées dans l'API des java beans, cette API qui a été crée par Sun en collaboration avec d'autres de l'industrie énonce les regèles que doivent suivre les développeurs de logiciels pour crées des composants logiciels indépendants et réutilisables, comme beaucoup de composants logiciels, les JavaBeans renferment un état et un comportement. Par l'utilisateur de balise de JSP conçues pour les JavaBeans dans leurs pages web, les développeurs de contenus peuvent tirer profit de la puissance de java en ajoutant des éléments dynamiques à leurs pages, sans pour autant écrire la moindre ligne de code java. Rappelons tout d'abord les principales caractéristiques des composants JavaBeans avant de détailler leur utilisation dans les JSP.

- Les conteneurs de JavaBeans :

Un conteneur JavaBeans est une application, un environnement ou un langage de programmation qui permet aux développeurs de faire appel à des composants, de les configurer et d'accéder a leurs données et a leurs méthodes, les applications qui se servent directement des composants JavaBeans sont exclusivement écrites en java mais les conteneurs permettent de les utilisés a des niveaux conceptuels plus élevés. Pour cela, les composants java beans exposent leurs fonctionnalités et leur comportement au conteneur peut ainsi les manipuler d'une façon plus intuitive. Le conteneur java beans définit ses propres critères de présentation et d'interaction avec le composant et écrit lui-même le code java qui en résulte.

Si vous avez déjà utilisé les outils visuels beans box de Sun, Visual age pour java d'IBM, visuel café de webgain ou d'autres outils de développement java, vous avez

déjà manipulé des composants. Ces ateliers de développements fonctionnent avec des conteneurs JavaBeans qui permettent de manipuler visuellement les composants. Ainsi peut-on par le simple déplacement d'icônes définir les caractéristiques des composants java, leur comportement, leur lien avec d'autres objets, et bâtir une application entière. L'application ainsi définie générer tout le code java nécessaire. Les conteneurs de JSP, d'une façon similaire, permettent aux développeurs créer des applications java basées sur le web sans récrire de code java. L'interaction avec les composants dans les JSP se fait par l'intermédiaire de balises qui peuvent se mêler a du code HTML classique. [DKK01]

- **Les propriétés de composants :**

Les conteneurs permettent de manipuler les composants en termes de propriétés (ce sont des attributs de composants JavaBeans identifiés par un nom, qui maintiennent son état de contrôlent son comportement). Un composant est défini par ces propriétés sans lesquelles il serait pratiquement inutilisable. Les propriétés d'un composant JavaBeans peuvent être modifiées pendant l'exécution par le conteneur pour contrôler les spécificités de son comportement. Les valeurs de propriétés constituent le seul mécanisme que le conteneur utilise pour mettre les JavaBeans à la disposition du développeur.
A titre d'exemple, supposons que l'on possède un JavaBeans watherBean qui détient des informations sur les conditions atmosphériques du moment et des prévisions météorologiques. Le JavaBeans peut récupérer ces informations en accèdent aux serveurs du National weather service ou les extraire d'une base de données, sachant que tout comme l'utilisateur de JavaBeans, il nous est d'aucune utilité de savoir comment ce composant obtient ses informations. Tout ce qui nous intéresse en tant que développeurs, c'est que weatherBean soit capable de nous fournir des informations telles que les températures actuelles, les maximales attendues ou les risques de précipitations. Chacun de ces éléments d'information est proposé au conteneur de JavaBeans sous forme d'une propriété du composant dont le »s valeurs sont accessibles par la page ou l'application web.

Chaque composant possédera un ensemble de propriétés en fonction des informations qu'il contient .on peut le personnaliser en initialisant soi-même les

valeurs des ses propriétés. Le créateur du JavaBeans va imposer des restrictions sur chacune de ses propriétés, ce qui lui permet de contrôler l'accès qui en fait. Une propriété peut être accessible en lecture seulement, en écriture seulement ou en mise à jour (lecture/écriture).c'est cette notion d'accessibilité qui permet au concepteur du JavaBeans d'imposer des restrictions sur la façon de l'utiliser. Ainsi, dans l'exemple weatherBean, cela n'aura aucun sens de permettre aux développeurs de modifier la valeur de la propriété du composant qui indique la température maximale de la journée. Cette information est gérée par le JavaBeans et ne devrait être accessible qu'en lecture. D'autre part, si le java beans est doté d'une propriété qui contient le code postal d'une région dont on connaître la météo, il serait normal de permettre aux développeurs de le spécifier. Une telle propriété serait accessible en, lecture/écriture.

- **Les propriétés liées et les propriétés de déclenchement :**

Certaines propriétés sont utilisées comme des déclencheurs, pour déclencher un comportement ou renvoyer des comptes rendus. La lecture de telles propriétés ou leur modification a pour conséquence immédiate la réalisation par un composant JavaBeans d'une certaine action sur le serveur : il peut s'agir mettre à jour les valeurs d'autres propriétés ou de lancer une autre tâche sur le serveur. La mise à jour de la valeur de la propriété de code postal par exemple pourrait conduire le JavaBeans à consulter les services du *National Weather Service* pour s'informer sur les conditions atmosphériques qui correspondent au nouveau code postal. Il mettra ensuite à jour ses autres propriétés relatives à la météo en conséquence. Dans ce cas, les propriétés qui ont trait à la météo et au code postal sont considérées comme des propriétés liées parce que la modification de la valeur de l'une d'entre elles met à jour les valeurs des autres.

- **Les propriétés indexées**

Il est également possible pour une propriété unique de mémoriser plusieurs valeurs. Ces propriétés sont dites *indexées* car chaque valeur enregistrée dans la propriété est accessible via un numéro d'indice qui pointe sur

la valeur souhaitée. On peut par exemple réclamer la première valeur de la liste, la troisième ou la vingtième. Ainsi, notre WeatherBean pourrait avoir une propriété qui prenne en charge les prévisions météorologiques pour les cinq jours à venir. Cependant, tous les conteneurs de JavaBeans ne fournissent pas un mécanisme simple pour manipuler directement ces propriétés multivaluées. Par exemple, les balises de Java-Beans des JSP ne reconnaissent pas les propriétés indexées. Il faut utiliser en lieu et place des scriptlets, des expressions JSP ou les balises JSP personnalisées (détaillées aux chapitres 13 et 14) pour pouvoir accéder à de telles propriétés.

- **Les types de données des propriétés :**

Les propriétés des JavaBeans peuvent être utilisées pour prendre en charge une quantité importante d'informations. Par exemple, les propriétés de WeatherBean peuvent mémoriser des informations très diverses comme les températures, les risques de précipitations, les prévisions météo, les codes postaux, etc. Chaque propriété d'un composant JavaBeans ne peut contenir qu'un type particulier de données. Ses valeurs ont un type Java, qui est utilisé en interne par le composant et dans le code Java généré par le conteneur des JavaBeans. Les propriétés peuvent bien entendu supporter n'importe quel type primitif de Java tel que int (entier simple précision) ou double (entier double précision), ainsi que des objets Java tels que String et Date. Elles peuvent également mémoriser des objets définis par les utilisateurs, voire même d'autres JavaBeans. Les propriétés indexées mémorisent généralement un tableau de saleurs qui ont le même type de données.

Le conteneur de JavaBeans détermine la façon dont il convient de manipuler les valeurs de Propriétés d'un composant. On va référencer les valeurs des propriétés par leur type de données en utilisant des scriptlets et des expressions Ainsi, si une propriété contient des valeurs de type «entier on ne peut obtenir et déposer que des Valeurs entières. Cependant, avec les balises Beans, on traite chaque propriété comme si elle ne contenait que du texte String). Lorsqu'on initialise la saleur d'une propriété d'un JavaBeans, on lui transmet du texte. De même, lorsqu'on effectue une

lecture du contenu d'une propriété, on va récupérer du texte indépendamment du type de données interne utilisé dans le JavaBeans. Cette stratégie orientée texte permet de manipuler d'une façon qui soit simple les balises Bean de JSP et se marie bien avec HTML.

Le conteneur de JSP effectue toutes les conversions de type nécessaires. Par exemple, lorsqu'on initialise une propriété de type entier, le conteneur de JSP fait les appels Java nécessaires pour 'convertir les suites de caractères numériques qu'on lui a fournies en une valeur entière. Bien entendu, ce processus de conversion nous oblige à transmettre les valeurs textuelles appropriées de façon que Java puisse les convertir correctement dans le type de données natif. Si une propriété prend en charge des valeurs à virgule flottante, par exemple. Une erreur est générée si on essaie de lui affecter des valeurs comme banane, pain, cent ou (3,9).

De. Concepteurs astucieux de JavaBeans peuvent ainsi contrôler les valeurs des propriétés en acceptant des valeurs de type chaîne de caractères pour des propriétés qui ne sont pas de ce type et en effectuant eux-mêmes les conversions. Cette technique doit être utilisée pour toute valeur qui n'est ni une chaîne de caractères ni un type primitif de Java. Par conséquent, il pourrait être parfaitement autorisé d'attribuer à une propriété de type entier la valeur cent si le concepteur du JavaBeans l'a préparée à une telle affectation.

4. Les différentes fonctions des composants JavaBeans :

Les JavaBeans peuvent être répartis selon trois grandes catégories fonctionnelles : les composants graphiques utilisés dans les interfaces utilisateur graphiques. Les composants de données qui permettent l'accès aux informations, et les composants de services (appelés aussi *worker beans)* qui peuvent effectuer des tâches ou des calculs spécifiques. Bien entendu. Un JavaBeans *peut* appartenir *à* plusieurs de *ces catégories à la fois.*

- **Les composants graphiques**

En raison du développement des composants graphiques. Cette utilisation des composants JavaBeans est une des plus répandues. Ce sont des éléments tels que les champs de textes, les listes de sélection ou tout objet graphique utile pour la conception d'interface utilisateur. En regroupant des composants graphiques dans des JavaBeans, les environnements de développement Jasa peuvent profiter des avantages offerts par le support de la programmation graphique de ces composants. Les développeurs peuvent créer leurs interfaces simplement en assemblant les éléments souhaités. Toutefois, comme les composants graphiques ont été conçus pour s'exécuter avec les applications Java graphiques, ils ne sont pas compatibles avec les JSP qui sont plus orientées vers la conception d'interfaces HTML.

- **Les composants de données**

On peut avec ces JavaBeans accéder aux données que le composant lui-même n'a pas forcément la capacité de collecter ou de générer. Le calcul et la collecte des données mémorisées dans les JavaBeans de données sont de la responsabilité d'un autre composant ou service plus contple.se. Ces JavaBeans ne sont accessibles qu'en lecture. Ils permettent donc de récupérer des données mais pas de les modifier.

Certains JavaBeans de données permettent d'initialiser des propriétés afin de contrôler le format et le filtre des données avant de les retourner via d'autres propriétés. Par exemple, un **AccountStatusBean** pourrait être doté d'une propriété currentType qui contrôlerait si la propriété associée au solde du compte retourne les données en dollars, en livres ou en francs suisses. Les JavaBeans de données, en raison de leur simplicité, sont utiles pour normaliser l'accès aux informations en fournissant une interface stable.

- **Les composants de services**

Comme leur nom l'indique, ces JavaBeans permettent d'accéder à un comportement ou un service particulier. C'est la raison pour laquelle ils sont quelquefois appelés les JavaBeans travailleurs. Ils peuvent extraire des informations des hases de données, effectuer des calculs ou formater des informations. Comme il n'y a pas d'autre possibilité pour interagir avec un JavaBeans que de le faire par ses propriétés, on sa accéder à ses services de la même façon. Dans une conception classique, on va initialiser les valeurs de certaines propriétés qui contrôlent

5. Architecture des applications web :

Quelle que soit la complexité d'une application web, il est toujours conseillé d'analyser son architecture selon trois logiques (figure 3-2) :

- ✓ La couche de présentation, qui présente des résultats à l'utilisateur et en détermine l'apparence ;
- ✓ La couche de contrôle, qui contrôle le flot d'exécution de l'application ; et
- ✓ La couche de logique applicative « modèle», qui gère les données de l'application, exécute les traitements et communique directement avec les ressources sous-jacentes.

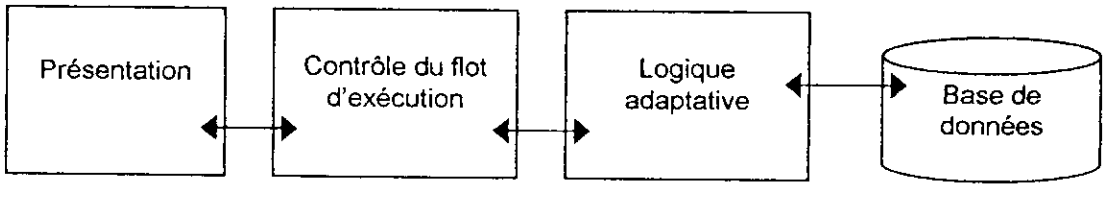


Figure 3-2 architecture des applications web par niveaux logique

5.1 Architecture orientée servlets :

C'est une approche qui consiste en effet à ne confier aux page JSP que les aspects de présentation, en délèguent les aspects « back-end» de flot d'exécution et de logique applicative à une ou plusieurs servlets. Les requêtes sont alors directement dirigées aux pages de présentation JSP via une servlet, qui effectue toutes les action nécessaire au fonctionnement de l'application. Une servlet peut jouer l'un ou plusieurs des trois rôles suivants :

- Exécuter des actions pour le compte d'une page JSP, par exemple soumettre une commande ;
- Livrer à une page JSP des données pour quelles soient affichées, par exemple un enregistrement de base de données ;
-Contrôler l'enchaînement des pages JSP d'une même application.

Après avoir effectué son traitement, une servlet transfère la requête vers la page JSP appropriée, voire vers une page HTML statique (voire figure 3-3)

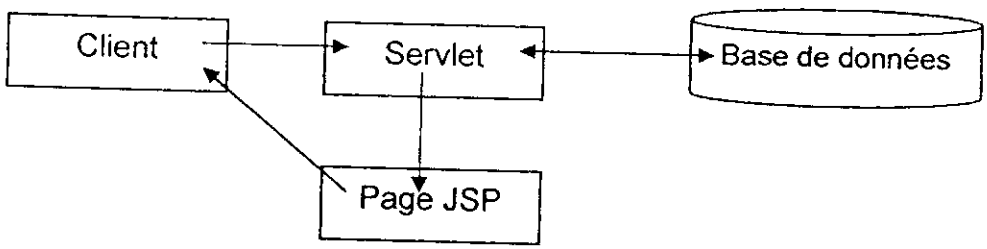


Figure 3-3 Enchaînement d'exécution d'une application basée sur des servlet

Ceux qui connaissent le pattern de médiateur (gestion de comportement) s'apercevront qu'il s'agit d'appliquer le même principe, mais à des pages et composants JSP plutôt qu'a des objet java. Le principe de ce pattern est de créer un composants de l'application interagissent, entre eux et avec les sources de données, cette approche permet de découpler la relation qui existait entre les pages jsp :on peut les coder indépendamment sans forcément inclure les détails de fonctionnement des autres pages. En outre, la séparation entre présentation et logique applicative est encor plus nette. Plus concrètement, il s'agit de minimiser la

quantité de traitements effectués par les pages elles-mêmes, pour s'appuyer sur des servlets dédiées. Le code JSP frontal est moins complexe car il se limite désormais au simple affichage de données ou à la collecte de données utilisateur.
De même, les servlets n'ont à contenir aucun détail de présentation des informations ; elle gère que le contrôle de l'exécution et la production des données à passer aux pages jsp à des fins de présentation à l'utilisateur. [DKK01]

6. Conclusion :

Un composant JavaBeans est une classe java qui obéit à des conventions de nommage et de conception simples qui précisées dans les spécifications JavaBeans Ces composants ne doivent pas nécessairement étendre une classe de base spécifique ni implémenter une interface particulière. Une classe qui applique les conventions JavaBeans et qui est manipulée en respectant ces conventions est un composant JavaBeans.

Chapitre 4

Analyse et conception du système

I. INTRODUCTION :

La modélisation et la spécification orienté objet est une méthode pour penser et écrire des problèmes organiser autour de concepts du monde réel.

- ✓ Comprendre des problèmes.
- ✓ Communiquer avec des experts
- ✓ Modéliser avec des organisations
- ✓ Préparer la documentation
- ✓ Concevoir des programmes et des bases de données.

Le processus comporte les étapes suivantes :

- on réalise un modèle analytique des aspects essentiels de l'application, sans aucun priori d'implémentation. Ce modèle comporte les objets essentiels du domaine applicatif avec leur relation.
- ensuite en prend des décision de conception et on affine le modèle initial pour préciser et optimiser l'implémentation.
- Les objets applicatifs ainsi écrits forment le cadre de la programmation à venir.
- Enfin, on traduit le modèle dans un langage de programmation quelconque

1.1. Qu'est ce que l'orienter objet ?

On déclare orienté objet un logiciel organiser autour d'entités groupant des données arbitraire (comme les struct c, et les record pascal) et des procédures.

- on appelle encapsulation le processus d'agrégation des ces données et les processus d'autres caractéristiques sont habituellement requises des objets au sens propre.
  - l'identité
  - la classification
  - le polymorphisme
  - l'héritage

1.1.1 l'identité

Les objets sont des entités qui peuvent être distinguées. Notamment, deux objets distincts peuvent avoir exactement les mêmes données membres.

1.1.2 la classification

Les objets ayant des attributs (données et comportements) similaire sont groupés dans une classe. La classe décrit tous les caractéristique des objets de sont type, une classe qui abstrait un objet du mondes réel intègre les données qui semblent pertinentes en fonction du degré de détail voulu.

1.1.3 le polymorphisme

La même fonction peut avoir des effets différents sur des objets de classes distinctes.

1.1.4 l'héritage

Des caractéristiques partagées par différentes classes peuvent être décrites dans une classe intermédiaire dont les autres héritent, ce qui permet de les décrire par les seules modifications
Qu'elle apporte au schéma de base .l'héritage apporte une grande puissance d'expression et d'abstraction au modèle objet.

II. La methode de concetion OMT etl e langage de modelisation UML:

1. Object Modeling Technique (OMT)

C'est une méthode d'analyse orientée objet pour le développement des système d'information. OMT apporte deux choses [CEE00] :

- une méthodologie pour le développement orienté objet.
- une notation graphique pour la communication de concepts orientés objet. Dans cette partie nous utiliserons le langage de modélisation UML.

1.1. Méthodologie orientée objet

La méthodologie consiste en la réalisation d'un modèle du domaine applicatif, complété par des détails d'implémentation au cours de la phase de spécification et conception du système le processus comporte les étapes suivantes : analyse, modélisation du système, modélisation des objets, programmation.

1.1.1. Analyse

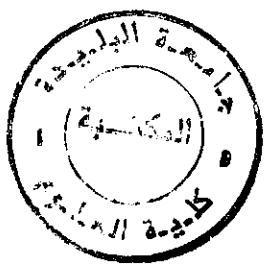
- à partir d'un énoncé du besoin, l'analyste construit (avec le demandeur) un modèle du monde réel qui fasse apparaître ses propriétés pertinentes.
- le modèle issu de l'analyse est une description précise, concise, de ce que le client demande, non de comment on peut le faire.
- les objets du modèle sont des objets du domaine, jamais des objets issus de l'informatique.
- une bonne analyse peut être comprise et critiquée par des experts du domaine qui ne sont pas des informaticiens.
- une bonne analyse ne comporte aucune décision d'implantation [LAU96].

1.1.2. Modélisation du système

- l'ingénieur (system designer en anglais) prend des décisions de haut niveau sur l'architecture du système.
- pendant cette phase, le système cible est organisé en sous systèmes pertinents relativement au modèle initial et aux choix d'architecture.
- l'ingénieur décide des caractéristiques à optimiser, définit une stratégie pour aborder le problème, et fait une première évaluation des besoins en ressources humaines [LAU96].

1.1.3. Modélisation des objets

- l'ingénieur décrit un modèle détaillé basé sur l'analyse et le modèle du système, qui respecte les choix stratégiques.
- l'accent est ici porté sur les structures de données et les algorithmes nécessaires pour réaliser chaque classe.
- les classes initiales restent pertinentes, mais elles sont complétées par les classes d'implémentation, choisies en fonction de besoins concrets en performance exprimés pour le système final.
- les objets initiaux et les objets informatiques sont décrits dans le même langage, bien qu'ils figurent dans des plans conceptuels distincts [LAU96].



1.1.4. Programmation

- dans cette phase bien connue, les classes et relations décrites précédemment sont traduites dans un langage de programmation.
- la programmation devrait être une part mineure de l'ensemble du processus, les décisions importantes ayant déjà été prises.

- le langage cible influence toujours la conception dans une certaine mesure, mais celle ci ne devrait jamais dépendre de détails trop pointus d'un langage donné.

- la programmation doit obéir à des règles de contrôle qualité suffisantes pour :
  - garantir la traçabilité des programmes par rapport à la spécification (vérification).
  - aboutir à un résultat suffisamment réutilisable, extensible, portable ... [LAU96].

2. Unfied modeling language (UML)

2.1 Introduction a l'UML:

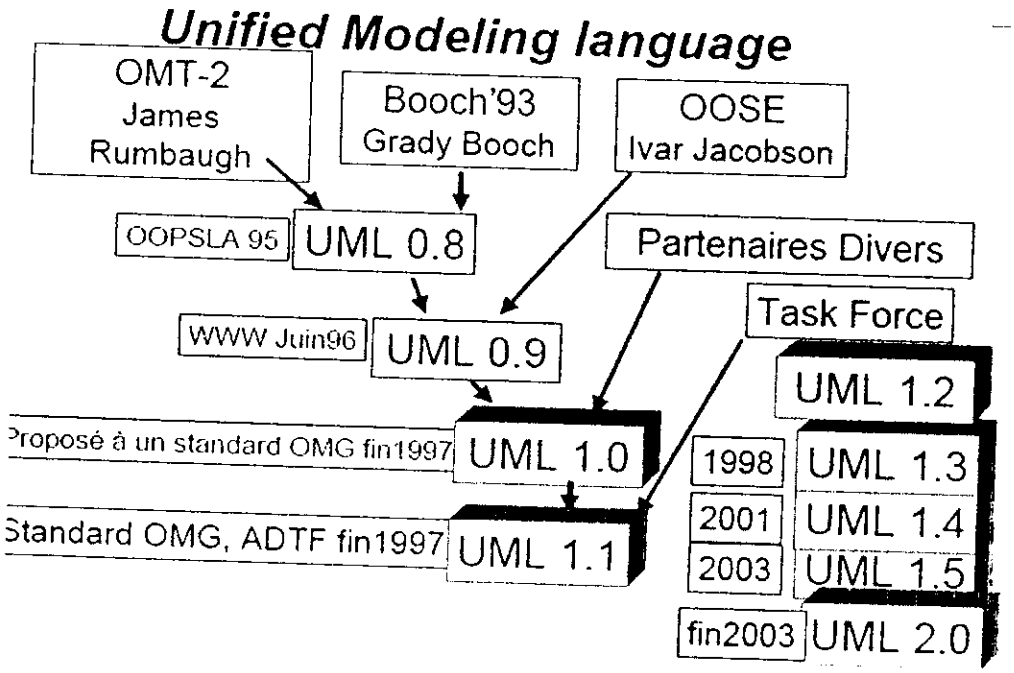


Figure 4-1 origines de uml [ROO]

Pourquoi modéliser

. Un modèle est une simplification de la réalité qui permet de mieux comprendre le système à développer.

. Il permet

- De visualiser le système comme il est ou comme il devrait l'être.
- De valider le modèle vis à vis des clients
- De spécifier les structures de données et le comportement du système.
- De fournir un guide pour la construction du système.
- De documenter le système et les décisions prises.

Qu'apporte la modélisation objet

. Plus grande indépendance du modèle par rapport aux fonctionnalités demandées.

. Des fonctionnalités peuvent être rajoutées ou modifiées, le modèle objet ne change pas.

. Plus proche du monde réel.

Les objectifs d'UML

• Représenter des systèmes entiers

• Etablir un couplage explicite entre les concepts et les artefacts exécutables

• Prendre en compte les facteurs d'échelle

• Créer un langage de modélisation utilisable à la fois par les humains et les machines

Recherche d'un langage commun unique

– Utilisable par toutes les méthodes

– Adapté à toutes les phases du développement

– Compatible avec toutes les techniques de réalisation

UML un langage

. UML n'est pas une méthode

. UML est un langage de modélisation objet

. UML a été adopté par toutes les méthodes objet

. UML est dans le domaine public, c'est une norme

UML un langage pour

. Visualiser

. Chaque symbole graphique a une sémantique

. Spécifier De manière précise et complète, sans ambiguïté,

. Construire Les classes, les relations SQL peuvent être générées automatiquement

. Documenter Les différents diagrammes, notes, contraintes,

Exigences seront présentés dans un document.

UML et les domaines d'utilisation

. Systèmes d'information des entreprises

. Les Banques et les services financiers

. Télécommunications

. Transport

. Défense et aérospatiale

. Scientifique

. Applications distribuées par le WEB

Les 9 diagrammes en UML

UML est un langage graphique et repose sur neuf types de diagrammes. Chacun de ces diagrammes utilise le même principe : les concepts sont représentés par des symboles, et les relations entre les concepts sont représentées par des lignes qui relient les symboles. Le vocabulaire et la grammaire d'UML sont très réduits.

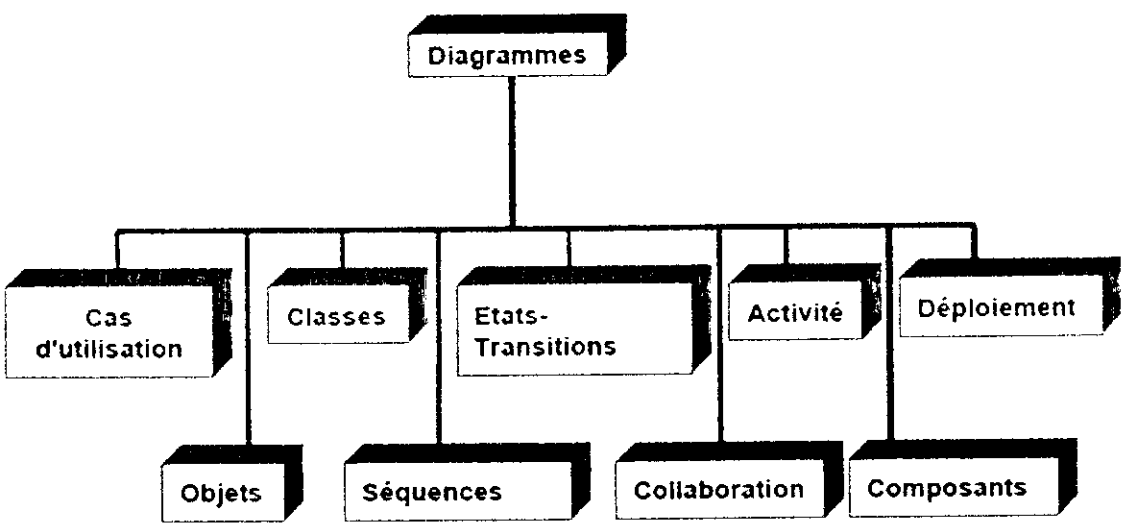
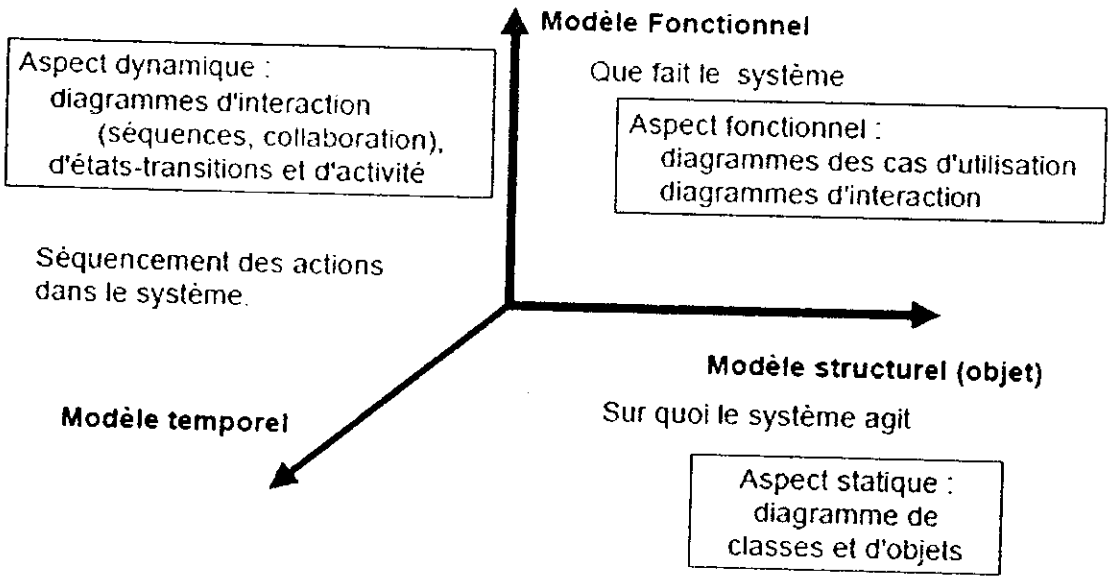


Figure 4-2 les 9 diagrammes de l'uml [ROO]

Les trois composantes d'une modélisation



Phases de la modélisation

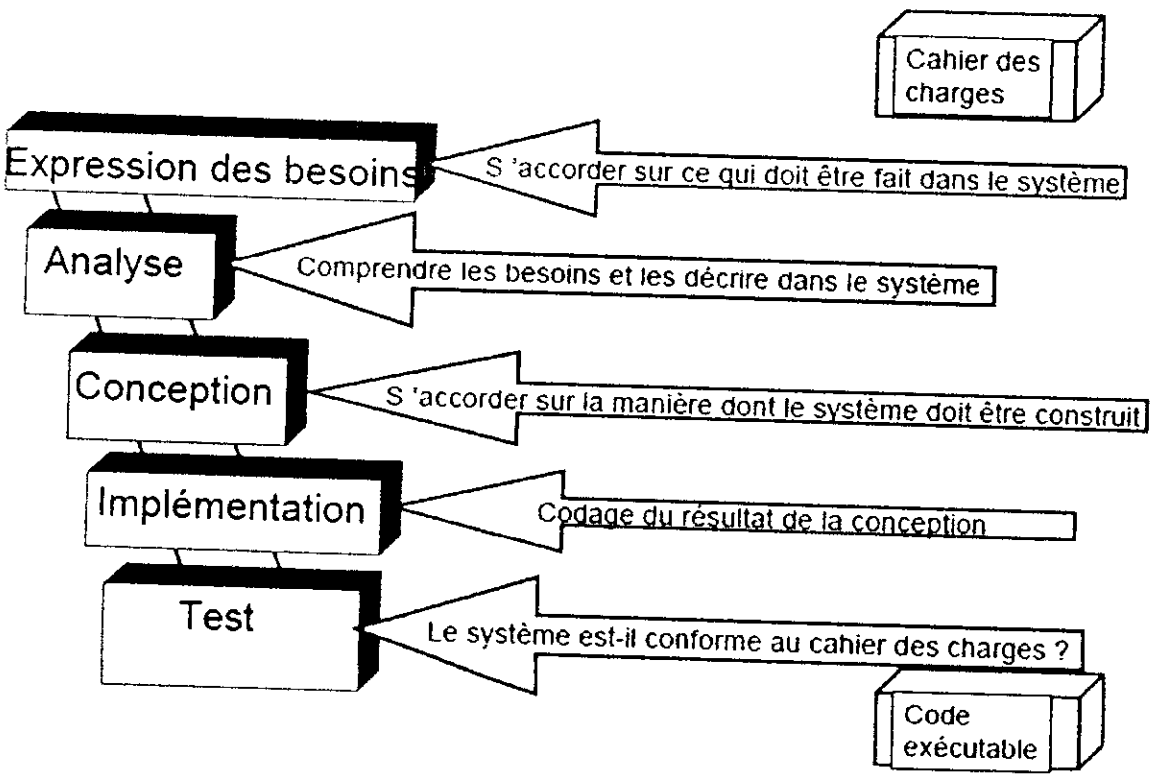


Figure 4-3 Phases de la modélisation [ROO]

2.2 Les diagramme de l'UML

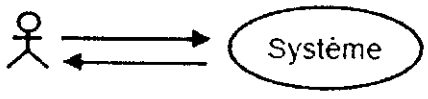
2.2.1 Le diagramme des Use Cases ou des cas d'utilisation

-> Ce que doit faire le système

Sans spécifier comment il le fait

But des Use Cases

Les cas d'utilisation représentent les Fonctionnalités que le système doit savoir faire. Chaque cas d'utilisation décrit un ensemble d'interactions successives d'une entité en dehors du système (utilisateur) avec le système lui même pour réaliser une fonctionnalité.



Les Uses Cases permettent :

• De connaître le comportement du système sans spécifier comment ce comportement sera réalisé.

• De définir les limites précises du système

• Au développeur de bien comprendre l'attente des utilisateurs et les experts du domaine.

De plus les Use Cases sont :

• Des instruments de validation et de test du système en cours et en fin de construction.

Modèle des cas d'utilisations

. Un diagramme de cas d'utilisation définit :

. le système

. les acteurs

. les cas d'utilisations

. les liens entre acteurs et cas d'utilisations

. Un modèle de cas d'utilisation se définit par

. des diagrammes de cas d'utilisation

. une description textuelle des scénarios d'utilisation

. une description de ces scénarios par

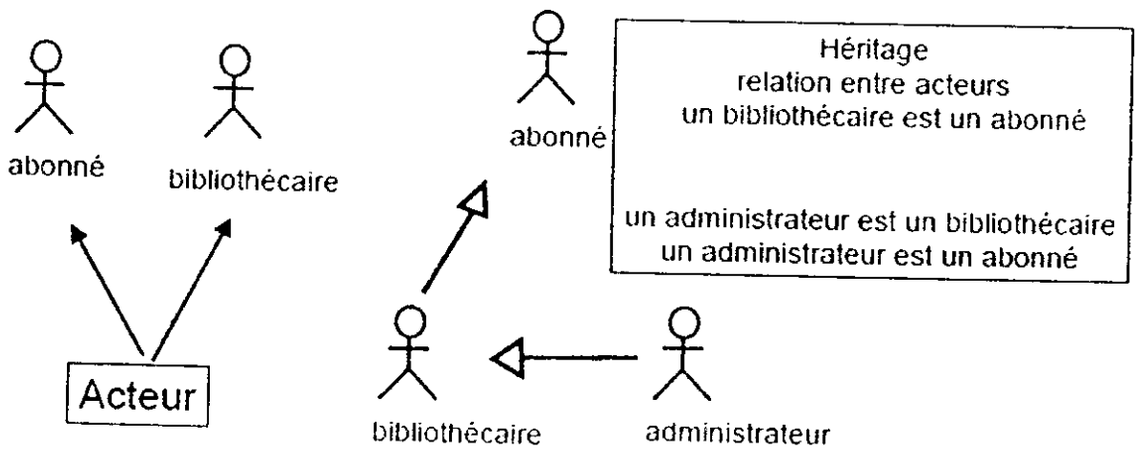
. les diagrammes de séquences

. les diagrammes de collaboration

Les Acteurs

. Un acteur représente une personne ou un périphérique qui joue un rôle (interagit) avec le système.

. Relation entre acteurs : généralisation (héritage)



Les cas d'utilisation (use-case)

. Un cas d'utilisation est un moyen de représenter les différentes possibilités d'utiliser un système.

. Il exprime toujours une suite d'interactions entre un acteur et l'application.

. Il définit une fonctionnalité utilisable par un acteur.

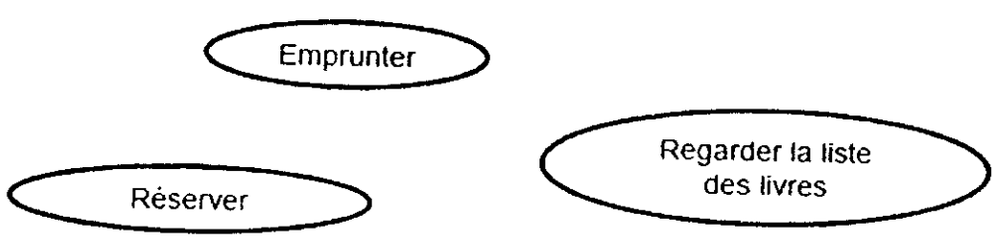
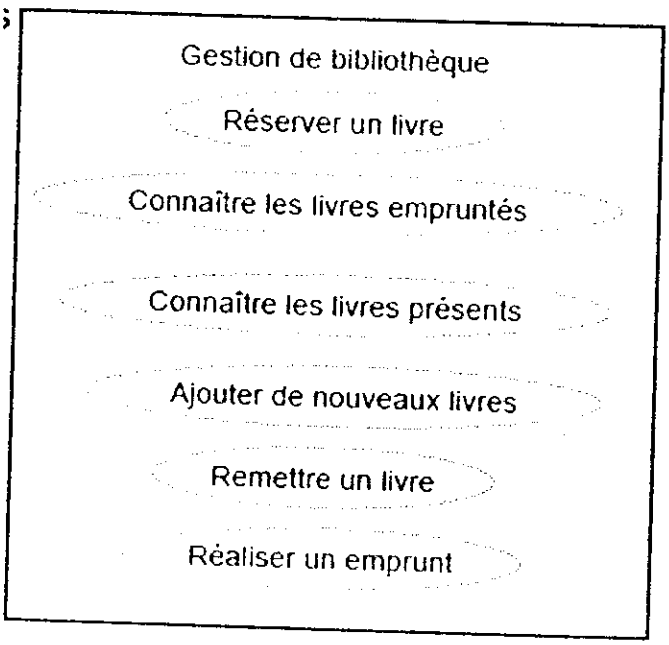


Figure 4-4 les cas d'utilisation

Le système

. Le système définit l'application informatique, il ne contient donc pas les acteurs, mais les cas d'utilisation et leurs associations



Exemple

Les diagrammes de cas d'utilisation

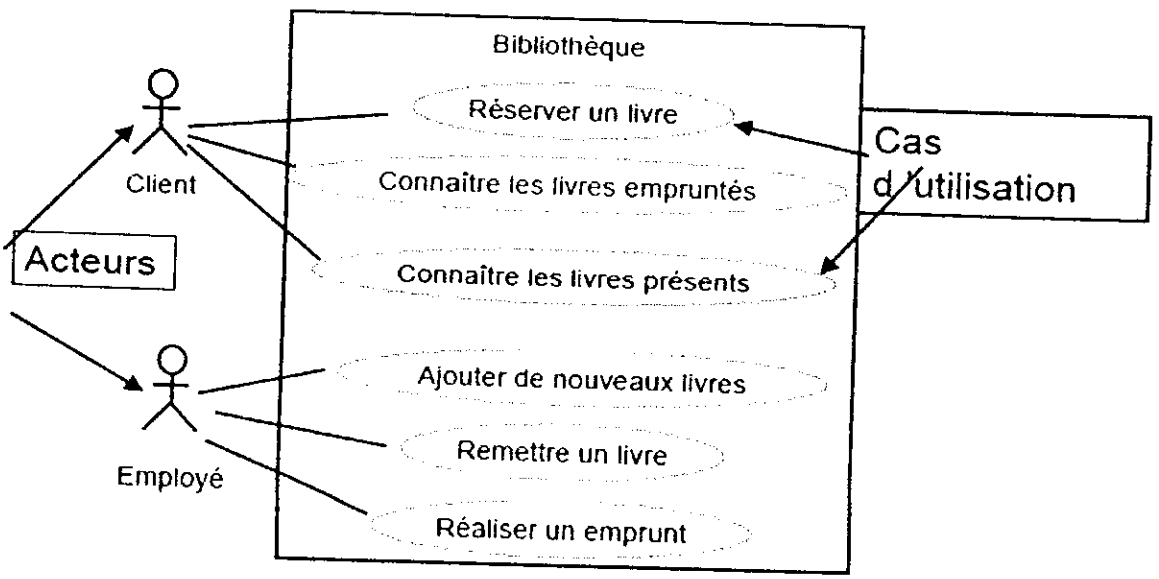


Figure 4-5 diagrammes de cas d'utilisation

Scénarios d'un cas d'utilisation

. La description d'un cas d'utilisation se fait par des scénarios qui définissent la suite logique des interactions qui constituent ce cas.

. On peut définir des scénarios simples ou des scénarios plus détaillés faisant intervenir les variantes, les cas d'erreurs, etc.

. Cette description se fait de manière simple, par un texte compréhensible par les personnes du domaine de l'application.

. Elle précise ce que fait l'acteur et ce que fait le système

. La description détaillée pourra préciser les contraintes de l'acteur et celles du système.

Exemple Scénarios d'un cas d'utilisation

description simplifiée

Le client se présente devant un terminal:

- (1) Le système affiche un message d'accueil.
- (2) Le client choisit l'opération réservation parmi les différentes opérations proposées.
- (3) Le système lui demande de s'authentifier.
- (4) Le client donne son identification (nom, mot de passe).
- (5) Le système lui demande de choisir un livre.
- (6) Le client précise le livre qu'il désire.
- (7) Le système lui précise si un exemplaire du livre lui est réservé.

2.2.2 Les diagrammes de séquence

Ces diagrammes représentent également la dynamique de fonctionnement du système également. En revanche, la représentation temporelle des événements est mise en avant plutôt que la représentation spatiale. [FAP00]

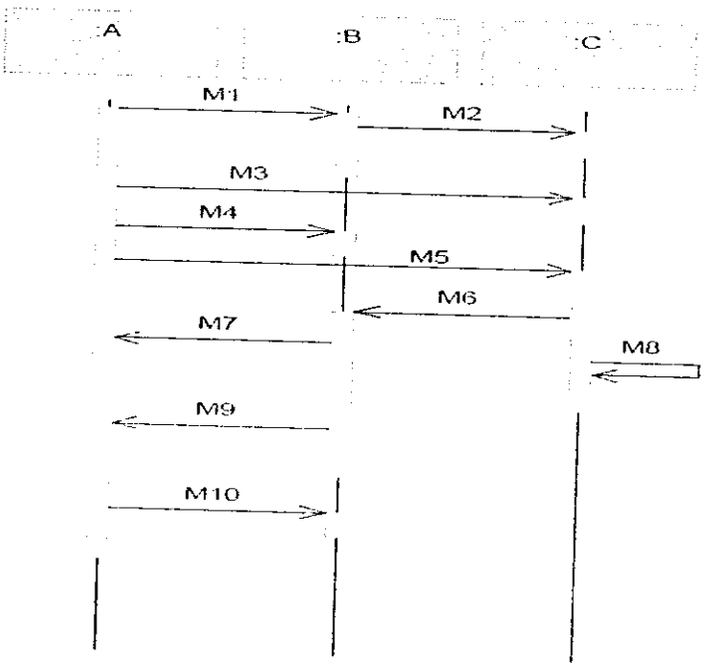


Figure 4-6 diagramme de séquence

Scénario par diagramme de séquences

. Suite aux descriptions textuelles, le scénario peut être représenté en utilisant un diagramme de séquences.

Le diagramme de séquences permet:

.de visualiser l'aspect temporel des interactions

.de connaître le sens des interactions (acteur vers système ou contraire)

Exemple

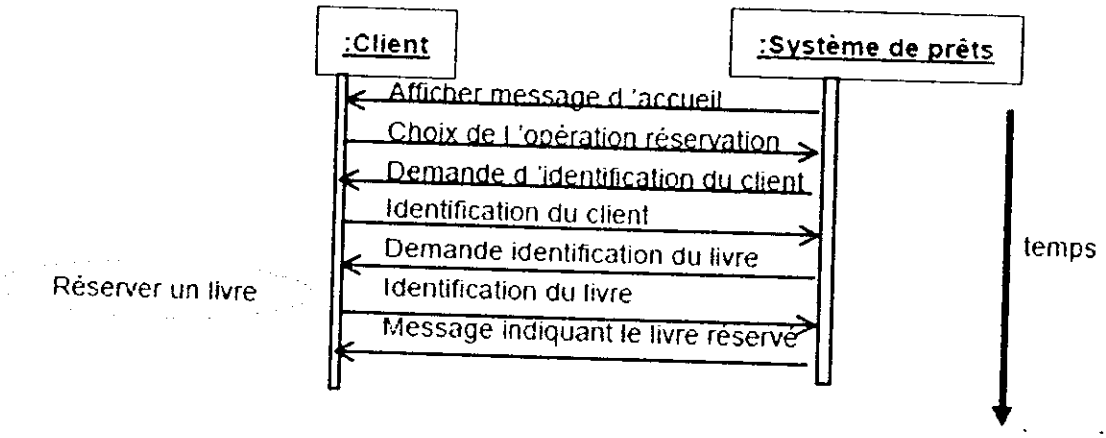


Figure 4-7 exemples de diagramme de séquence [ROO]

2.2.3 Les diagrammes de collaboration

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait

figurer les envois de messages, annotés par leur ordre d'apparition. [FAP00]

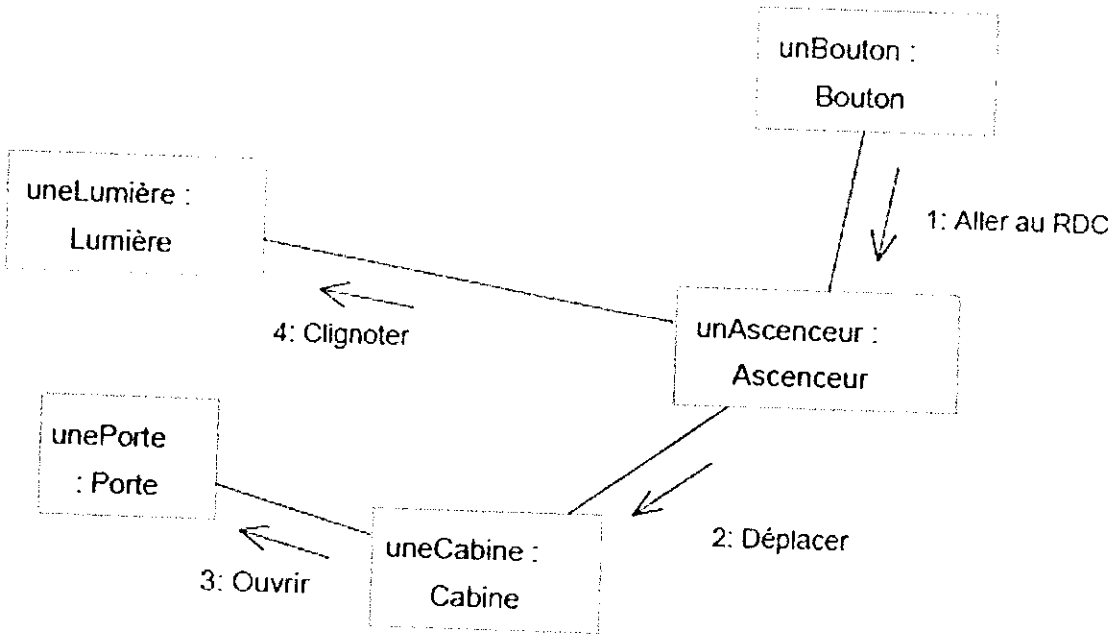


Figure 4-8 diagramme de collaboration

2.2.4 Les diagrammes de classes

Ces diagrammes décrivent l'architecture du système; on y représente les classes et les relations entre classes, qu'elles soient d'héritage, d'agrégation ...

La classe

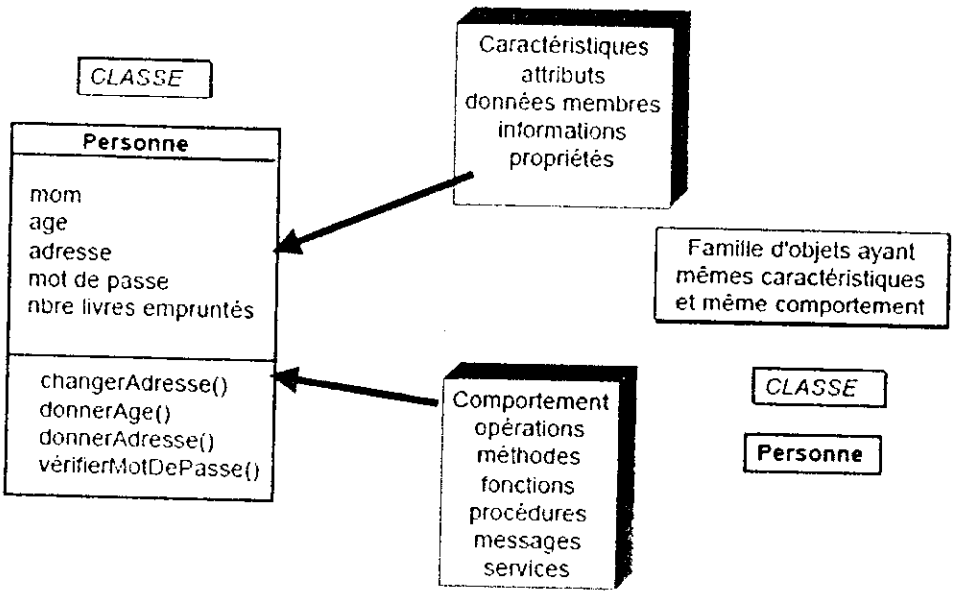
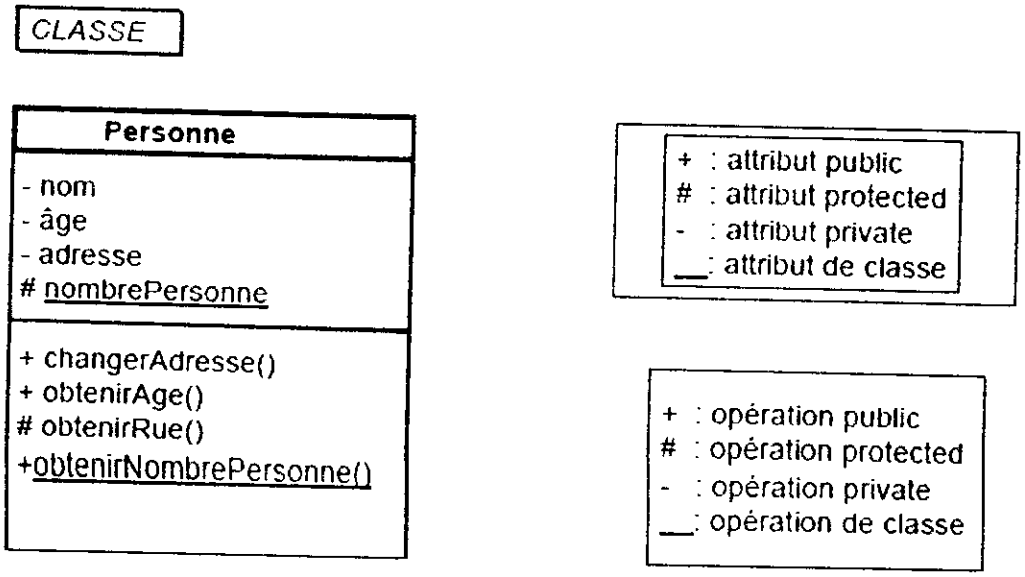


Figure 4-9 les classes

Protection des attributs et des Opérations



Exemple

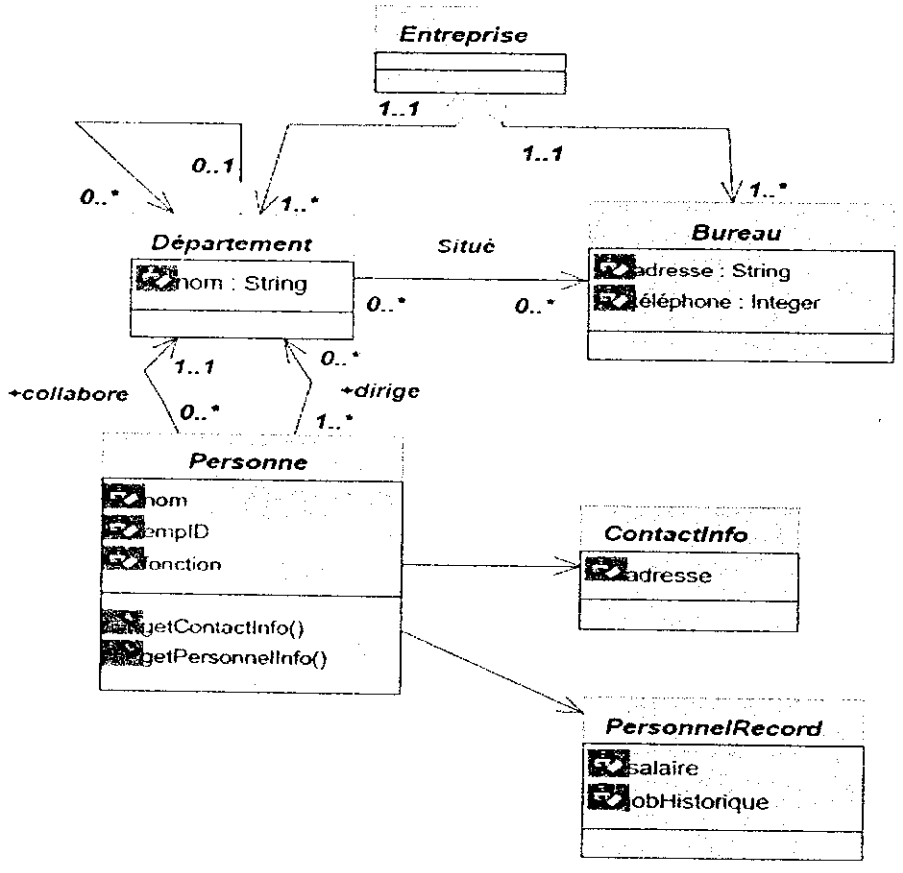


Figure 4-10 diagramme de classes [FAP00]

Résumé

On a présenté une méthode de conception OMT et un langage de modélisation d'un system d'information par objets. Le langage de modélisation UML respecte les trois étapes de l'étude : l'étape statique, dynamique, fonctionnelle. Statique pour aboutir a un modèle objet, dynamique pour décrire la réalisation des objets du system aux événements et les itération entre les objets eux-mêmes, et enfin, l'étape fonctionnelle qui comprend des contraintes entre les valeurs, les information de contrôles et de structure de d'objet appartenant respectivement au modèle dynamique et au modèle objet.
Nous allons utiliser cette méthode pour la conception de notre system

3. Analyse et conception :

Pour concevoir notre système, nous avons choisis la méthode de conception « OMT » qui sera appliquée en adoptant le langage de modélisation objet UML.
La technique de modélisation par objet (uml) associe trois modèles liés mais distinct : le modèle objet, le modèle dynamique et le modèle fonctionnel.
Le modèle objet décrit la structure des données sur lesquelles le modèle dynamique et le modèle fonctionnel opèrent.
Le modèle dynamique décrit la structure de contrôle des objets, met en évidence les décisions qui dépendent de la valeur des objets et invoque des fonctions.
Le modèle fonctionnel décrit les fonctions invoquées par les opérations du modèle objet et du modèle dynamique.

3.1. Analyse

3.1.1. ANALYSE DES BESOINS :

La base de données doit être accessible via les réseaux intranet et Internet, donc elle assure un accès non limité dans le temps et le lieu avec une gestion de ressources.
Le système doit assurer aux agents des services tel que : consultation, recherche..., et ceci selon les privilèges qu'on leur donne lors de leurs inscription.

- Le système a le contrôle des ressources : ajout, modification, suppression de données.
- Contrôle des abonnés : inscription d'un utilisateur, lister les utilisateurs, modifier les coordonnées d'un utilisateur,...
- Attribution des privilèges aux abonnées : simple consultation, consultation et recherche....

Les besoins selon l'utilisateur du système seront :

Agent :

- Consulter un son répertoire,
- Rechercher des informations selon des critères bien spécifiques.
- Contacter un agent service,

Administrateur:

- Lister les utilisateurs,
- Supprimer un privilège à un utilisateur,
- Affecter un privilège a un utilisateur.

Agents du service :

- Consulter la base de données
- Contacter un utilisateur,
- Faire des Recherches.
- Modifier les informations propres à un utilisateur.

Un administrateur est considéré aussi comme utilisateur avec privilèges donc il a accès à l'ensemble des fonctionnalistes du système

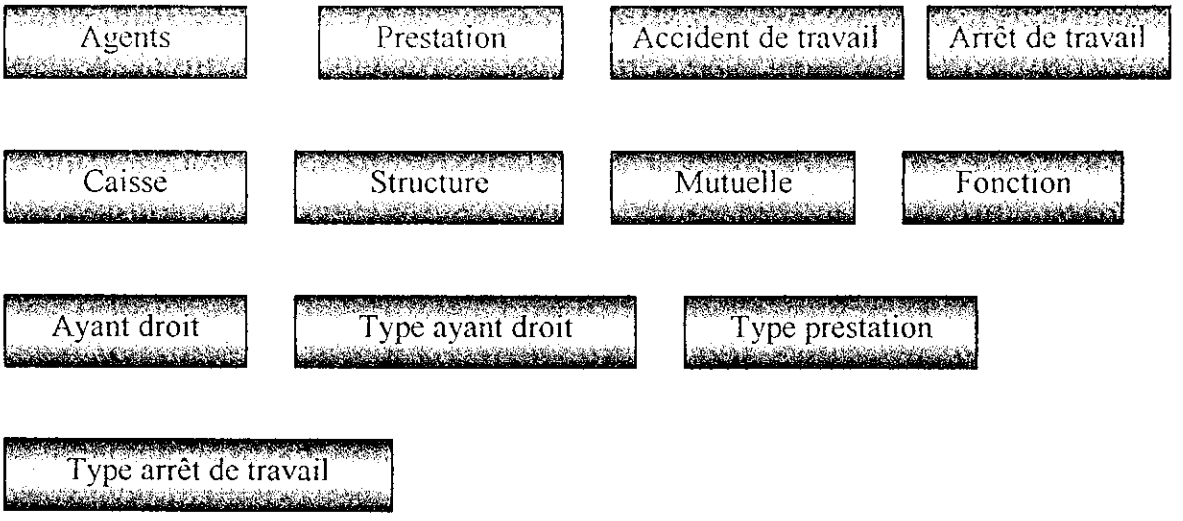
La phase d'analyse, première phase de la modélisation OMT, a pour objectif de décrie de manière précise, concise, correcte et compréhensible un modèle du monde réel.
L'analyse ne se préoccupe pas des solutions mais des questions, elle identifie le quoi faire et l'environnement d'un système sans décrire le comment qui est le propre de la conception.

3.1.2. Modèle objet

Un modèle objet saisit la structure statique d'un système. On montre les objets, les relations entre les objets et les attributs qui caractérisent chaque classe. Les étapes nécessaires à la construction, de ce modèle sont :

- identifications des classes.
- Préparation du dictionnaire de données.
- Identification des attributs de chaque classe.
- Identification des associations.

3.1.2.1 Identification des classes



3.1.2.2. Dictionnaire de données

- **Identification des classes**

Agents : ensemble du personnels de la direction générale de SONATRACH

Prestation : ensemble des prestations déposés par les agents

Accident de travail : les accidents de travail enregistrer

Arrêt de travail : les arrêt de travail qui on été déposer au service

Caisse : c'est les caisses où sont affiliés les agents de la direction générale SONATRACH

Structure : ensemble de structure de la direction ou appartient les agents

Mutuelle : la mutuelle où sont affiliés les agents de la direction

Fonction : les fonction qu'exerce l'agent

Ayant droit : les personnes qui on le droit a la prestation.

Type ayant droit : type d'ayant droit a la prestation.

Type de prestation : type de prestations sociales.

Type arrêt de travail : types d'arrêts de travail répertorie.

- **Identification des attributs :**

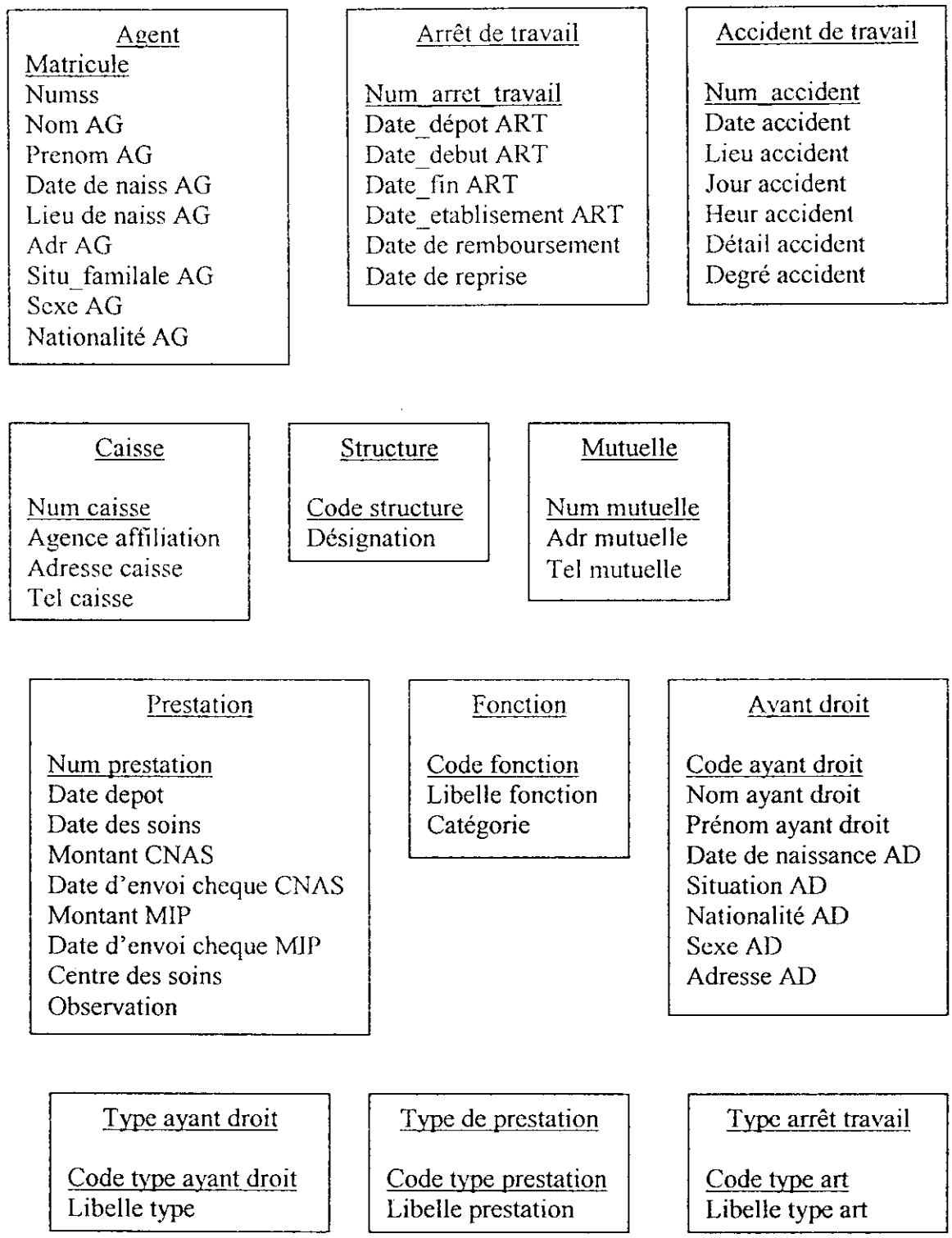


Figure 4.11 Identification des attributs

- **Identification des associations :**

Tableau 4.1. Identification des associations

association	Dimension	cardinalité	Classes-intervenant
exerce	2	1..*- 1	Agent - fonction
A pour benif	2	* - 1	Prestation - ayant droit
Est de type	2	1.* - 1	Ayant droit - type ayant droit
Est de type1	2	1.* - 1	Prestation - type prestation
Est de type2	2	1.* - 1	Arrêt de travail - type arrêt de travail
Présente	2	* - 1	Agent - prestation
Appartient	2	1.* - 1	Agent - structure
Affilie	2	1.* - 1	Agent - caisse
Affilie1	2	* - *	Agent - mutuel
Peut avoir	2	1.* - *	Agent - accident de travail
Dépôt arrêt de travail	2	1 - *	Agent - arrêt de travail

- **exerce :** chaque agent exerce une fonction
- **a pour benif :** pour chaque prestation a pour bénéficiaire des ayant droit
- **est de type :** chaque ayant droit appartiens a un type
- **est de type1 :** chaque prestation présenté appartiens a un type de prestation
- **est de type2 :** chaque arrêt de travail déposé appartiens a un type de travail
- **présente :** l'agent présente une prestation
- **appartiens :** un agent appartiens a une structure
- **affilie :** un agent affilié a une caisse
- **Affilie1 :** agent est affilié a une mutuelle
- **Peu avoir :** un agent peut avoir un accident de travail
- **Dépôt arrêt de travail :** agent déposé un arrêt de travail

3.1.2.3 Diagramme de classes :

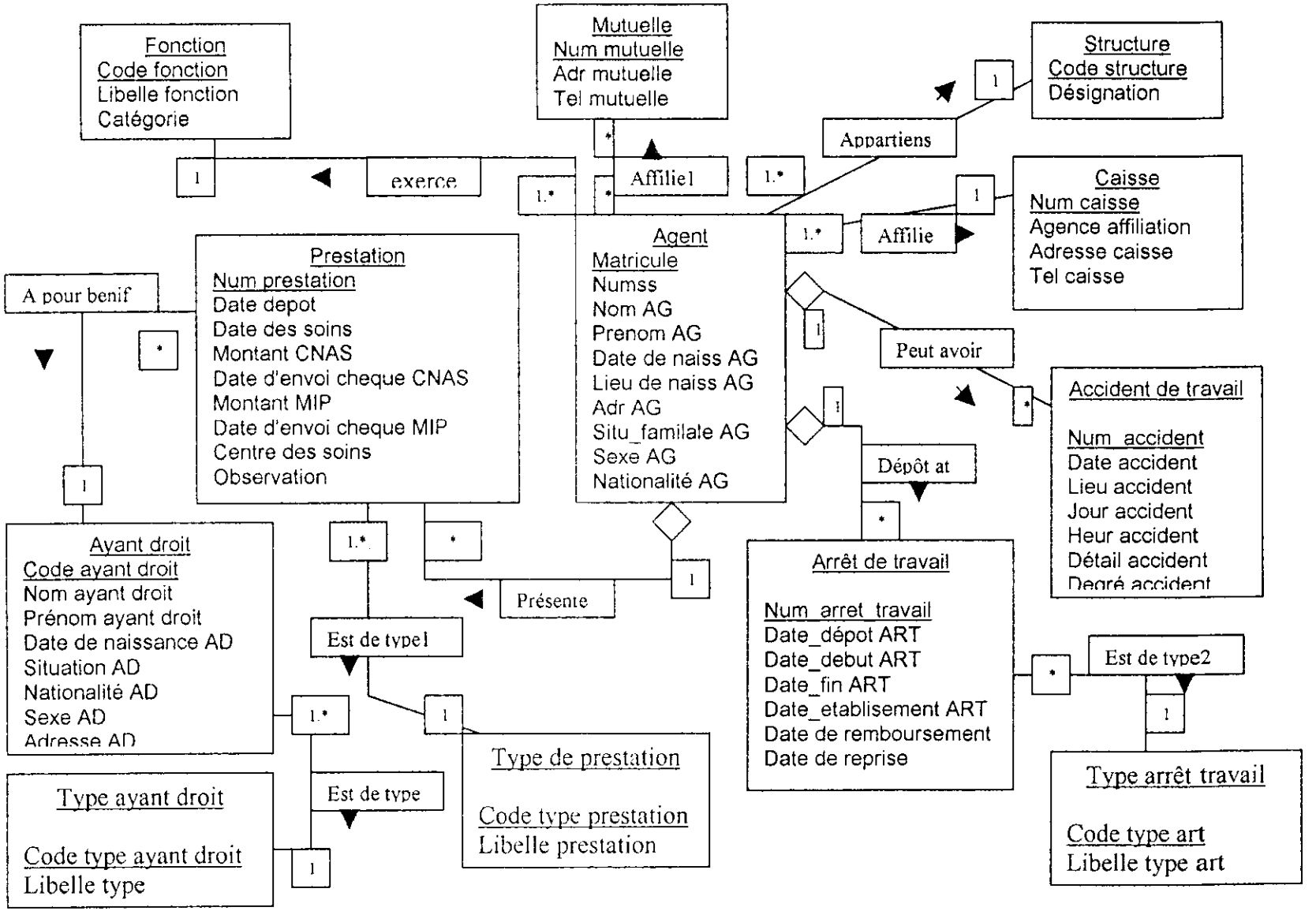


Figure 4.12. Diagramme des classes

3.1.3. Model fonctionnelle :

Le model fonctionnelle décrit l'aspect fonctionnelle du system cette phase consiste a rependre a la question suivante "que fait le system " et pour cela on utilise le diagramme "use case". Les étapes nécessaires à la construction, de ce modèle sont :

- le système
. les acteurs
. les cas d'utilisations
. les liens entre acteurs et cas d'utilisations

- Les acteurs :

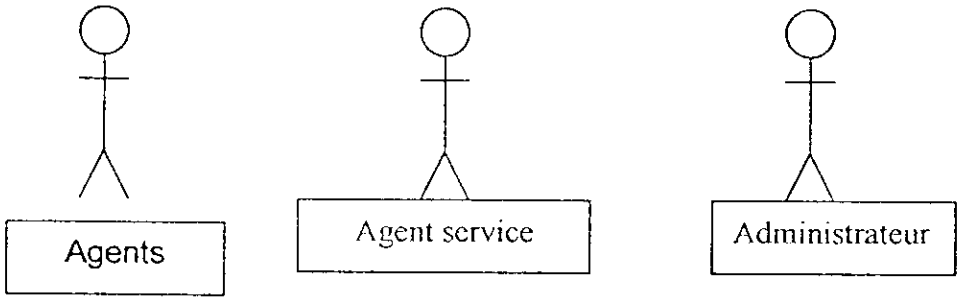


Figure 4-13 les acteurs

- Les cas d'utilisation (use case) :

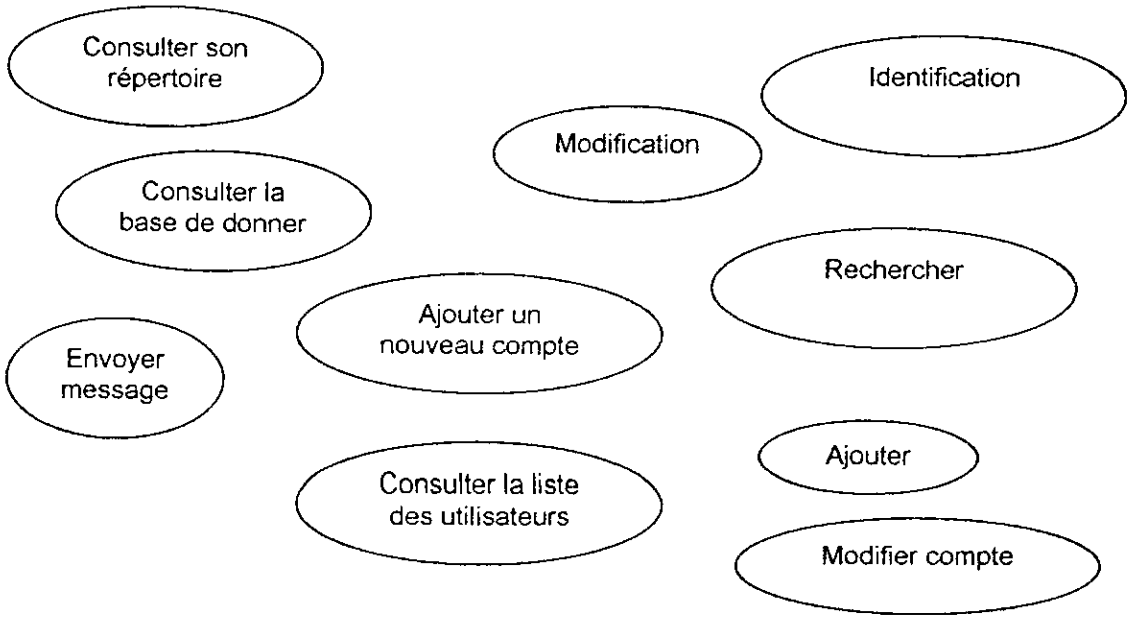
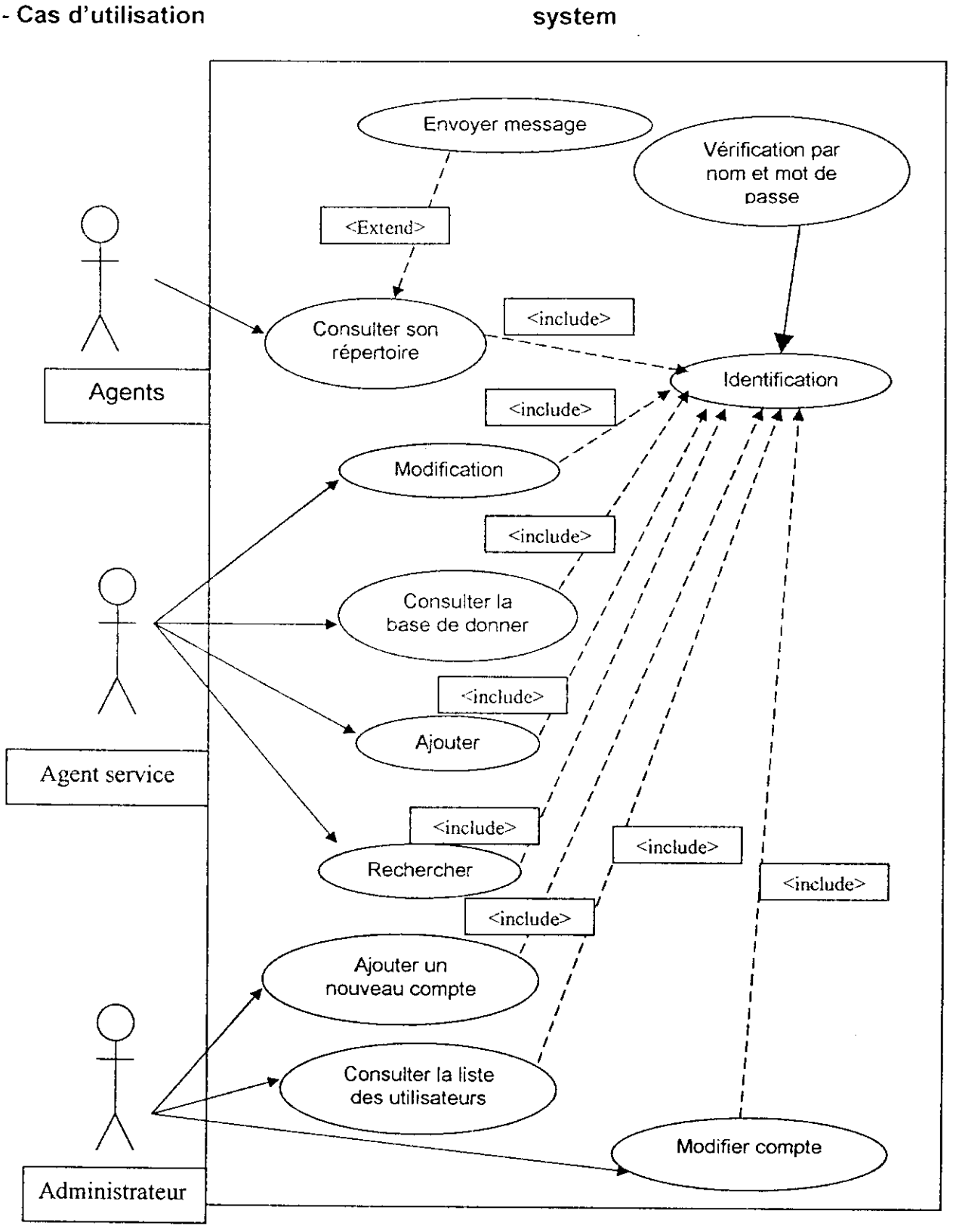


Figure 4-14 les cas d'utilisation

- Cas d'utilisation



Les acteurs

Figure 4-15 diagramme de cas d'utilisation

3.1.4. Modèle dynamique :

le modèle dynamique décrit des concepts traitant du flot de contrôle des interactions et des séquences d'opérations dans un système d'objet actif (concurrente). [CEE00]

- **Les scénarios par diagramme de séquence**

Le diagramme de séquences permet:

.de visualiser l'aspect temporel des interactions

.de connaître le sens des interactions (acteur vers système ou contraire)

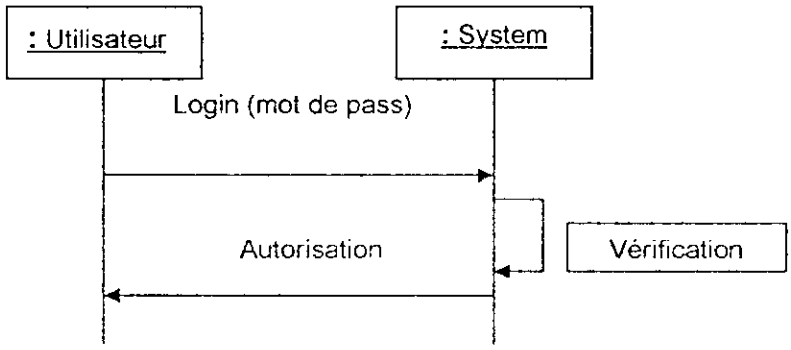


Figure 4-16 Identifications des utilisateurs

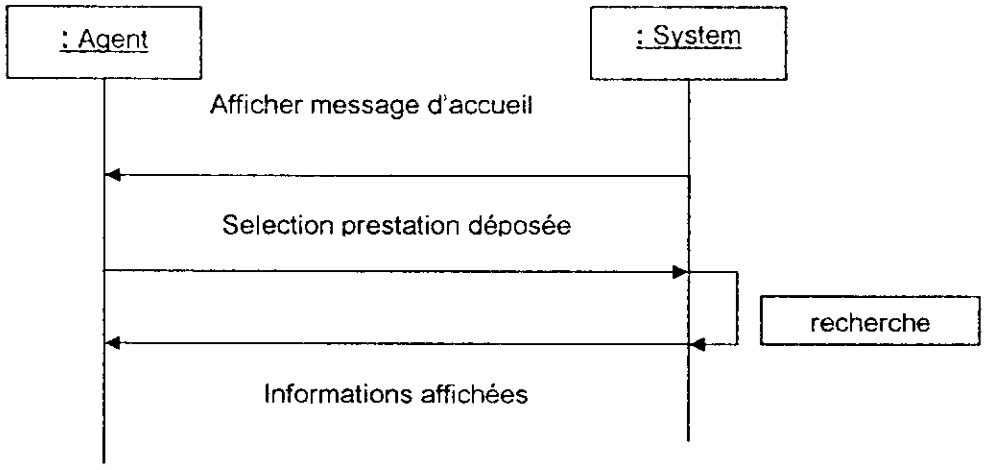


Figure 4-17 Consulter les prestations relatives a une personne

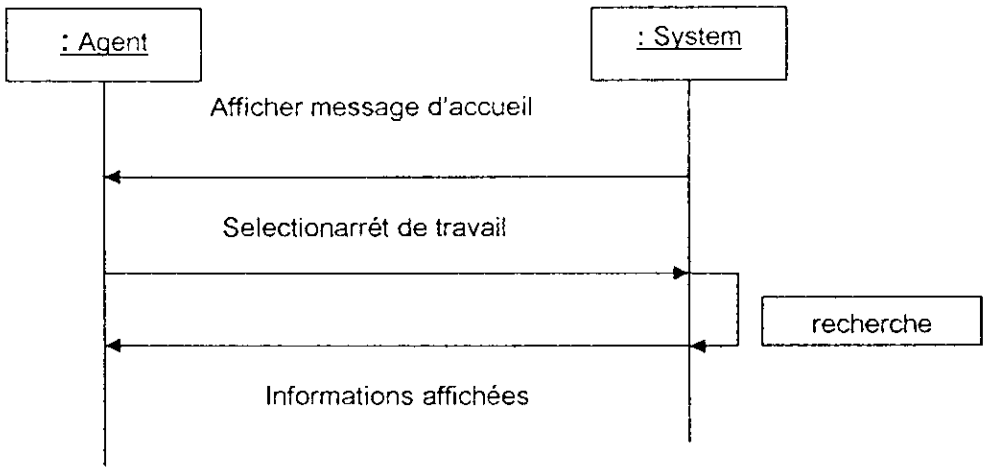


Figure 4-18 Consulter arrêt de travail relatif a une personne

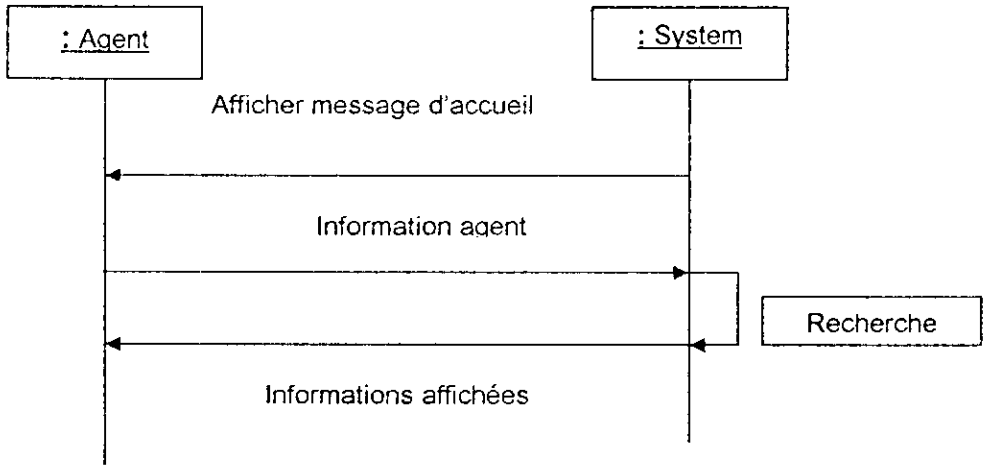


Figure 4-19 Consulter les informations sur l'agent

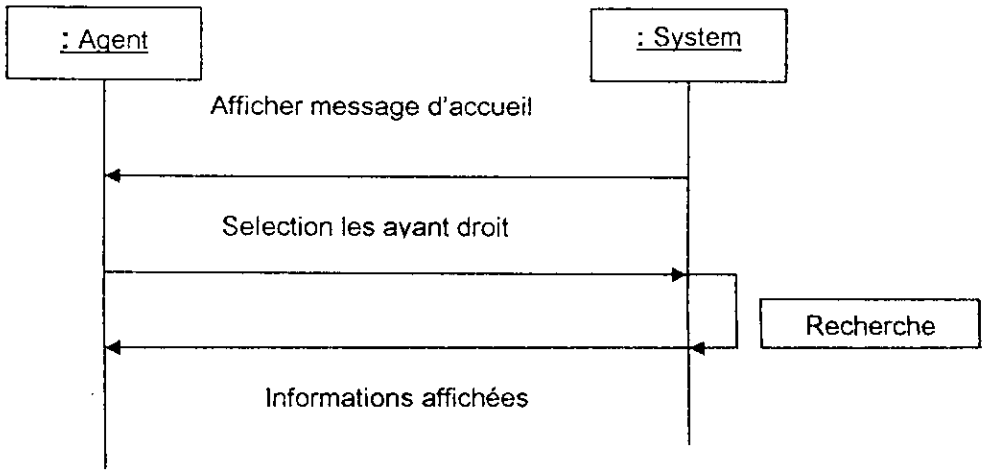


Figure 4-20 Consulter les ayant droit relatif a une personne

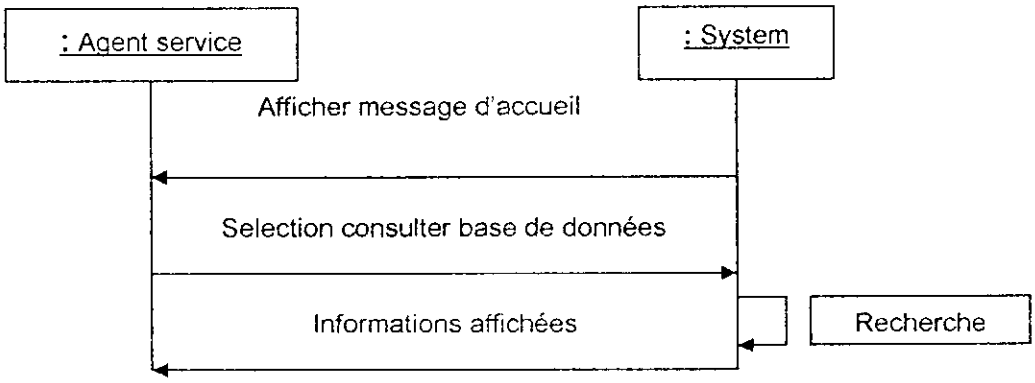


Figure 4-21 Consulter la base de données

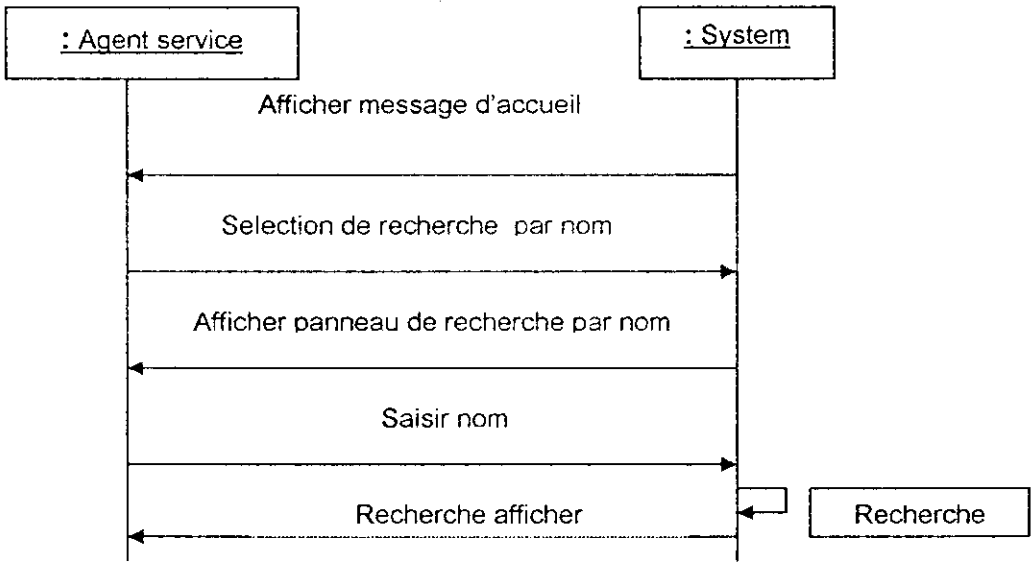


Figure 4-22 Recherche d'un agent par nom

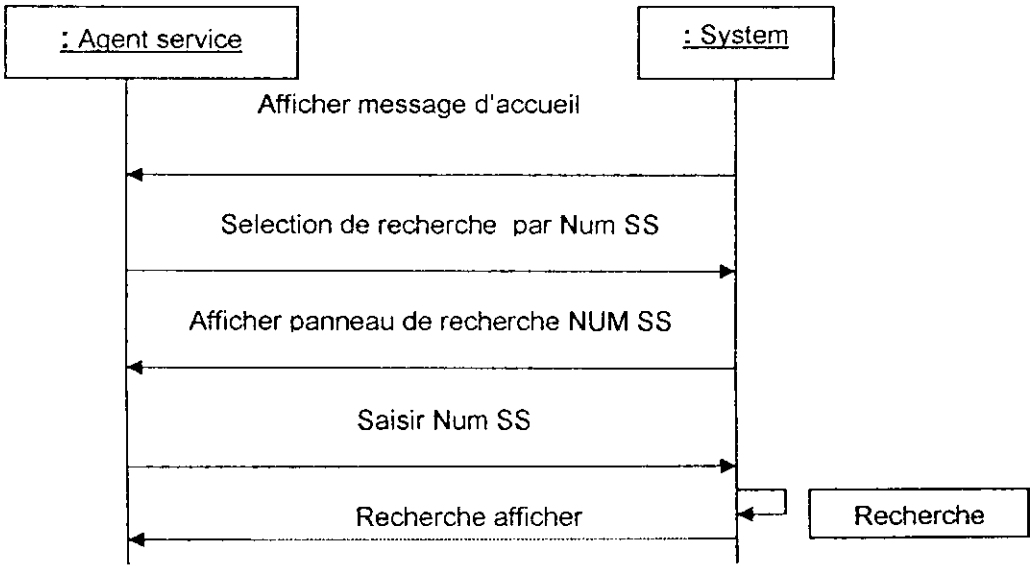


Figure 4-23 Recherche pare Numéro de sécurité social

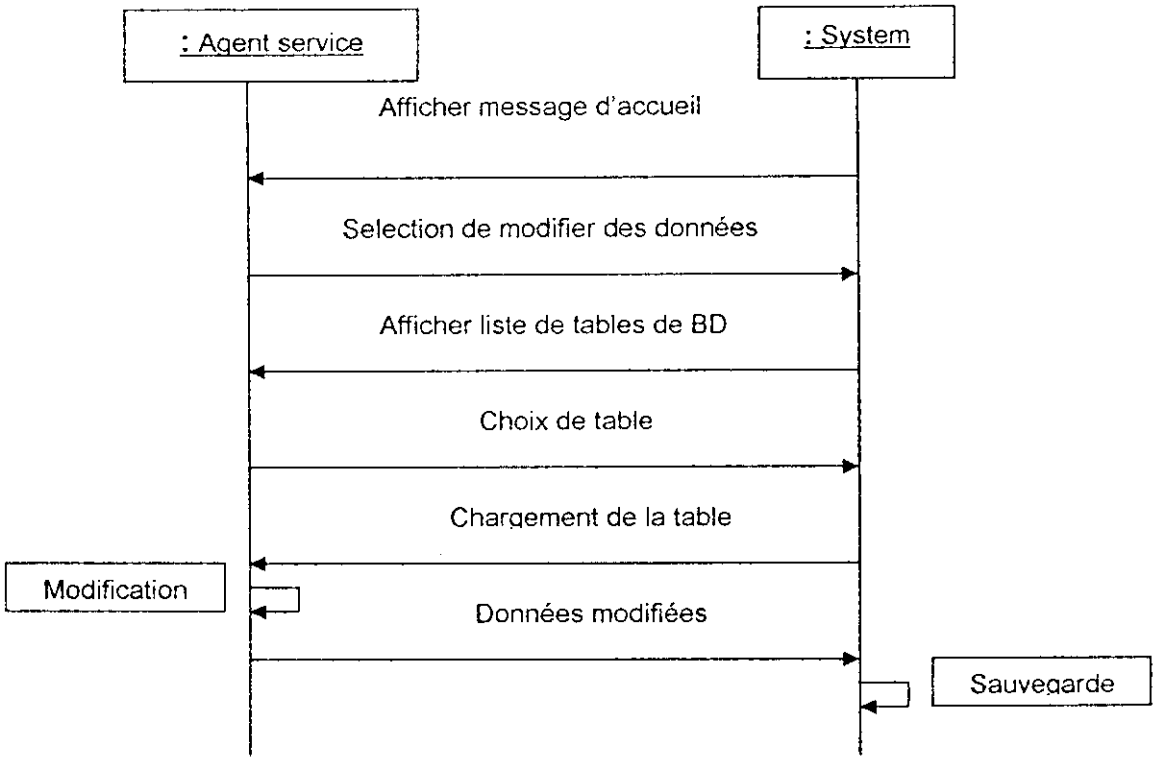


Figure 4-24 Modifier des données de la base

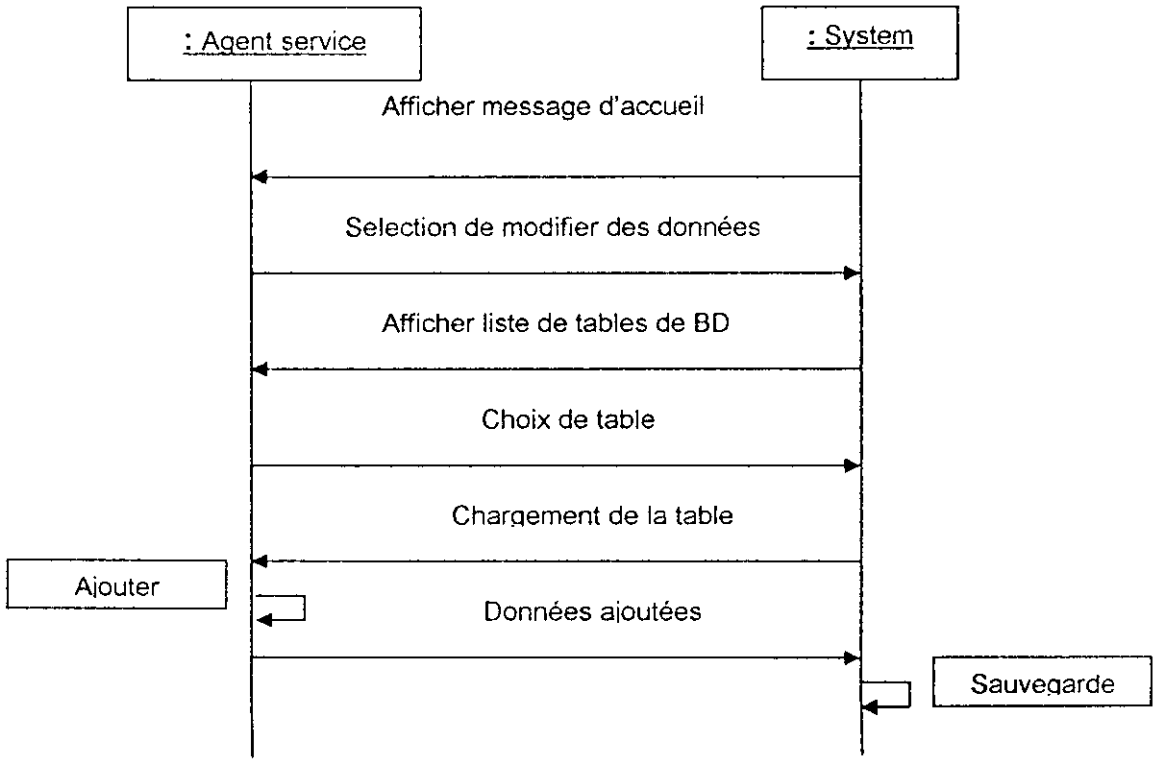


Figure 4-25 Ajouter des données a la base

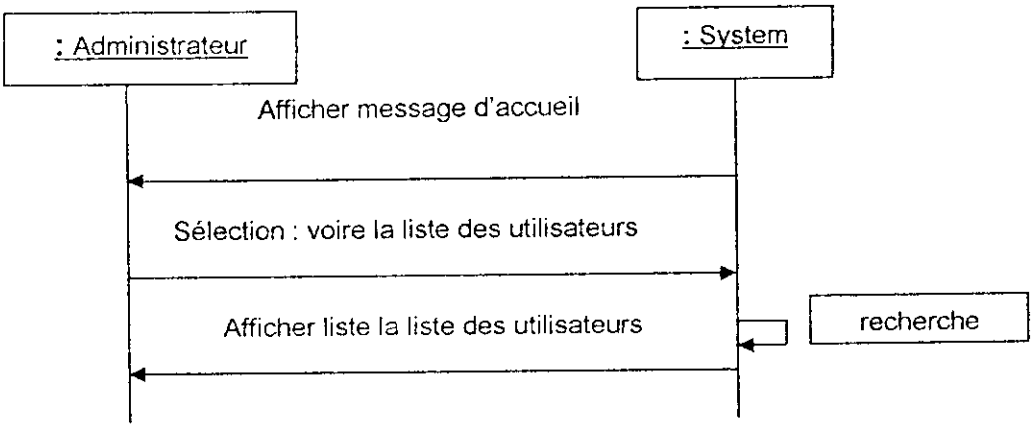


Figure 4-26 Consulter la liste des utilisateurs

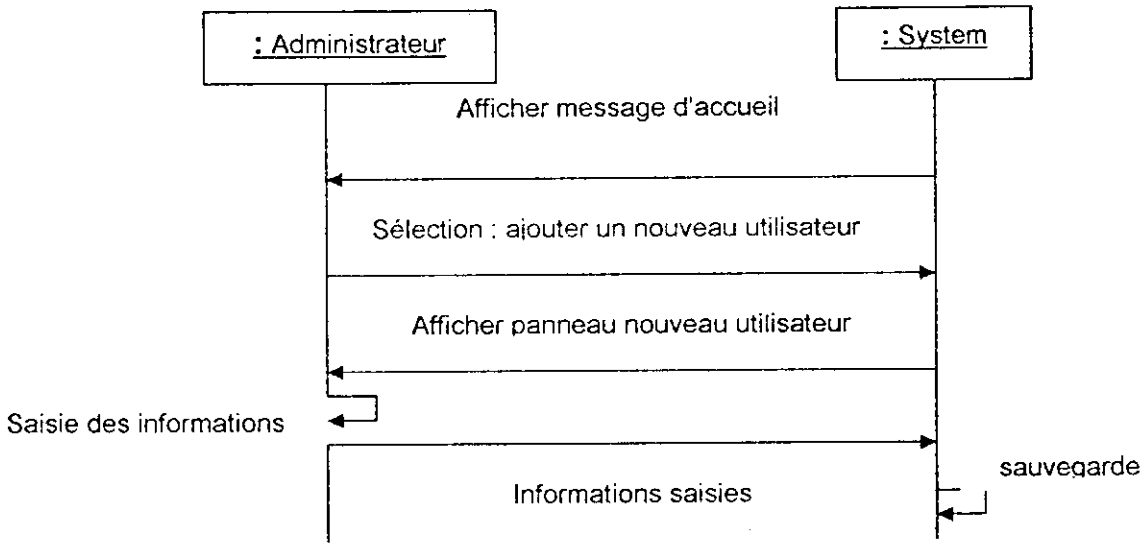


Figure 4-27 Ajouter un utilisateur

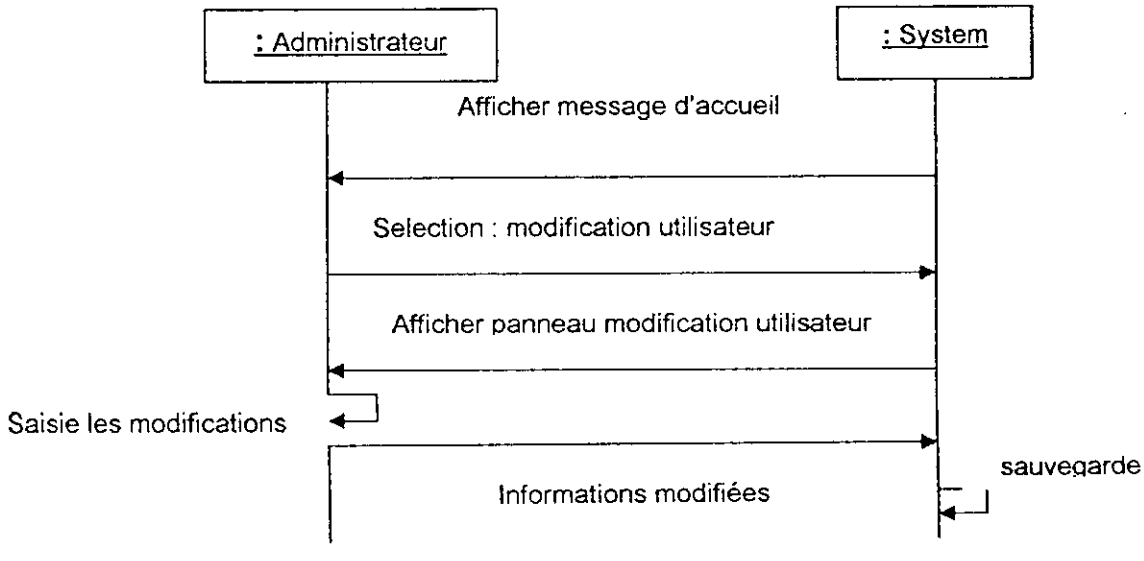


Figure 4-28 Modification utilisateur

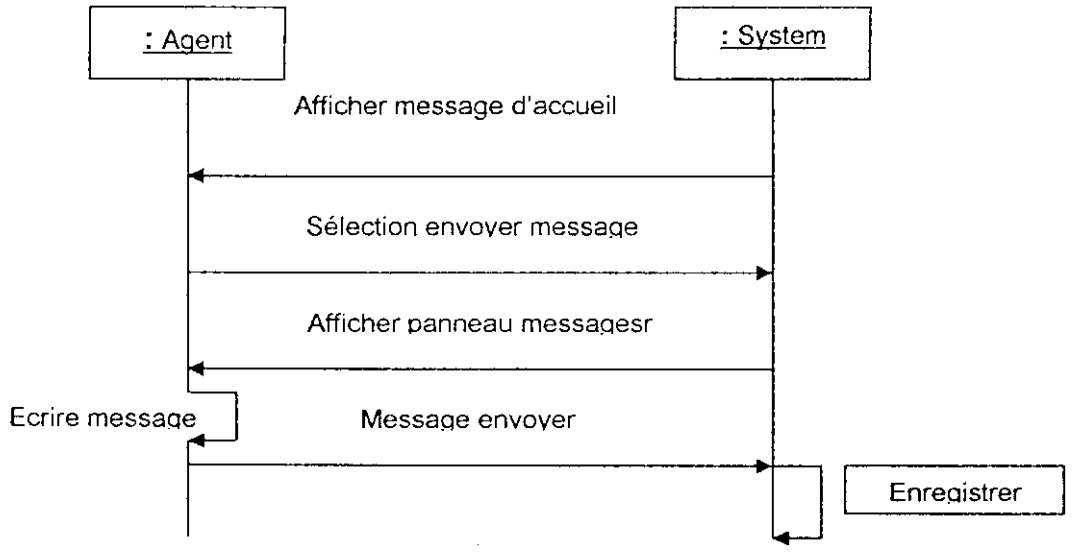


Figure 4-29 Envoyer message

- **Diagrammes de collaborations**

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait Figurer les envois de messages.

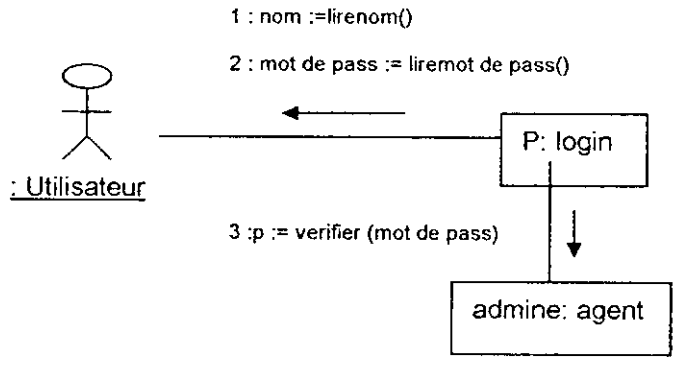


Figure 4-30 Diagramme de collaboration –indentification-

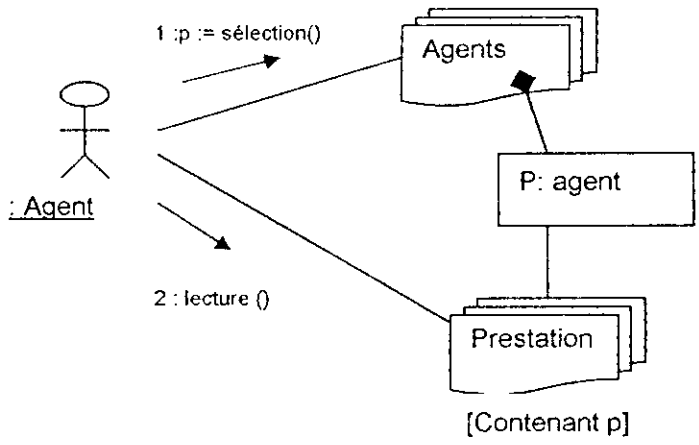


Figure 4-31 Digramme de collaboration de consultation prestation relatif a une personne

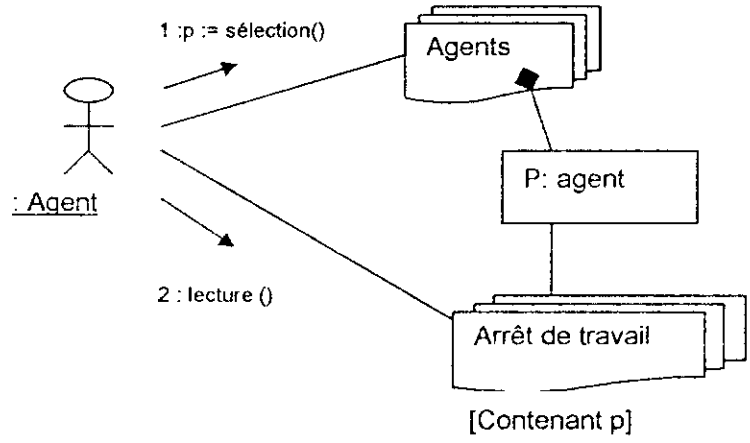


Figure 4-32 Diagramme de collaboration consulter arrêt de travail relatif à une personne

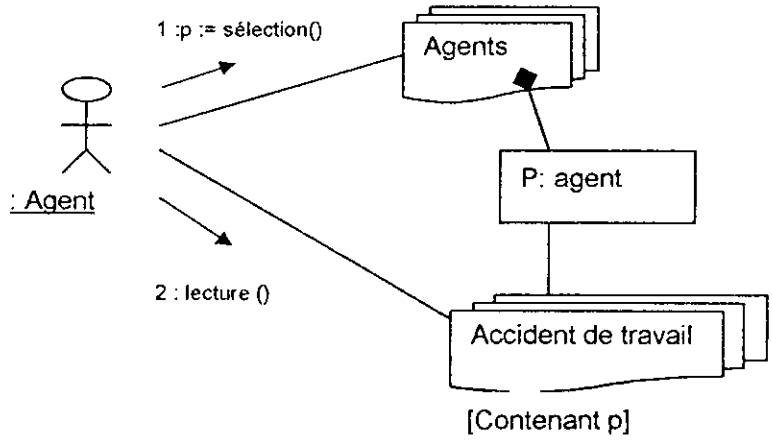


Figure 4-33 Diagramme de collaboration consulter accident de travail relatif à une personne

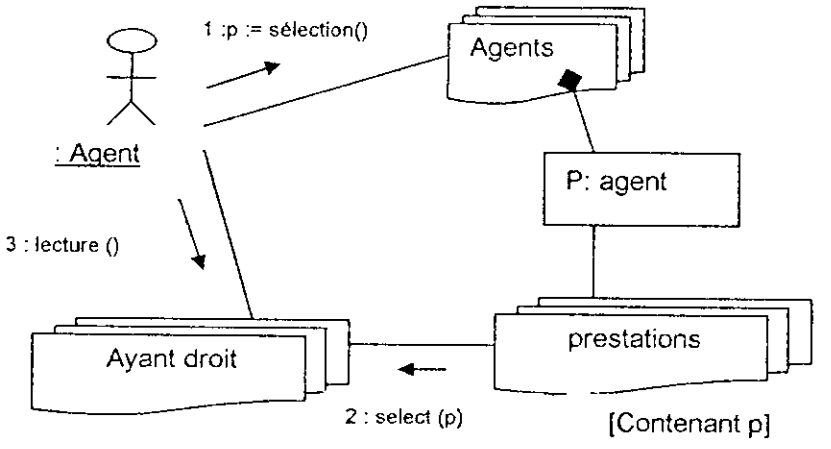


Figure 4-34 Diagramme de collaboration consulter ayant droit relatif à une personne

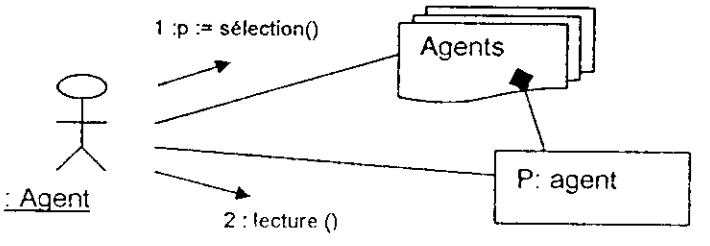


Figure 4-35 Diagramme de collaboration consulter information sur un agent

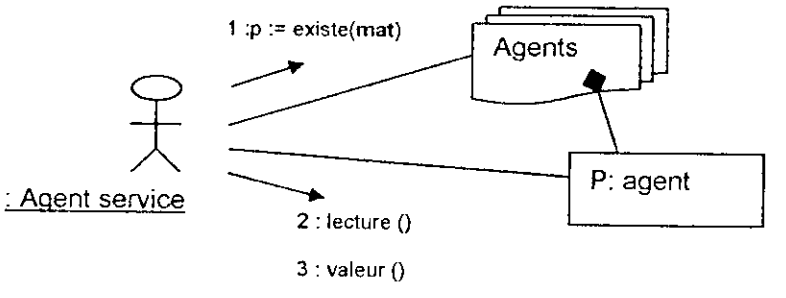


Figure 4-36 Recherche information sur un agent par matricule

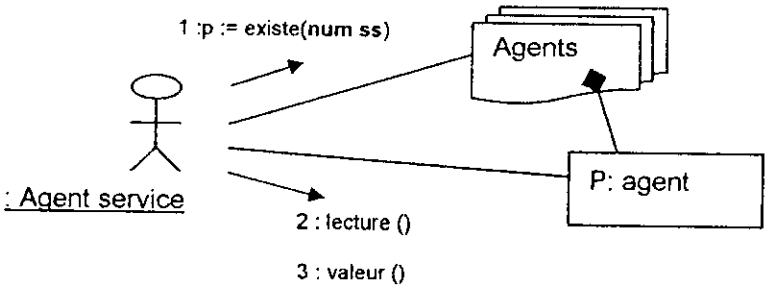


Figure 4-37 Recherche information sur un agent par numéro de sécurité sociale

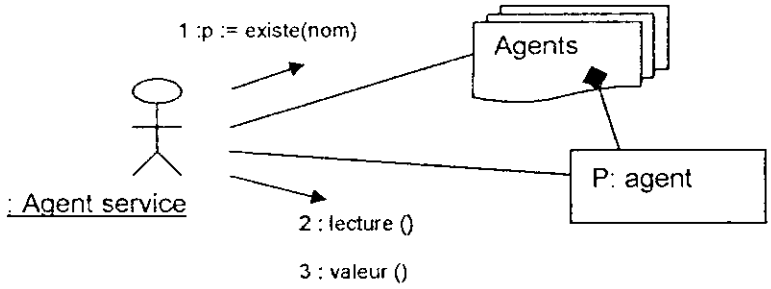


Figure 4-38 Recherche information sur un agent par nom

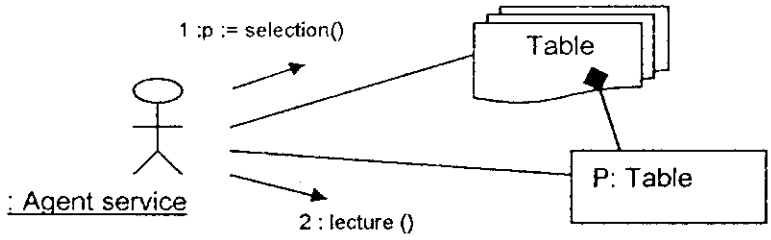


Figure 4-39 Consulter la base de données

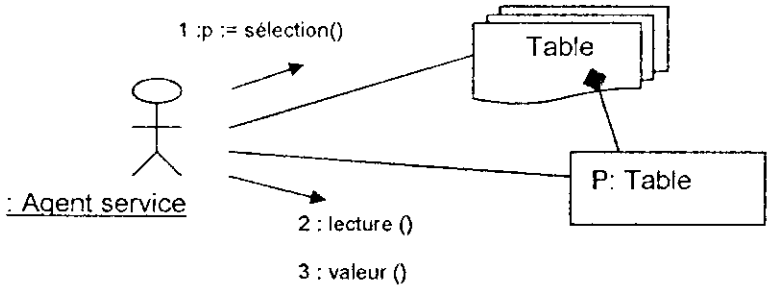


Figure 4-40 Modification de données de la base

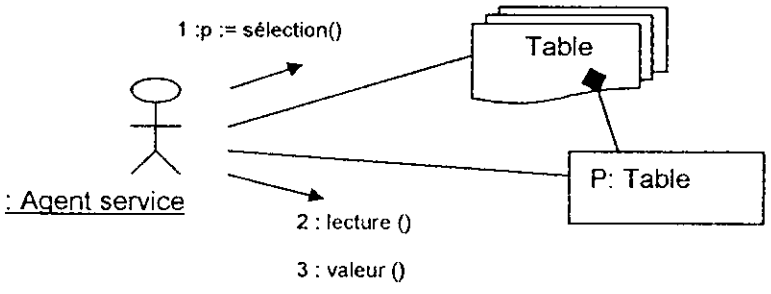


Figure 4-41 Ajouter des données à la base

4. CONCEPTION DE SYSTEME

4.1. Règles de passage entre modèle objet et modèle tables relationnelles

➢ Représentation des classes d'objets en tables :

- chaque classe est représentée par un ou plusieurs tables.

➢ Représentation des associations d'objets en tables :

- Chaque association plusieurs à plusieurs est représentée par une table distincte.
- Une association un à plusieurs est représentée par une table distincte ou peut être enfouie comme clé étrangère dans la table pour l'une ou l'autre des classes. Pour les associations un à plusieurs ou un à un, il n'y a pas de cycle, on dispose de l'option supplémentaire qui consiste à ranger l'association et les objets liés dans une seule table.

Ayant conscience que cela peut introduire une redondance et violer des formes normales.

- Les noms de rôles sont incorporés en tant que partie du nom de l'attribut de la clé étrangère.
- Les associations n-aires (n>2) se représentent par des tables distinctes. Ce qui aide parfois à promouvoir une association n-aires en une classe.
- Une association qualifiée se représente en une table distincte avec au moins trois attributs, la clé primaire de chaque classe liée est le qualificatif.
- Les agrégations suivent les mêmes règles que les associations.

➢ **Représentation de la généralisation de l'héritage simple en tables :**

- On représente chaque superclasse et chaque sous-classe par une table.
- S'il n'y a de table de superclasse, les attributs sont dupliqués dans chaque table de sous-classe.
- S'il n'y a de table de sous-classe, on apporte tous les attributs de sous-classe dans la superclasse.

4.2. Traduction du modèle objet en base de données relationnelles

En appliquant les règles de passage au modèle objet, on obtient la représentation logique de la base de données qui se représente comme suit :

Nom fichier : AGENT		
Nom champ	**type**	**longueur**
Matricule	AN	06
Num sécurité sociale	N	12
Nom AG	A	15
Prénom AG	A	15
Date de naiss AG	D	08
Lieu de naiss AG	AN	15
Adr AG	A	01
Situation_familale AG	AN	25
Sexe AG	A	01
Nationalité AG	A	15
Code fonction	AN	10
Num caisse	AN	05
N° affiliation	N	06
Date affilmiation	N	12
	D	08

Tableau 4-2- Représentations logique de la classe agents

Nom fichier : TYPE PRESTATION		
Nom champ	**type**	**longueur**
Code type prestation	A	03
libelle	AN	15

Tableau 4-3 Représentations logique de la classe type prestation

Nom fichier : ARRET DE TRAVAIL		
Nom champ	**type**	**longueur**
Num_arret_travail	N	06
Date_dépot ART	D	08
Date_debut ART	D	08
Date_fin ART	D	08
Date_etablisement ART	D	08
Date de remboursement	D	08
Date de reprise	D	08
Code type arrêt de travail	A	03
matricule	AN	06

Tableau 4-4 Représentations logique de la classe arrêt de travail

Nom fichier : ACCIDENT DE TRAVAIL		
Nom champ	**type**	**longueur**
Num_accident	N	03
Date accident	D	08
Lieu accident	AN	25
Jour accident	A	07
Heur accident	N	05
Détail accident	AN	15
Degré accident	AN	20
matricule	AN	06

Tableau 4-5 Représentations logique de la classe accident de travail

Nom fichier : STRUCTURE		
Nom champ	**type**	**longueur**
Code structure	AN	05
Désignation	A	25

Tableau 4-6 Représentations logique de la classe structure

Nom fichier : PRESTATION		
Nom champ	**type**	**Longueur**
Num prestation	N	08
Date depot	D	08
Date des soins	D	08
Montant CNAS	N	07
Date d'envoi cheque CNAS	D	08
Montant MIP	N	07
Date d'envoi cheque MIP	D	08
Centre des soins	AN	15
Observation	A	20
Code ayant droit	N	02
Code type prestation	A	03
matricule	AN	06

Tableau 4-7 Représentations logique de la classe prestation

Nom fichier : FONCTION		
Nom champ	**type**	**longueur**
Code fonction	AN	10
Libelle	AN	20
catégorie	A	02

Tableau 4-8 Représentations logique de la classe fonction

Nom fichier : AYANT DROIT		
Nom champ	**type**	**longueur**
Code ayant droit	N	02
Nom ayant droit	A	15
Prénom ayant droit	A	15
Date de naissance AD	D	08
Situation AD	A	01
Nationalité AD	A	15
Sexe AD	A	01
Adresse AD	AN	25
Code ayant droit	A	03

Tableau 4-9 Représentations logique de la classe ayant droit

Nom fichier : TYPE ARRET DE TRAVAIL		
Nom champ	**type**	**longueur**
Code type arrêt de travail	A	03
Libelle type arrêt de travail	AN	20

Tableau 4-10 Représentations logique de la classe type arrêt de travail

Nom fichier : TYPE AYANT DROIT		
Nom champ	**type**	**longueur**
Code type ayant droit	A	03
libelle	AN	20

Tableau 4-11 Représentations logique de la classe type ayant droit

Nom fichier : CAISSE		
Nom champ	**type**	**longueur**
Num caisse	N	06
Agence affiliation	N	06
Adresse affiliation	AN	30
tel	N	09

Tableau 4- 12 Représentations logique de la classe caisse

4.3. CONCEPTION DES OBJETS

Chaque table de schéma conceptuel est représentée par la définition d'un objet au niveau du modèle objet externe, en les raffinant avec l'ajout des méthodes qui sont directement déduites des actions du modèle dynamique ou des traitements du modèle fonctionnel et en respectant les règles de passage énoncées précédemment. La conception des objets du système est définie par les prototypes des classes suivantes :

classe : AGENT	
Attribut	**méthode**
Matricule	Sélection ()
Num sécurité sociale	Image ()
Nom AG	Valeur ()
Prénom AG	Existe (string)
Date de naiss AG	Existe (double)
Lieu de naiss AG	
Adr AG	
Situation_familale AG	
Sexe AG	
Nationalité AG	
Code fonction	
Num caisse	
N° affiliation	
Date affilmiation	

Tableau 4-13 prototype de la classe agents

classe : prestation	
Attribut	**méthode**
Num prestation	Sélection ()
Date dépôt	Image ()
Date des soins	Valeur ()
Montant CNAS	Existe (double)
Date d'envoi cheque CNAS	
Montant MIP	
Date d'envoi cheque MIP	
Centre des soins	
Observation	
Code ayant droit	
Code type prestation	
matricule	

Tableau 4-14 prototype de la classe prestation

classe : arrêt de travail	
Attribut	**méthode**
Num_arret_travail	Sélection ()
Date_dépot ART	Image ()
Date_debut ART	Valeur ()
Date_fin ART	Existe (double)
Date_etablisement ART	
Date de remboursement	
Date de reprise	
Code type arrêt de travail	
matricule	

Tableau 4-15 prototype de la classe arrêt de travail

classe : accident de travail	
Attribut	**méthode**
Num_accident	Sélection ()
Date accident	Image ()
Lieu accident	Valeur ()
Jour accident	Existe (double)
Heur accident	
Détail accident	
Degré accident	
matricule	

Tableau 4-16 prototype de la classe accident de travail

5. CONCLUSION

L'objectif de la modélisation est de comprendre le problème, son domaine d'application et de construire une conception correcte avant de passer à l'implémentation.

Les trois étapes de l'étude ont été étudiées, l'étape statique, dynamique et fonctionnelle. Reste à entamer l'implémentation.

Chapitre 5

Implémentations et résultats

1. Introduction :

Au cours de notre étude, plusieurs résultats ont été obtenu, notamment le modèle dynamique, le modèle fonctionnel, la conception des objets du système.
L'ensemble des ces résultats a été soumis à l'approbation de la direction SONATRACH.
Les résultats de cette étude feront l'objet de projection sur machine, selon l'outil choisi pour le niveau physique et permettront de rendre la solution opérationnelle.

2. Environnement technique de développement

2.1 Présentation des langages de programmation utilisés

Pour l'implémentation et la gestion de la base de données, et le développement de l'application on a utilisé les langages suivants :

2.1.1. Le langage SQL

Pour la mise en place et la gestion de la base de données, le langage utilisé est le SQL, donc parfaitement compatible avec la plate forme choisie, de plus il est entièrement conçu pour le web.
Le SQL est u langage de requête structuré (**S**tructured **Q**uery **L**angage) destiné à communiquer avec des bases de données relationnelles implémentées dans des SGBDR
Le langage SQL permet d'effectuer divers opération comme la création, l'extraction, la modification, la suppression, la fusion, etc..., sur des collections de données, par l'intermédiaire d'instructions particulière appelées des commande, souvent assistées d'ailleurs par des clauses ou des options.

2.1.2. Les JSP :

Les JSP (Java Server Pages) sont un standard permettant de développer des applications Web interactives, c'est-à-dire dont le contenu est dynamique. C'est-à-dire qu'une page web JSP (repérable par l'extension .jsp) aura un contenu pouvant être différent selon certains paramètres (des informations stockées dans une base de données,

Les JSP sont intégrables au sein d'une page Web en HTML à l'aide de balises spéciales permettant au serveur Web de savoir que le code compris à l'intérieur de ces balises doit être interprété afin de renvoyer du code HTML au navigateur du client.
Les JSP permettent donc d'écrire facilement des servlets, en incluant dans des balises spécifiques le code JSP au sein du fichier HTML. De cette façon, elles fournissent une technologie rapide afin de créer des pages dynamiques.
De plus, les JSP étant basées sur Java côté serveur, elles possèdent toutes les caractéristiques faisant la force de Java :

les JSP sont multithreadées,
les JSP sont portables,
les JSP sont orientées objet,
les JSP sont sûres,

2.2. Implémentation

La base de données a été implémentée avec oracle 9i :

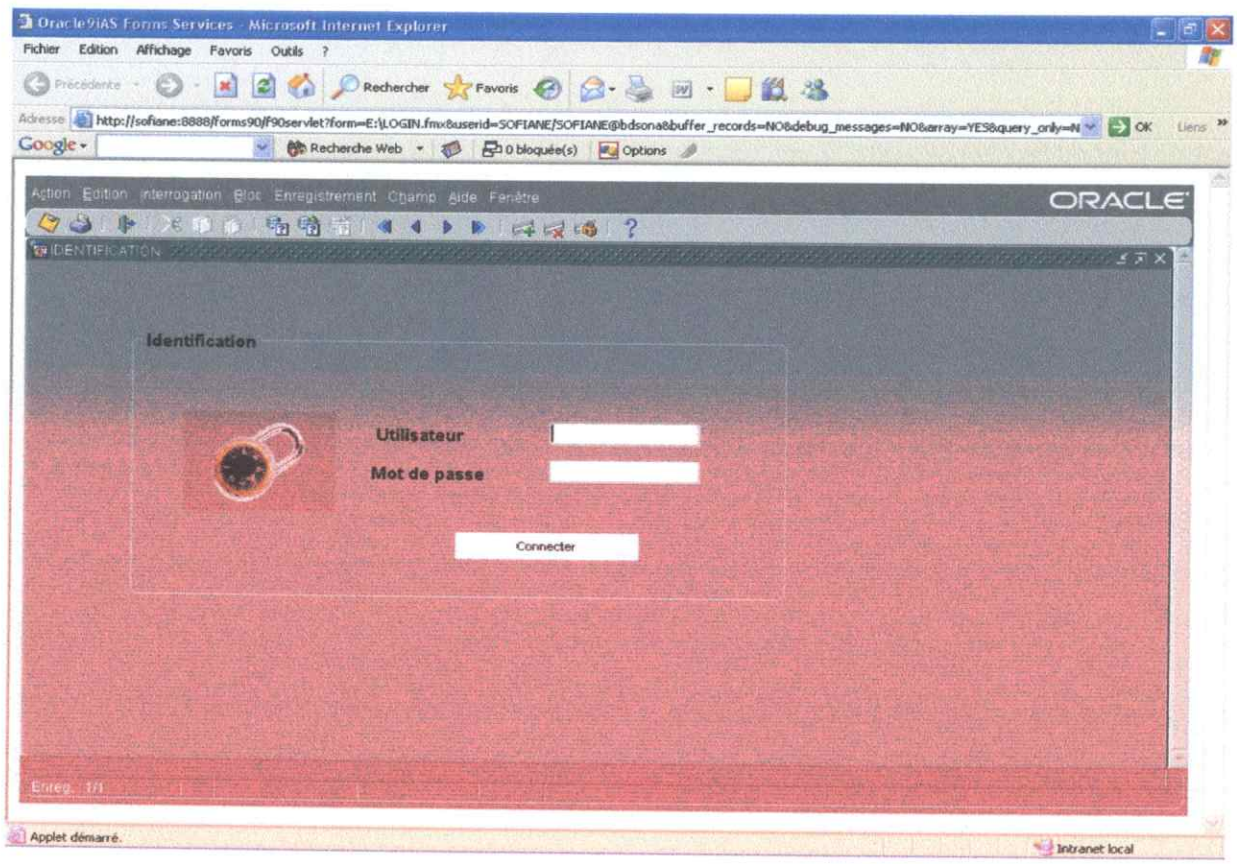
2.2.1 Oracle 9i :

C'est un system relationnel de gestion de base de données Multi-Utilisateurs à haute performance tournant sous le système d'exploitation Windows (XP/NT/2000 serveur.) ou linux, il est conçu pour être exploité sur un ordinateur serveur et être le composant serveur dans un environnement trois-tiers

2.3. Test de l'application :

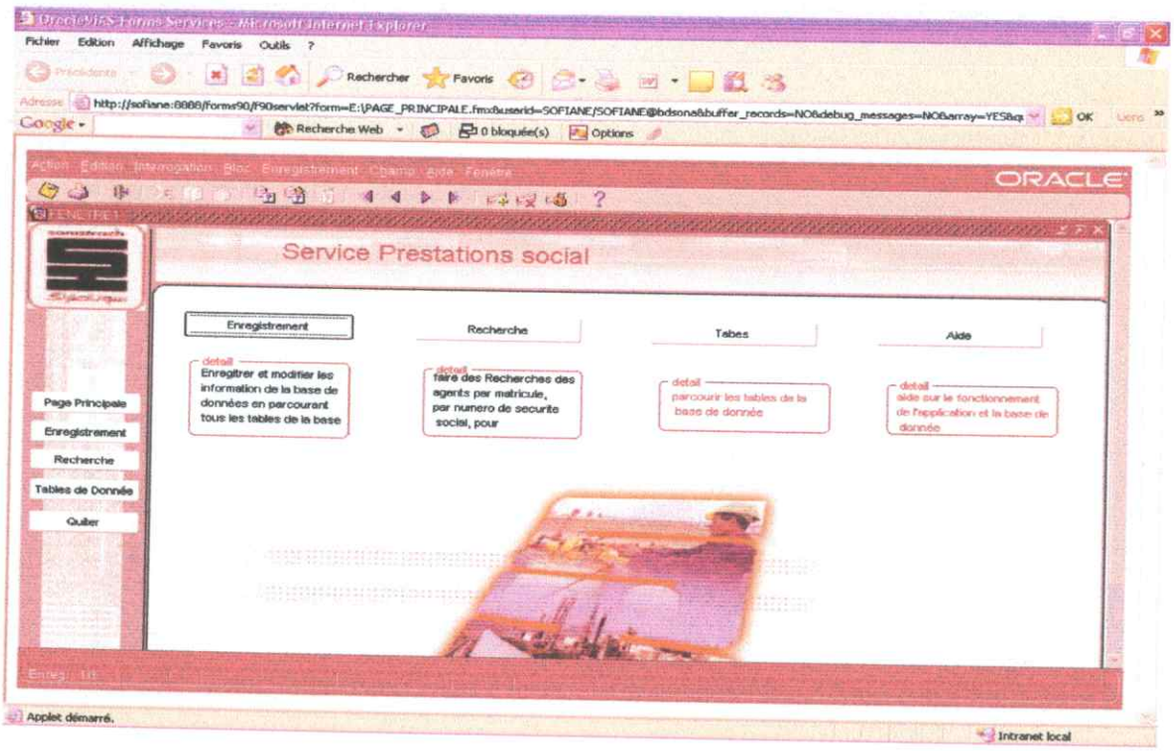
L'utilisateur de notre application web peu faire le suivi des dossiers des prestations sociales en utilisant l'intranet de la société ou l'Internet.

L'utilisateur doit saisir son nom et son mot de passe.

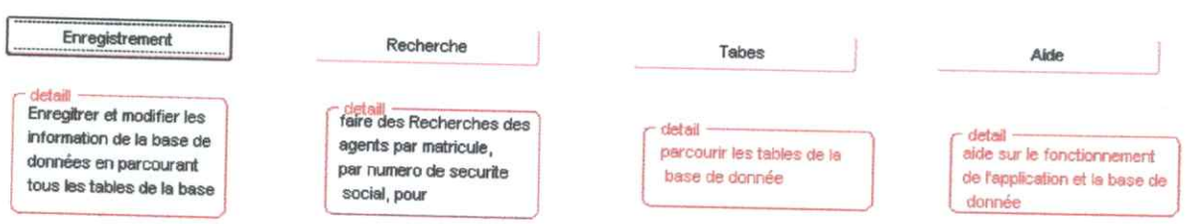


Apres la saisie un composant s'occupe de vérifier la validation des informations en entrée, en cas d'erreur, un message s'affiche pour l'indiquer

Si les données sont valides, l'utilisateur aura une page contenant toutes les opérations qu'il peut effectuer, dans se cas c'est l'agent du service qui est identifiée.

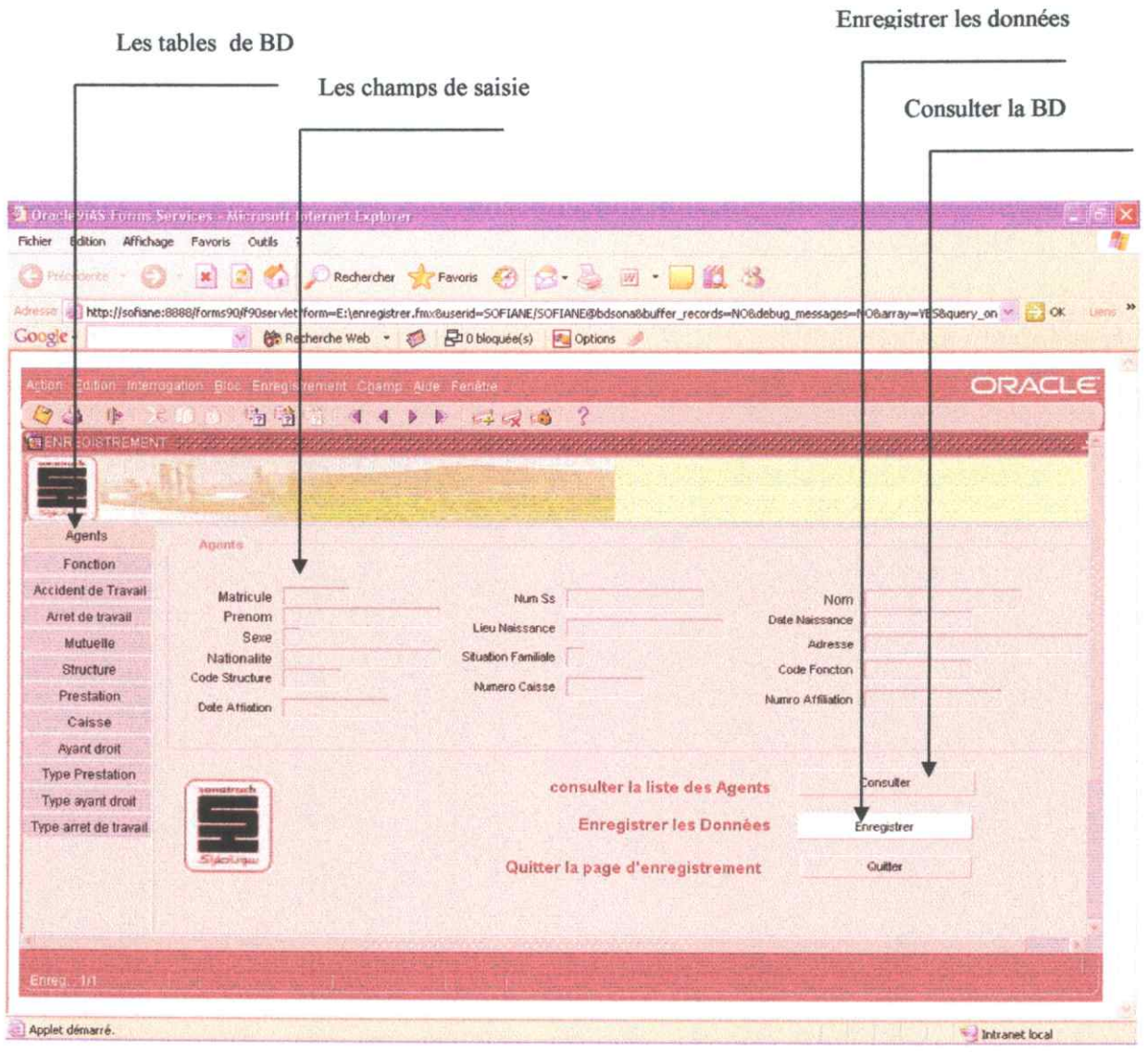


La page principale permet d'accéder aux différentes fonctions de l'application, dans ce qui suit nous allons détailler chaque sous partie.

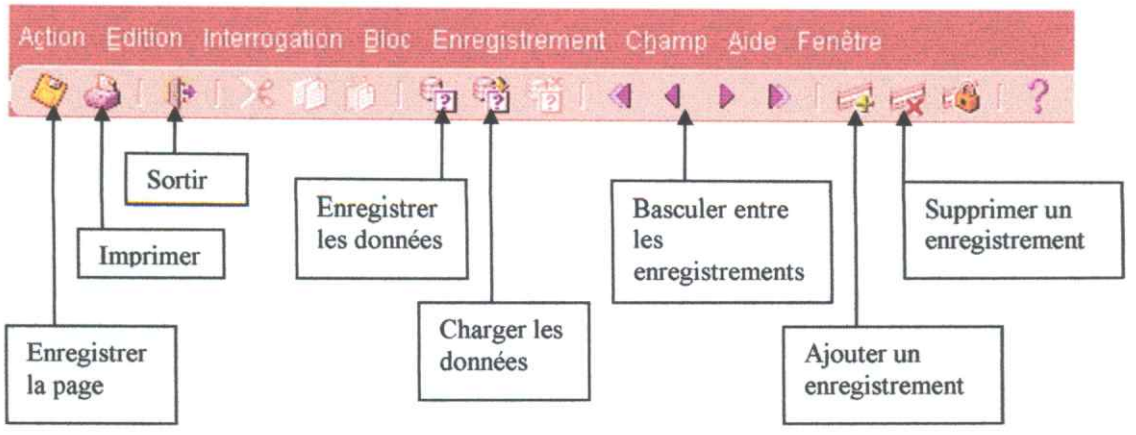


2.3.1. Page Enregistrement :

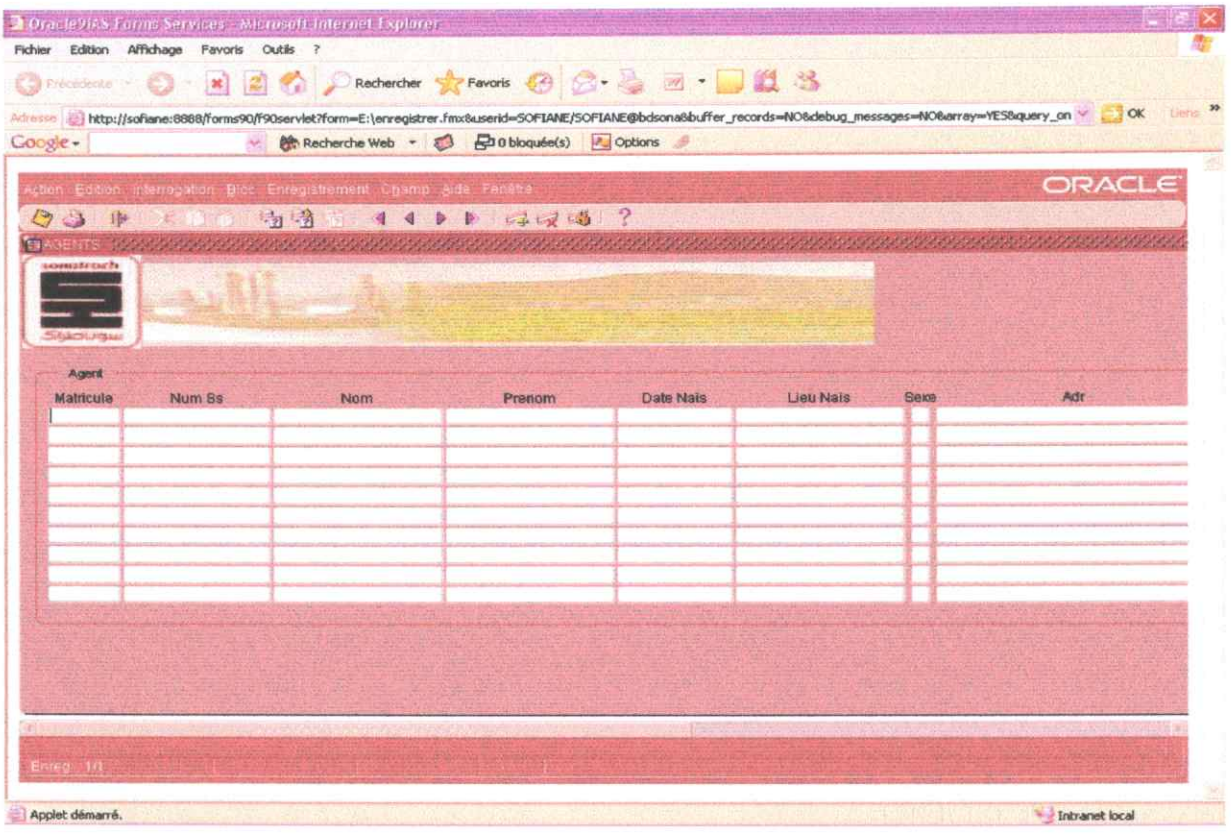
La page enregistrement permet de saisir les données dans la base et de consulter, en va détailler cette partie :



- zone de contrôle : permet de charger, supprimer ou modifier les données

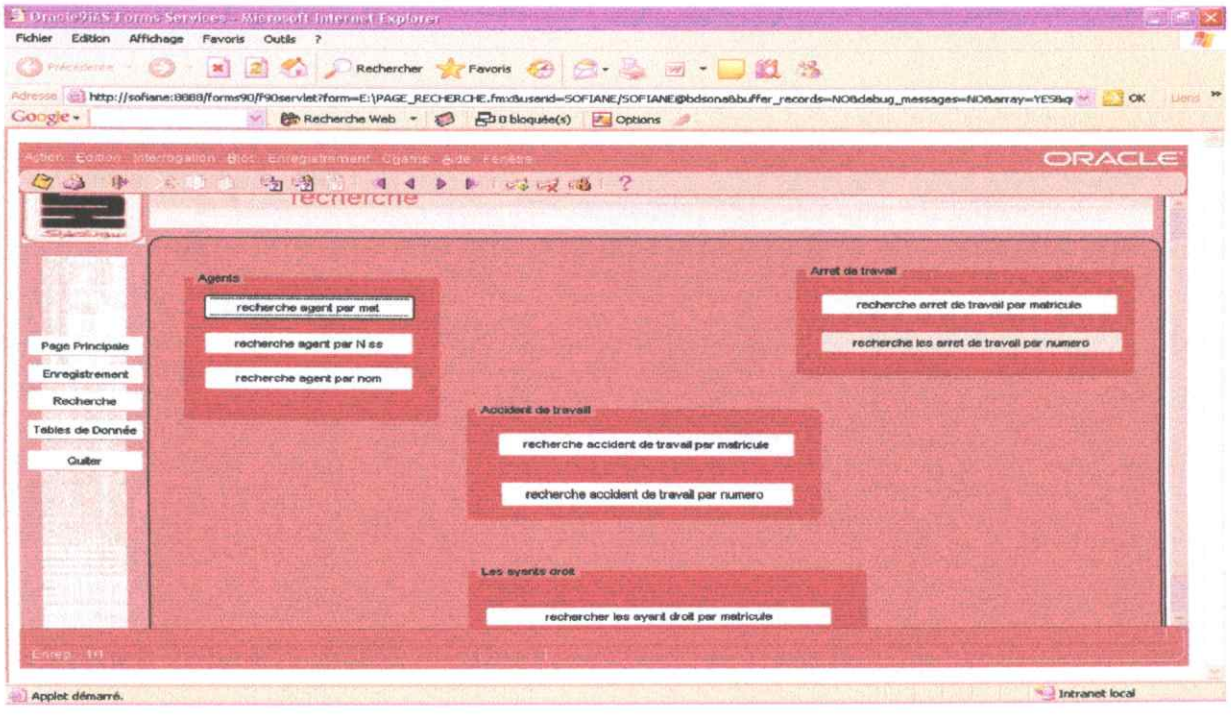


En choisissant de consulter le BD on aura la page suivante :

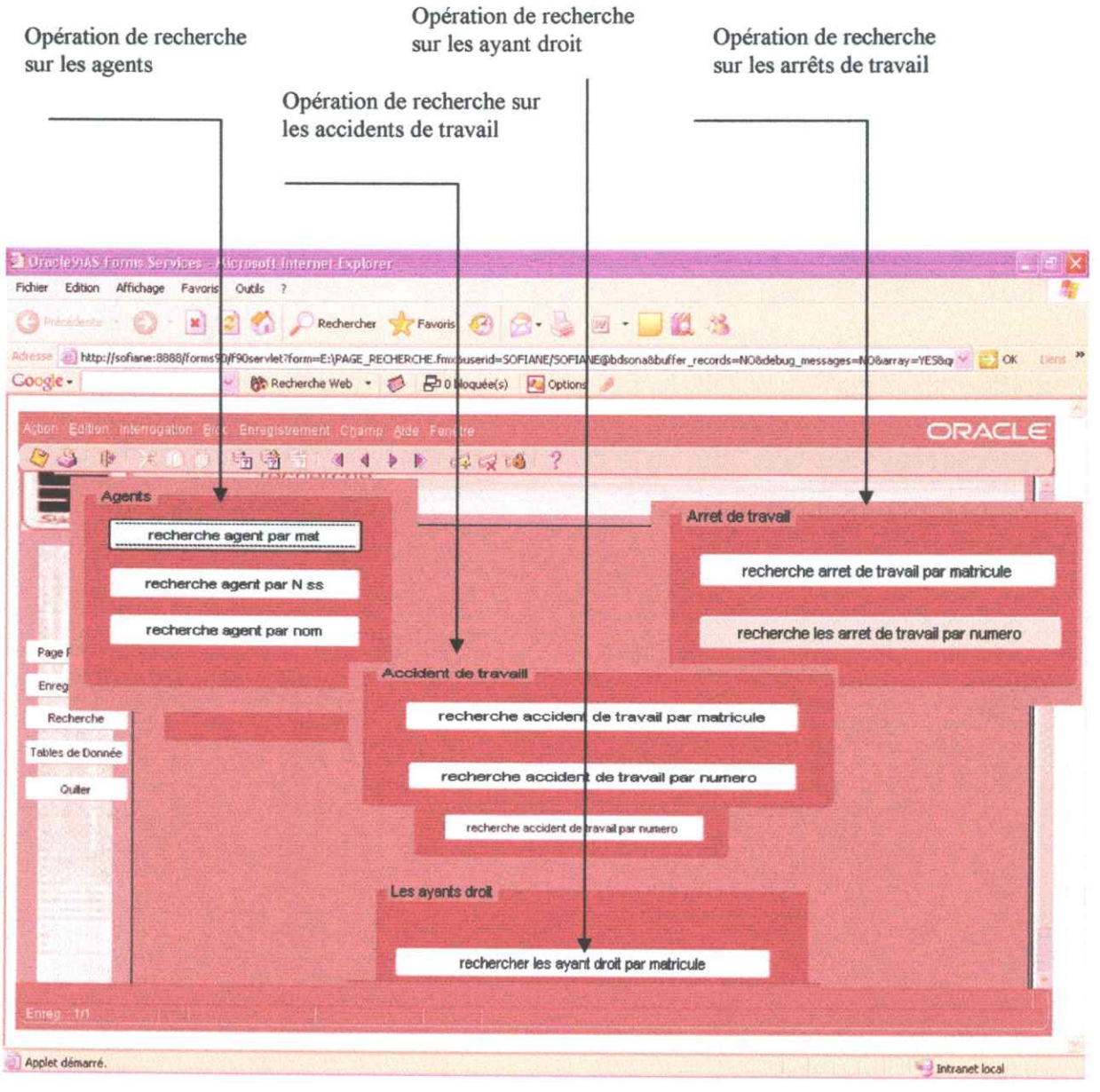


2.3.2. Page de recherche :

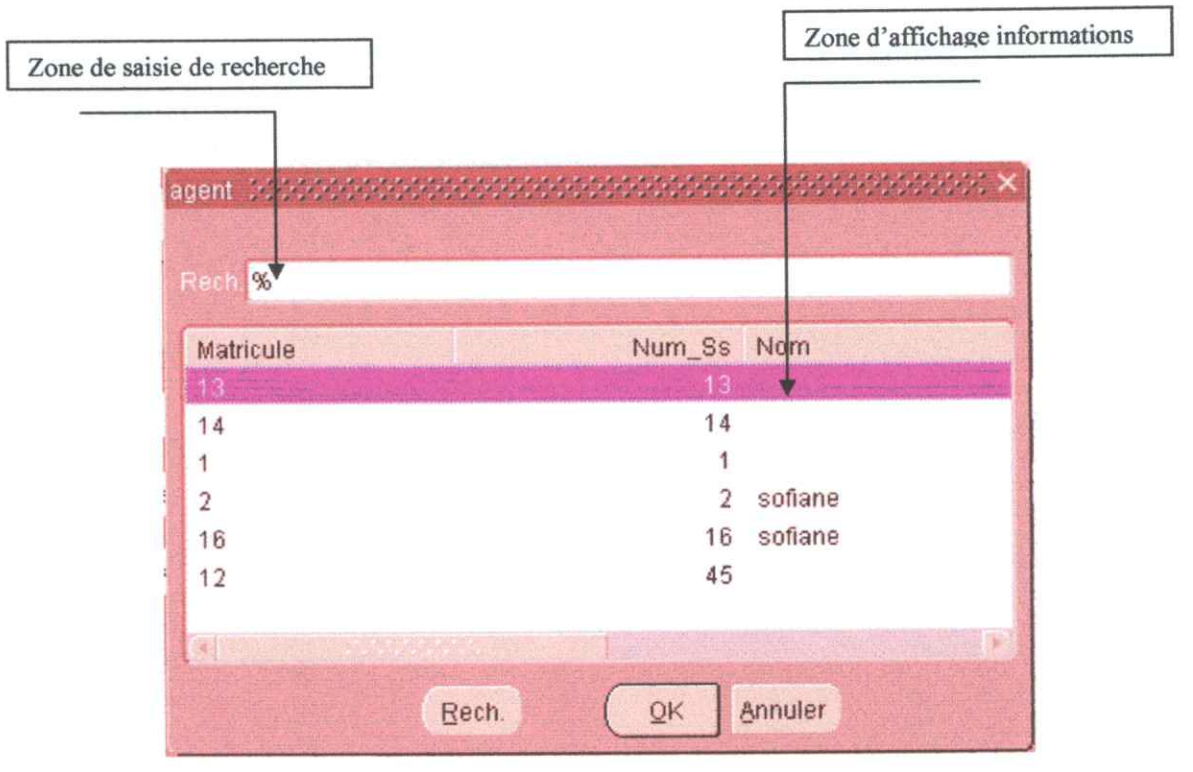
La page de recherche permet de faire des recherches ciblées notamment sur les agents, les arrêts de travail, les accidents de travail, etc....



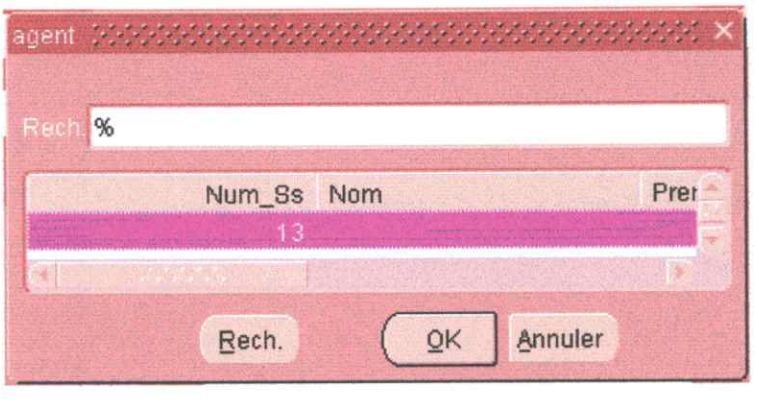
En détaillera cette page et son fonctionnement dans ce qui va suivre :



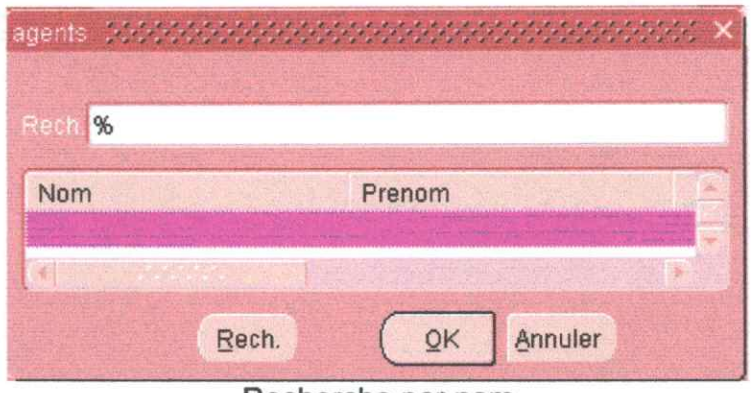
On choisissant l'anglet agent la figure suivante apparaît, et on pourra faire des recherche concernant les agents, comme recherche par matricule ou par numéro de sécurité sociale, etc..



Recherche par matricule.



Recherche par numéro de sécurité sociale.



Recherche par nom

3. conclusion :

Dans cette phase nous sommes arrivés à faire une application web pour le service prestations sociales qui fait le suivi et la gestion du personnelles de direction en appliquant les techniques d'accès aux bases de données ,et on utilisant les outils de développement tel que Oracle9i, Oracle IDS, JSP,etc....

Conclusion générale :

Notre travail ayant pour objectif la conception et la réalisation d'une base de données pour le service prestations sociales fonctionnant sur le réseau Internet et intranet, nous avons réalisé l'application en utilisant une architecture dite architecture trois-tiers, et une approche composant comme technique d'accès a la base de données a partir d'un serveur d'applications

A partir de ce serveur d'applications nous avons élaboré des composants JavaBeans qui effectuent des tâches bien spécifiques telle que charger, enregistrer des informations dans un serveur de données, faire une connection persistante a la base de données, etc.... puis nous avons entamé l'étape d'analyse qui nous a permis d'aboutir aux résultats attendus.

De l'approche méthodologique et du langage de modélisation adopté nous avons réalisés une base données au moyens de la méthode OMT et du langage UML avec le SGBD oracle 9i sous l'environnement Windows. Concernant la partie présentation nous avons utilisés les JSP et Oracle IDS

Grâce à la technique d'accès aux données a partir d'un serveur d'applications par l'utilisation des entités JavaBeans nous avons obtenu les résultats escomptés. Ces résultats ont été soumis à l'appréciation des responsables concernés de la direction générale de SONATRACH et nous espérons qu'ils constitueront des arguments valables pour qu'il soit intégré dans leur système.

Annexes

A. Internet

B. Le SQL

C. Présentation de Sonatrach

1 - INTRODUCTION

Contrairement aux idées reçues, Internet n'est pas un phénomène récent. C'est en 1969 que le gouvernement américain (plus précisément le DOD : Department Of Defense, soit le ministère de la défense) décide de mettre en place un système (Arpanet) pour relier les ordinateurs entre eux à des fins de défense nationale. En 1986, la NSF (National Science Foundation) et la NASA décident d'interconnecter les universités américaines. Très rapidement, ce système sera étendu à de nombreuses universités étrangères. Depuis 1992, Internet est devenu accessible au grand public.

Internet repose sur trois principaux types de services :

- La messagerie (e-mail ou mél en français) qui permet l'échange de documents et courriers électroniques. Un aspect intéressant de la messagerie est la possibilité de s'inscrire à une mailing list, c'est à dire émettre et recevoir des messages à une communauté de personnes ayant des centres d'intérêt communs.
- Les forums de discussions (newsgroups) qui permettent à plusieurs individus de se regrouper autour d'un même thème.
- Le Web permet d'accéder à des pages via un navigateur (browser). Ces pages contiennent du texte, des données, des fichiers, des images, du son, ...

Depuis son début, Internet connaît une croissance très forte. Le nombre de serveurs a presque doublé sur un an (source : http://www.nw.com/zone/host-count-history), passant de 19 à 38 millions en juillet 1998. Selon IDC, les dépenses liées au développement de sites passeront de 211 milliards de dollars cette année à un montant estimé de 954 Milliards de dollars en 2002. Les dépenses spécifiques au commerce en ligne passeront, toujours sur la même période, de 17 à 167 Milliards de dollars. Enfin, selon le groupe Forrester (http://www.forrester.com), le nombre de sites doublerait tous les 53 jours.

Toutes les activités informationnelles de la vie courante peuvent être réalisées sur Internet :

Consulter des bibliothèques, des rapports, des études, voir des expositions, visiter des écoles, contacter des experts, écouter la radio, voir la télévision, lire un journal ou un livre, "rencontrer" des gens et échanger des idées, ... L'information sur Internet est très, voire trop abondante.

2 - OUVERTURE A INTERNET

Internet, et son petit frère à l'échelle de l'entreprise intranet, deviennent aujourd'hui des éléments incontournables du système d'information. Le succès récent de cette technologie relativement ancienne est du à l'apparition du World Wide Web (WWW) en 1989, permettant à un utilisateur d'accéder au réseau Internet à travers un navigateur (browser). Ce dernier lui permet de visualiser les informations sous une forme ergonomique, sans avoir besoin de connaissances en informatique.
Les pages visualisées ne sont pas stockées sur le poste client mais sont envoyées, à la demande, par un serveur Web. Elles sont écrites en langage HTML (Hyper Text Markup Langage) et contiennent du texte formaté, des liens vers d'autres documents ou d'autres parties de la page présentée et des images.

2.1 - ACCES A DES PAGES STATIQUES

Affichage de rapports statiques. L'outil d'aide à la décision doit permettre de générer les rapports sous une forme HTML, afin de les visualiser depuis un navigateur et de les mettre à disposition des autres utilisateurs, par exemple sur un serveur d'entreprise, Intranet. Les pages définies sont alors des pages statiques, contenant à la fois la présentation et les données et non mises à jour dynamiquement. Afin de présenter aux utilisateurs des informations récentes, l'outil doit permettre de régénérer automatiquement les rapports à intervalles réguliers (par exemple tous les jours) ou après chaque chargement de données dans le Data Warehouse.
Automatisme du lien entre l'outil d'aide à la décision et le serveur Web. Le serveur Web devient ici un serveur d'entreprise, permettant simplement de stocker des documents et de les mettre à la disposition de toute personne possédant un navigateur Internet et ayant le droit d'accéder à ces pages. Pour que cette solution soit viable, il faut que l'outil soit capable de mettre automatiquement les pages HTML à disposition sur le serveur Web.
Même si elle apparaît sommaire, cette fonctionnalité peut être assez intéressante. Par exemple, dans le cadre d'une entreprise et d'un réseau intranet, les rapports élémentaires peuvent être mis à la disposition de l'ensemble des utilisateurs qui n'ont

ainsi pas à maîtriser et à employer un outil d'aide à la décision pour accéder aux informations qui les intéressent.
Mais, au delà de ces fonctionnalités statiques, il est également nécessaire d'accéder directement aux données à travers le navigateur et de générer des pages HTML dynamiquement, à la demande de l'utilisateur.

2.2 - ACCES A DES PAGES DYNAMIQUES

Au delà de l'envoi de pages statiques, le serveur Web est aujourd'hui capable de créer dynamiquement des pages, à la demande de l'utilisateur. Ceci peut se faire à travers des scripts CGI (Common Gateway Interface). Ils vont alors se charger d'interroger la base de données. Des interfaces plus évoluées sont proposées par Netscape avec NSAPI et par Microsoft avec ISAPI. Elles sont plus performantes et, contrairement à CGI, ne nécessitent pas la création d'un processus séparé à chaque exécution de script. Par exemple, ISAPI fait appel à des DLLs et non à des exécutables. Plus performantes, elles sont cependant moins fiables car un problème survenant lors de l'exécution d'un script risque de provoquer l'arrêt du processus qui est alors également celui du serveur Web (on parle de démon HTTP).
D'autre part, le langage Javascript permet de joindre des programmes à des pages HTML, afin de soulager le serveur et d'exécuter certains programmes au niveau client. Il peut s'agir, par exemple, de programmes permettant de ontrôler si l'utilisateur a bien renseigné tous les champs obligatoires dans un formulaire, avant de l'envoyer vers le serveur, afin d'éviter des aller et retours inutiles.
Enfin, le langage Java permet de créer de petites applications (appelées des applets) qui pourront être chargées directement sur le poste client et exécutées à partir du navigateur, il faut pour cela que ce dernier soit compatible Java.
Au delà du simple partage de rapports, la génération de pages dynamiques est une caractéristique indispensable. L'utilisateur doit pouvoir formuler ses requêtes et récupérer les résultats à travers son navigateur Internet.
D'autre part, il est nécessaire de lui laisser manipuler les données, par exemple dans le cas d'outils permettant d'effectuer de l'analyse multidimensionnelle, de naviguer dans les données.

Au delà d'applications " clé en main " mises à la disposition de l'utilisateur et lui permettant de manipuler les données dans le cadre qui lui a été imparti, ces outils devraient permettre à l'utilisateur de définir ses requêtes aussi librement qu'il le fait avec l'outil, de même pour la valorisation des résultats. Ceci permet alors de mettre à la disposition de l'ensemble des utilisateurs les données de l'entreprise, évite les coûts et les efforts d'installation et de mise à niveau des produits, l'application étant alors basée sur le serveur.

1. Présentation de SQL

SQL signifie Structured Query Language c'est-à-dire Langage d'interrogation structuré.

En fait SQL est un langage complet de gestion de bases de données relationnelles. Il a été conçu par IBM dans les années 70. Il est devenu le langage standard des systèmes de gestion de bases de données (SGBD) relationnelles (SGBDR).

C'est à la fois :

_ un langage d'interrogation de la base (ordre SELECT)

_ un langage de manipulation des données (LMD; ordres UPDATE, INSERT, DELETE)

_ un langage de définition des données (LDD ; ordres CREATE, ALTER, DROP),

_ un langage de contrôle de l'accès aux données (LCD ; ordres GRANT, REVOKE).

Le langage SQL est utilisé par les principaux SGBDR: DB2, Oracle, Informix, Ingres, RDB,... Chacun de ces SGBDR a cependant sa propre variante du langage. Ce support de cours présente un noyau de commandes disponibles sur l'ensemble de ces SGBDR, et leur implantation dans Oracle Version 7.

2. Normes SQL

SQL a éte normalisé dès 1986 mais les premières normes, trop incomplètes, ont été ignorées par les éditeurs de SGBD.

La norme actuelle SQL-2 (appelée aussi SQL-92) date de 1992. Elle est acceptée par tous mais les SGBD relationnels qui dominent actuellement le marché (en particulier Oracle) ne sont toujours pas totalement adaptés à cette norme.

SQL-2 définit trois niveaux :

_ Full SQL (ensemble de la norme)

_ Intermediate SQL

_ Entry Level (ensemble minimum à respecter pour se dire à la norme SQL-2)

3. Utilitaires associés

Comme tous les autres SGBD, Oracle comprend plusieurs utilitaires qui facilitent l'emploi du langage SQL et le développement d'applications de gestion s'appuyant sur une base de données relationnelle. En particulier SQL-FORMS facilite grandement la réalisation des traitements effectués pendant la saisie ou la modification des données en interactif par l'utilisateur. Il permet de dessiner les écrans de saisie et d'indiquer les traitements associés à cette saisie. D'autres utilitaires permettent de décrire les états de sorties imprimés, de sauvegarder les données, d'échanger des données avec d'autres logiciels, de travailler en réseau ou de constituer des bases de données réparties entre plusieurs sites.

Ce cours se limitant strictement à l'étude du langage SQL, nous n'étudierons pas tous ces utilitaires. Nous verrons les commandes essentielles d'un seul utilitaire, SQLPLUS, qui facilite l'utilisation interactive du langage SQL par un utilisateur. Ces commandes permettent de modifier les ordres SQL et de constituer des fichiers de commandes SQL que l'on peut réutiliser ensuite.

Les commandes du langage SQL peuvent être tapées directement au clavier par l'utilisateur ou elles peuvent être incluses dans un programme écrit dans un langage de troisième génération (Cobol, Langage C, Fortran, Ada,...) grâce à un précompilateur fourni par Oracle.

4. SGBD Oracle

Oracle est un SGBD (système de gestion de bases de données) édité par la société du même nom (Oracle Corporation - http://www.oracle.com/), leader mondial des bases de données.

La société *Oracle Corporation* a été créée en 1977 par Lawrence Ellison, Bob Miner, et Ed Oates. Elle s'appelle alors *Relational Software Incorporated (RSI)* et commercialise un Système de Gestion de Bases de données relationnelles (SGBDR ou RDBMS pour *Relational Database Management System*) nommé *Oracle.*

En 1979, le premier prototype (RDBMS - RSI1) intégrant la séparation des espaces d'adressage entre les programmes utilisateurs et le noyau Oracle est commercialisé. Cette version est entièrement développée en langage assembleur. La seconde version (RDBMS - RSI2) est un portage de l'application sur d'autres plates-formes.

En 1983 la troisième version apporte des améliorations au niveau des performances et une meilleure prise en charge du SQL. Cette version est entièrement codée en langage C. A la même époque RSI change de raison sociale et devient *Oracle*.

En 1984 la première version d'Oracle (Oracle 4) est commercialisée sur les machines IBM.

En 1985 Oracle 5 permet une utilisation client-serveur grâce au middleware *SQL*Net*.

En 1986 Oracle a été porté sur la plateforme 8086.

En 1988 Oracle 6 est disponible sur un grand nombre de plates-formes et apporte de nombreuses nouvelles fonctionnalités ainsi qu'une amélioration notable des performances.

En 1991, Oracle 6.1 propose une option *Parallel Server* (dans un premier temps sur la DEC VAX, puis rapidement sur de nombreuses autres plates-formes).

En 1992, Oracle 7 sort sur les plates-formes UNIX (elle ne sortira sur les plates-formes Windows qu'à partir de 1995). Cette version permet une meilleure gestion de la mémoire, du CPU et des entrées-sorties. La base de données est accompagnée d'outils d'administration (SQL*DBA) permettant une exploitation plus aisée de la base. En 1997, la version Oracle 7.3 (baptisée *Oracle Universal Server*) apparaît, suivie de la version 8 offrant des capacités objet à la base de données

Oracle est écrit en langage C et est disponible sur de nombreuses plates-formes matérielles (plus d'une centaine) dont :

- AIX (IBM)
- Solaris (Sun)
- HP/UX (Hewlett Packard)

- Windows NT (Microsoft)

Oracle depuis la version 8.0.5 est disponible sous Linux

Les versions d'Oracle

Oracle se décline en plusieurs versions

- Oracle Server **Standard**, une version comprenant les outils les plus courants de la solution Oracle. Il ne s'agit pas pour autant d'une version bridée...
- Oracle Server **Enterprise Edition**

Les fonctionnalités d'Oracle

Oracle est un SGBD permettant d'assurer :

- La définition et la manipulation des données
- La cohérence des données
- La confidentialité des données
- L'intégrité des données
- La sauvegarde et la restauration des données
- La gestion des accès concurrents
-

Les composants d'Oracle

Outre la base de données, la solution Oracle est un véritable environnement de travail constitué de nombreux logiciels permettant notamment une administration graphique d'Oracle, de s'interfacer avec des produits divers et d'assistants de création de bases de données et de configuration de celles-ci.

On peut classer les outils d'Oracle selon diverses catégories :

- Les outils d'administration
- Les outils de développement
- Les outils de communication
- Les outils de génie logiciel
- Les outils d'aide à la décision

Les outils d'administration d'Oracle

Oracle est fourni avec de nombreux outils permettant de simplifier l'administration de la base de données. Parmi ces outils, les plus connus sont :

- Oracle Manager (SQL*DBA)
- NetWork Manager
- Oracle Enterprise Manager
- Import/Export : un outil permettant d'échanger des données entre deux bases Oracle

Outils de développement d'Oracle

Oracle propose également de nombreux outils de développement permettant d'automatiser la création d'applications s'interfaçant avec la base de données. Ces outils de développement sont :

- Oracle Designer
- Oracle Developer
- SQL*Plus : une interface interactive permettant d'envoyer des requêtes SQL et PL/SQL à la base de données. SQL*Plus permet notamment de paramétrer l'environnement de travail (formatage des résultats, longueur d'une ligne, nombre de lignes par page, ...)
- Oracle Developper : il s'agit d'une suite de produits destinés à la conception et à la création d'applications client-serveur. Il est composé de 4 applications :
    - Oracle Forms (anciennement SQL*Forms) : un outil permettant d'interroger la base de données de façon graphique sans connaissances préalables du

langage SQL. SQL*Forms permet ainsi de développer des applications graphiques (fenêtres, formulaires, ...) permettant de sélectionner, modifier et supprimer des données dans la base.

- Oracle Reports (SQL*ReportWriter) : un outil permettant de réaliser des états
- Oracle Graphics : un outil de génération automatique de graphiques dynamiques pour présenter graphiquement des statistiques réalisées à partir des données de la base
- Procedure Builder : un outil permettant de développer des procédures, des fonctions et des packages

Outils de programmation

Oracle dispose d'un grand nombre d'interfaces (API) permettant à des programmes écrits dans divers langages de s'interfacer avec la base de données en envoyant des requêtes SQL. Ces interfaces (appelées précompilateurs) forment une famille dont le nom commence par *PRO** :

- Pro*C
- Pro*Cobol
- Pro*Fortran
- Pro*Pascal
- Pro*PLI
- ...

3. Les commandes SQL

En SQL, les commandes sont dites de format libre. Cela signifie qu'il n'existe aucune règle indiquant qu'un mot donné doit commencer à une position particulière de la ligne. Cette manière d'écriture est plus lisible.

Quelques commandes SQL :

CREATE Table : permet de créer une table,
DROP Table : pour supprimer une table,
INSERT : en ajoutant des lignes aux colonnes, on encadre les valeurs par des apostrophes,
SELECT : permet d'afficher les données sélectionnées,
UPDATE : pour modifier la valeur d'une donnée,
DELETE : pour supprimer une enregistrement,

Des conditions sont associées à ces commandes grâce à la clause **WHERE**, elle est utilisée pour retrouver les enregistrements qui répondent à une certaine condition. Cette condition est appelée simple. En connectant plusieurs conditions simples par l'utilisation des opérateurs : **AND**, **OR** et **NOT**, on aura les conditions combinées.
La clause **IN** est une manière concise de formuler certaines conditions.

L'opérateur **BETWEEN** n'est pas une fonctionnalité essentielle de SQL, mais il rend toutefois certaines commandes SELECT plus simples.
Dans la plupart des cas, les conditions nécessitent une concordance exacte. Pour cela, l'opérateur **LIKE** avec un caractère joker est utilisé.

L'ordre des lignes dans une table est sans importance dans un SGBD. D'un point de vue pratique, lors de l'interrogation d'une base de données relationnelle, le résultat s'affiche sans ordre prédéfini. Les lignes peuvent s'afficher dans l'ordre choisis spécifié à l'aide de la clause **ORDER BY**.

SQL dispose de fonctions pour calculer :
la somme : en utilisant la fonction **SUM**,
la moyenne : grâce à la fonction **AVG**,
ainsi pour compter, on utilise **COUNT**,
et pour trouver la valeur du maximum et de minimum, en introduisant les fonctions **MAX** et **MIN** respectivement.
L'opérateur **DISTINCT** n'est pas une fonction, mais il peut néanmoins se révéler utile dans certaines situations, couplé avec la fonction **COUNT**.

La clause GROUP BY permet de grouper des données dans un ordre particulier puis de calculer des statistiques si nécessaire.
La clause HAVING est utilisée pour les groupes, en effet, elle fait pour les groupes ce que la commande WHERE fait pour lignes.

En SQL, on réalise la jointure entre tables en incluant une condition dans la clause WHERE de façon à s'assurer que les colonnes en concordance contiennent des valeurs égales. Pour réaliser la jointure, on utilise des sous requêtes avec **IN** et **EXISTS**.

I. PRESENTATION GENERALE

I.1. Présentation de l'organisme d'accueil (SONATRACH)

I.1.1. Dénomination :

Société Nationale pour la recherche, la production, le Transport, la transformation et la Commercialisation des Hydrocarbures.

I.1.2. Forme juridique :

Entreprise publique et économique a sa création, elle est transformée, sans création d'une nouvelle personne morale, en une société par actions (SPA) par le décret présidentiel N°98-48 du 11 Février 1998.

I.1.3. Siége social :

Le siège social de SONATRACH est à Hydra ALGER, sis Djenane El Malik. Il peut être transféré en tout autre lieu du territoire national par délibération de l'assemblée générale.

I.1.4. Capital social :

SONATRACH dispose d'un capital social de **245.000.000.000 (deux cents quarante cinq milliards de dinars)** réparti en deux cent quarante cinq mille actions d'un million de dinars chacune, entièrement et exclusivement souscrit et libellé par l'état.

I.2. Historique de la SONATRACH

L'entreprise nationale SONATRACH (Société nationale de transport de transformation et de commercialisation des hydrocarbures) a été créée le **31/12/1963 (décret 63-491)** pour assurer la responsabilité de la production du transport et la commercialisation des hydrocarbures.

Les missions et les prérogatives de l'entreprise nationale SONATRACH ont été élargies le **22 septembre 1966 (décret 66-626)** ; ainsi ses missions qui se limitaient à l'origine au transport, la transformation et la commercialisation des hydrocarbures ont été élargies à tous les domaines de l'industrie pétrolière, à savoir la prospection, la recherche, la production, le transport, la transformation et la commercialisation des hydrocarbures.

Depuis le **24 février 1971**, date de nationalisation des hydrocarbures, l'entreprise a pris en charge l'ensemble du domaine minier et s'est vue confier le développement de toutes les branches de l'industrie pétrolière.

La SONATRACH est passée de 33 agents en 1964 à 103.000 vers la fin des années 80. Pour assurer une meilleure gestion et améliorer les performances dans le cadre de la politique nationale pour la réorganisation de l'économie du pays, elle entreprend sa restructuration pour donner naissance à 17 entreprises industrielles.

Actuellement, la SONATRACH compte un effectif de 36.000 agents environ et conserve pour sa part la charge des opérations de recherche, de production, de transport par canalisation, de traitement, conditionnement et liquéfaction des hydrocarbures liquides et gazeux.

Dans le cadre de la restructuration décidé en 1982, la SONATRACH a fait l'objet d'un découpage qui a donner naissance à treize entreprises parmi lesquelles, NAFTAL, NAFTEC, ENTP, ASMIDAL, ENSP, ...etc.

I.3. Missions principales de la SONATRACH

Sous l'autorité d'un Directeur Général, la SONATRACH a notamment pour missions essentielles :

- Le développement, la conservation et la valorisation des réseaux énergétiques sur tout le territoire national.

- La reconstitution et l'augmentation des réserves d'hydrocarbures.

- L'intensification des efforts d'exploitation et capitalisation des études réalisées dans ce domaine, pour une meilleure connaissance du sous-sol et la mise en évidence des réserves d'hydrocarbures.

- La diversification des marchés et des produits destinés a l'exportation.

- L'approvisionnement énergétique national à moyen terme, compte-rendu des réserves nationales.

- L'adaptation de l'outil commercial aux exigences du marché énergétique pour une meilleure maîtrise de ses mécanismes et des performances commerciales accrus.

- Le développement la maîtrise et la maintenance des complexes de production, de transport et de conditionnement des hydrocarbures.

- Le développement des techniques modernes de gestion nationale par le biais de la formation continue.

Figure 1-1 Organigramme de l'ensemble de l'entreprise de SONATRACH

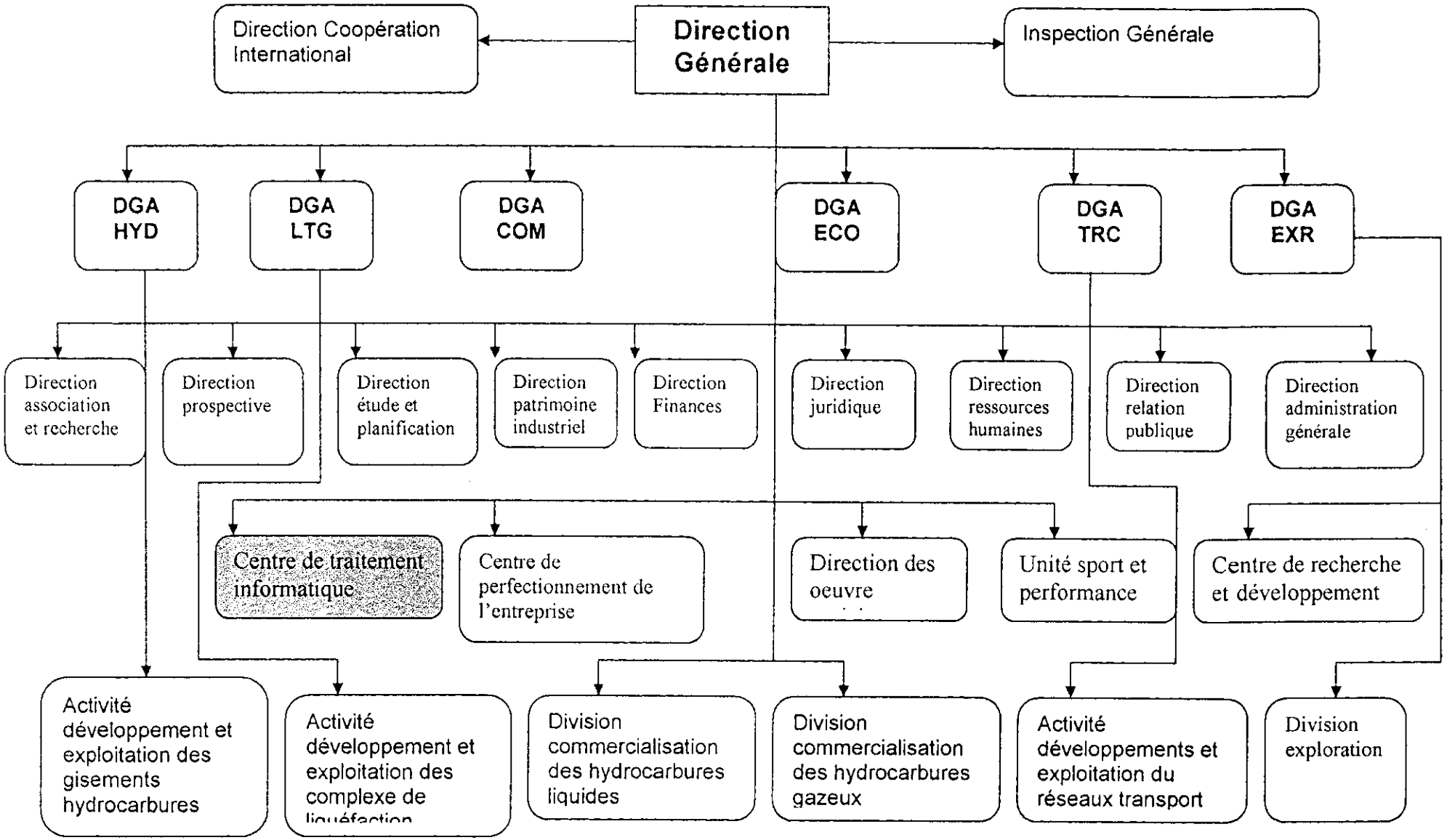
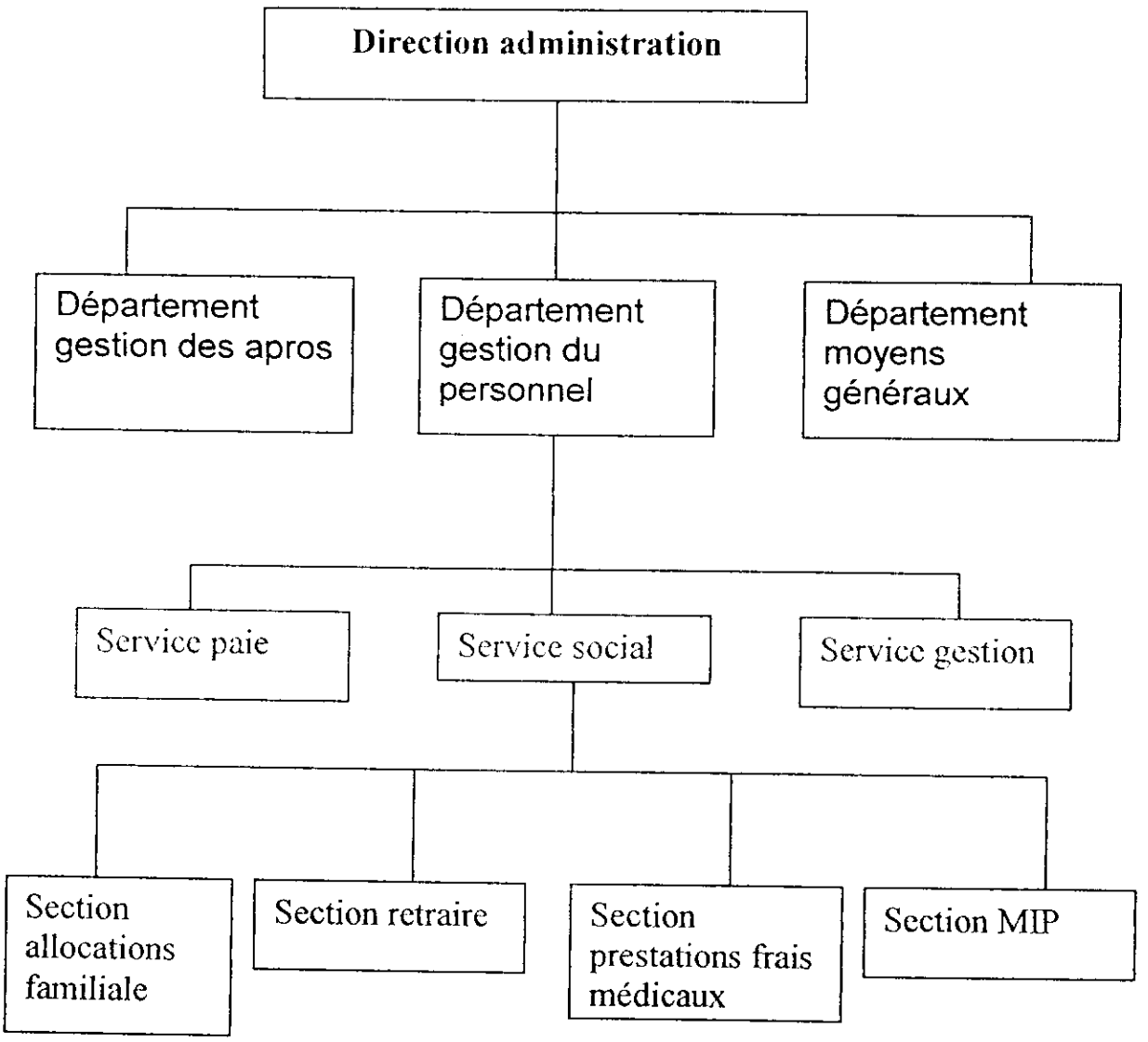


Figure 1-2 Organigramme de la structure d'étude



Note

Web[3.1]

WWW (World Wide Web): Encore appelé Web. En anglais, web signifie tissage ou toile d'araignée, en référence à la topologie du réseau Internet. Le CERN (Centre Européen de Recherches Nucléaires) fut l'acteur principal du WWW. Par la suite, le suivi du WWW fût pris en charge par le W3C (World Wide Web Consortium).

CGI[3.2]

CGI (Common Gateway Interface) : Nous verrons par la suite que cette interface qui présente l'avantage d'être standard et reconnue par tous, n'est pas très performante et tend à être progressivement remplacée.

SGML[3.3]

SGML (Standard General Markup Language) : Standard de description de document permettant de se concentrer sur le contenu plutôt que sur la forme.

TCP/IP[3.4]

HTTP n'est pas du tout lié au protocole TCP/IP et on pourrait envisager de faire fonctionner une version adaptée à un protocole comme IPX.

ISAPI[3.5]

ISAPI est l'API d'extension des serveurs Web de Microsoft.

NSAPI[3.6]

NSAPI est l'API d'extension des serveurs Web de Netscape.

contraignant[3.7]

Pas question, en effet, d'utiliser le click droit pour les applications devant fonctionner sur Macintosh, par exemple.

OAK[3.8]

OAK signifie chêne en anglais.

graphiques[3.9]

Les traditionnelles classes AWT, mais aussi, depuis la version 1.2 du JDK (Java Development Kit), les classes SWING permettant de réaliser des interfaces graphiques indépendantes du système hôte.

JDBC[3.10]

JDBC (Java Data Base Connectivity) : API fédérant la connexion des programmes Java à des bases de données.

RMI[3.11]

RMI (Remote Method Invocation) : Mécanisme d'appel de méthode distante utilisé par les programmes Java.

CORBA[3.12]

CORBA (Common Object Request Broker Architecture) : Architecture objet de l'OMG visant à l'intégration d'applications par la communication entre objets sur le réseau de l'entreprise.

Références bibliographiques :

[LEF97] Alain LEFEBURE. Eyrolles, 1997
"Intranet clent-serveur universel"

[ROO] Robert OGOR
"Modélisation avec UML".

[PRF03] Pascal Roques , Franck Vallée 2003
"UML en action, de l'analyse des besoins a la conception en java ".

[FAP00] Fabrice POPINEAU 2000
"Introduction a la conception orientée objet ".

[LAU96] Laurent HENOCQUE - Ecole supérieur d'ingénieur de luming, 1996
"Introduction a la méthode OMT "

[DKK01] Duane K. FIELDS, Mark A. KOLB 2001.
" JSP java server page "

[JMF97] Jean-Marie FAVRE 1997
"Java Beans"

[GER01] George Reese ,2001
" JDBC et JAVA Guide du programmeur "

[GIB03] Gilles Briard avec la collaboration de la société Digora, 2003
" Oracle 9i sous windows "

Les sites web:

www.sun.com

www.oracle.com

cedric.babault.free.fr

www.abcdoc.net

www.java-source.com

www.moteurprog.com

www.eunis.org

www.orsys.fr

www.eclipse.org

jakarta.apache.org

www.lesjsp.com

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique.

Université Saad Dahlab, Blida
USDB.

Faculté des sciences.
Département informatique.

Mémoire pour l'obtention
d'un diplôme d'ingénieur d'état en informatique.
Option : system d'information

Sujet :

Conception et réalisation d'une base de données prestations sociales accessible via le web en architecture trois-tiers

Présenté par : Mr. Zeraoui Sofiane
Mr. Benhalima Mohamed

Promoteur : Mr. A. Cherif Zahar

Organisme d'accueil : SONATRACH (Direction Générale)

Soutenance le : , devant le jury composé de :

Nom. Président du jury, grade, organisme	Président
Nom. Examinateur 1, grade, organisme	Examinateur
Nom. Examinateur 2, grade, organisme	Examinateur

2005/2006

REMERCIEMENT

Mes remerciements vont à tout ce qui ont contribué, de près ou de loin à mener à terme ce travail, notamment les enseignants qui m'ont instruit tout au long de mon long parcours.

Sans oublier notre promoteur Mr CHERIF ZAHAR pour ces orientations et conseils fructueux.

Je remercie chaleureusement tout le personnel de la SONATRACH et plus particulièrement
Mr TAYEBI responsable du service prestations sociales et
Mr A.DAIMALLAH responsable de la formation et de l'infographie.

DÉDICACES

J'ai le grand plaisir de dédier le fruit de mes années d'études aux êtres qui me sont les plus chers, mes parents.

A ma mère qui s'est données tant de mal pour moi et qui ma offert amour et soutien depuis mon premier souffle.

A mon père pour avoir mis tous les moyens a ma disposition et qui ma encouragé et soutenu pendant mes études

A mes chère sœurs et mon frère qui m'ont données chaque un une aide très précieuse a leurs façons.

Et toutes les personnes que j'aime et qui m'aime.

A mon binôme Mohamed.

A mes chères amis, Mr Goumari, youcef, tarik, azouaou, ibrahim, yacine, chaoukette, hamza noureddine, lyes, souad, keireddine, abdelhak, zaki, ali, mahfoud, et tout mes amis de la promo 2005/2006

Zeraoui Sofiane

DÉDICACES

Je dédie ce mémoire :

A mes très chers parents qui mon beaucoup soutenu et encouragé durant mes années d'études

A ma grand mère et mon grand père

A ma Sœur Djamila et ses fils Wissam et Nabila

A mes frères Zoubir et Zakaria

A mes amis

A tous mes amis de la promo 2005/2006

Benhalima Mohamed

Résumé :

Ce document et les annexes qui l'accompagnent constituent le résultat de notre travail qui a consisté en la conception et la réalisation d'une base de données accessible via le web

Après analyse des informations recueillies auprès du service prestations sociales de la direction générale de Sonatrach, nous avons utilisé OMT&UML comme méthodologie pour la conception de notre système dont l'objectif est de faire le suivi des dossiers social du personnel de la direction générale de SONATRACH en se servant de la technologie Internet.

Dans cette perspective nous avons utilisé l'architecture trois-tiers, que nous avons détaillée au chapitre 3 du présent document.

En effet, à partir d'un serveur d'applications on accède à la base de données par l'approche composant avec les entités JavaBeans.

Mots Clés : OMT, UML, Architecture trois-tiers, l'approche composant, JavaBeans.

Sum up:

This document and the enclosures which go with constitute the result of our work which consisted in the conception and the realisation of a data base accessible via the Web.

After the analyse of the collected information from the social insurance benefits service of the general management of SONATRACH, we have used the OMT and UML as a methodology for the conception of our system which objective is to trace the social files of the general management of SONATRACH using the Internet technology.

For this outlook we used the three-third architecture that we have detailed in chapter3 of the present document.

In deed, from an application sever we accede to the data base by the component approach with the JavaBeans entities

Key Words : OMT, UML, Architecture trois-tiers, l'approche composant, JavaBeans.

SOMMAIRE

Chapitre 3 : L'approche composant

Chapitre 4 : Analyse et conception du système

Chapitre 5 : Implémentation et résultats

Liste des figures :

Liste des tableaux

INTRODUCTION GENERALE

SONATRACH en tant que l'une des principales sociétés pétrolières dans le monde n'a pas voulu rester en marge des avancées technologiques dans le domaine de l'Internet. Compte tenu que l'entreprise soit éparpillée un peu partout sur le territoire algérien et du fait que ce n'est pas toujours facile de faire le suivi des dossiers du personnel surtout si on est toujours en déplacement. Le besoin d'une application web pour le service prestations sociales se fait ressentir. La mission est de mettre en place cette application dans intranet de la société reliant tout d'abord les sites de l'entreprise à Alger, puis la rendre présente sur le web.

Le web s'est développé comme un hypertexte sur le réseau Internet pour permettre facilement l'accès à des fichiers chaînés, rapidement le besoin de couplage avec les bases de données est apparu. Trois raisons au moins motivent ce besoin :

- ✓ L'introduction du clients-serveur à présentation universelle (architecture trois-tiers)
- ✓ La génération des sites web dynamique composés à partir de template HTML et de données extraites de bases.
- ✓ Le commerce électronique, qui nécessite la gestion de catalogues et de transactions en base de données.

Notre projet d'étude consiste en la conception et la réalisation d'une base de données prestations sociales, accessible via le web en architecture 3-tiers pour le siége de la SONATRACH.

Pour faire ce travaille, il existe plusieurs techniques d'accès aux bases de données à partir d'un serveur d'applications ,notamment , l'approche par page comme le PHP, l'approche par scriptlette avec les page JSP ou encore l'approche par composant utilisant les JavaBeans dans les pages JSP.

Dans cette perspective nous avons retenu pour notre projet cette dernière approche pour les raisons suivantes :

- profit des avantages de la portabilité et la puissance de java.
- Utilisation d'une technologie récente de conception de sites dynamiques à savoir le JSP (java Server page).

Notre mémoire s'articule essentiellement autour de cinq chapitres :

- Dans le premier chapitre sera présenté l'étude du domaine où la société souhaite améliorer son fonctionnement
- dans le deuxième chapitre, sera mis l'accent sur les system d'informations en architecture trois-tiers et les bases de données&web
- dans le troisième chapitre, sera présenté l'approche composant, et la persistance des objets java avec les entités Beans du J2EE.
- Dans le quatrième chapitre, sera présenté le langage de modélisation objet UML. Et une méthode de conception OMT, puis l'analyse et la conception du system.
- Dans le dernier chapitre Nous présenterons les langages de programmations utilisés ainsi que la description détaillée de notre application

Enfin, une conclusion général et des perspectives clôturant notre travail.

Chapitre 1

Etude de l'existant

1. INTRODUCTION

Toute l'imagination et tout le savoir-faire de l'analyste seront sans emploi, s'ils ne peuvent s'exercer sur une base réaliste.

Pour que l'application soit mise à la disposition de ceux qui ont besoin et à qui elle est destinée, l'étude de l'existant est le point de passage obligé qui matérialise le premier contact avec un domaine ignoré, c'est la première étape dans notre étude, elle nous permet de découvrir en détail le domaine où la société souhaite améliorer son fonctionnement.

Cette étude a pour but de nous permettre de:

1/ connaître en détail l'architecture Internet du siège SONATRACH.

2/ connaître les fonctions des services « prestation sociale » ainsi que les flux d'informations externe et interne

3/ voir en détail les postes de travails et les fonctions de service prestations sociales.

Avant d'être un réseau ou un ensemble de technologies, Internet est un standard mondial de communication

Il offre la possibilité

- de gagner en productivité dans la gestion interne et dans les relations interentreprises, et
- d'ouvrir aux entreprises des marchés nouveaux.

SONATRACH, groupe pétrolier International, étant la plus importante entreprise nationale, doit utiliser cette technologie pour ne pas être à la marge du progrès.

En effet, Les technologies Internet concernent toutes les fonctions de l'entreprise :

- ✓ vendre
- ✓ se faire connaître
- ✓ trouver des partenaires
- ✓ faire de la veille technologique et de l'intelligence économique
- ✓ transmettre des documents écrits, sonores ou vidéo

✓ conduire des projets, faxer, téléphoner, participer à des bourses, travailler en réseau....

2. ARCHITECTURE INTERNET

- **DMZ**

  Contenant les serveurs devant être perçus de l'extérieur.

- **Cisco PIX F**

  • Système (hard & soft) dédié sécurisé en temps réel,

  • Nombreuses options de connexion LAN :

  – Ethernet

  – Fast Ethernet

  – Token Ring ET FDDI

  • Support de 250.000 connexions simultanées.

- **Serveur proxy**

Avec l'utilisation du serveur proxy Les performances peuvent être encore améliorées ; L'enregistrement des sites WEB dans sa mémoire cache rendent la connexion plus rapide.

PROXY serveur tournant sur un port précis en attente de connexion cliente, capable d'accepter et de traiter plusieurs demandes de la part de plusieurs clients simultanément, Son rôle est de récupérer les informations sur les requêtes de ses clients et de les transmettre à PROXYREQUETE pour les traitements et le renvoie de la réponse.

- **routeur**

Internet résulte de l'interconnexion de différents réseaux physiques par des machines appelées routeurs. Chaque réseau, contenant un ensemble d'hôtes, est connecté, éventuellement, à un routeur. Le routeur permet de déterminer l'adresse physique de destinataire et de faire circuler les messages (paquets).

- **Switch**

Les ponts permettent d'étendre les possibilités du LAN : nombre de stations, distance, confidentialité, taux de défaillance. Un pont multiport est appelé commutateur (switch). Ils permettent également d'optimiser les débits. Les ponts sont des ordinateurs complets travaillant au niveau de la couche 3 et sont souvent multi protocoles. Un pont reçoit les trames circulant sur les segments raccordés, les analyses. Il récupère ainsi les numéros Ethernet des stations actives sur un segment. En fonction du destinataire, il émet ou rejette la trame. Un pont compte comme une station sur chaque segment. On mesure la qualité d'un pont par son taux de filtrage et son taux de transfert. Le maximum théorique de ces deux quantités est14 880 paquets de 64 octets par seconde.

La figure 1.1 illustre l'architecture Internet de siége Sonagraphe ; chaque Vlan correspond à une direction de siége SONATRACH c'est-à-dire architecturé en Vlan fonctionnels (Virtual Local Area Network) correspondants aux différentes structures qui y sont hébergées,

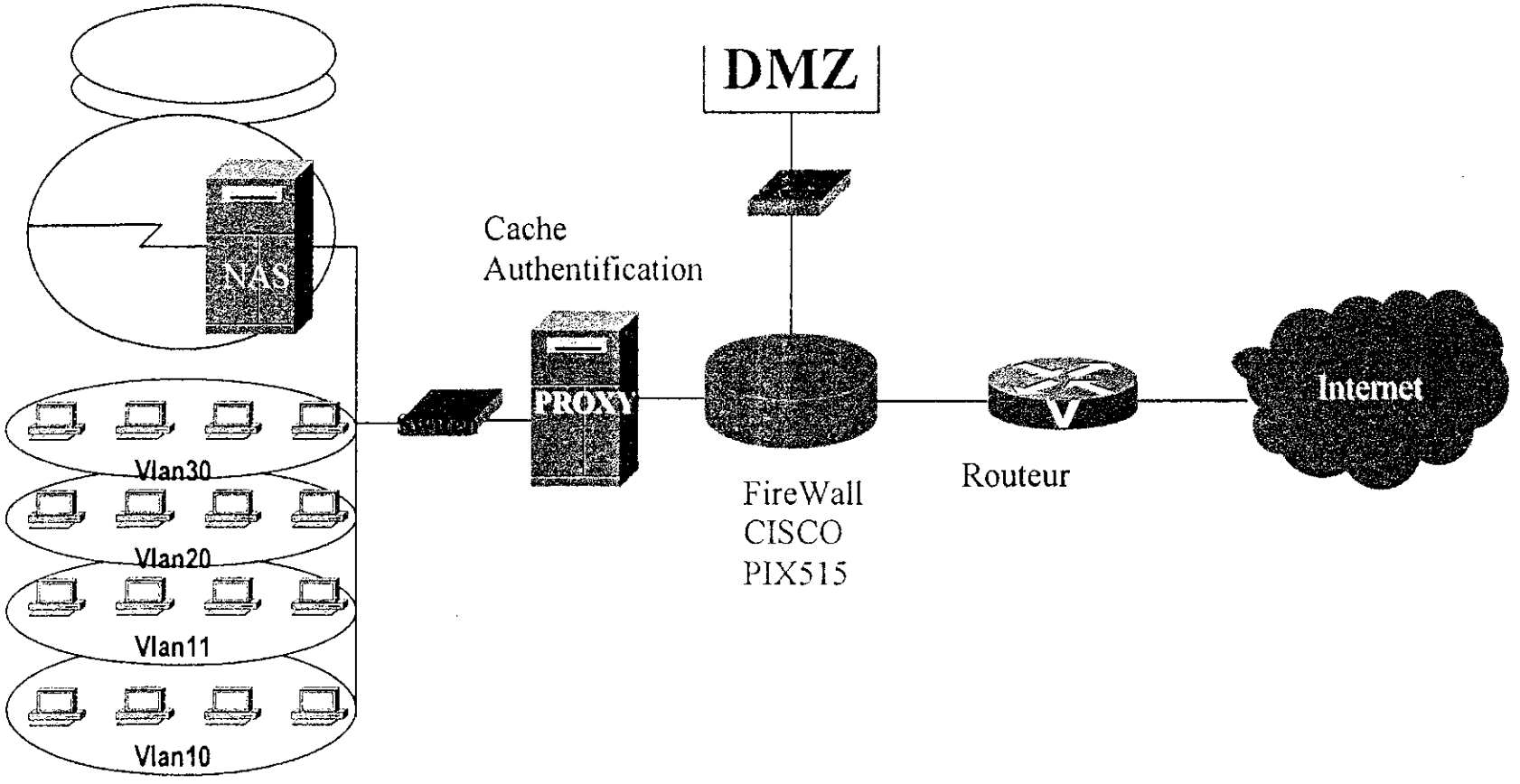


Figure 1.1. Architecture Internet de siége SONATRACH

3. Présentation de service prestation sociale :

Notre but est de mettre en place une application web en vue de la relation d'un system d'information pour le suivi des dossiers social du personnel de la direction générale de SONATRACH.

De ce fait l'application doit rependre aux besoins du service prestations sociales qui accompli entre autre les tâches suivantes :

- Les remboursements médicaux.
- Les allocations familiales
- La mutuelle
- La retraite.

3.1. Flux D'information externe :

3.1.1 Le Schéma de Flux D'information externes

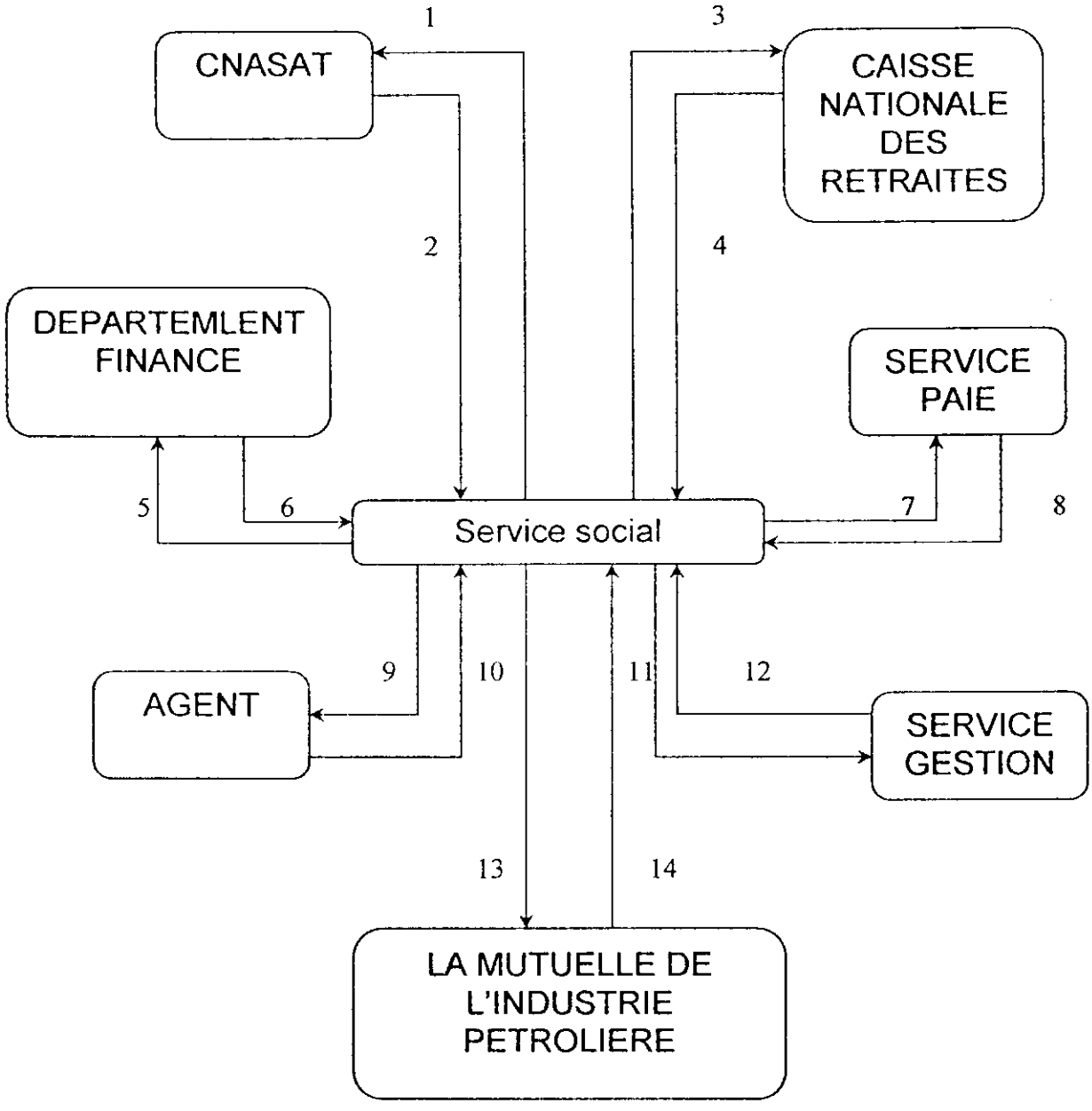


Figure 1.2 flux d'informations externe

3.1.2. Description du Flux d'informations Externes

N° DE LIAISON	DESIGNATION
01	- Envoi de la déclaration d'emploi à la CNAS. - Envoi des bordereaux d'exploitation à la CNAS. - Envoi des dossiers frais médicaux et arrêts de travail à la CNAS. - Envoi des bordereaux de paiement des allocations familiales, états des impayés et les bordereaux de réclamation à la CNAS. - Envoi des pièces de renouvellement et des situations nouvelles à la CNAS. - Envoi des dossiers indemnités journalières à la CNAS. - Envoi des lettres types à la CNAS.
02	- Réception des bordereaux d'exploitation. - Réception des cartes d'assurance. - Réception des chèques barrés. - Réception des décomptes de la CNAS. - Réception des récapitulations de la CNAS. - Réception des bordereaux de paiement des allocations familiales et des bordereaux d'exploitation (indemnités journalières ou remboursement des frais médicaux) avec accusés de réception.
03	- Envoi des dossiers de mise en retraite à la CNR ainsi que la lettre type.
04	- Réception de la notification de la CNR.

05	- Envoi des bordereaux de paiement internes au département finance. - Envoi des chèques barrés au département finance. - Envoi des récapitulations au département finance. - Envoi des bordereaux de versement des cotisations au département finance.
06	- Réception des notes internes du département finance ainsi que les lettres types.
07	- Envoi des bordereaux de paiement internes au service paie.
08	Réception des journaux de paie du service paie.
09	- Envoi des cartes d'assurance aux intéressés. - Envoi des décomptes à l'agent. - Envoi des convocations à l'agent.
10	- réception des dossiers frais médicaux et arrêts de travail. - réception des dossiers allocations familiales. - Réception des fiches familiales, certificats de scolarité, bulletin de naissance, extrait de décès et une copie de jugement. - Réception de mise en retraite des agents. - Réception des pièce : facture cliniques, acte des mariage, prescription médicale pour la circoncision.
11	- envoi de la photocopie de la notification au service de gestion.
12	- Réception d'une fiche de renseignement du service de gestion. - Réception des décisions du service de gestion.
13	- envoi des document MIP + décomptes CNAS + les pièces + bordereaux nominatif de prestation a la MIP. - Envoi de chèque barré et du bordereau de versement de

	cotisation a la MIP.
14	- réception de bordereau de versements de cotisation avec accusé de réception.

Tableau 1.1 Flux d'informations Externes

3.2. Flux d'information interne :

3.2.1 Schéma de flux d'informations internes

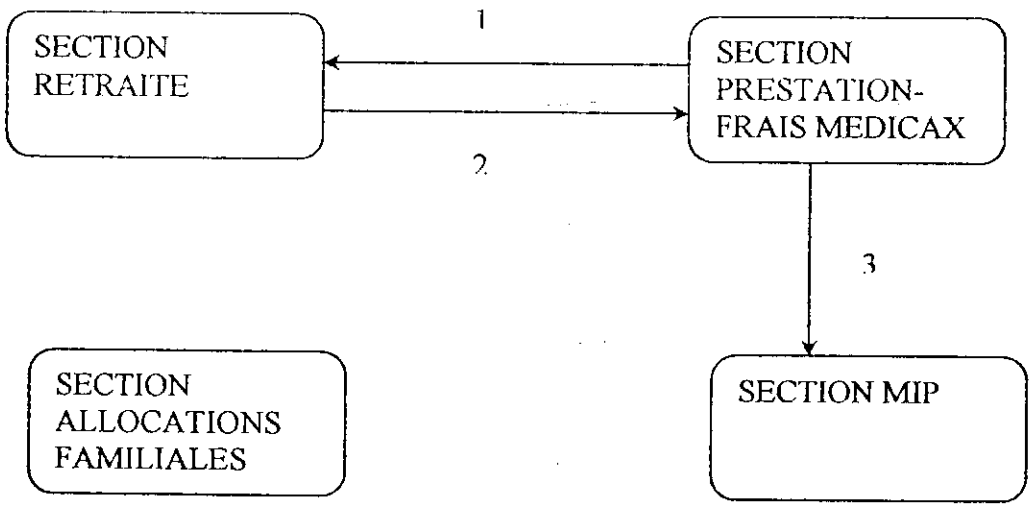


Figure 1.3 flux d'information interne

3.2.2. DESCRIPTION DU FLUX D'INFORMATIONS INTERNES

N^0 DE LIAISON	DESIGNATION
1	- envoi des bordereaux d'exploitation des arrêts de travail a la section retraite. - Envoi des déclarations d'accidents de travail et des bordereaux d'exploitation avec accusé de réception.
2	- Envoi des arrêts de travail a la section prestation frais médicaux. - Envoi des déclarations d'accident de travail, des infirmités et des questionnaires a la section prestations frais médicaux.
3	- Envoi des décomptes frais médicaux a la section *MIP*

Tableau 1.2 Flux d'informations internes

3.2.3. Fiche de Fonction des postes de travail :

Fiche de Fonction du Poste 01 :

Désignation : chef section prestations / assurance sociale

Section : retraite

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribuées

- indemnités journalières
- retraite
- capitale décès

b- tache à accomplir :

N°	Tâches	fréquences	remarque
1	- consultation des butins de paie des agents	-1fois/mois	-envoyer par le service de paie
2			
3	- exploitation des arrêts de travail	-1fois/mois	
4	- établissement des dossiers des indemnités journalières	-1fois/mois	
5	- Remplissage des fiches médicales	-1fois/mois	
6	- Etablissement des bordereau e paiement des prestation	-1fois/mois	-en 3exemplaires
7	- envoi des dossiers des indemnités journalières	-1fois/mois	
8	- mise à jour des fiches médicales	-1fois/mois	
9	- Envoi de la récapitulation et du chèque et des décomptes au service finance	-1fois/mois	
10	- établissement des bordereaux d'envoi pour les envoyer au service paie	-1fois/mois	
11	- recensement des agents partant en retraite	- 1fois/an	
12	- Envoi des convocations aux agents partant en retraite	- 1fois/an	
13	- envoi des dossiers de la mise en retraite à la CNR	- 1fois/an	
14	- Envoi d'une note interne et d'une notification au service GESTION	-irrégulier	

15	- aviser la CNR de la cessation de travail des retraités - Positionnement des retraités ainsi que tous les renseignements les concernant sur le registre du suivi personnel mis en retraite	-irrégulier -irrégulier	

Tableau 1.3 Fonction du Poste 01

Fiche de Fonction du Poste 2 :

Désignation : charger de gestion

Section : MIP

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribuées :

- remboursement MIP (20%)
- Cotisation

b- tâches accomplir :

N°	Tâches	fréquences	remarque
1	- établissement des décomptes de prestation	-1fois/semaine	
2	- établissements des bordereaux nominatifs de prestation	-1fois/semaine	
3	- envoi des décomptes et des bordereaux a la MIP	-1fois/semaine	
4	- établissement des bordereaux d'envoi	-1fois/mois	
5	- classement des décomptes	-1fois/mois	-par direction et par ordre alphabétique
6	- remplissage des fiches d'adhésion	-aléatoire	
7	- consultation des journaux de paie	-1fois/mois	
8	- établissement des bordereaux de versement de cotisation	-1fois/mois	
9	- envoi des chèques, bordereaux de cotisation et des bordereaux d'exploitation a la MIP	-1fois/mois	

Tableau 1.4 Fonction du Poste 02

Fiche de Fonction du Poste 3

Désignation : cadre de gestion administration

Section : prestation – frais médicaux

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.S)

– Fonction :

a- Tâches attribuées

- remboursements des frais médicaux
- immatriculation

b- tâches à accomplir :

N^0	Tâches	Fréquences	Remarques
1	- positionnements des dossiers sur le fichier du suivi des remboursements des frais médicaux	1/semaine	-Par N^0 SS
2	- établissement des bordereaux 'exploitation	1/semaine	-Par N^0 SS
3	- envoi des dossiers et des bordereaux a la CNAS	1/semaine	-Pour la collectivité
4	- classements des décomptes	1/semaine	-Pour la
5	- mise à jour du fichier du suivi des remboursements des frais médicaux	1/semaine	collectivité
6	- classements des décomptes par direction et par ordre alphabétique	1/mois	
7	- calcule du montant globale de chaque décompte	1/mois	
8	- établissement des bordereaux de paiements	1/mois	
9	- envoi des bordereaux de paiements aux services PAIE et FINANCE	1/mois	
10	- envoi des décomptes aux intéressés	1/mois	
11	- remplissage des fiches signalétique	Aléatoire	
12	- remplissage des déclarations d'emploi du travailleur	Aléatoire	
13	- envoi des déclarations d'emploi du travailleur a la CANS	1/semaine	
14	- positionnement des N^0 de sécurité sociale sur les fiches signalétiques	Aléatoire	
15	- envoi des cartes d'assurance aux intéressés	Aléatoire	

Fiche de fonction du poste 4

Désignation : cadre de gestion allocation familiale

Section : allocation familiale

Service : social

Département : personnel siége (D.P.S)

Direction : administration générale (D.A.G)

- Fonction :

a- tâches attribués :

- s'occupe des allocations familiales

b- tâches à accomplir :

N^0	Tâches	Fréquences	Remarque
1	- envoi des convocations a l'intéressé	Aléatoire	
2	- établissements des attestations de salaire et de travail	Aléatoire	
3	- établissements des fiches d'allocations	Aléatoire	
4	- établissements des bordereaux d'exploitations	1/mois	
5	- établissements de bordereaux de réclamations ou additif	1/mois	
6	- établissement des notes internes	1/mois	
7	- envoi des bordereaux	1/mois	
8	- envoi des notes internes	Aléatoire	
9	- envoi des pièces de renouvellement et des situations nouvelles a la CNAS	1/mois	

Tableau 1.5 Fonction du Poste 04

4. Conclusion :

On a pu découvrir en détail les moyens et l'infrastructure où la société souhaite fonctionner notre application. Ensuite, on a mis en évidence les fonctions du service « prestations sociales » en définissant les postes de travail, les tâches qui lui sont attribuées, et les flux d'informations qui circulent entre eux.

Notre but dans le chapitre suivant est de détailler les systèmes d'information en architecture trois-tiers et les bases de données &web, car l'approche composant avec les JavaBeans s'applique dans le domaine des bases de données&web

Chapitre 2

Système d'information
&
Architecture trois-tiers

1. Introduction :

Dans ce chapitre nous allons exposer la notion des systèmes d'informations distribuées et l'architecture trois-tiers ainsi que les bases de données&web.

L'objectif premier d'un système d'information quel qu'il soit est de permettre à plusieurs utilisateurs d'accéder aux mêmes informations. Pour cela il faut donc regrouper les informations utilisées par l'entreprise. En terme technique, cela se traduit par la centralisation des données au sein d'une base de données. L'évolution des systèmes d'information s'est donc basée sur une meilleure subdivision entre les tâches à réaliser pour permettre l'exploitation de ces données par les utilisateurs finaux. Ceci permet de structurer plus efficacement les informations ce qui entraîne à la fois une meilleure organisation de l'entreprise et une meilleure efficacité technique. Cette subdivision a été facilitée par l'avènement des technologies orientées objets qui s'appliquent aussi bien au modèle client-serveur qu'au modèle Internet. Ces technologies permettent une séparation entre les différents composants du système. Il devient alors possible de réaliser de nouvelles architectures permettant la mise à disposition des informations sous différentes formes tout en diminuant les temps de développement. Ces technologies permettent également de faire collaborer une grande diversité de systèmes. On parle alors d'architecture distribuée. Il est ainsi possible de présenter des données en provenance d'un mainframe mélangées à des données en provenance d'un SGBDR, le tout étant affiché dans un browser sur la même page HTML. [LEF97]

2. Système d'information et l'architecture trois-tiers :

Tout système d'information nécessite la réalisation de trois groupes de fonctions: le stockage des données, la logique applicative et la présentation. Ces trois parties sont indépendantes les unes des autres: on peut ainsi vouloir modifier la présentation sans modifier la logique applicative. La conception de chaque partie doit également être indépendante, toutefois la conception de la couche la plus basse est utilisée dans la couche d'au dessus. Ainsi la conception de la logique applicative se base sur le modèle de données, alors que la conception de la présentation dépend de la logique applicative [LEF97]

2.1. Trois tâches importantes

- **Stockage et accès aux données**

Le système de stockage des données a pour but de conserver une quantité plus ou moins importantes de données de façon structurée. On peut utiliser pour cette partie des systèmes très variés qui peuvent être des systèmes de fichiers, des mainframes, des systèmes de bases de données relationnelles, etc. Le point commun entre tous ces systèmes est qu'ils permettent le partage des données qu'ils contiennent via un réseau. La méthode d'accès à ces données dépendra du type d'organisation de ces données. Dans le cas d'une base de données relationnelle, l'accès peut se faire par des API qui dépendent du langage et de l'environnement. Ainsi en JAVA l'accès se fait via JDBC, alors qu'en C++ il se fera à l'aide d'ODBC. Quel que soit l'API, le langage SQL est utilisé.

- **Logique applicative**

La logique applicative est la réalisation informatique du mode de fonctionnement de l'entreprise. Cette logique constitue les traitements nécessaires sur l'information afin de la rendre exploitable par chaque utilisateur. Les utilisateurs peuvent avoir des besoins très variés et évolutifs. Il devient alors nécessaire de permettre l'évolution du système sans pour autant devoir tout reconstruire. Cette partie utilise les données pour les présenter de façon exploitable par l'utilisateur.

Il convient donc de bien identifier les besoins des utilisateurs afin de réaliser une logique applicative utile tout en structurant les données utilisées.

- **Présentation**

La présentation est la partie la plus immédiatement visible pour l'utilisateur. Elle a donc une importance primordiale pour rendre attrayante l'utilisation de l'informatique. Son évolution a été très importante depuis les débuts de l'informatique. Depuis les terminaux en mode texte connectés à des mainframes jusqu'au HTML de nos jours en passant par les applications graphiques développées en client serveur, il y a eu beaucoup de chemin parcouru. Différents types d'interfaces demeurent intéressantes. En effet l'ergonomie d'un site Intranet HTML n'est pas forcément idéale pour tous les types d'applications. Il peut être intéressant de proposer plusieurs types d'interface pour une seule logique applicative. Par exemple une entreprise disposant d'un site de commerce électronique peut proposer un accès à la liste de ses produits sur Internet en HTML mais disposer d'une interface d'administration réalisée à l'aide d'une applet graphique [LEF97]

2.2. Principe de l'architecture trois-tiers

Le principe d'une architecture trois-tiers est relativement simple: il consiste à séparer la réalisation des trois parties vues précédemment (stockage des données, logique applicative, présentation). Nous avons déjà pu entrevoir la possibilité de séparer la conception de ces trois subdivisions, ici il s'agit de séparer leur implantation. Tout comme dans le client-serveur cette séparation signifie qu'il est possible de déployer chaque partie sur un serveur indépendant, toutefois cela n'est pas obligatoire. La mise en place de ce type d'architecture permet dans tous les cas une plus grande évolutivité du système. Il est ainsi possible de commencer par déployer les deux serveurs sur la même machine, puis de déplacer le serveur applicatif sur une autre machine lorsque la charge devient excessive. Les éléments permettant la réalisation classique d'un système en architecture trois tiers sont les suivants:

- système de base de donnée relationnel (SGBDR) pour le stockage des données
- serveur applicatif pour la logique applicative
- navigateur web pour la présentation

Il est important de remarquer que l'essentiel du travail de développement sera implanté au niveau du serveur applicatif. Le SGBDR nécessitera un travail d'administration surtout dans le cas d'une quantité de données importante. Le travail de conception de la base de donnée sera la pierre angulaire du système. En effet l'ensemble du développement s'appuiera sur cette conception. Le navigateur web nécessitera la programmation de code spécifique permettant de gérer l'affichage par ce navigateur. Ce code sera placé sur le serveur applicatif pour permettre une mise à jour sans nécessiter de nouveaux déploiements.

2.3. Avantages de cette architecture

Cette architecture se développe actuellement au sein des entreprises grâce aux nombreux avantages qu'elle présente. Malgré la différence évidente entre une architecture trois tiers et un système client-serveur (l'apparition d'un serveur pour la logique applicative), le système reste basé sur les technologies éprouvées détaillées précédemment (aspect relationnel et transaction). La logique applicative est déplacée au niveau du serveur d'application mais reste programmée à l'aide des mêmes technologies liées aux bases de données relationnelles. En particulier l'utilisation du langage SQL reste jusqu'à présent la solution la plus intéressante au niveau de la qualité logicielle. Elle présente à la fois une grande fiabilité, une bonne disponibilité, une excellente évolutivité,... Toutefois il faut prendre en compte deux facteurs importants: d'une part le choix du SGBDR (ils n'ont pas tous les même qualités), d'autre part la qualité des programmes utilisant la base de données (aussi bien au niveau de la conception que de la programmation). L'avantage principal d'une architecture multi-tiers est la facilité de déploiement. L'application en elle même n'est déployée que sur la partie serveur (serveur applicatif et serveur de base de données). Le client ne nécessite qu'une installation et une configuration minime. En effet il suffit d'installer un navigateur web compatible avec l'application pour que le client puisse accéder à l'application, ce navigateur étant par ailleurs souvent installé par défaut sur toutes les machines. Cette facilité de déploiement aura pour conséquence non seulement de réduire le coût de déploiement mais aussi de permettre une évolution régulière du système. Cette

évolution ne nécessitera que la mise à jour de l'application sur le serveur applicatif. Ceci est très important car cette évolutivité est un des problèmes majeurs de l'informatique. Le troisième avantage est l'amélioration de la sécurité. Dans un système client-serveur tous les clients accédaient à la base de données ce qui la rendait vulnérable. Avec une architecture multi-tiers l'accès à la base n'est effectué que par le serveur applicatif. Ce serveur est le seul à connaitre la façon de se connecter à cette base. Il ne partage aucune des informations permettant l'accès aux données, en particulier le login et le password de la base. Il est alors possible de gérer la sécurité au niveau de ce serveur applicatif, par exemple en maintenant la liste des utilisateurs avec leurs mots de passe ainsi que leurs droits d'accès aux fonctions du système. On peut même améliorer encore la sécurité par la mise en place d'une architecture réseau interdisant totalement l'accès au serveur de base de données pour les utilisateurs finaux. La mise en place de firewall correctement configuré permettra ceci. Dans le cas de la mise en place d'un workflow au sein de l'entreprise Kallisto, l'avantage le plus intéressant est sans aucun doute l'évolutivité du système. En effet les besoins actuels sont relativement vastes, toutefois la réalisation totale semble relativement longue et difficile à réaliser d'un seul coup. De plus des problèmes risquent de nécessiter une mise à jour répétée du système. Un autre aspect intéressant est la possibilité d'utiliser le système en Extranet, ce qui chez Kallisto permettra de faciliter la communication entre le siège social situé à Lyon et l'agence de Paris [LEF97]

3. Les bases de données et le web

3.1. La révolution Internet :

S'il est un phénomène qui a marqué le monde de l'informatique ces dernières années, c'est bien celui d'Internet.
Ce réseau mondial, créé en 1969 par l'armée américaine, puis utilisé par les chercheurs et autres scientifiques, a connu une croissance phénoménale auprès du grand public avec l'introduction du *World Wide Web*[3.1] en 1989. Ce dernier permet

de publier simplement des informations richement mises en forme et pouvant même, par la suite, contenir des données multimédia.

La véritable révolution du WWW réside dans son caractère universel, rendu possible par l'utilisation de standards reconnus.

3.1.1. Les standards d'Internet

L'universalité du Web repose sur des standards simples et admis par tous :

- HTML, pour la description des pages disponibles sur le Web,
- HTTP, pour la communication entre navigateur et serveur Web,
- TCP/IP, le protocole réseau largement utilisé par les systèmes Unix,
- CGI[3.2], l'interface qui permet de déclencher à distance des traitements sur les serveurs Web.

- HTML (HyperText Markup Langage)

HTML est le langage de description de pages hypertexte utilisé par le World Wide Web, il est issu de SGML[3.3].

Une page HTML est composée de son contenu propre (du texte plus ou moins richement mis en forme) et de références statiques vers d'autres sources d'informations (images, liens vers d'autres documents...).

La consultation d'une page HTML n'implique donc que très rarement le chargement du seul fichier décrivant la page, il s'accompagne en général de celui de nombreux fichiers annexes. Ces chargements mettent en oeuvre le protocole HTTP.

- HTTP

HTTP est un protocole réseau applicatif (dernier niveau du modèle OSI) sans connexion utilisé pour l'échange des données sur le Web. En fait, une connexion HTTP est créée pour chaque requête et ne dure que pendant l'exécution de cette dernière.

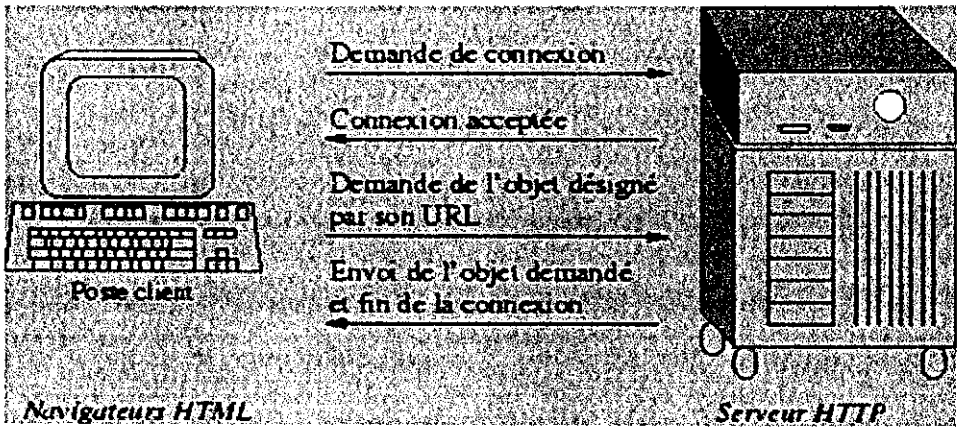


Figure 2.1: Le fonctionnement de base de HTTP[LEF97]

Le protocole HTTP, en temps que protocole réseau applicatif, s'appuie sur un protocole de transport indépendant qui, dans le cadre d'Internet, est TCP/IP[3,4].

- TCP/IP (Transmission Control Protocol / Internet Protocol)

TCP/IP est un protocole réseau de niveau trois et quatre (réseau et transport) qui s'est largement imposé sur les systèmes Unix, puis sur Internet, et fait aujourd'hui figure de standard universel.

Si le fonctionnement de TCP/IP s'adapte bien à la topologie et à la qualité de service du réseau Internet, on ne peut en dire autant du mariage avec le protocole HTTP. En effet, TCP/IP utilise un mécanisme de ``démarrage lent'' afin d'éviter les engorgements du réseau. Ce mécanisme permet à la machine émettrice d'ouvrir progressivement une fenêtre de congestion en doublant le nombre de paquets émis à chaque aller-retour. En général, la courte durée des échanges HTTP ne permet pas à la fenêtre de congestion d'atteindre la largeur de bande fournie par le réseau local [LEF97].

- **CGI (Common Gateway Interface)**

CGI est un standard permettant d'écrire des extensions compatibles avec la grande majorité des serveurs HTTP. Ces extensions permettent l'exécution d'une action par le serveur à la demande d'un client.

Ce mécanisme relativement simple, voire même rustique, entraîne l'exécution d'un processus propre à chaque invocation, ce qui est très consommateur de ressources.

De ce fait, des extensions comme ISAPI[3.5], NSAPI[3.6] ou les servlets Java sont souvent préférés au standard CGI.

3.1.2. Adaptation à l'entreprise : Intranet

Aucun des mécanismes mis en oeuvre par Internet n'est exempt de défaut et il est relativement simple de trouver plus performant. En fait, la force de l'ensemble repose essentiellement dans son universalité.

La notion d'Intranet est née de l'intégration des principes d'Internet et des technologies déployées dans l'entreprise :

- on utilise le réseau local de l'entreprise,
- les données sont toujours gérées par un SGBD,
- les mécanismes utilisés pour interroger le SGBD sont toujours les mêmes.

3.2. Répartition des traitements

L'architecture trois tiers, encore appelée client-serveur de deuxième génération ou client-serveur distribué, sépare l'application en trois niveaux de service distincts :

- **premier niveau :** l'affichage et les traitements locaux (contrôles de saisie, mise en forme de données...) sont pris en charge par le poste client,
- **deuxième niveau :** les traitements applicatifs globaux sont pris en charge par le service applicatif,

- **troisième niveau** : les services de base de données sont pris en charge par un SGBD.

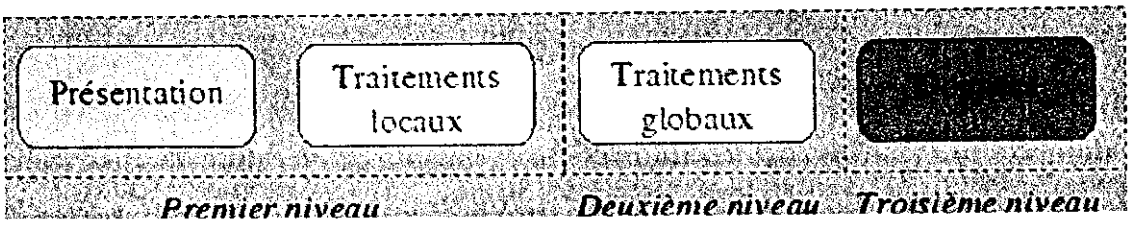


Figure 2.2: Le découpage d'une application en pavés fonctionnels indépendants

Tous ces niveaux étant indépendants, ils peuvent être implantés sur des machines différentes, de ce fait :

- le poste client ne supporte plus l'ensemble des traitements, il est moins sollicité et peut être moins évolué, donc moins coûteux,
- les ressources présentes sur le réseau sont mieux exploitées, puisque les traitements applicatifs peuvent être partagés ou regroupés (le serveur d'application peut s'exécuter sur la même machine que le SGBD),
- la fiabilité et les performances de certains traitements se trouvent améliorées par leur centralisation,
- il est relativement simple de faire face à une forte montée en charge, en renforçant le service applicatif.

Dans le cadre d'un Intranet, le poste client prend la forme d'un simple navigateur Web, le service applicatif est assuré par un serveur HTTP et la communication avec le SGBD met en oeuvre les mécanismes bien connus des applications client-serveur de la première génération.

Ce type d'architecture fait une distinction nette entre deux tronçons de communication indépendants et délimités par le serveur HTTP :

Le premier tronçon relie le poste client au serveur Web pour permettre l'interaction avec l'utilisateur et la visualisation des résultats. On l'appelle **circuit froid** et n'est composé que de standards (principalement HTML et HTTP). Le serveur Web tient le rôle de ``façade HTTP'',

- le deuxième tronçon permet la collecte des données, il est aussi appelé **circuit chaud**. Les mécanismes utilisés sont comparables à ceux mis en oeuvre pour une application deux tiers. Ils ne franchissent jamais la façade HTTP et, de ce fait, peuvent évoluer sans impacter la configuration des postes clients.

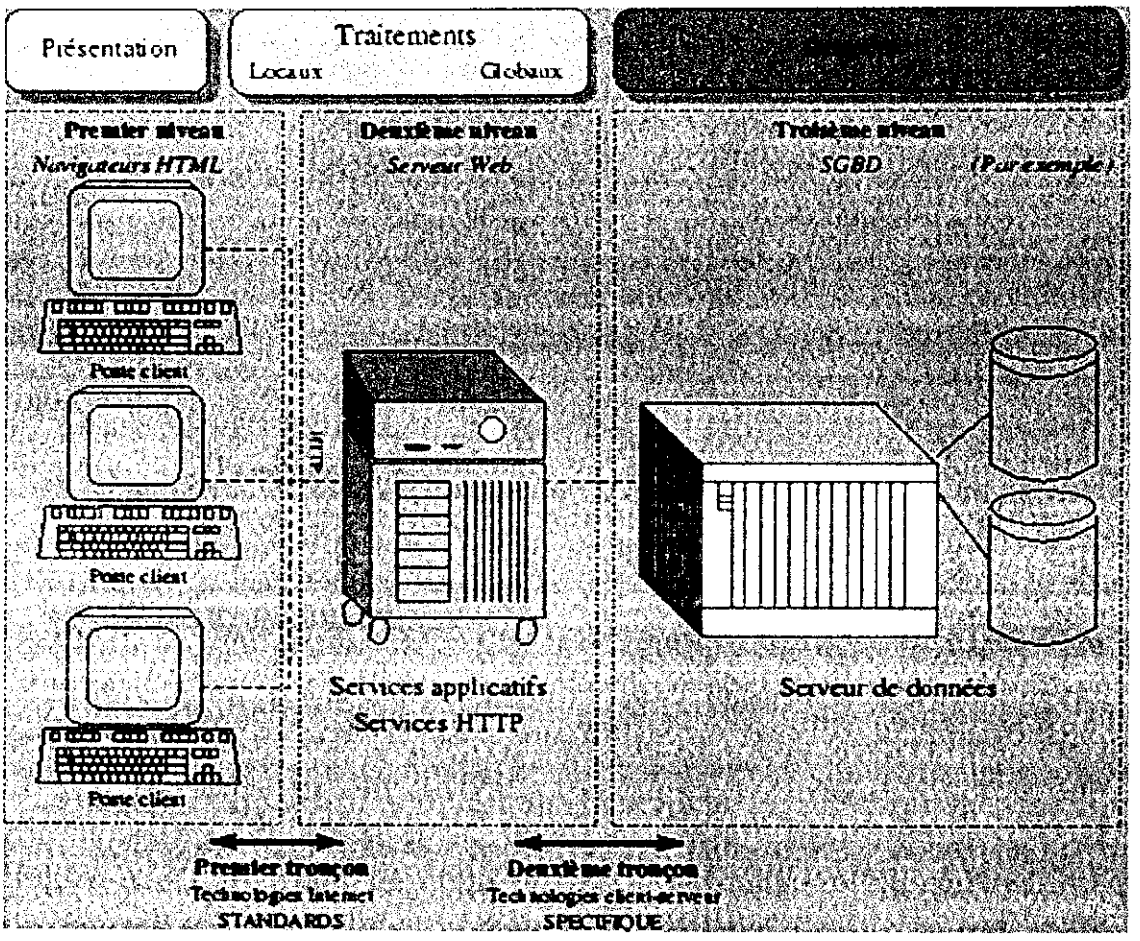


Figure 2.3: Répartition des couches applicatives dans une architecture trois tiers

3.3. Le client léger

3.3.1. Présentation

Dans l'architecture trois tiers, le poste client est communément appelé client léger ou *Thin Client*, par opposition au client lourd des architectures deux tiers. Il ne prend en charge que la présentation de l'application avec, éventuellement, une partie de logique applicative permettant une vérification immédiate de la saisie et la mise en forme des données. Il est souvent constitué d'un simple navigateur Internet.

Le poste client ne communique qu'avec la façade HTTP de l'application et ne dispose d'aucune connaissance des traitements applicatifs ou de la structure des données exploitées. Les évolutions de l'application sont donc possibles sans nécessiter de modification de la partie cliente.

Par exemple, un internaute se connectant à *www.yahoo.fr* pour effectuer une recherche provoque, sans même le savoir, l'exécution de traitements sur le serveur. Si ces traitements évoluent, ce qui doit arriver relativement souvent, le client continuera à utiliser le service sans se rendre compte des changements (sauf s'ils lui apportent de nouveaux services).

De plus, ce même internaute peut se connecter au serveur en utilisant tout type de poste client disposant d'un navigateur compatible HTML (PC sous Windows, Macintosh, Station Unix, WebPhone...).

On voit donc ici la force des architectures trois tiers par rapport au client-serveur de première génération. Le déploiement est immédiat, les évolutions peuvent être transparentes pour l'utilisateur et les caractéristiques du poste client sont libres.

3.3.2. Ergonomie

- Utilisation d'HTML

Les pages HTML, même avec l'aide de langages de script, sont loin d'atteindre les possibilités offertes par l'environnement Windows :

- le multifenêtrage n'est pas facile à mettre en oeuvre,
- le déroulement de l'application doit se faire séquentiellement,
- les pages affichées sont relativement statiques,
- le développement multiplateforme peut être contraignant[3.7].
- l'ergonomie de l'application est limitée aux possibilités du navigateur.

Pour ces raisons, certaines applications ne sont pas réalisables dans une architecture de type Intranet (infocentres, applications bureautiques...).

Dans les autres cas, l'appauvrissement de l'interface utilisateur est souvent le gage d'une plus grande facilité de prise en main de l'application.

Il est rare en effet de devoir suivre une formation pour apprendre à se servir d'un site Web particulier. La plupart du temps, une formation générale à l'ergonomie des sites Web suffit.

Cette perte de richesse peut donc se transformer en avantage, à condition de respecter une charte graphique et ergonomique cohérente pour toutes les applications Intranet d'une entreprise.

Il est possible d'aller au delà des possibilités offertes par le langage HTML en y introduisant des applets Java ou des contrôles ActiveX. Nous ne parlerons pas ici des modules d'extension du navigateur, encore appelés *plug-in*, qui étaient en vogue avant l'arrivée des solutions Java et ActiveX, car cette solution est trop contraignante à déployer.

- **Utilisation de Java**

Java est un langage de développement orienté objet et multiplateforme introduit par Sun en 1995. Il s'agit de l'adaptation à Internet du langage OAK[3.8], initialement étudié pour les environnements de petite taille. Il permet, entre autre, d'écrire de petites applications, appelées *applet*, pouvant être intégrées à des pages HTML pour en enrichir le contenu.

Le caractère multiplateforme de Java se prête bien à une utilisation sur Internet, où les caractéristiques des postes clients ne sont pas maîtrisées. Il repose sur l'utilisation d'un interpréteur de pseudo-code Java, pompeusement appelé ``machine virtuelle".

Les programmes Java ne sont pas compilés en code machine, mais en pseudo-code Java uniquement compréhensible par la machine virtuelle. Cette dernière interprète le code et se charge de lier les modules au moment de l'exécution, en les téléchargeant si nécessaire. La liaison dynamique des modules au moment de l'exécution permet d'optimiser le trafic réseau, puisqu'on ne charge que le strict nécessaire.

Pour ces raisons, un programme Java s'exécute plus lentement que son équivalent compilé. La compilation à la volée des programmes Java et plus encore, la technologie d'optimisation dynamique de code *HotSpot* de Sun, permettent de réduire l'écart de performance avec des programmes compilés.

Java propose aujourd'hui une large palette de composants graphiques[3.9] et multimédia qui permettent d'atteindre la richesse fonctionnelle des applications Windows.

Une applet Java est aussi capable d'exploiter directement un serveur de données en utilisant JDBC[3.10] ou de faire appel à des procédures distantes en utilisant RMI[3.11] ou CORBA[3.12]. Nous verrons ces mécanismes par la suite.

4. Conclusion

Etant donné que notre travail est basé sur les bases de données et le Web nous avons mis l'accent sur l'architecture trois-tiers et ses trois niveaux ainsi que l'environnement logiciel qui permettent a cette architecture de fonctionné sur le Web, pour la conception d'applications web et des pages dynamiques.
Notre but dans le chapitre suivant est d'étudier et de détailler l'approche composante avec les JavaBeans.

Chapitre 3

L'approche composant

1. Introduction :

Notre objectif dans ce chapitre est de présenter la réalisation technique de l'architecture trois-tires, et l'approche que nous avons adoptés pour la réalisation de notre application ; puis on verra en détail les composants JavaBeans et leurs fonctionnements ainsi que l'architecture orientée servlets.

2. Outils pour la réalisation de system d'information en architecture trois-tiers :

Nous avons donc vu ce qu'est une architecture trois-tiers ainsi que ses avantages. Nous allons maintenant présenter les choix techniques effectués pour la réalisation de cette architecture. En fait, il existe actuellement plusieurs solutions permettant ce type d'architecture. Ces technologies possèdent chacune leurs avantages et leurs inconvénients. On peut ainsi construire un système multi-tiers basé uniquement sur des technologies Microsoft comme ODBC, ASP, SQL Server... Ces technologies sont relativement efficaces mais ont pour principal inconvénient de n'être pas du tout portables. Kallisto utilise également des technologies spécifiques à Oracle basées sur Oracle Application Server, le serveur applicatif d'Oracle. Cette technologie est particulièrement efficace car elle permet de n'utiliser que des API natives à Oracle. Toutefois les systèmes développés avec ces outils ne pourront pas fonctionner avec autre chose. Cet inconvénient n'est pas obligatoirement important puisqu' Oracle fonctionne tout de même sur un grand nombre de systèmes (la plupart des UNIX et Windows). La solution choisie ici est basée sur le langage JAVA. Elle utilise une base de données Oracle, un serveur applicatif compatible J2EE nommé WebLogic, et un navigateur web comme client. Le JAVA étant un langage 100% portable, le développement du système ne dépendra absolument pas de la machine sur laquelle il fonctionnera. De plus WebLogic suivant les dernières spécifications de J2EE, les développements réalisés pourront fonctionner sur de nombreux autres serveurs applicatifs.

Oracle

Afin de stocker les données il fallait également une base de données. La société Kallisto étant spécialisée dans les bases Oracle, ce choix s'imposait. J'ai d'ailleurs pu profiter d'une formation interne dispensée la semaine de mon arrivée par un expert Oracle (10 ans d'expérience). Oracle est un système de gestion de base de données relationnelle très connu. Il est réputé pour être performant, fiable,... Oracle est en fait la référence en matière de base de données relationnelle. Les possibilités d'administration sont importantes et permettent de gérer des bases de tailles importantes (ce qui n'est pas primordial dans notre cas).

Java 2 Enterprise Edition (J2EE)

Le langage JAVA est un langage objet qui présente de nombreux avantages. Le premier d'entre eux est de supporter les notions de la programmation orienté-objet (encapsulation, héritage, classe, objet). Il permet une programmation relativement simple sans se préoccuper de notions complexes présentes en C++ (indirection de pointeurs, fonctions virtuelles,...). Mais ces intérêts sont relativement mineurs comparés à l'apport que représente les interfaces de programmation standardisés proposées par Sun. Ces interfaces, couplées à la compilation en un ByteCode exécutable sur une machine virtuelle, donnent aux applications un niveau de portabilité maximale. Dans notre cas nous sommes intéressés par le développement d'une application Intranet utilisant des accès à une base de données. Il existe en JAVA un grand nombre d'API destinées à réaliser ce type de programmation et regroupées sous le terme J2EE (Java 2 Enterprise Edition). Ces API forment ce qu'on nomme un Framework. Il s'agit de proposer des interfaces guidant le développeur vers une architecture prédéterminée. On profite donc à la fois d'un ensemble d'outils indispensables au développement mais aussi d'un guide pour l'élaboration d'une méthode de développement. Ce guide reste suffisamment flexible pour nous permettre d'adapter la structure de notre application à nos besoins. L'objectif majeur de J2EE est la réalisation d'applications en architecture distribuée. Les technologies JAVA intégrées dans cette plate-forme sont:

- Enterprise JavaBeans (EJB)
- Common Object Request Broker Architecture (CORBA[3.12])
- Java Servlets 2.1
- Java Server Pages 1.1 (JSP)
- Java Message Service (JMS)
- Java Transaction API (JTA)
- JavaMail 1.1
- Java Database Connectivity 2.0 (JDBC[3.10])
- Java Naming and Directory Interface 1.2 (JNDI)
- eXtensible Markup Language (XML)

3. l'approche composant :

Les composants sont des éléments indépendants et réutilisables qui encapsulent un comportement applicatif ou des données en un paquetage distincte. Tels des dispositifs de type <boites noire>, ils effectuent des opérations sans révéler leurs mécanismes internes. Comme une interface abstraite fait le lien entre le comportement applicatif et l'implémentation, il est ainsi épargné aux utilisateurs les détails complexes d'un code touffu. Les fonctionnalités apportées n'augmentent pas la complexité globale de l'application. En outre, les composants ne sont pas exclusifs à une application ni a une utilisation unique .ils utilisées comme des briques élémentaires dans des projets qui parfois,n'ont aucun rapport entre eux.
L'abstraction et la réutilisation sont les deux principes essentiels de la conception orientée composant. La figure 3-1 illustre cette approche en montrant comment on peut combiner des composants logiciels indépendants. [DKK01]

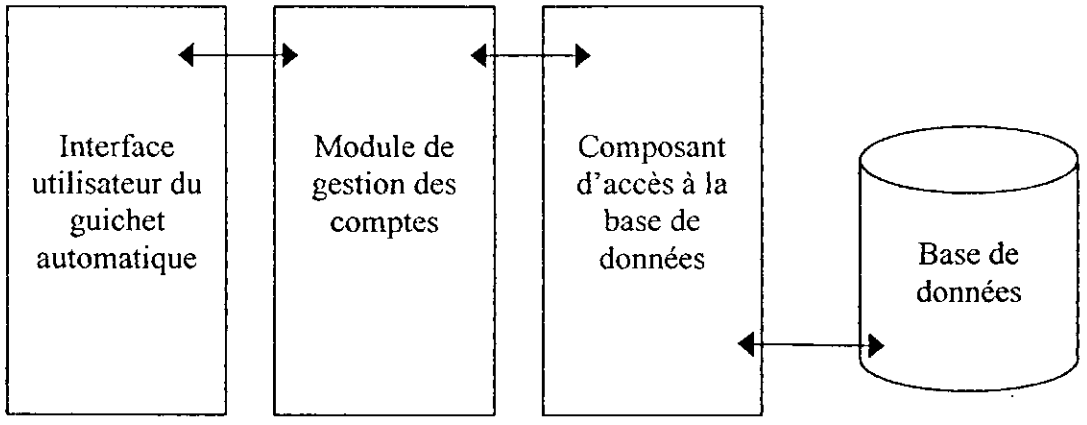


Figure 3-1 une application basée sur les composants

Les composants sont donc des éléments logiciels réutilisables qui peuvent être reliés pour construire une application. Un bon modèle de composants minimise le travail de codage des relations entre les différents éléments de l'application. Le principe des architectures de composants consiste à utiliser une interface commune pour manipuler des objets avec des outils de développement, la seule nécessité qui incombe à l'outille étant de prendre encharge l'interface.

3.1. Avantages des architectures à base de composants

Lorsqu'un architecte entreprend la conception d'une nouvelle maison, il se base sur des composants qui vent lui permettre tout à la fois de gagner du temps et de réduire le coûts engager, plutôt que de concevoir l'installation électrique à zéro, il va par exemple utiliser des composants déjà existants. Ainsi cet architecte ne va-t-il pas concevoir des systèmes de climatisations personnalisés, mais plutôt va-t-il les sélectionner à partir de modèles déjà disponibles sur le marché. Selon toute probabilité, cet architecte ne possédera pas le savoir-faire ni les ressource nécessaires qui lui permettraient de concevoir de tels systèmes. Inversement, le constructeur d'appareils de climatisation n'a aucun savoir-faire en construction de bâtiments. L'approche orienté composants permet à l'architecte et à l'entrepreneur de concentrer leurs efforts sur ce que, chacun de leur coté, savent le mieux faire. Il en va de même en informatique. Les architectures par composants offrent la possibilité de masquer la complexité des composants donnés par l'adjonction d'une interface, à travers laquelle le composant réagit avec son environnement ou avec d'autres composants.

On peut utiliser la même argumentation pour illustrer le concept de réutilisation et d'interchangeabilité. L'entreprise de construction a opté pour un climatiseur fixe par des écrous standard et fonctionnent sur une tension électrique standard. Le propriétaire de la maison pourra ainsi cet appareil par un nouveau modèle plus performant, et sans qu'il doive reconstruire toute la maison. La normalisation des environnements et des méthodes de conception a permis d'obtenir des systèmes souples, faciles à maintenir. Les composants logiciels ont été conçus pour opérer dans ces environnements spécifiques. L'existence de règles prédéfinies qui

régissent leur interaction avec les autres composants augmente d'autant leur interchangeabilité. [DKK01]

3.2. Fonctionnement des composants JavaBeans

Le composants JavaBeans sont des composants logiciels écrits en java, conformes aux spécifications exposées dans l'API des java beans, cette API qui a été crée par Sun en collaboration avec d'autres de l'industrie énonce les regèles que doivent suivre les développeurs de logiciels pour crées des composants logiciels indépendants et réutilisables, comme beaucoup de composants logiciels, les JavaBeans renferment un état et un comportement. Par l'utilisateur de balise de JSP conçues pour les JavaBeans dans leurs pages web, les développeurs de contenus peuvent tirer profit de la puissance de java en ajoutant des éléments dynamiques à leurs pages, sans pour autant écrire la moindre ligne de code java. Rappelons tout d'abord les principales caractéristiques des composants JavaBeans avant de détailler leur utilisation dans les JSP.

- Les conteneurs de JavaBeans :

Un conteneur JavaBeans est une application, un environnement ou un langage de programmation qui permet aux développeurs de faire appel à des composants, de les configurer et d'accéder a leurs données et a leurs méthodes, les applications qui se servent directement des composants JavaBeans sont exclusivement écrites en java mais les conteneurs permettent de les utilisés a des niveaux conceptuels plus élevés. Pour cela, les composants java beans exposent leurs fonctionnalités et leur comportement au conteneur peut ainsi les manipuler d'une façon plus intuitive. Le conteneur java beans définit ses propres critères de présentation et d'interaction avec le composant et écrit lui-même le code java qui en résulte.

Si vous avez déjà utilisé les outils visuels beans box de Sun, Visual age pour java d'IBM, visuel café de webgain ou d'autres outils de développement java, vous avez

déjà manipulé des composants. Ces ateliers de développements fonctionnent avec des conteneurs JavaBeans qui permettent de manipuler visuellement les composants. Ainsi peut-on par le simple déplacement d'icônes définir les caractéristiques des composants java, leur comportement, leur lien avec d'autres objets, et bâtir une application entière. L'application ainsi définie générer tout le code java nécessaire. Les conteneurs de JSP, d'une façon similaire, permettent aux développeurs créer des applications java basées sur le web sans récrire de code java. L'interaction avec les composants dans les JSP se fait par l'intermédiaire de balises qui peuvent se mêler a du code HTML classique. [DKK01]

- **Les propriétés de composants :**

Les conteneurs permettent de manipuler les composants en termes de propriétés (ce sont des attributs de composants JavaBeans identifiés par un nom, qui maintiennent son état de contrôlent son comportement). Un composant est défini par ces propriétés sans lesquelles il serait pratiquement inutilisable. Les propriétés d'un composant JavaBeans peuvent être modifiées pendant l'exécution par le conteneur pour contrôler les spécificités de son comportement. Les valeurs de propriétés constituent le seul mécanisme que le conteneur utilise pour mettre les JavaBeans à la disposition du développeur.
A titre d'exemple, supposons que l'on possède un JavaBeans watherBean qui détient des informations sur les conditions atmosphériques du moment et des prévisions météorologiques. Le JavaBeans peut récupérer ces informations en accèdent aux serveurs du National weather service ou les extraire d'une base de données, sachant que tout comme l'utilisateur de JavaBeans, il nous est d'aucune utilité de savoir comment ce composant obtient ses informations. Tout ce qui nous intéresse en tant que développeurs, c'est que weatherBean soit capable de nous fournir des informations telles que les températures actuelles, les maximales attendues ou les risques de précipitations. Chacun de ces éléments d'information est proposé au conteneur de JavaBeans sous forme d'une propriété du composant dont le »s valeurs sont accessibles par la page ou l'application web.

Chaque composant possédera un ensemble de propriétés en fonction des informations qu'il contient .on peut le personnaliser en initialisant soi-même les

valeurs des ses propriétés. Le créateur du JavaBeans va imposer des restrictions sur chacune de ses propriétés, ce qui lui permet de contrôler l'accès qui en fait. Une propriété peut être accessible en lecture seulement, en écriture seulement ou en mise à jour (lecture/écriture).c'est cette notion d'accessibilité qui permet au concepteur du JavaBeans d'imposer des restrictions sur la façon de l'utiliser. Ainsi, dans l'exemple weatherBean, cela n'aura aucun sens de permettre aux développeurs de modifier la valeur de la propriété du composant qui indique la température maximale de la journée. Cette information est gérée par le JavaBeans et ne devrait être accessible qu'en lecture. D'autre part, si le java beans est doté d'une propriété qui contient le code postal d'une région dont on connaître la météo, il serait normal de permettre aux développeurs de le spécifier. Une telle propriété serait accessible en, lecture/écriture.

- **Les propriétés liées et les propriétés de déclenchement :**

Certaines propriétés sont utilisées comme des déclencheurs, pour déclencher un comportement ou renvoyer des comptes rendus. La lecture de telles propriétés ou leur modification a pour conséquence immédiate la réalisation par un composant JavaBeans d'une certaine action sur le serveur : il peut s'agir mettre à jour les valeurs d'autres propriétés ou de lancer une autre tâche sur le serveur. La mise à jour de la valeur de la propriété de code postal par exemple pourrait conduire le JavaBeans à consulter les services du *National Weather Service* pour s'informer sur les conditions atmosphériques qui correspondent au nouveau code postal. Il mettra ensuite à jour ses autres propriétés relatives à la météo en conséquence. Dans ce cas, les propriétés qui ont trait à la météo et au code postal sont considérées comme des propriétés liées parce que la modification de la valeur de l'une d'entre elles met à jour les valeurs des autres.

- **Les propriétés indexées**

Il est également possible pour une propriété unique de mémoriser plusieurs valeurs. Ces propriétés sont dites *indexées* car chaque valeur enregistrée dans la propriété est accessible via un numéro d'indice qui pointe sur

la valeur souhaitée. On peut par exemple réclamer la première valeur de la liste, la troisième ou la vingtième. Ainsi, notre WeatherBean pourrait avoir une propriété qui prenne en charge les prévisions météorologiques pour les cinq jours à venir. Cependant, tous les conteneurs de JavaBeans ne fournissent pas un mécanisme simple pour manipuler directement ces propriétés multivaluées. Par exemple, les balises de Java-Beans des JSP ne reconnaissent pas les propriétés indexées. Il faut utiliser en lieu et place des scriptlets, des expressions JSP ou les balises JSP personnalisées (détaillées aux chapitres 13 et 14) pour pouvoir accéder à de telles propriétés.

- **Les types de données des propriétés :**

Les propriétés des JavaBeans peuvent être utilisées pour prendre en charge une quantité importante d'informations. Par exemple, les propriétés de WeatherBean peuvent mémoriser des informations très diverses comme les températures, les risques de précipitations, les prévisions météo, les codes postaux, etc. Chaque propriété d'un composant JavaBeans ne peut contenir qu'un type particulier de données. Ses valeurs ont un type Java, qui est utilisé en interne par le composant et dans le code Java généré par le conteneur des JavaBeans. Les propriétés peuvent bien entendu supporter n'importe quel type primitif de Java tel que int (entier simple précision) ou double (entier double précision), ainsi que des objets Java tels que String et Date. Elles peuvent également mémoriser des objets définis par les utilisateurs, voire même d'autres JavaBeans. Les propriétés indexées mémorisent généralement un tableau de saleurs qui ont le même type de données.

Le conteneur de JavaBeans détermine la façon dont il convient de manipuler les valeurs de Propriétés d'un composant. On va référencer les valeurs des propriétés par leur type de données en utilisant des scriptlets et des expressions Ainsi, si une propriété contient des valeurs de type «entier on ne peut obtenir et déposer que des Valeurs entières. Cependant, avec les balises Beans, on traite chaque propriété comme si elle ne contenait que du texte String). Lorsqu'on initialise la saleur d'une propriété d'un JavaBeans, on lui transmet du texte. De même, lorsqu'on effectue une

lecture du contenu d'une propriété, on va récupérer du texte indépendamment du type de données interne utilisé dans le JavaBeans. Cette stratégie orientée texte permet de manipuler d'une façon qui soit simple les balises Bean de JSP et se marie bien avec HTML.

Le conteneur de JSP effectue toutes les conversions de type nécessaires. Par exemple, lorsqu'on initialise une propriété de type entier, le conteneur de JSP fait les appels Java nécessaires pour 'convertir les suites de caractères numériques qu'on lui a fournies en une valeur entière. Bien entendu, ce processus de conversion nous oblige à transmettre les valeurs textuelles appropriées de façon que Java puisse les convertir correctement dans le type de données natif. Si une propriété prend en charge des valeurs à virgule flottante, par exemple. Une erreur est générée si on essaie de lui affecter des valeurs comme banane, pain, cent ou (3,9).

De. Concepteurs astucieux de JavaBeans peuvent ainsi contrôler les valeurs des propriétés en acceptant des valeurs de type chaîne de caractères pour des propriétés qui ne sont pas de ce type et en effectuant eux-mêmes les conversions. Cette technique doit être utilisée pour toute valeur qui n'est ni une chaîne de caractères ni un type primitif de Java. Par conséquent, il pourrait être parfaitement autorisé d'attribuer à une propriété de type entier la valeur cent si le concepteur du JavaBeans l'a préparée à une telle affectation.

4. Les différentes fonctions des composants JavaBeans :

Les JavaBeans peuvent être répartis selon trois grandes catégories fonctionnelles : les composants graphiques utilisés dans les interfaces utilisateur graphiques. Les composants de données qui permettent l'accès aux informations, et les composants de services (appelés aussi *worker beans)* qui peuvent effectuer des tâches ou des calculs spécifiques. Bien entendu. Un JavaBeans *peut* appartenir *à* plusieurs de *ces catégories à la fois.*

- **Les composants graphiques**

En raison du développement des composants graphiques. Cette utilisation des composants JavaBeans est une des plus répandues. Ce sont des éléments tels que les champs de textes, les listes de sélection ou tout objet graphique utile pour la conception d'interface utilisateur. En regroupant des composants graphiques dans des JavaBeans, les environnements de développement Jasa peuvent profiter des avantages offerts par le support de la programmation graphique de ces composants. Les développeurs peuvent créer leurs interfaces simplement en assemblant les éléments souhaités. Toutefois, comme les composants graphiques ont été conçus pour s'exécuter avec les applications Java graphiques, ils ne sont pas compatibles avec les JSP qui sont plus orientées vers la conception d'interfaces HTML.

- **Les composants de données**

On peut avec ces JavaBeans accéder aux données que le composant lui-même n'a pas forcément la capacité de collecter ou de générer. Le calcul et la collecte des données mémorisées dans les JavaBeans de données sont de la responsabilité d'un autre composant ou service plus contple.se. Ces JavaBeans ne sont accessibles qu'en lecture. Ils permettent donc de récupérer des données mais pas de les modifier.

Certains JavaBeans de données permettent d'initialiser des propriétés afin de contrôler le format et le filtre des données avant de les retourner via d'autres propriétés. Par exemple, un **AccountStatusBean** pourrait être doté d'une propriété currentType qui contrôlerait si la propriété associée au solde du compte retourne les données en dollars, en livres ou en francs suisses. Les JavaBeans de données, en raison de leur simplicité, sont utiles pour normaliser l'accès aux informations en fournissant une interface stable.

- **Les composants de services**

Comme leur nom l'indique, ces JavaBeans permettent d'accéder à un comportement ou un service particulier. C'est la raison pour laquelle ils sont quelquefois appelés les JavaBeans travailleurs. Ils peuvent extraire des informations des hases de données, effectuer des calculs ou formater des informations. Comme il n'y a pas d'autre possibilité pour interagir avec un JavaBeans que de le faire par ses propriétés, on sa accéder à ses services de la même façon. Dans une conception classique, on va initialiser les valeurs de certaines propriétés qui contrôlent

5. Architecture des applications web :

Quelle que soit la complexité d'une application web, il est toujours conseillé d'analyser son architecture selon trois logiques (figure 3-2) :

- ✓ La couche de présentation, qui présente des résultats à l'utilisateur et en détermine l'apparence ;
- ✓ La couche de contrôle, qui contrôle le flot d'exécution de l'application ; et
- ✓ La couche de logique applicative « modèle», qui gère les données de l'application, exécute les traitements et communique directement avec les ressources sous-jacentes.

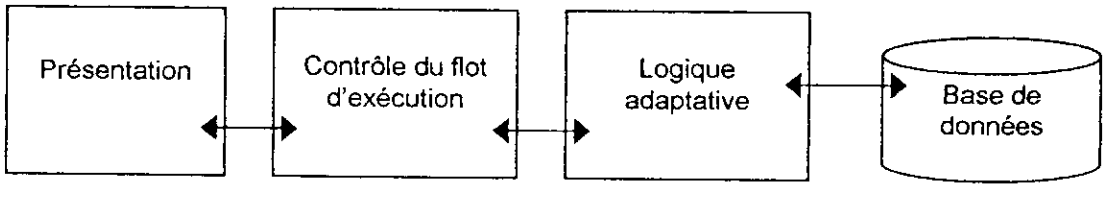


Figure 3-2 architecture des applications web par niveaux logique

5.1 Architecture orientée servlets :

C'est une approche qui consiste en effet à ne confier aux page JSP que les aspects de présentation, en délèguent les aspects « back-end» de flot d'exécution et de logique applicative à une ou plusieurs servlets. Les requêtes sont alors directement dirigées aux pages de présentation JSP via une servlet, qui effectue toutes les action nécessaire au fonctionnement de l'application. Une servlet peut jouer l'un ou plusieurs des trois rôles suivants :

- Exécuter des actions pour le compte d'une page JSP, par exemple soumettre une commande ;
- Livrer à une page JSP des données pour quelles soient affichées, par exemple un enregistrement de base de données ;
-Contrôler l'enchaînement des pages JSP d'une même application.

Après avoir effectué son traitement, une servlet transfère la requête vers la page JSP appropriée, voire vers une page HTML statique (voire figure 3-3)

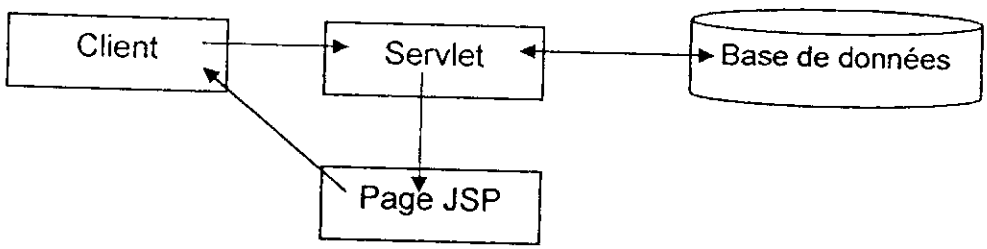


Figure 3-3 Enchaînement d'exécution d'une application basée sur des servlet

Ceux qui connaissent le pattern de médiateur (gestion de comportement) s'apercevront qu'il s'agit d'appliquer le même principe, mais à des pages et composants JSP plutôt qu'a des objet java. Le principe de ce pattern est de créer un composants de l'application interagissent, entre eux et avec les sources de données, cette approche permet de découpler la relation qui existait entre les pages jsp :on peut les coder indépendamment sans forcément inclure les détails de fonctionnement des autres pages. En outre, la séparation entre présentation et logique applicative est encor plus nette. Plus concrètement, il s'agit de minimiser la

quantité de traitements effectués par les pages elles-mêmes, pour s'appuyer sur des servlets dédiées. Le code JSP frontal est moins complexe car il se limite désormais au simple affichage de données ou à la collecte de données utilisateur.
De même, les servlets n'ont à contenir aucun détail de présentation des informations ; elle gère que le contrôle de l'exécution et la production des données à passer aux pages jsp à des fins de présentation à l'utilisateur. [DKK01]

6. Conclusion :

Un composant JavaBeans est une classe java qui obéit à des conventions de nommage et de conception simples qui précisées dans les spécifications JavaBeans Ces composants ne doivent pas nécessairement étendre une classe de base spécifique ni implémenter une interface particulière. Une classe qui applique les conventions JavaBeans et qui est manipulée en respectant ces conventions est un composant JavaBeans.

Chapitre 4

Analyse et conception du système

I. INTRODUCTION :

La modélisation et la spécification orienté objet est une méthode pour penser et écrire des problèmes organiser autour de concepts du monde réel.

- ✓ Comprendre des problèmes.
- ✓ Communiquer avec des experts
- ✓ Modéliser avec des organisations
- ✓ Préparer la documentation
- ✓ Concevoir des programmes et des bases de données.

Le processus comporte les étapes suivantes :

- on réalise un modèle analytique des aspects essentiels de l'application, sans aucun priori d'implémentation. Ce modèle comporte les objets essentiels du domaine applicatif avec leur relation.
- ensuite en prend des décision de conception et on affine le modèle initial pour préciser et optimiser l'implémentation.
- Les objets applicatifs ainsi écrits forment le cadre de la programmation à venir.
- Enfin, on traduit le modèle dans un langage de programmation quelconque

1.1. Qu'est ce que l'orienter objet ?

On déclare orienté objet un logiciel organiser autour d'entités groupant des données arbitraire (comme les struct c, et les record pascal) et des procédures.

- on appelle encapsulation le processus d'agrégation des ces données et les processus d'autres caractéristiques sont habituellement requises des objets au sens propre.
  - l'identité
  - la classification
  - le polymorphisme
  - l'héritage

1.1.1 l'identité

Les objets sont des entités qui peuvent être distinguées. Notamment, deux objets distincts peuvent avoir exactement les mêmes données membres.

1.1.2 la classification

Les objets ayant des attributs (données et comportements) similaire sont groupés dans une classe. La classe décrit tous les caractéristique des objets de sont type, une classe qui abstrait un objet du mondes réel intègre les données qui semblent pertinentes en fonction du degré de détail voulu.

1.1.3 le polymorphisme

La même fonction peut avoir des effets différents sur des objets de classes distinctes.

1.1.4 l'héritage

Des caractéristiques partagées par différentes classes peuvent être décrites dans une classe intermédiaire dont les autres héritent, ce qui permet de les décrire par les seules modifications
Qu'elle apporte au schéma de base .l'héritage apporte une grande puissance d'expression et d'abstraction au modèle objet.

II. La methode de concetion OMT etl e langage de modelisation UML:

1. Object Modeling Technique (OMT)

C'est une méthode d'analyse orientée objet pour le développement des système d'information. OMT apporte deux choses [CEE00] :

- une méthodologie pour le développement orienté objet.
- une notation graphique pour la communication de concepts orientés objet. Dans cette partie nous utiliserons le langage de modélisation UML.

1.1. Méthodologie orientée objet

La méthodologie consiste en la réalisation d'un modèle du domaine applicatif, complété par des détails d'implémentation au cours de la phase de spécification et conception du système le processus comporte les étapes suivantes : analyse, modélisation du système, modélisation des objets, programmation.

1.1.1. Analyse

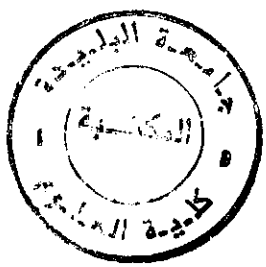
- à partir d'un énoncé du besoin, l'analyste construit (avec le demandeur) un modèle du monde réel qui fasse apparaître ses propriétés pertinentes.
- le modèle issu de l'analyse est une description précise, concise, de ce que le client demande, non de comment on peut le faire.
- les objets du modèle sont des objets du domaine, jamais des objets issus de l'informatique.
- une bonne analyse peut être comprise et critiquée par des experts du domaine qui ne sont pas des informaticiens.
- une bonne analyse ne comporte aucune décision d'implantation [LAU96].

1.1.2. Modélisation du système

- l'ingénieur (system designer en anglais) prend des décisions de haut niveau sur l'architecture du système.
- pendant cette phase, le système cible est organisé en sous systèmes pertinents relativement au modèle initial et aux choix d'architecture.
- l'ingénieur décide des caractéristiques à optimiser, définit une stratégie pour aborder le problème, et fait une première évaluation des besoins en ressources humaines [LAU96].

1.1.3. Modélisation des objets

- l'ingénieur décrit un modèle détaillé basé sur l'analyse et le modèle du système, qui respecte les choix stratégiques.
- l'accent est ici porté sur les structures de données et les algorithmes nécessaires pour réaliser chaque classe.
- les classes initiales restent pertinentes, mais elles sont complétées par les classes d'implémentation, choisies en fonction de besoins concrets en performance exprimés pour le système final.
- les objets initiaux et les objets informatiques sont décrits dans le même langage, bien qu'ils figurent dans des plans conceptuels distincts [LAU96].



1.1.4. Programmation

- dans cette phase bien connue, les classes et relations décrites précédemment sont traduites dans un langage de programmation.
- la programmation devrait être une part mineure de l'ensemble du processus, les décisions importantes ayant déjà été prises.

- le langage cible influence toujours la conception dans une certaine mesure, mais celle ci ne devrait jamais dépendre de détails trop pointus d'un langage donné.

- la programmation doit obéir à des règles de contrôle qualité suffisantes pour :
  - garantir la traçabilité des programmes par rapport à la spécification (vérification).
  - aboutir à un résultat suffisamment réutilisable, extensible, portable ... [LAU96].

2. Unfied modeling language (UML)

2.1 Introduction a l'UML:

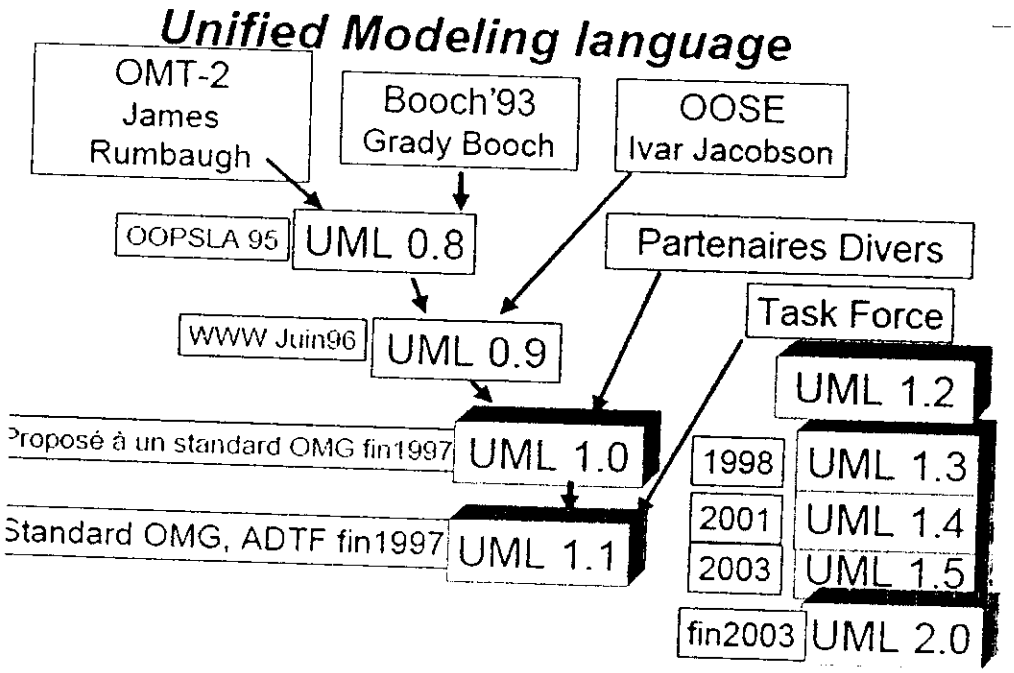


Figure 4-1 origines de uml [ROO]

Pourquoi modéliser

. Un modèle est une simplification de la réalité qui permet de mieux comprendre le système à développer.

. Il permet

- De visualiser le système comme il est ou comme il devrait l'être.
- De valider le modèle vis à vis des clients
- De spécifier les structures de données et le comportement du système.
- De fournir un guide pour la construction du système.
- De documenter le système et les décisions prises.

Qu'apporte la modélisation objet

. Plus grande indépendance du modèle par rapport aux fonctionnalités demandées.

. Des fonctionnalités peuvent être rajoutées ou modifiées, le modèle objet ne change pas.

. Plus proche du monde réel.

Les objectifs d'UML

• Représenter des systèmes entiers

• Etablir un couplage explicite entre les concepts et les artefacts exécutables

• Prendre en compte les facteurs d'échelle

• Créer un langage de modélisation utilisable à la fois par les humains et les machines

Recherche d'un langage commun unique

– Utilisable par toutes les méthodes

– Adapté à toutes les phases du développement

– Compatible avec toutes les techniques de réalisation

UML un langage

. UML n'est pas une méthode

. UML est un langage de modélisation objet

. UML a été adopté par toutes les méthodes objet

. UML est dans le domaine public, c'est une norme

UML un langage pour

. Visualiser

. Chaque symbole graphique a une sémantique

. Spécifier De manière précise et complète, sans ambiguïté,

. Construire Les classes, les relations SQL peuvent être générées automatiquement

. Documenter Les différents diagrammes, notes, contraintes,

Exigences seront présentés dans un document.

UML et les domaines d'utilisation

. Systèmes d'information des entreprises

. Les Banques et les services financiers

. Télécommunications

. Transport

. Défense et aérospatiale

. Scientifique

. Applications distribuées par le WEB

Les 9 diagrammes en UML

UML est un langage graphique et repose sur neuf types de diagrammes. Chacun de ces diagrammes utilise le même principe : les concepts sont représentés par des symboles, et les relations entre les concepts sont représentées par des lignes qui relient les symboles. Le vocabulaire et la grammaire d'UML sont très réduits.

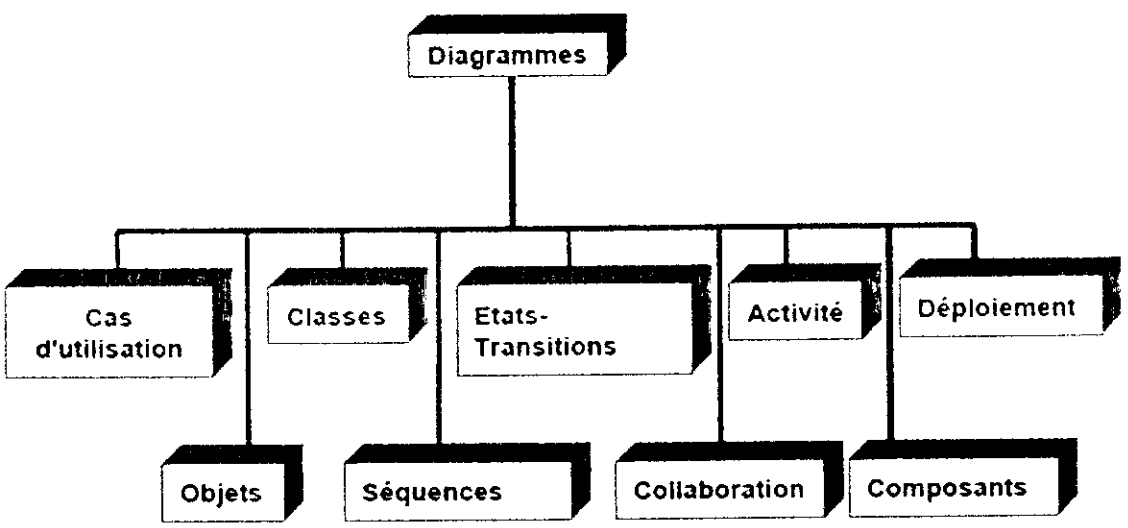
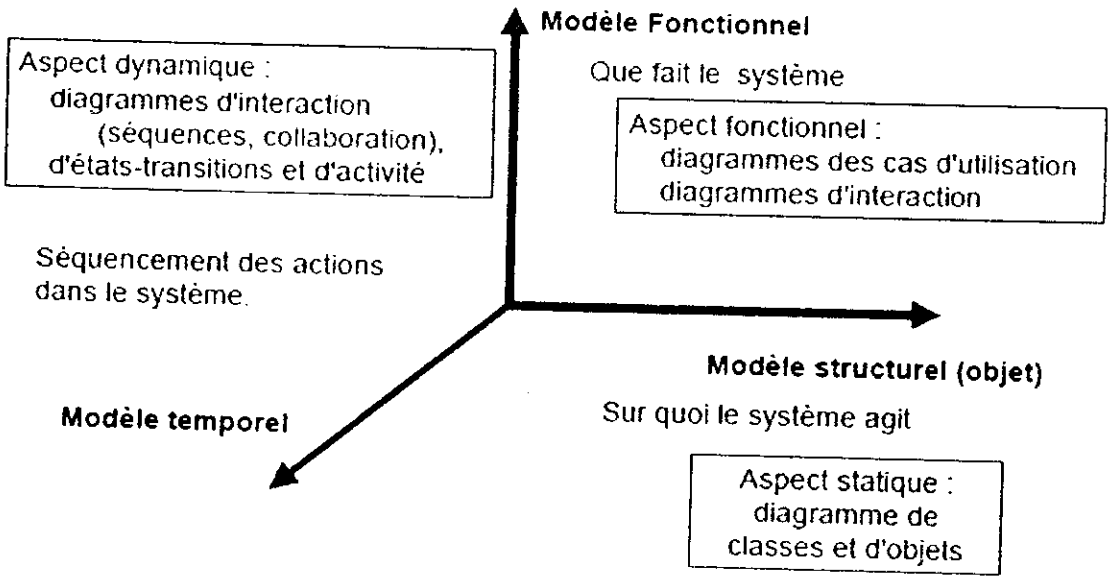


Figure 4-2 les 9 diagrammes de l'uml [ROO]

Les trois composantes d'une modélisation



Phases de la modélisation

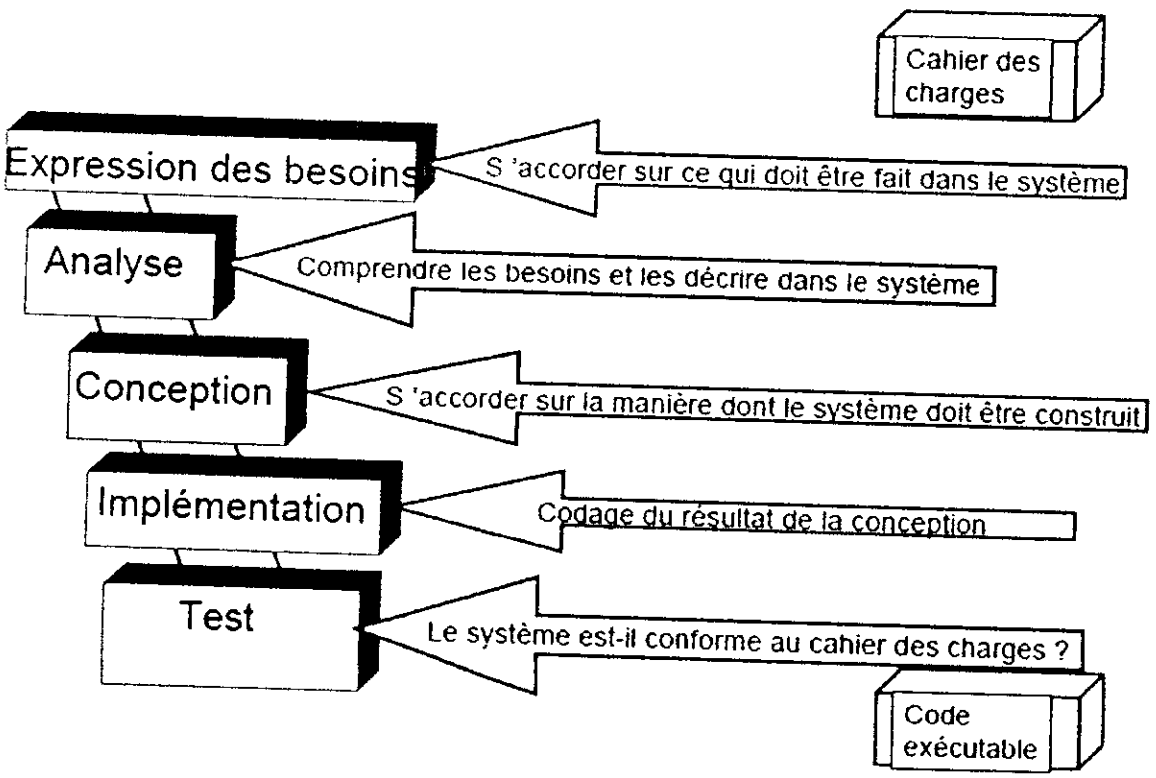


Figure 4-3 Phases de la modélisation [ROO]

2.2 Les diagramme de l'UML

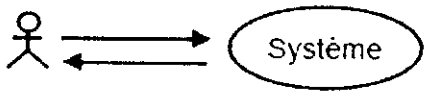
2.2.1 Le diagramme des Use Cases ou des cas d'utilisation

-> Ce que doit faire le système

Sans spécifier comment il le fait

But des Use Cases

Les cas d'utilisation représentent les Fonctionnalités que le système doit savoir faire. Chaque cas d'utilisation décrit un ensemble d'interactions successives d'une entité en dehors du système (utilisateur) avec le système lui même pour réaliser une fonctionnalité.



Les Uses Cases permettent :

• De connaître le comportement du système sans spécifier comment ce comportement sera réalisé.

• De définir les limites précises du système

• Au développeur de bien comprendre l'attente des utilisateurs et les experts du domaine.

De plus les Use Cases sont :

• Des instruments de validation et de test du système en cours et en fin de construction.

Modèle des cas d'utilisations

. Un diagramme de cas d'utilisation définit :

. le système

. les acteurs

. les cas d'utilisations

. les liens entre acteurs et cas d'utilisations

. Un modèle de cas d'utilisation se définit par

. des diagrammes de cas d'utilisation

. une description textuelle des scénarios d'utilisation

. une description de ces scénarios par

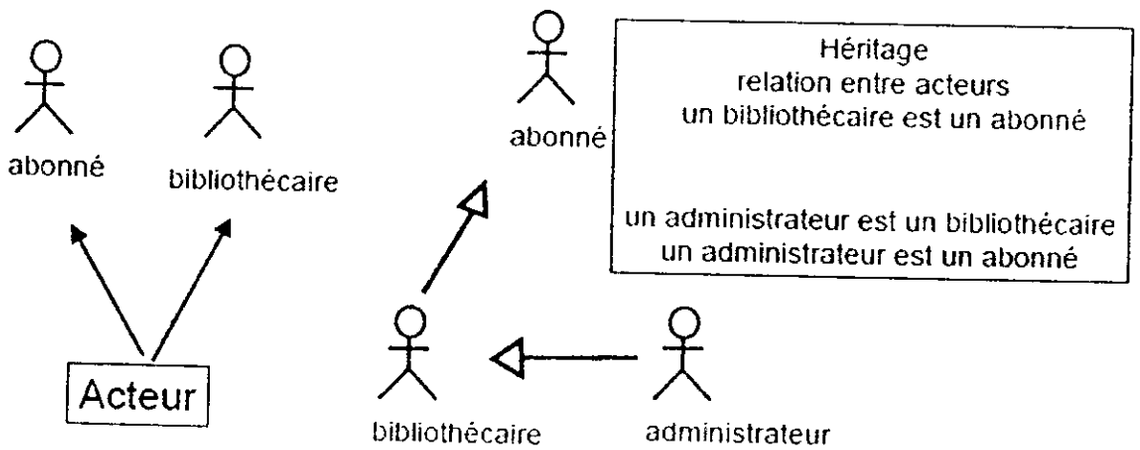
. les diagrammes de séquences

. les diagrammes de collaboration

Les Acteurs

. Un acteur représente une personne ou un périphérique qui joue un rôle (interagit) avec le système.

. Relation entre acteurs : généralisation (héritage)



Les cas d'utilisation (use-case)

. Un cas d'utilisation est un moyen de représenter les différentes possibilités d'utiliser un système.

. Il exprime toujours une suite d'interactions entre un acteur et l'application.

. Il définit une fonctionnalité utilisable par un acteur.

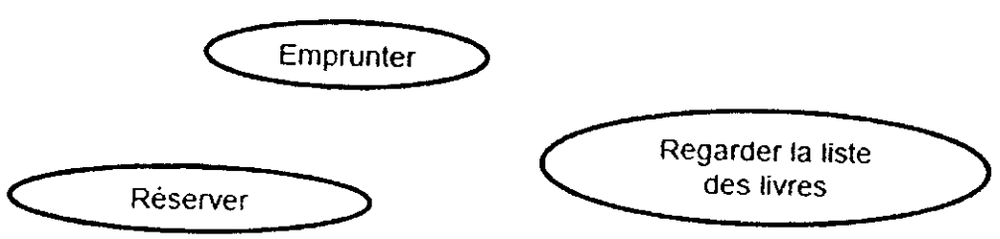
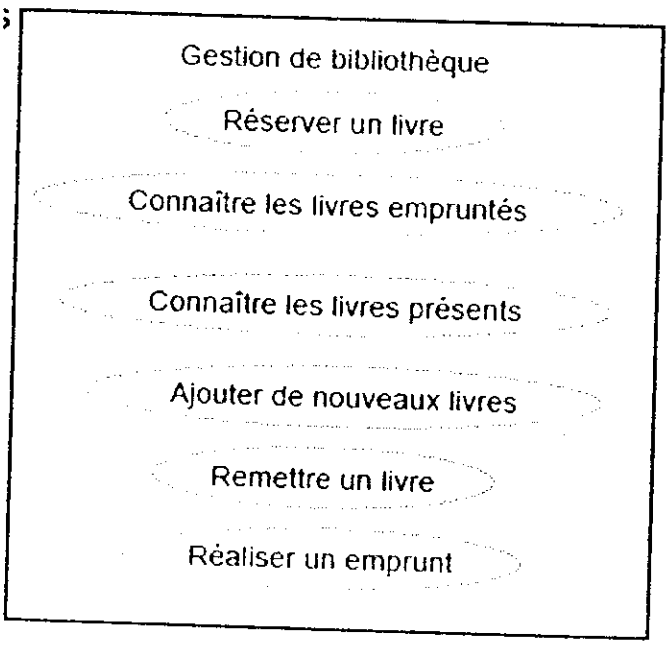


Figure 4-4 les cas d'utilisation

Le système

. Le système définit l'application informatique, il ne contient donc pas les acteurs, mais les cas d'utilisation et leurs associations



Exemple

Les diagrammes de cas d'utilisation

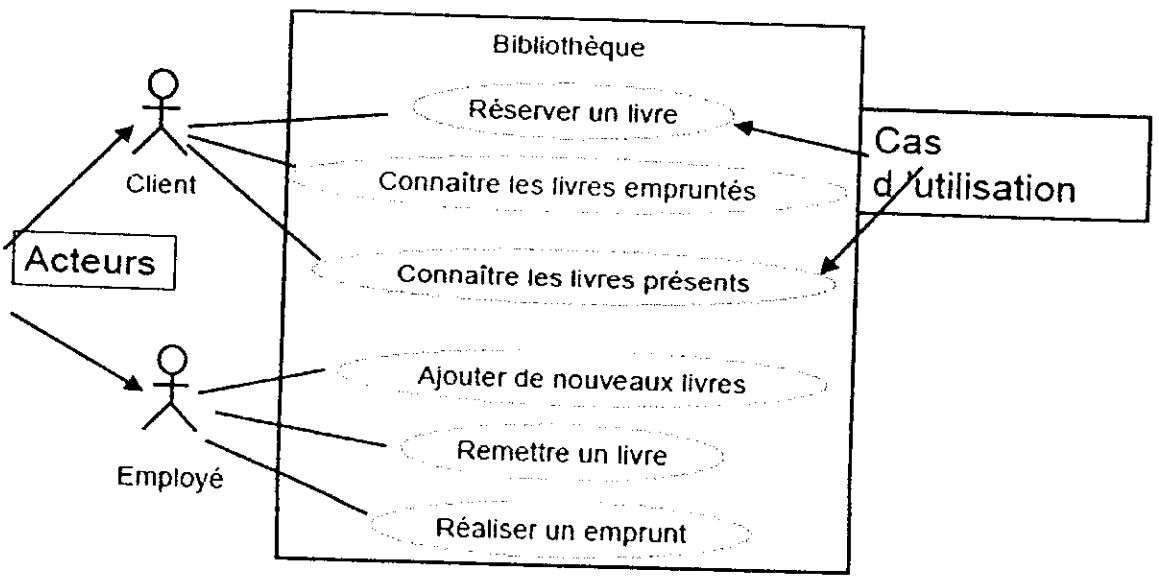


Figure 4-5 diagrammes de cas d'utilisation

Scénarios d'un cas d'utilisation

. La description d'un cas d'utilisation se fait par des scénarios qui définissent la suite logique des interactions qui constituent ce cas.

. On peut définir des scénarios simples ou des scénarios plus détaillés faisant intervenir les variantes, les cas d'erreurs, etc.

. Cette description se fait de manière simple, par un texte compréhensible par les personnes du domaine de l'application.

. Elle précise ce que fait l'acteur et ce que fait le système

. La description détaillée pourra préciser les contraintes de l'acteur et celles du système.

Exemple Scénarios d'un cas d'utilisation

description simplifiée

Le client se présente devant un terminal:

- (1) Le système affiche un message d'accueil.
- (2) Le client choisit l'opération réservation parmi les différentes opérations proposées.
- (3) Le système lui demande de s'authentifier.
- (4) Le client donne son identification (nom, mot de passe).
- (5) Le système lui demande de choisir un livre.
- (6) Le client précise le livre qu'il désire.
- (7) Le système lui précise si un exemplaire du livre lui est réservé.

2.2.2 Les diagrammes de séquence

Ces diagrammes représentent également la dynamique de fonctionnement du système également. En revanche, la représentation temporelle des événements est mise en avant plutôt que la représentation spatiale. [FAP00]

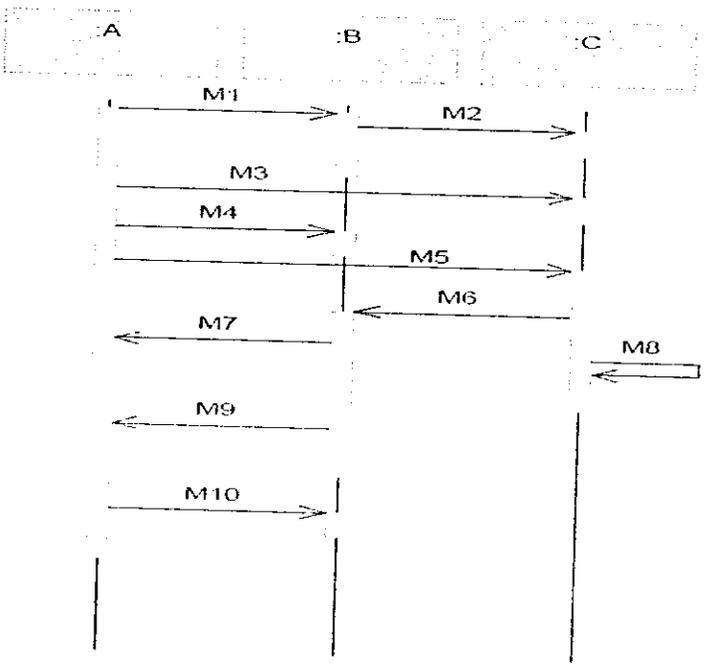


Figure 4-6 diagramme de séquence

Scénario par diagramme de séquences

. Suite aux descriptions textuelles, le scénario peut être représenté en utilisant un diagramme de séquences.

Le diagramme de séquences permet:

.de visualiser l'aspect temporel des interactions

.de connaître le sens des interactions (acteur vers système ou contraire)

Exemple

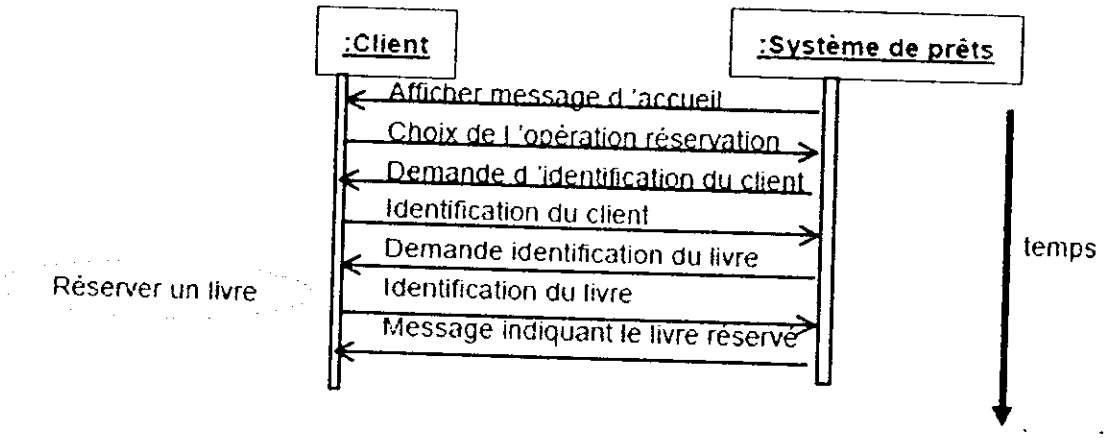


Figure 4-7 exemples de diagramme de séquence [ROO]

2.2.3 Les diagrammes de collaboration

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait

figurer les envois de messages, annotés par leur ordre d'apparition. [FAP00]

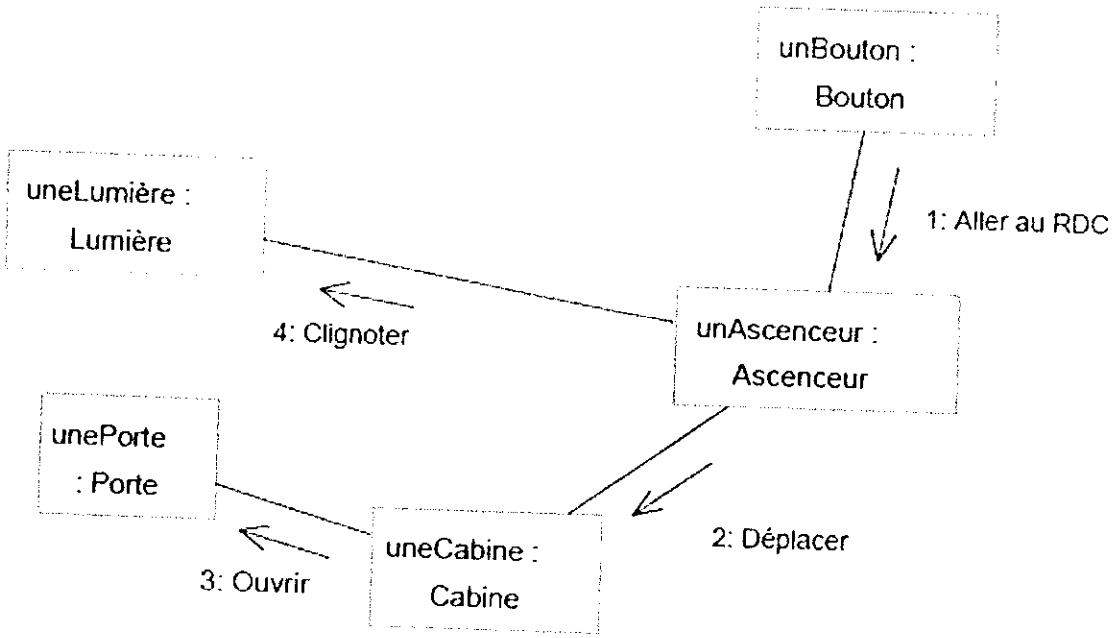


Figure 4-8 diagramme de collaboration

2.2.4 Les diagrammes de classes

Ces diagrammes décrivent l'architecture du système; on y représente les classes et les relations entre classes, qu'elles soient d'héritage, d'agrégation ...

La classe

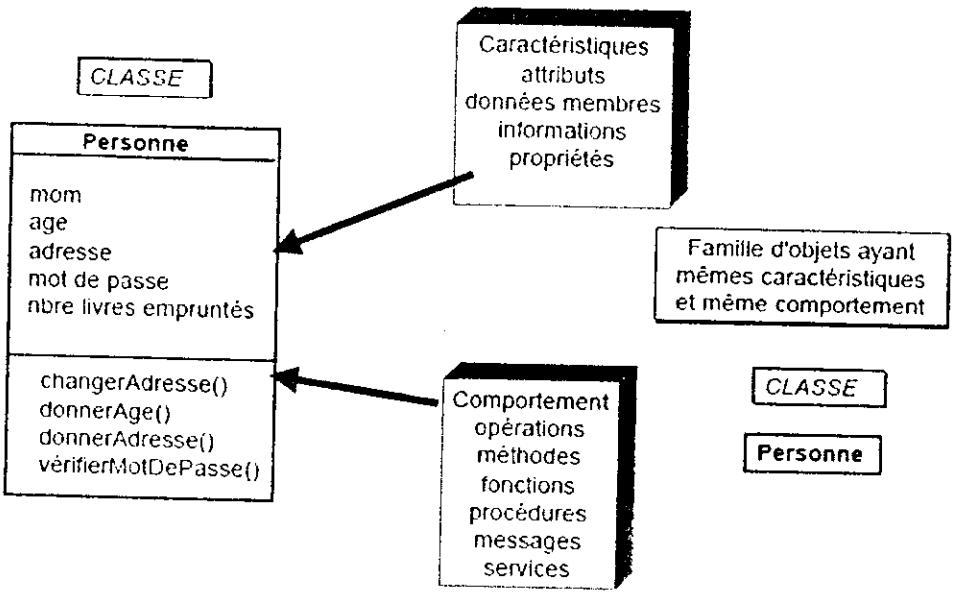
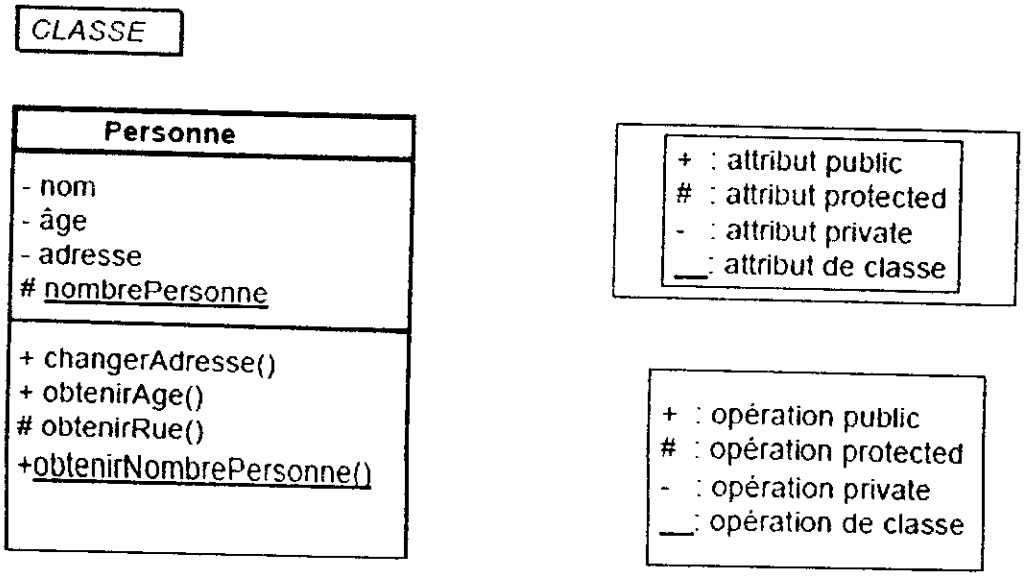


Figure 4-9 les classes

Protection des attributs et des Opérations



Exemple

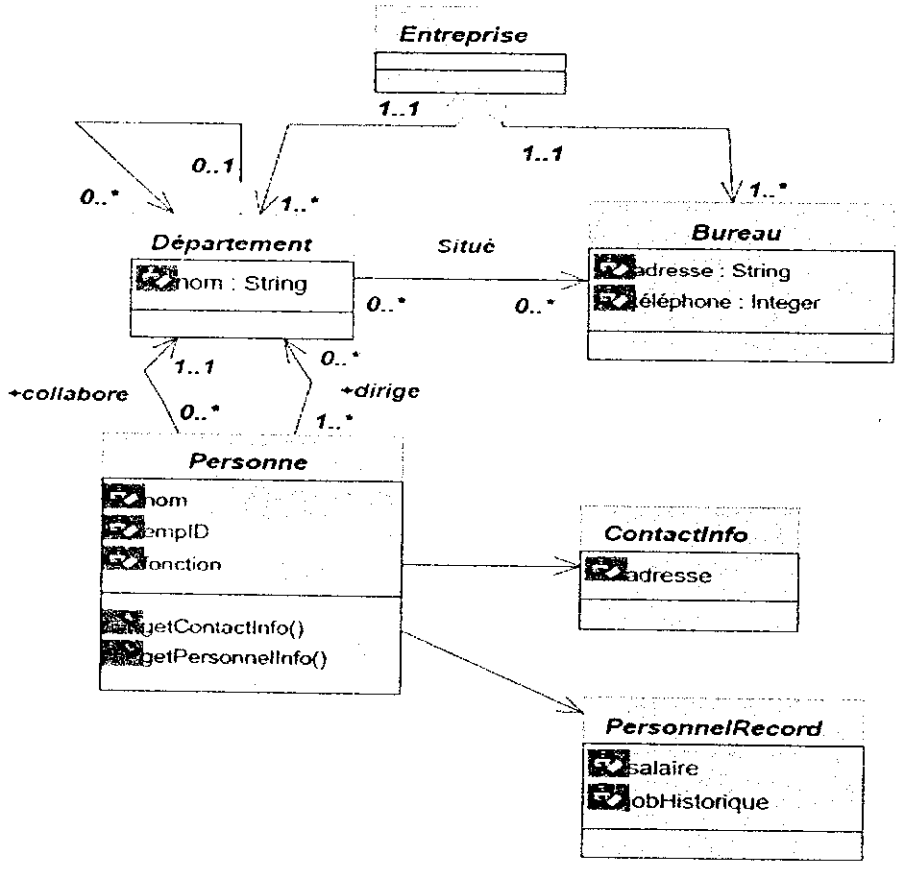


Figure 4-10 diagramme de classes [FAP00]

Résumé

On a présenté une méthode de conception OMT et un langage de modélisation d'un system d'information par objets. Le langage de modélisation UML respecte les trois étapes de l'étude : l'étape statique, dynamique, fonctionnelle. Statique pour aboutir a un modèle objet, dynamique pour décrire la réalisation des objets du system aux événements et les itération entre les objets eux-mêmes, et enfin, l'étape fonctionnelle qui comprend des contraintes entre les valeurs, les information de contrôles et de structure de d'objet appartenant respectivement au modèle dynamique et au modèle objet.
Nous allons utiliser cette méthode pour la conception de notre system

3. Analyse et conception :

Pour concevoir notre système, nous avons choisis la méthode de conception « OMT » qui sera appliquée en adoptant le langage de modélisation objet UML.
La technique de modélisation par objet (uml) associe trois modèles liés mais distinct : le modèle objet, le modèle dynamique et le modèle fonctionnel.
Le modèle objet décrit la structure des données sur lesquelles le modèle dynamique et le modèle fonctionnel opèrent.
Le modèle dynamique décrit la structure de contrôle des objets, met en évidence les décisions qui dépendent de la valeur des objets et invoque des fonctions.
Le modèle fonctionnel décrit les fonctions invoquées par les opérations du modèle objet et du modèle dynamique.

3.1. Analyse

3.1.1. ANALYSE DES BESOINS :

La base de données doit être accessible via les réseaux intranet et Internet, donc elle assure un accès non limité dans le temps et le lieu avec une gestion de ressources.
Le système doit assurer aux agents des services tel que : consultation, recherche..., et ceci selon les privilèges qu'on leur donne lors de leurs inscription.

- Le système a le contrôle des ressources : ajout, modification, suppression de données.
- Contrôle des abonnés : inscription d'un utilisateur, lister les utilisateurs, modifier les coordonnées d'un utilisateur,...
- Attribution des privilèges aux abonnées : simple consultation, consultation et recherche....

Les besoins selon l'utilisateur du système seront :

Agent :

- Consulter un son répertoire,
- Rechercher des informations selon des critères bien spécifiques.
- Contacter un agent service,

Administrateur:

- Lister les utilisateurs,
- Supprimer un privilège à un utilisateur,
- Affecter un privilège a un utilisateur.

Agents du service :

- Consulter la base de données
- Contacter un utilisateur,
- Faire des Recherches.
- Modifier les informations propres à un utilisateur.

Un administrateur est considéré aussi comme utilisateur avec privilèges donc il a accès à l'ensemble des fonctionnalistes du système

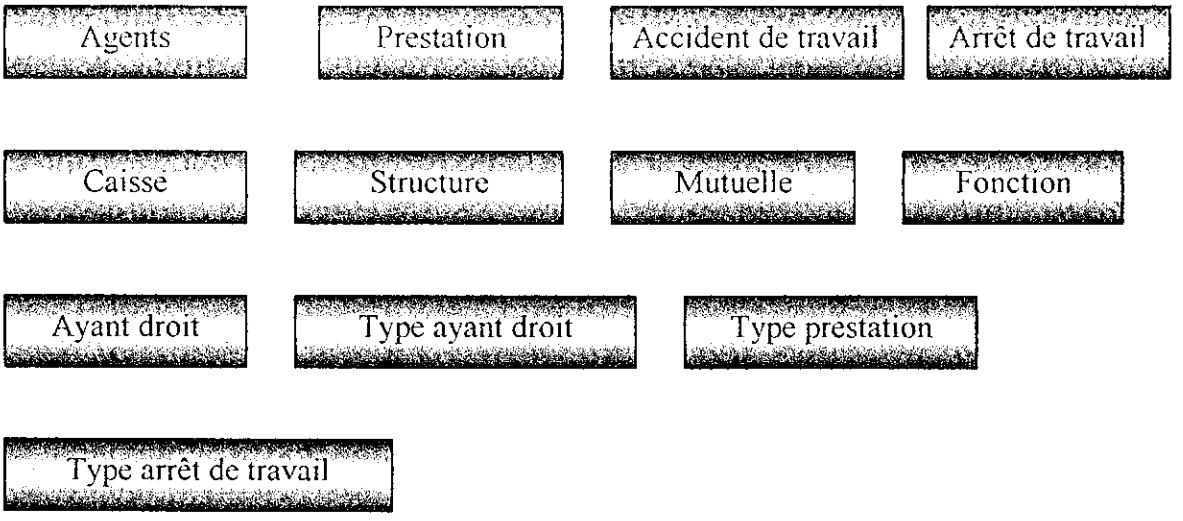
La phase d'analyse, première phase de la modélisation OMT, a pour objectif de décrie de manière précise, concise, correcte et compréhensible un modèle du monde réel.
L'analyse ne se préoccupe pas des solutions mais des questions, elle identifie le quoi faire et l'environnement d'un système sans décrire le comment qui est le propre de la conception.

3.1.2. Modèle objet

Un modèle objet saisit la structure statique d'un système. On montre les objets, les relations entre les objets et les attributs qui caractérisent chaque classe. Les étapes nécessaires à la construction, de ce modèle sont :

- identifications des classes.
- Préparation du dictionnaire de données.
- Identification des attributs de chaque classe.
- Identification des associations.

3.1.2.1 Identification des classes



3.1.2.2. Dictionnaire de données

- **Identification des classes**

Agents : ensemble du personnels de la direction générale de SONATRACH

Prestation : ensemble des prestations déposés par les agents

Accident de travail : les accidents de travail enregistrer

Arrêt de travail : les arrêt de travail qui on été déposer au service

Caisse : c'est les caisses où sont affiliés les agents de la direction générale SONATRACH

Structure : ensemble de structure de la direction ou appartient les agents

Mutuelle : la mutuelle où sont affiliés les agents de la direction

Fonction : les fonction qu'exerce l'agent

Ayant droit : les personnes qui on le droit a la prestation.

Type ayant droit : type d'ayant droit a la prestation.

Type de prestation : type de prestations sociales.

Type arrêt de travail : types d'arrêts de travail répertorie.

- **Identification des attributs :**

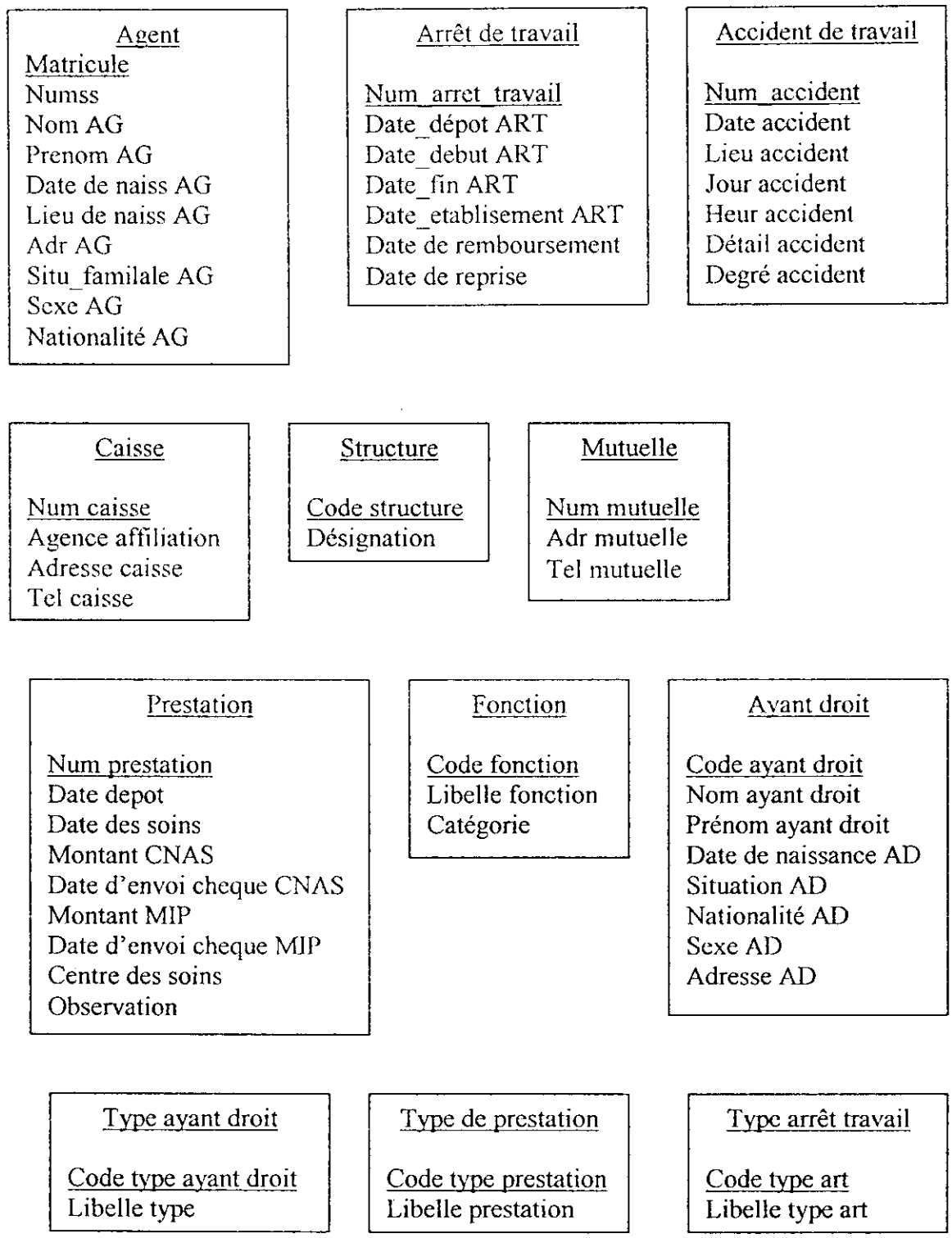


Figure 4.11 Identification des attributs

- **Identification des associations :**

Tableau 4.1. Identification des associations

association	Dimension	cardinalité	Classes-intervenant
exerce	2	1..*- 1	Agent - fonction
A pour benif	2	* - 1	Prestation - ayant droit
Est de type	2	1.* - 1	Ayant droit - type ayant droit
Est de type1	2	1.* - 1	Prestation - type prestation
Est de type2	2	1.* - 1	Arrêt de travail - type arrêt de travail
Présente	2	* - 1	Agent - prestation
Appartient	2	1.* - 1	Agent - structure
Affilie	2	1.* - 1	Agent - caisse
Affilie1	2	* - *	Agent - mutuel
Peut avoir	2	1.* - *	Agent - accident de travail
Dépôt arrêt de travail	2	1 - *	Agent - arrêt de travail

- **exerce :** chaque agent exerce une fonction
- **a pour benif :** pour chaque prestation a pour bénéficiaire des ayant droit
- **est de type :** chaque ayant droit appartiens a un type
- **est de type1 :** chaque prestation présenté appartiens a un type de prestation
- **est de type2 :** chaque arrêt de travail déposé appartiens a un type de travail
- **présente :** l'agent présente une prestation
- **appartiens :** un agent appartiens a une structure
- **affilie :** un agent affilié a une caisse
- **Affilie1 :** agent est affilié a une mutuelle
- **Peu avoir :** un agent peut avoir un accident de travail
- **Dépôt arrêt de travail :** agent déposé un arrêt de travail

3.1.2.3 Diagramme de classes :

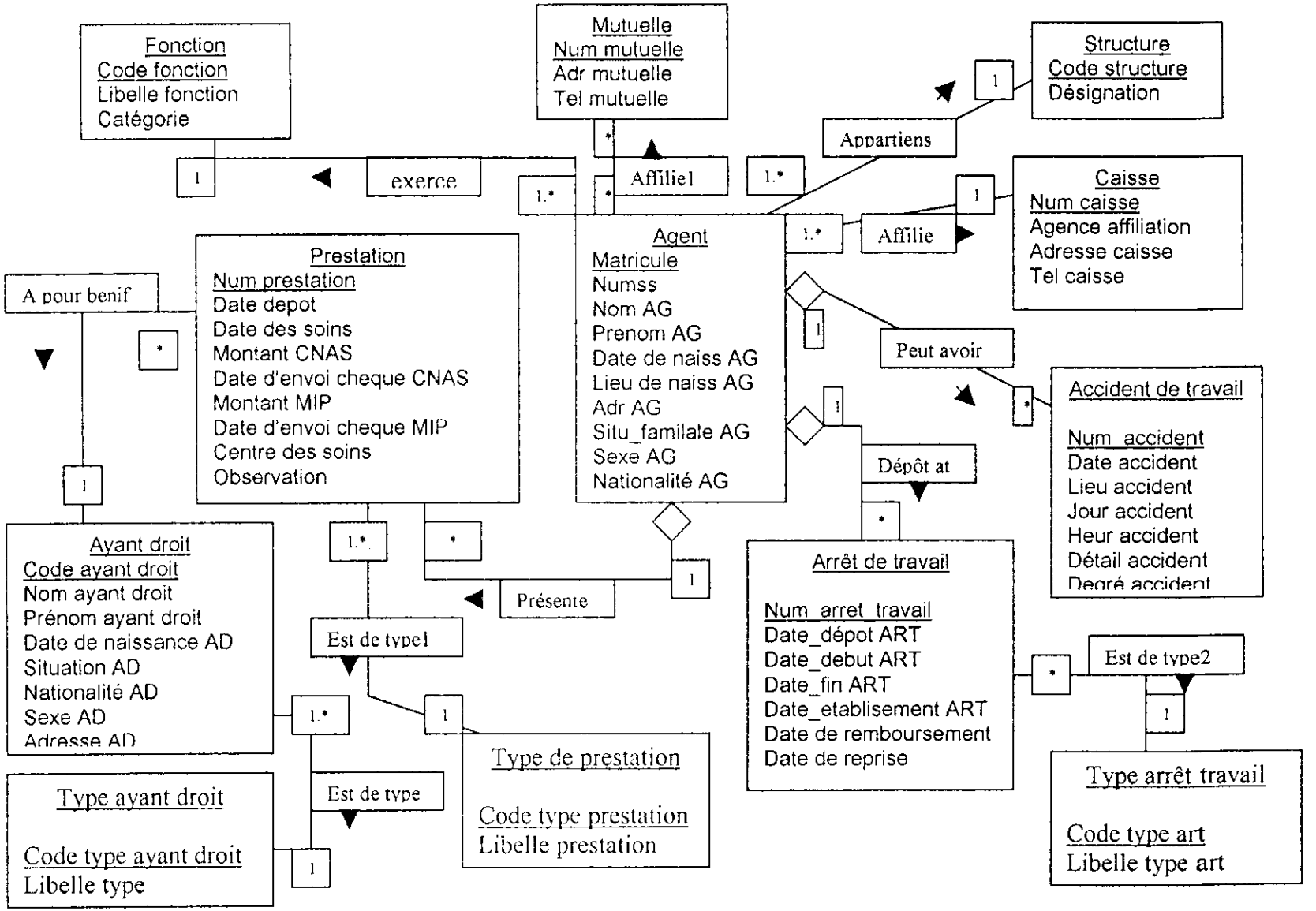


Figure 4.12. Diagramme des classes

3.1.3. Model fonctionnelle :

Le model fonctionnelle décrit l'aspect fonctionnelle du system cette phase consiste a rependre a la question suivante "que fait le system " et pour cela on utilise le diagramme "use case". Les étapes nécessaires à la construction, de ce modèle sont :

- le système
. les acteurs
. les cas d'utilisations
. les liens entre acteurs et cas d'utilisations

- Les acteurs :

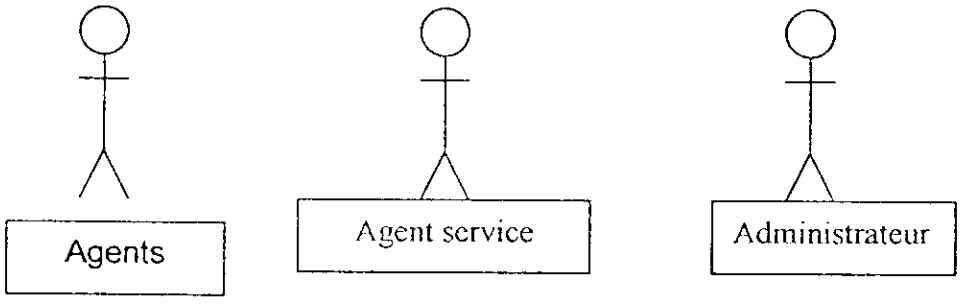


Figure 4-13 les acteurs

- Les cas d'utilisation (use case) :

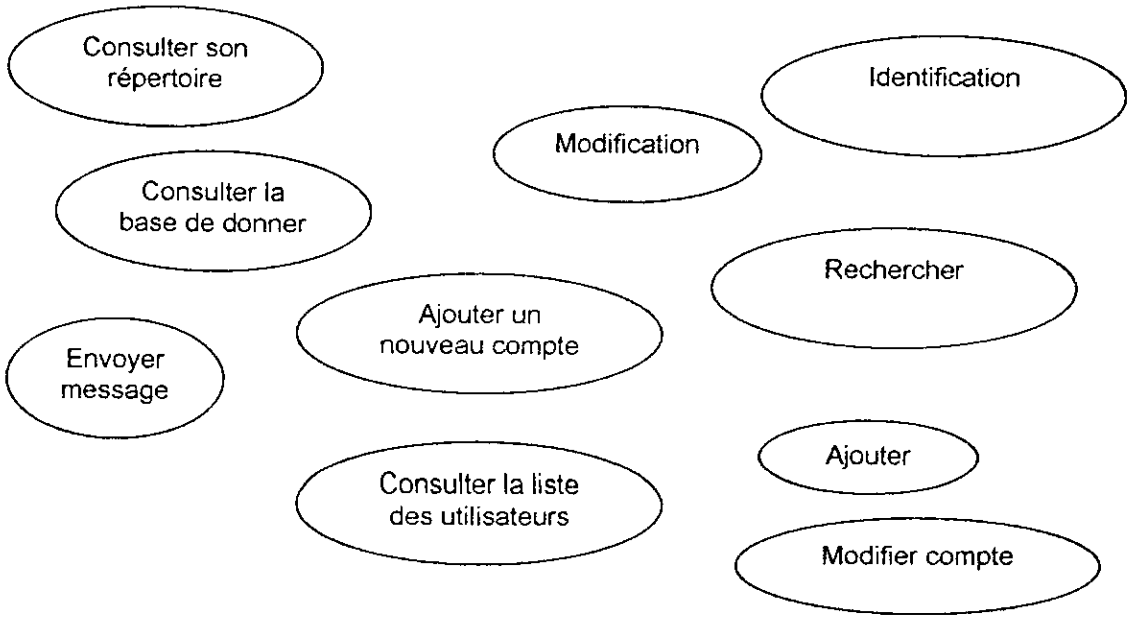
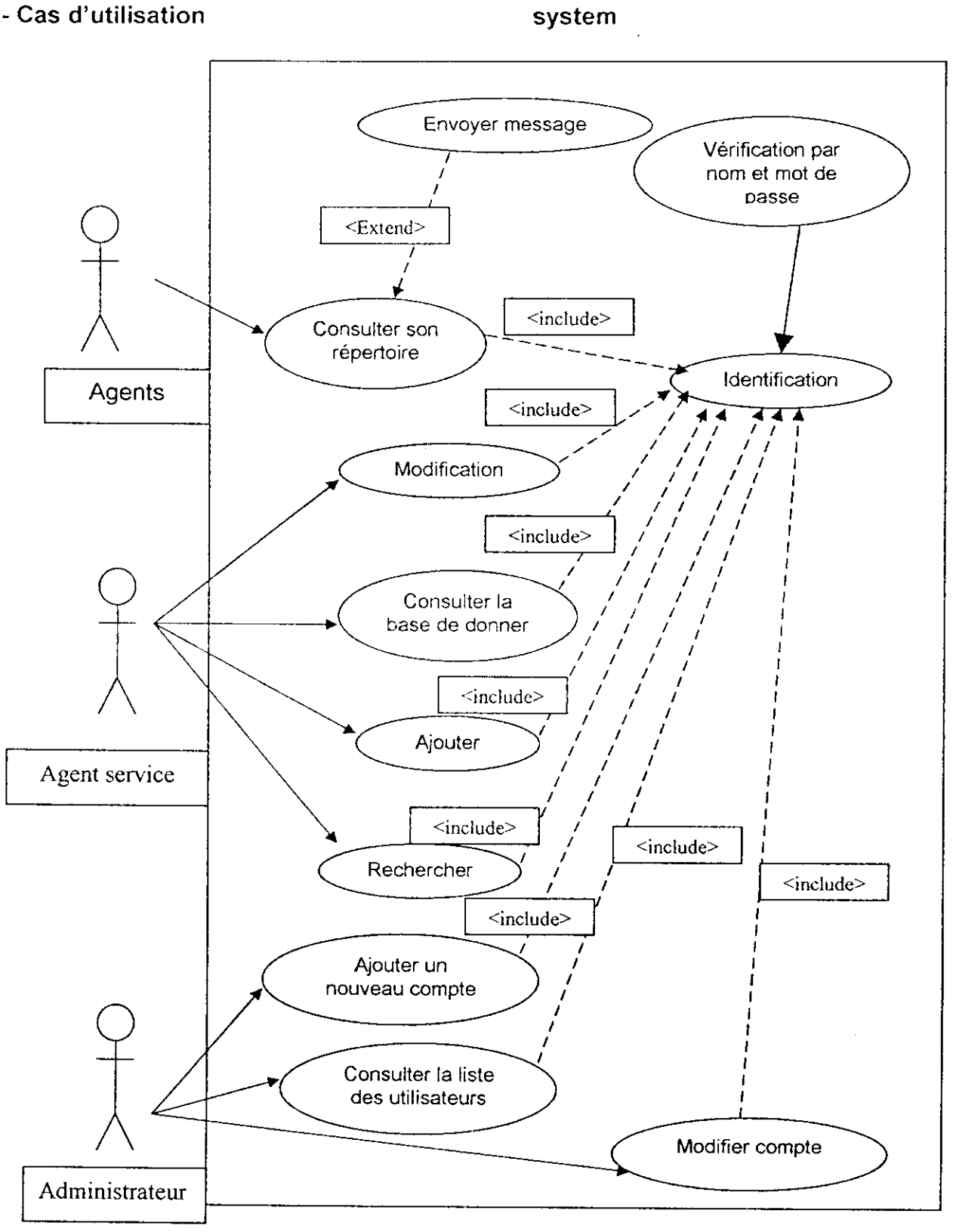


Figure 4-14 les cas d'utilisation

- Cas d'utilisation



Les acteurs

Figure 4-15 diagramme de cas d'utilisation

3.1.4. Modèle dynamique :

le modèle dynamique décrit des concepts traitant du flot de contrôle des interactions et des séquences d'opérations dans un système d'objet actif (concurrente). [CEE00]

- **Les scénarios par diagramme de séquence**

Le diagramme de séquences permet:

.de visualiser l'aspect temporel des interactions

.de connaître le sens des interactions (acteur vers système ou contraire)

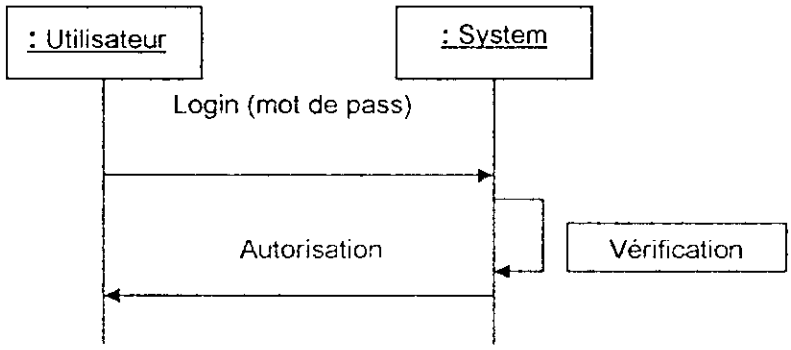


Figure 4-16 Identifications des utilisateurs

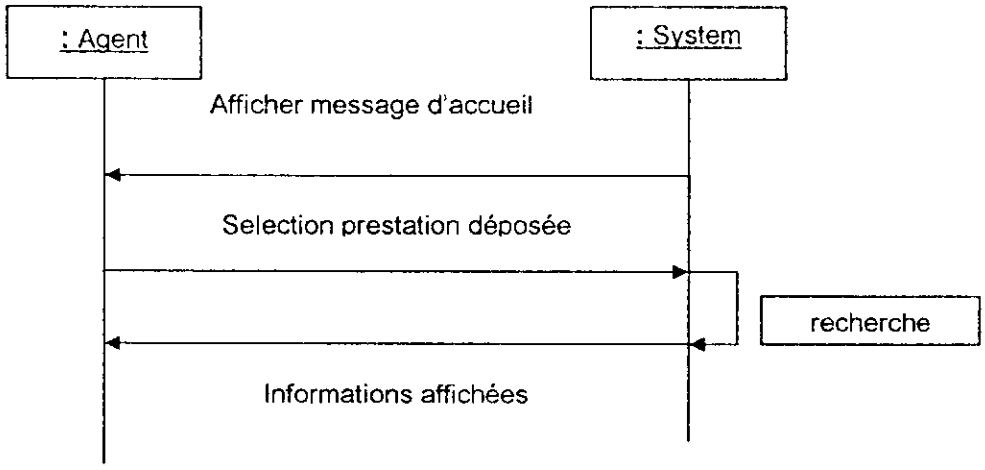


Figure 4-17 Consulter les prestations relatives a une personne

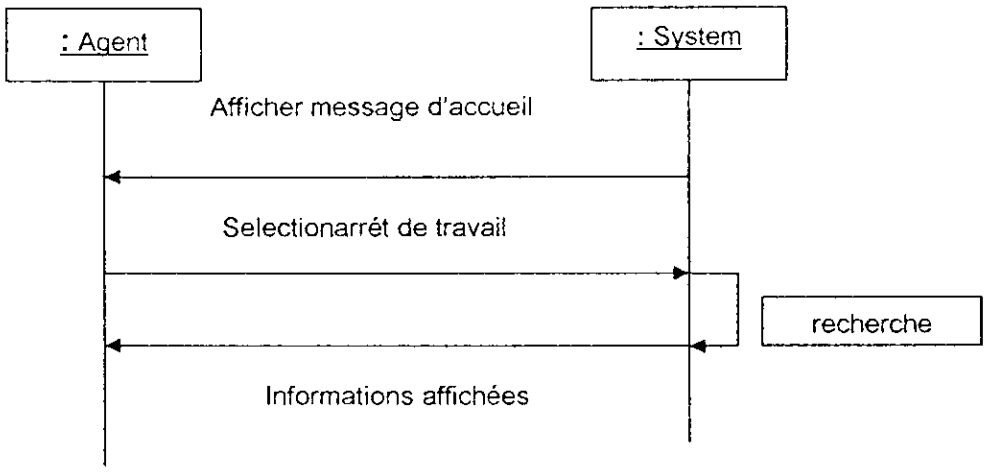


Figure 4-18 Consulter arrêt de travail relatif a une personne

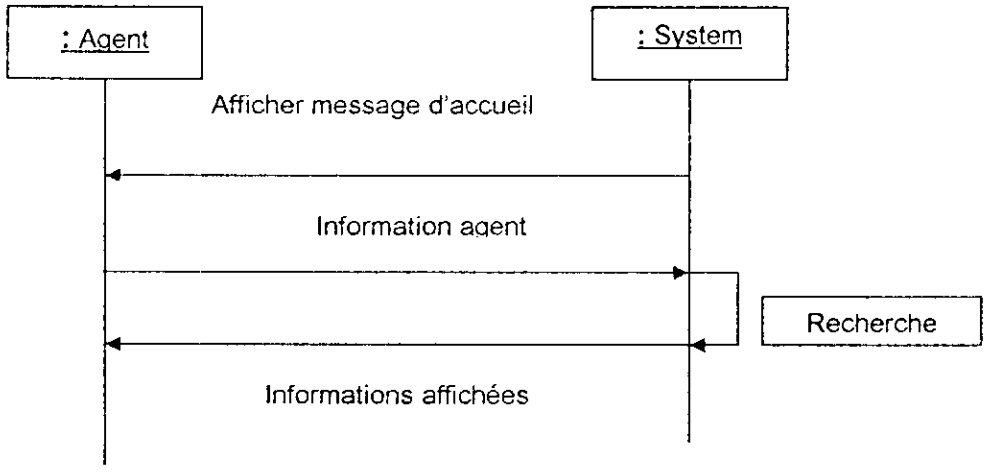


Figure 4-19 Consulter les informations sur l'agent

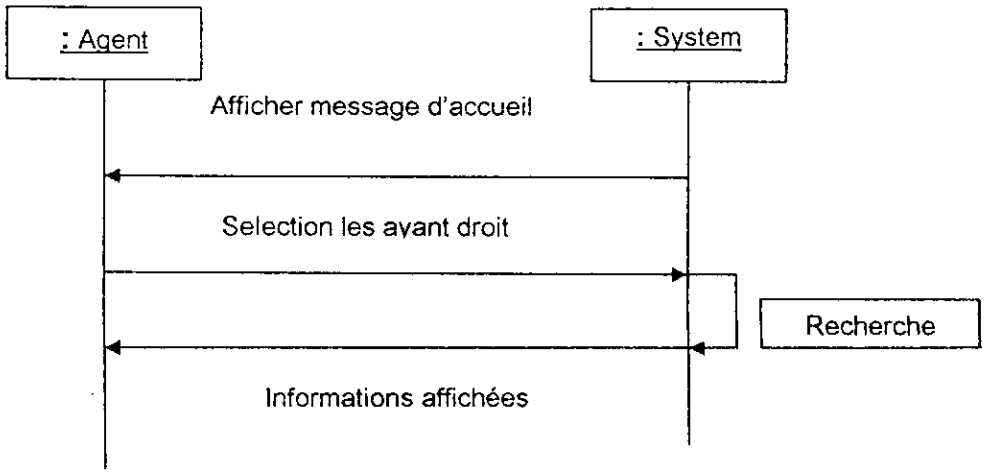


Figure 4-20 Consulter les ayant droit relatif a une personne

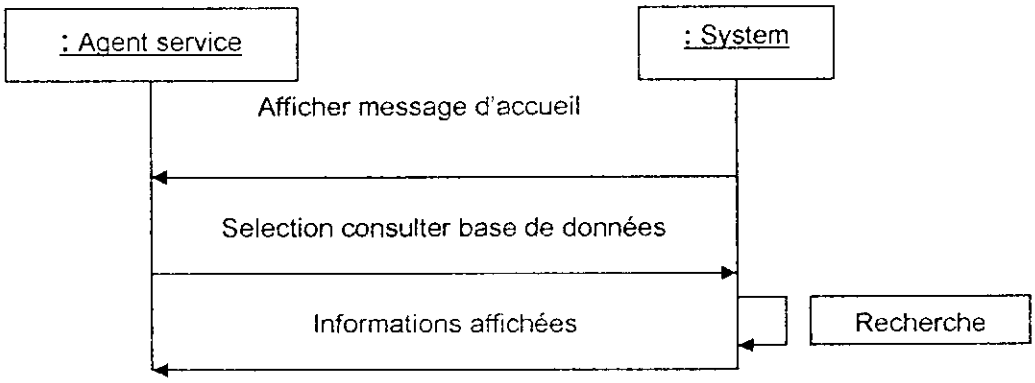


Figure 4-21 Consulter la base de données

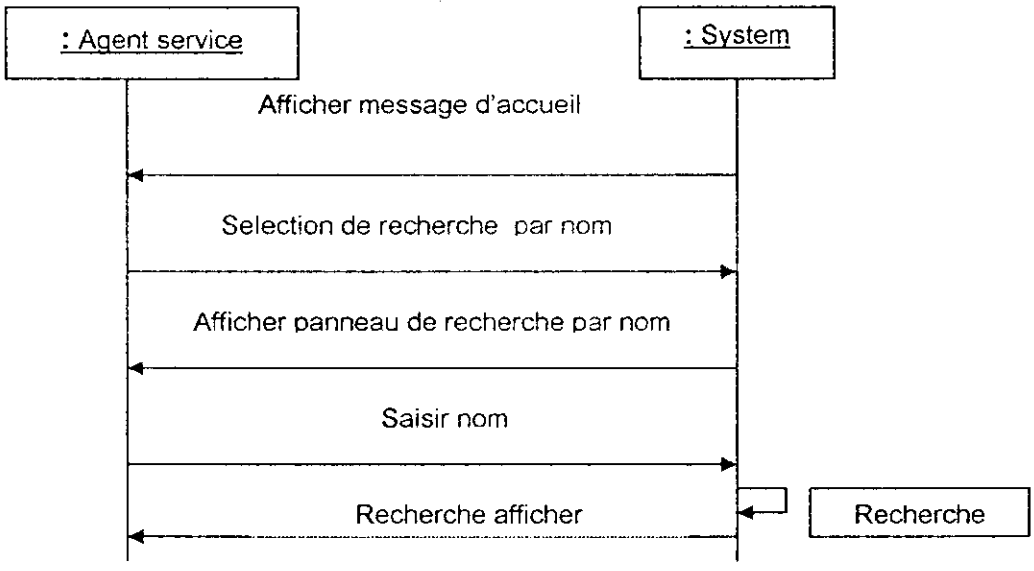


Figure 4-22 Recherche d'un agent par nom

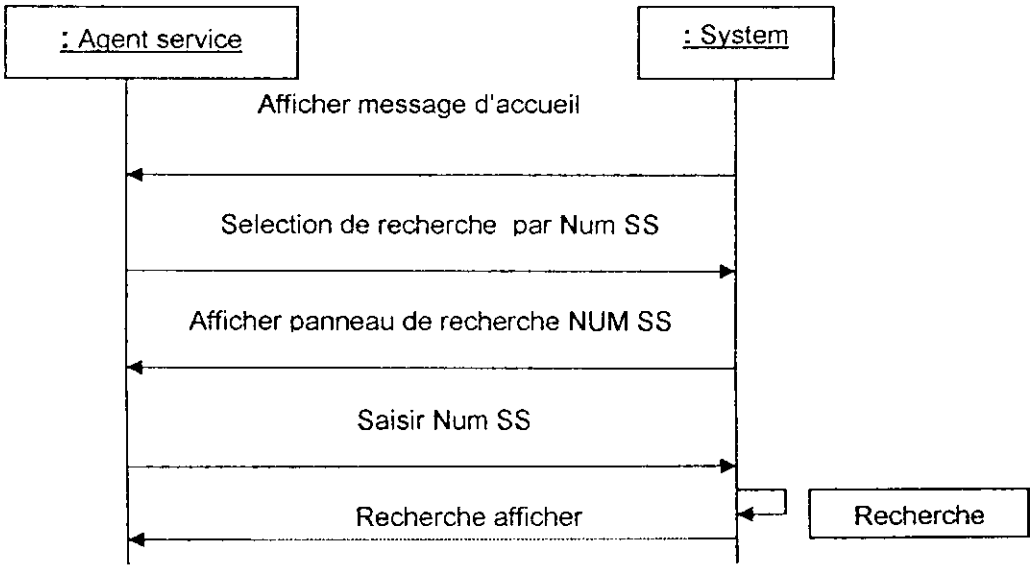


Figure 4-23 Recherche pare Numéro de sécurité social

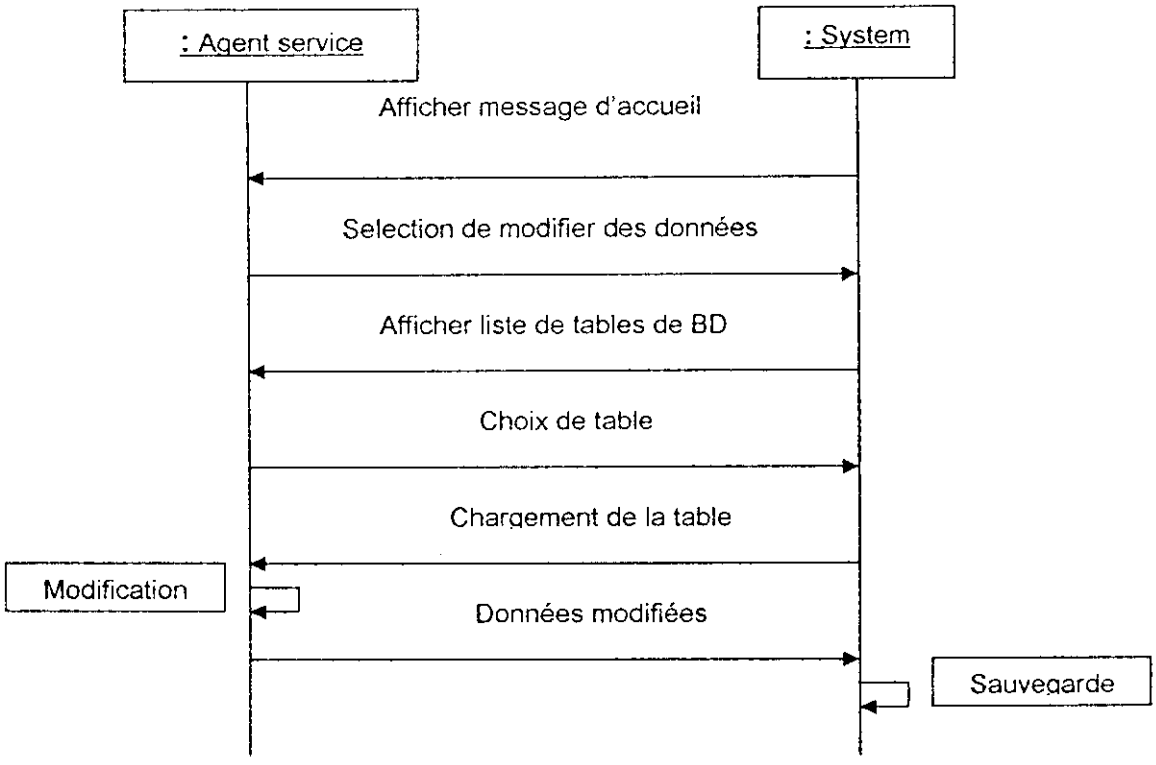


Figure 4-24 Modifier des données de la base

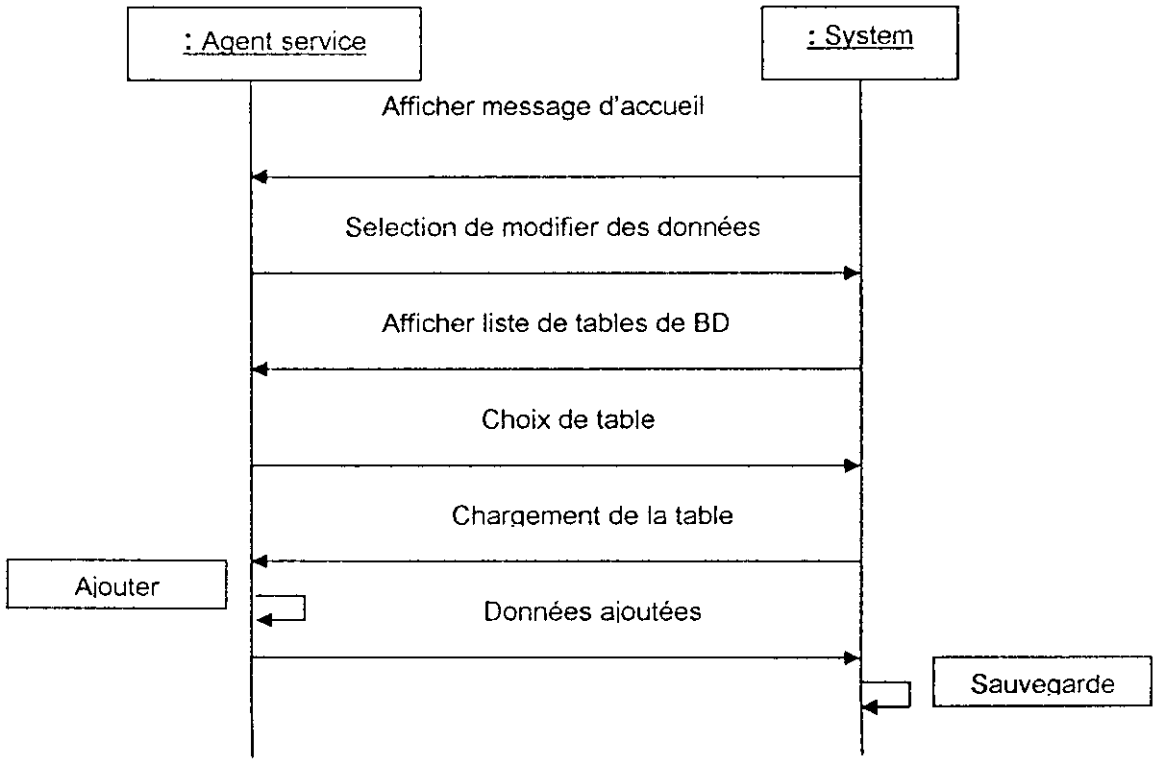


Figure 4-25 Ajouter des données a la base

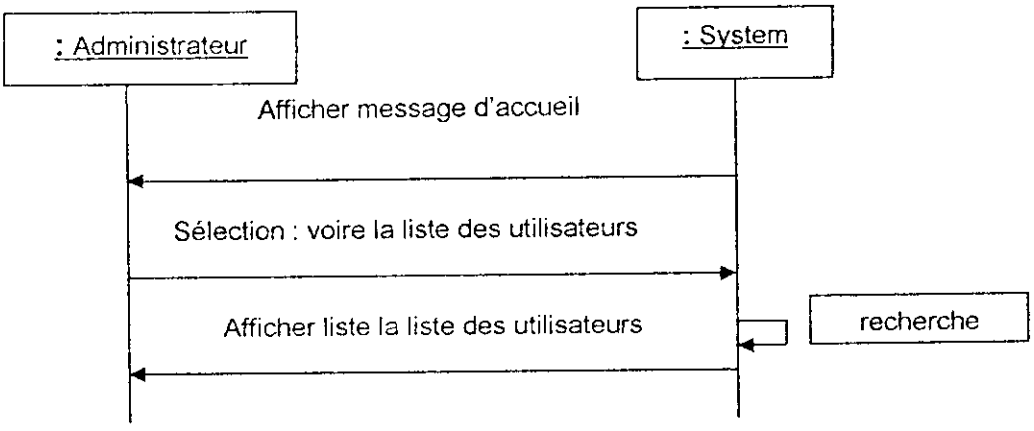


Figure 4-26 Consulter la liste des utilisateurs

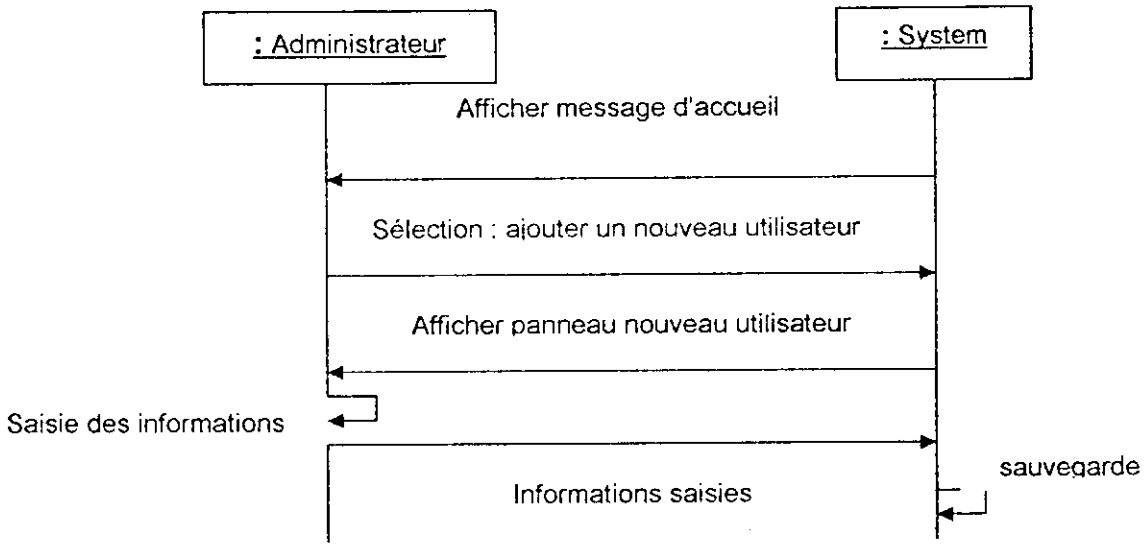


Figure 4-27 Ajouter un utilisateur

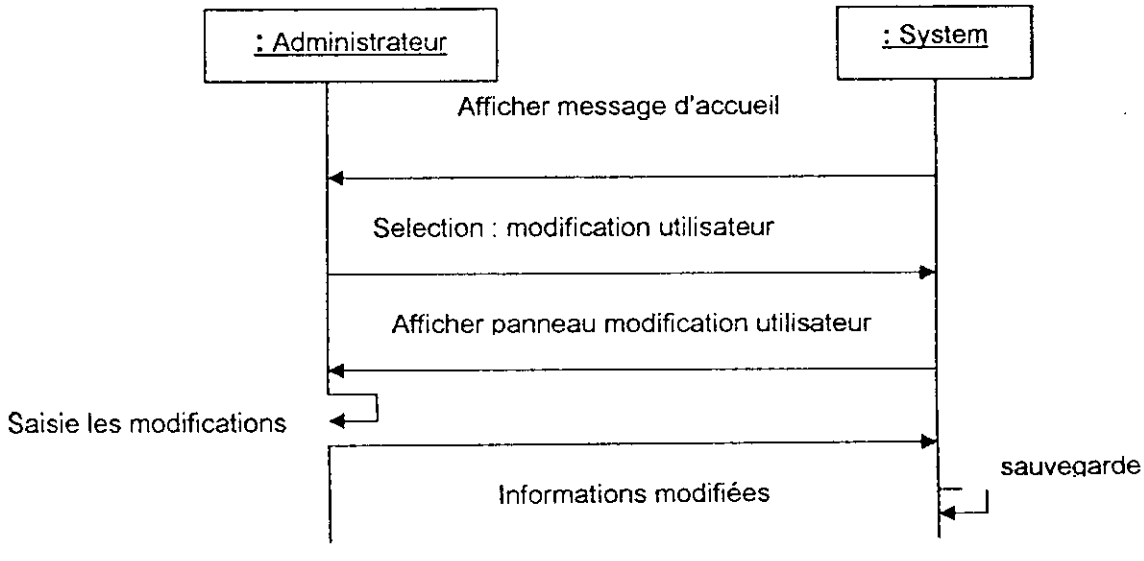


Figure 4-28 Modification utilisateur

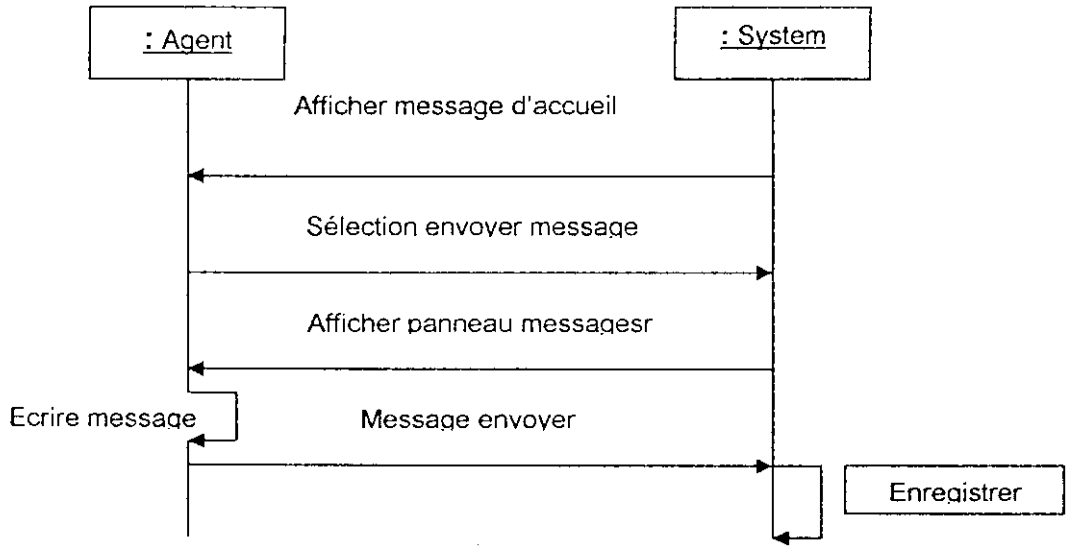


Figure 4-29 Envoyer message

- **Diagrammes de collaborations**

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait Figurer les envois de messages.

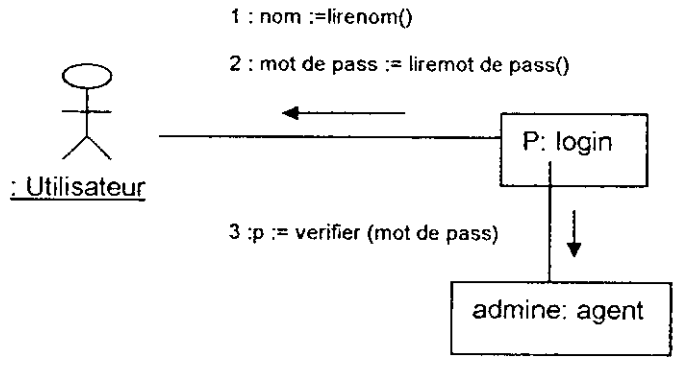


Figure 4-30 Diagramme de collaboration –indentification-

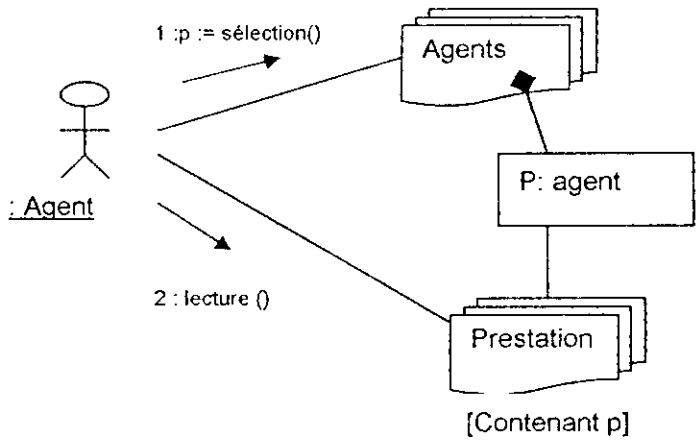


Figure 4-31 Digramme de collaboration de consultation prestation relatif a une personne

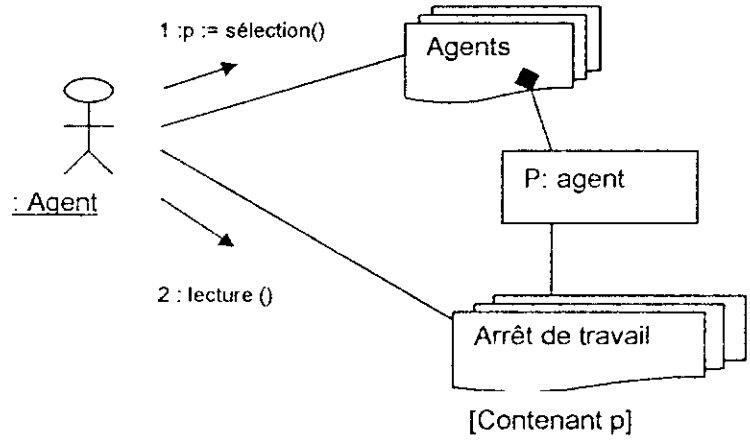


Figure 4-32 Diagramme de collaboration consulter arrêt de travail relatif à une personne

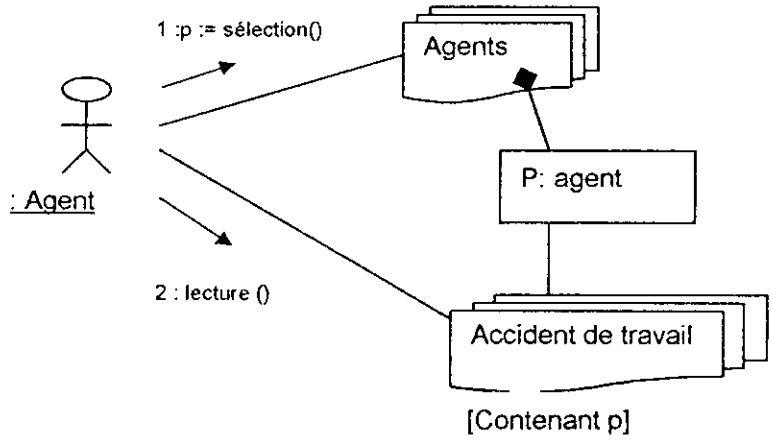


Figure 4-33 Diagramme de collaboration consulter accident de travail relatif à une personne

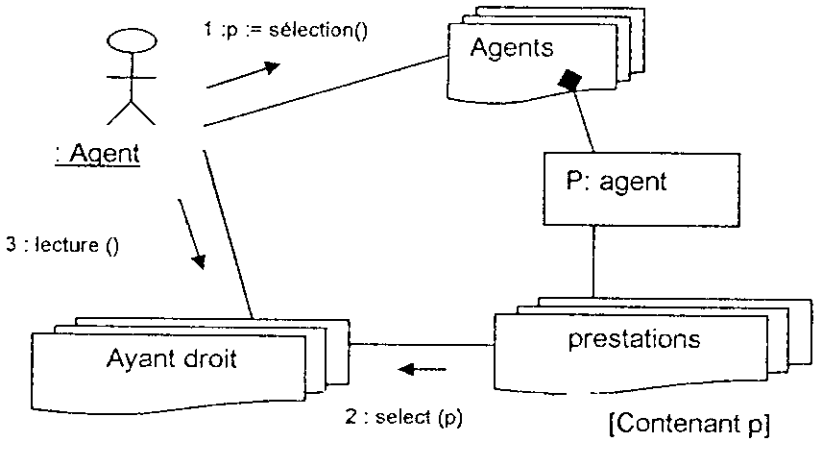


Figure 4-34 Diagramme de collaboration consulter ayant droit relatif à une personne

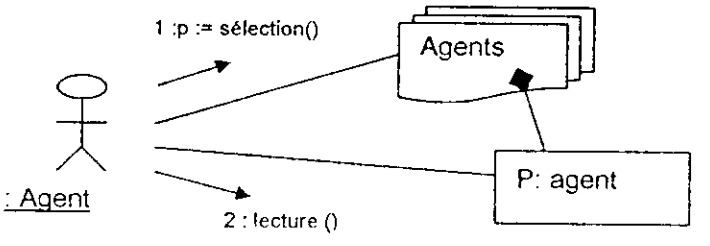


Figure 4-35 Diagramme de collaboration consulter information sur un agent

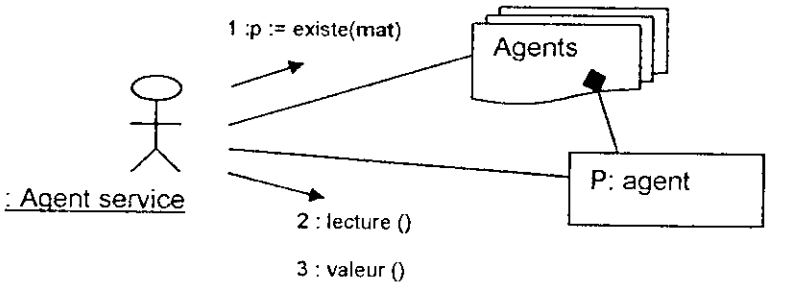


Figure 4-36 Recherche information sur un agent par matricule

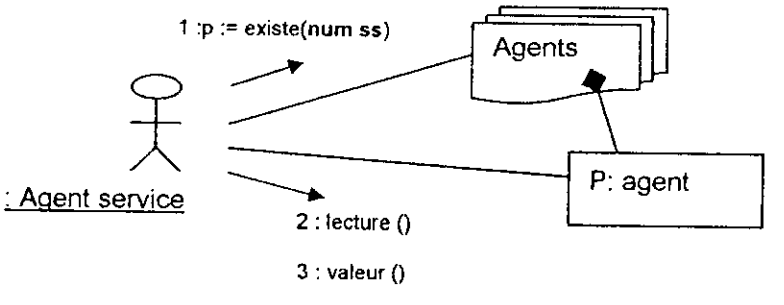


Figure 4-37 Recherche information sur un agent par numéro de sécurité sociale

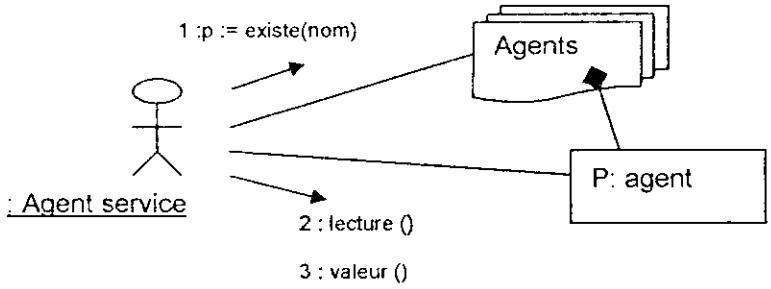


Figure 4-38 Recherche information sur un agent par nom

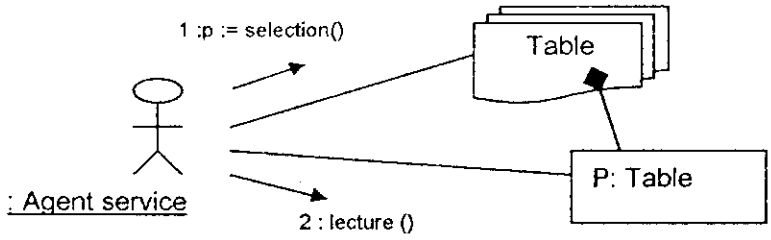


Figure 4-39 Consulter la base de données

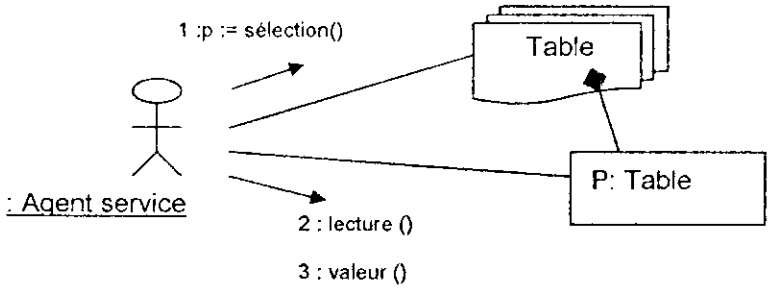


Figure 4-40 Modification de données de la base

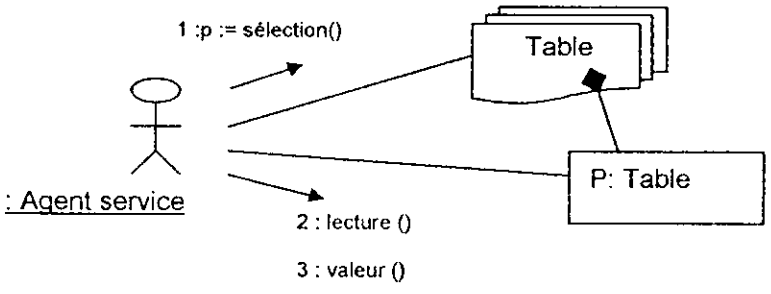


Figure 4-41 Ajouter des données à la base

4. CONCEPTION DE SYSTEME

4.1. Règles de passage entre modèle objet et modèle tables relationnelles

➢ Représentation des classes d'objets en tables :

- chaque classe est représentée par un ou plusieurs tables.

➢ Représentation des associations d'objets en tables :

- Chaque association plusieurs à plusieurs est représentée par une table distincte.
- Une association un à plusieurs est représentée par une table distincte ou peut être enfouie comme clé étrangère dans la table pour l'une ou l'autre des classes. Pour les associations un à plusieurs ou un à un, il n'y a pas de cycle, on dispose de l'option supplémentaire qui consiste à ranger l'association et les objets liés dans une seule table.

Ayant conscience que cela peut introduire une redondance et violer des formes normales.

- Les noms de rôles sont incorporés en tant que partie du nom de l'attribut de la clé étrangère.
- Les associations n-aires (n>2) se représentent par des tables distinctes. Ce qui aide parfois à promouvoir une association n-aires en une classe.
- Une association qualifiée se représente en une table distincte avec au moins trois attributs, la clé primaire de chaque classe liée est le qualificatif.
- Les agrégations suivent les mêmes règles que les associations.

➢ **Représentation de la généralisation de l'héritage simple en tables :**

- On représente chaque superclasse et chaque sous-classe par une table.
- S'il n'y a de table de superclasse, les attributs sont dupliqués dans chaque table de sous-classe.
- S'il n'y a de table de sous-classe, on apporte tous les attributs de sous-classe dans la superclasse.

4.2. Traduction du modèle objet en base de données relationnelles

En appliquant les règles de passage au modèle objet, on obtient la représentation logique de la base de données qui se représente comme suit :

Nom fichier : AGENT		
Nom champ	**type**	**longueur**
Matricule	AN	06
Num sécurité sociale	N	12
Nom AG	A	15
Prénom AG	A	15
Date de naiss AG	D	08
Lieu de naiss AG	AN	15
Adr AG	A	01
Situation_familale AG	AN	25
Sexe AG	A	01
Nationalité AG	A	15
Code fonction	AN	10
Num caisse	AN	05
N° affiliation	N	06
Date affilmiation	N	12
	D	08

Tableau 4-2- Représentations logique de la classe agents

Nom fichier : TYPE PRESTATION		
Nom champ	**type**	**longueur**
Code type prestation	A	03
libelle	AN	15

Tableau 4-3 Représentations logique de la classe type prestation

Nom fichier : ARRET DE TRAVAIL		
Nom champ	**type**	**longueur**
Num_arret_travail	N	06
Date_dépot ART	D	08
Date_debut ART	D	08
Date_fin ART	D	08
Date_etablisement ART	D	08
Date de remboursement	D	08
Date de reprise	D	08
Code type arrêt de travail	A	03
matricule	AN	06

Tableau 4-4 Représentations logique de la classe arrêt de travail

Nom fichier : ACCIDENT DE TRAVAIL		
Nom champ	**type**	**longueur**
Num_accident	N	03
Date accident	D	08
Lieu accident	AN	25
Jour accident	A	07
Heur accident	N	05
Détail accident	AN	15
Degré accident	AN	20
matricule	AN	06

Tableau 4-5 Représentations logique de la classe accident de travail

Nom fichier : STRUCTURE		
Nom champ	**type**	**longueur**
Code structure	AN	05
Désignation	A	25

Tableau 4-6 Représentations logique de la classe structure

Nom fichier : PRESTATION		
Nom champ	**type**	**Longueur**
Num prestation	N	08
Date depot	D	08
Date des soins	D	08
Montant CNAS	N	07
Date d'envoi cheque CNAS	D	08
Montant MIP	N	07
Date d'envoi cheque MIP	D	08
Centre des soins	AN	15
Observation	A	20
Code ayant droit	N	02
Code type prestation	A	03
matricule	AN	06

Tableau 4-7 Représentations logique de la classe prestation

Nom fichier : FONCTION		
Nom champ	**type**	**longueur**
Code fonction	AN	10
Libelle	AN	20
catégorie	A	02

Tableau 4-8 Représentations logique de la classe fonction

Nom fichier : AYANT DROIT		
Nom champ	**type**	**longueur**
Code ayant droit	N	02
Nom ayant droit	A	15
Prénom ayant droit	A	15
Date de naissance AD	D	08
Situation AD	A	01
Nationalité AD	A	15
Sexe AD	A	01
Adresse AD	AN	25
Code ayant droit	A	03

Tableau 4-9 Représentations logique de la classe ayant droit

Nom fichier : TYPE ARRET DE TRAVAIL		
Nom champ	**type**	**longueur**
Code type arrêt de travail	A	03
Libelle type arrêt de travail	AN	20

Tableau 4-10 Représentations logique de la classe type arrêt de travail

Nom fichier : TYPE AYANT DROIT		
Nom champ	**type**	**longueur**
Code type ayant droit	A	03
libelle	AN	20

Tableau 4-11 Représentations logique de la classe type ayant droit

Nom fichier : CAISSE		
Nom champ	**type**	**longueur**
Num caisse	N	06
Agence affiliation	N	06
Adresse affiliation	AN	30
tel	N	09

Tableau 4- 12 Représentations logique de la classe caisse

4.3. CONCEPTION DES OBJETS

Chaque table de schéma conceptuel est représentée par la définition d'un objet au niveau du modèle objet externe, en les raffinant avec l'ajout des méthodes qui sont directement déduites des actions du modèle dynamique ou des traitements du modèle fonctionnel et en respectant les règles de passage énoncées précédemment. La conception des objets du système est définie par les prototypes des classes suivantes :

classe : AGENT	
Attribut	**méthode**
Matricule	Sélection ()
Num sécurité sociale	Image ()
Nom AG	Valeur ()
Prénom AG	Existe (string)
Date de naiss AG	Existe (double)
Lieu de naiss AG	
Adr AG	
Situation_familale AG	
Sexe AG	
Nationalité AG	
Code fonction	
Num caisse	
N° affiliation	
Date affilmiation	

Tableau 4-13 prototype de la classe agents

classe : prestation	
Attribut	**méthode**
Num prestation	Sélection ()
Date dépôt	Image ()
Date des soins	Valeur ()
Montant CNAS	Existe (double)
Date d'envoi cheque CNAS	
Montant MIP	
Date d'envoi cheque MIP	
Centre des soins	
Observation	
Code ayant droit	
Code type prestation	
matricule	

Tableau 4-14 prototype de la classe prestation

classe : arrêt de travail	
Attribut	**méthode**
Num_arret_travail	Sélection ()
Date_dépot ART	Image ()
Date_debut ART	Valeur ()
Date_fin ART	Existe (double)
Date_etablisement ART	
Date de remboursement	
Date de reprise	
Code type arrêt de travail	
matricule	

Tableau 4-15 prototype de la classe arrêt de travail

classe : accident de travail	
Attribut	**méthode**
Num_accident	Sélection ()
Date accident	Image ()
Lieu accident	Valeur ()
Jour accident	Existe (double)
Heur accident	
Détail accident	
Degré accident	
matricule	

Tableau 4-16 prototype de la classe accident de travail

5. CONCLUSION

L'objectif de la modélisation est de comprendre le problème, son domaine d'application et de construire une conception correcte avant de passer à l'implémentation.

Les trois étapes de l'étude ont été étudiées, l'étape statique, dynamique et fonctionnelle. Reste à entamer l'implémentation.

Chapitre 5

Implémentations et résultats

1. Introduction :

Au cours de notre étude, plusieurs résultats ont été obtenu, notamment le modèle dynamique, le modèle fonctionnel, la conception des objets du système.
L'ensemble des ces résultats a été soumis à l'approbation de la direction SONATRACH.
Les résultats de cette étude feront l'objet de projection sur machine, selon l'outil choisi pour le niveau physique et permettront de rendre la solution opérationnelle.

2. Environnement technique de développement

2.1 Présentation des langages de programmation utilisés

Pour l'implémentation et la gestion de la base de données, et le développement de l'application on a utilisé les langages suivants :

2.1.1. Le langage SQL

Pour la mise en place et la gestion de la base de données, le langage utilisé est le SQL, donc parfaitement compatible avec la plate forme choisie, de plus il est entièrement conçu pour le web.
Le SQL est u langage de requête structuré (**S**tructured **Q**uery **L**angage) destiné à communiquer avec des bases de données relationnelles implémentées dans des SGBDR
Le langage SQL permet d'effectuer divers opération comme la création, l'extraction, la modification, la suppression, la fusion, etc..., sur des collections de données, par l'intermédiaire d'instructions particulière appelées des commande, souvent assistées d'ailleurs par des clauses ou des options.

2.1.2. Les JSP :

Les JSP (Java Server Pages) sont un standard permettant de développer des applications Web interactives, c'est-à-dire dont le contenu est dynamique. C'est-à-dire qu'une page web JSP (repérable par l'extension .jsp) aura un contenu pouvant être différent selon certains paramètres (des informations stockées dans une base de données,

Les JSP sont intégrables au sein d'une page Web en HTML à l'aide de balises spéciales permettant au serveur Web de savoir que le code compris à l'intérieur de ces balises doit être interprété afin de renvoyer du code HTML au navigateur du client.
Les JSP permettent donc d'écrire facilement des servlets, en incluant dans des balises spécifiques le code JSP au sein du fichier HTML. De cette façon, elles fournissent une technologie rapide afin de créer des pages dynamiques.
De plus, les JSP étant basées sur Java côté serveur, elles possèdent toutes les caractéristiques faisant la force de Java :

les JSP sont multithreadées,
les JSP sont portables,
les JSP sont orientées objet,
les JSP sont sûres,

2.2. Implémentation

La base de données a été implémentée avec oracle 9i :

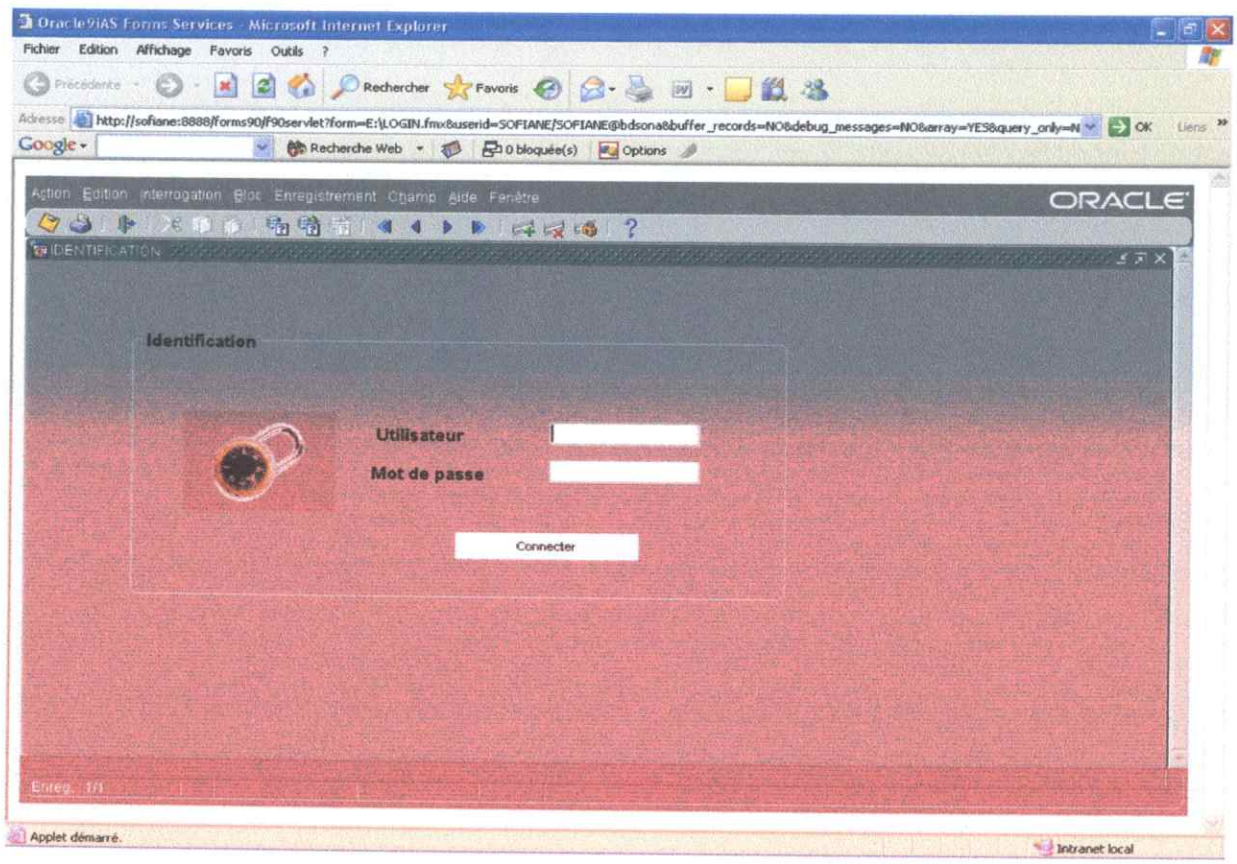
2.2.1 Oracle 9i :

C'est un system relationnel de gestion de base de données Multi-Utilisateurs à haute performance tournant sous le système d'exploitation Windows (XP/NT/2000 serveur.) ou linux, il est conçu pour être exploité sur un ordinateur serveur et être le composant serveur dans un environnement trois-tiers

2.3. Test de l'application :

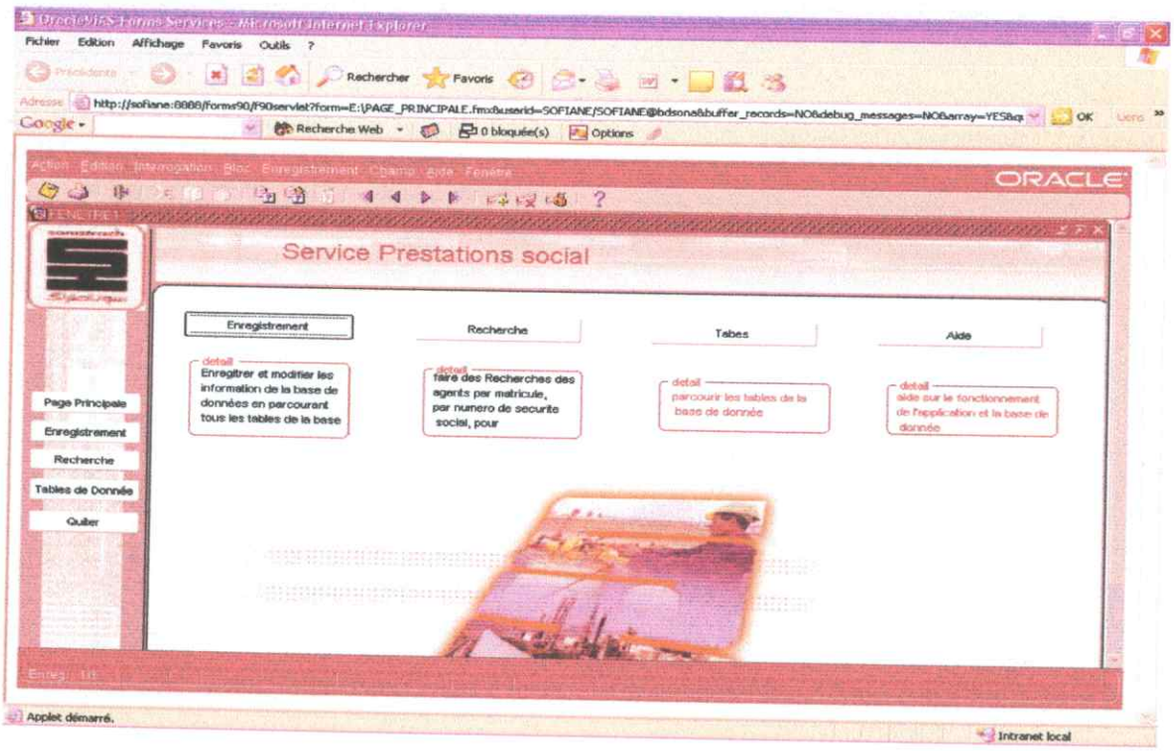
L'utilisateur de notre application web peu faire le suivi des dossiers des prestations sociales en utilisant l'intranet de la société ou l'Internet.

L'utilisateur doit saisir son nom et son mot de passe.

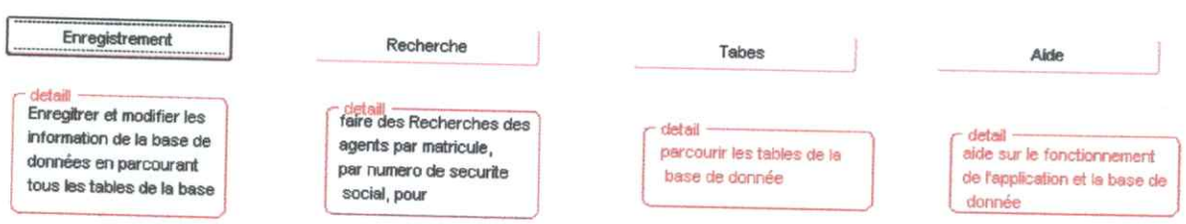


Apres la saisie un composant s'occupe de vérifier la validation des informations en entrée, en cas d'erreur, un message s'affiche pour l'indiquer

Si les données sont valides, l'utilisateur aura une page contenant toutes les opérations qu'il peut effectuer, dans se cas c'est l'agent du service qui est identifiée.

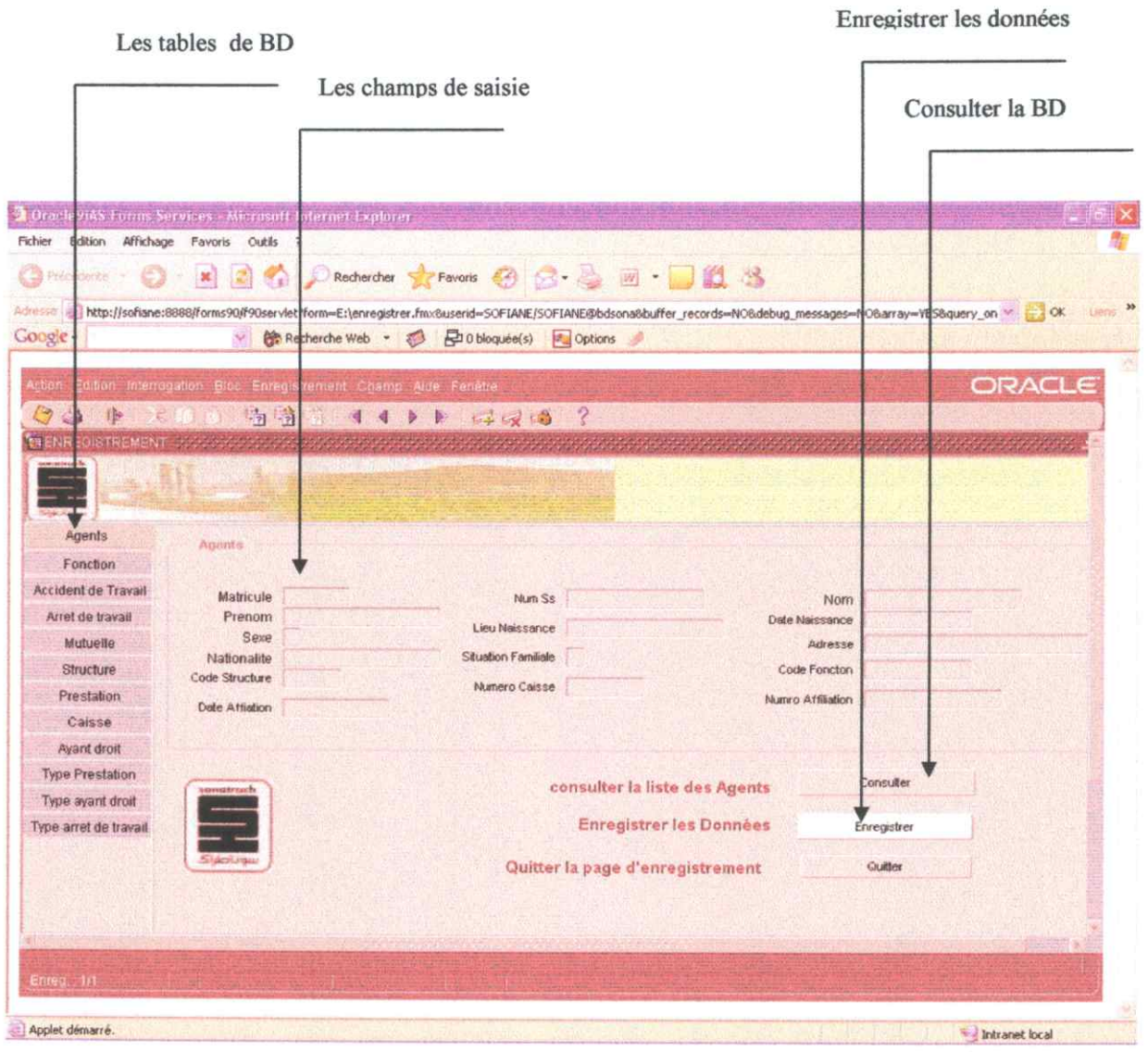


La page principale permet d'accéder aux différentes fonctions de l'application, dans ce qui suit nous allons détailler chaque sous partie.

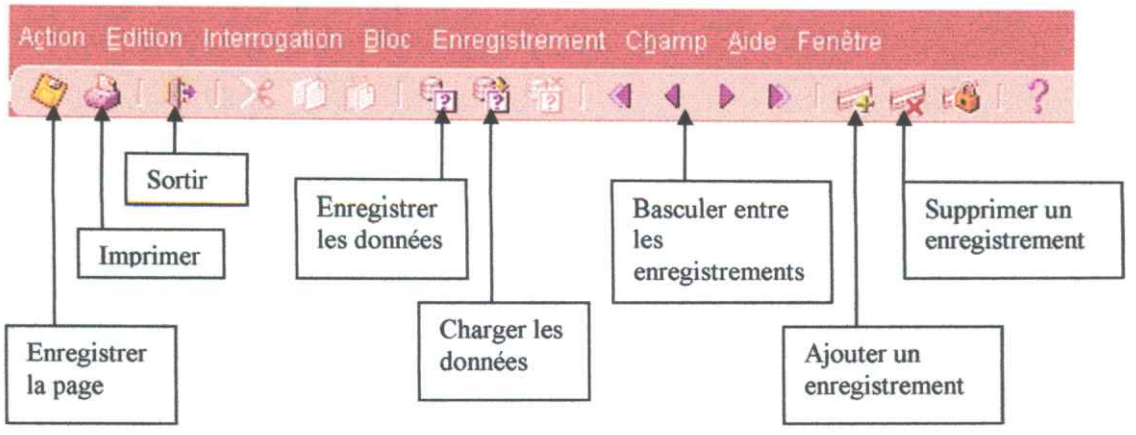


2.3.1. Page Enregistrement :

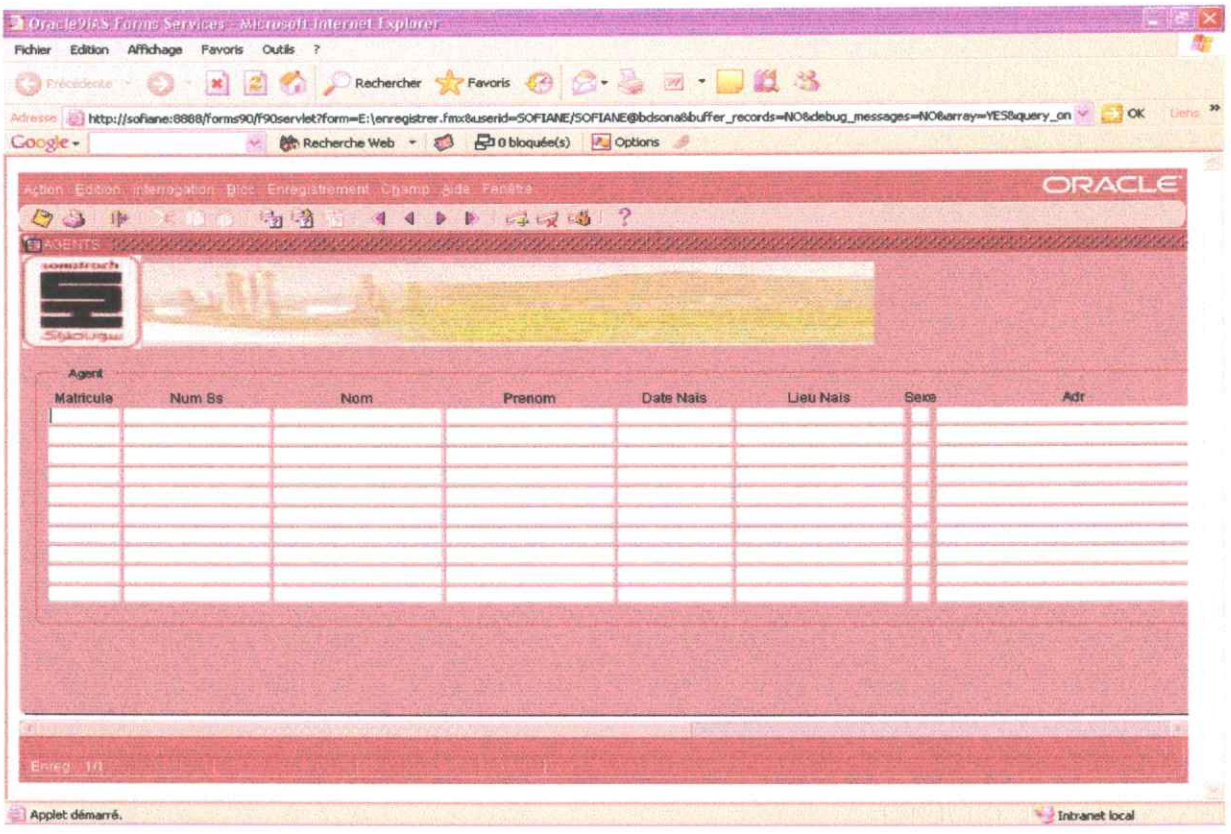
La page enregistrement permet de saisir les données dans la base et de consulter, en va détailler cette partie :



- zone de contrôle : permet de charger, supprimer ou modifier les données

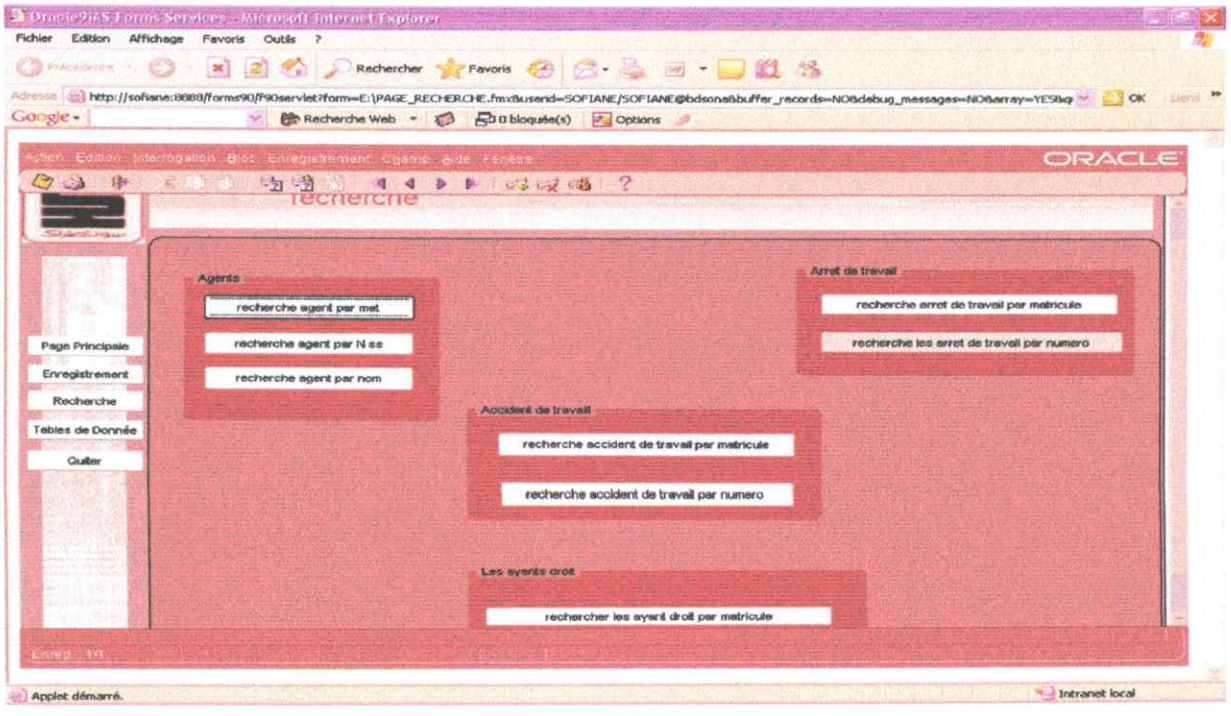


En choisissant de consulter le BD on aura la page suivante :

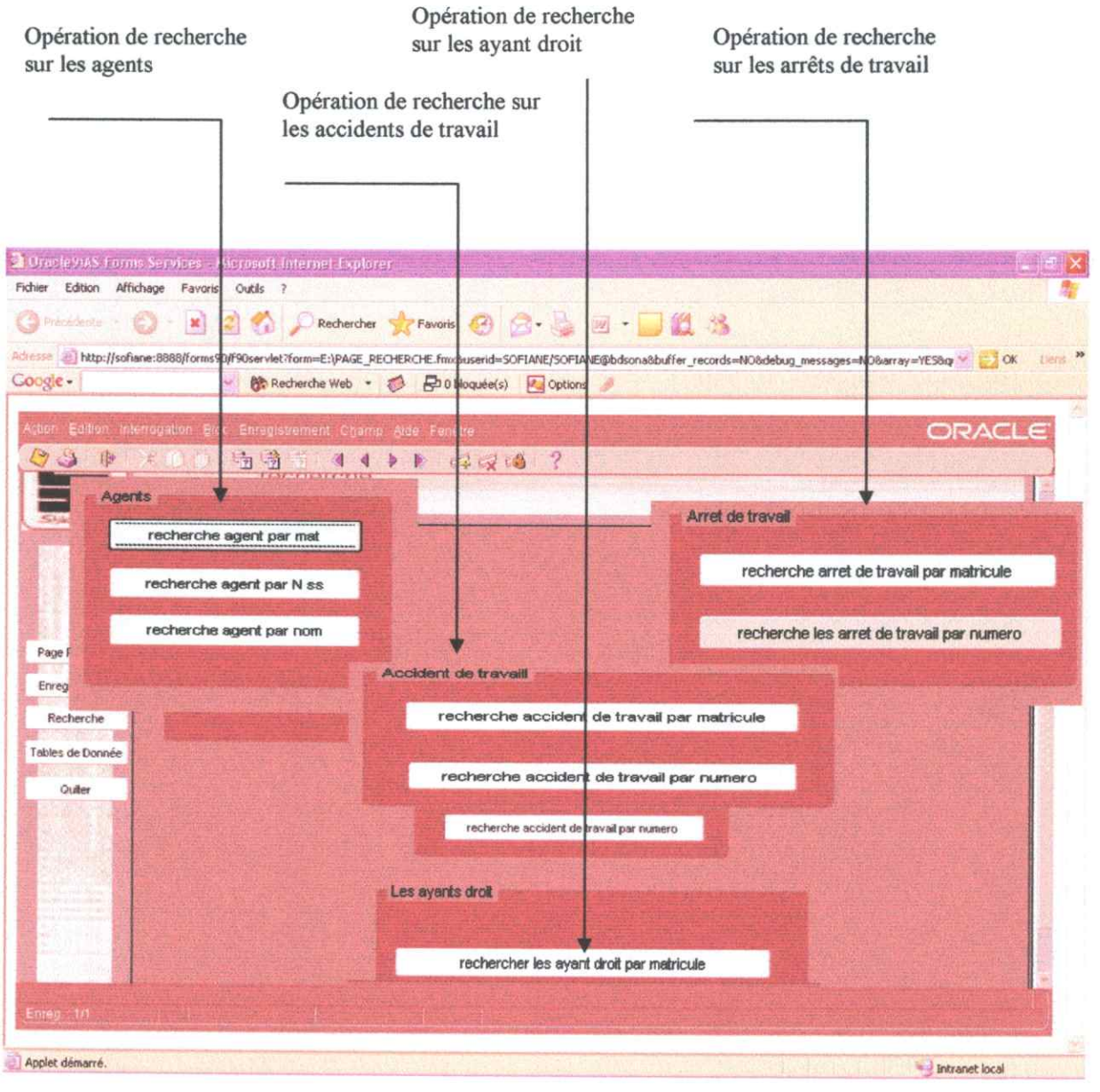


2.3.2. Page de recherche :

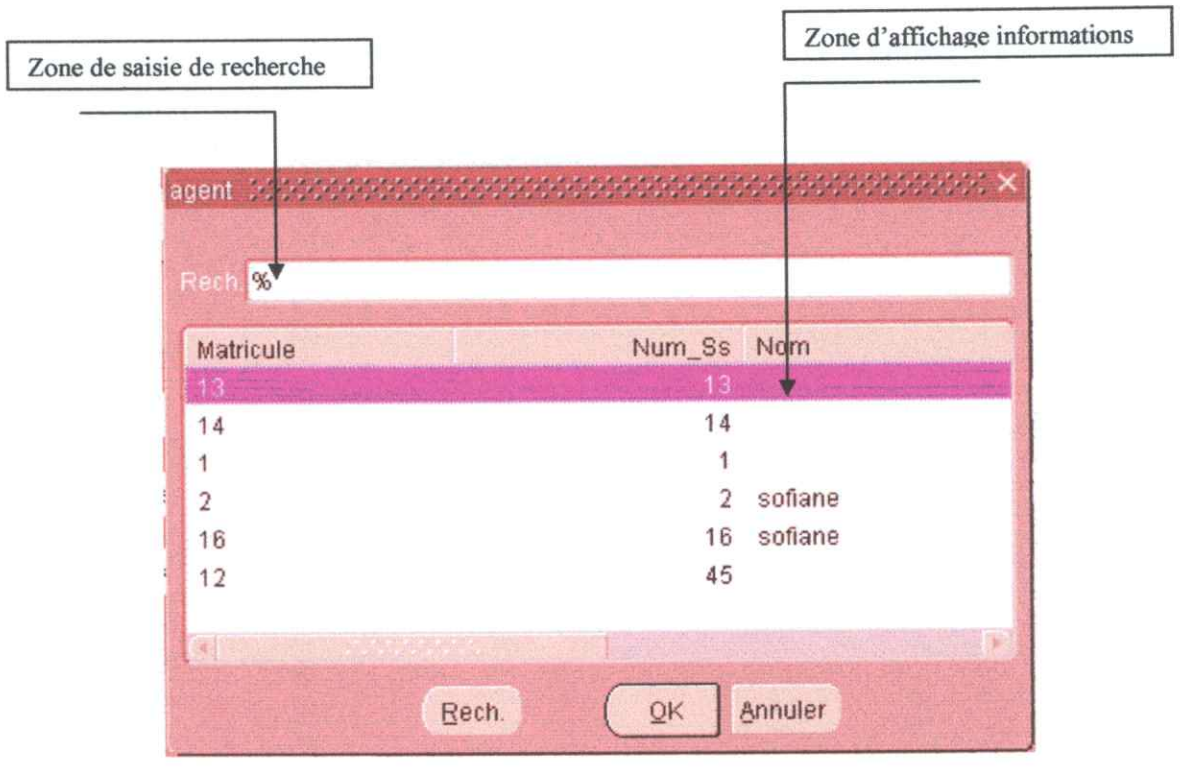
La page de recherche permet de faire des recherches ciblées notamment sur les agents, les arrêts de travail, les accidents de travail, etc....



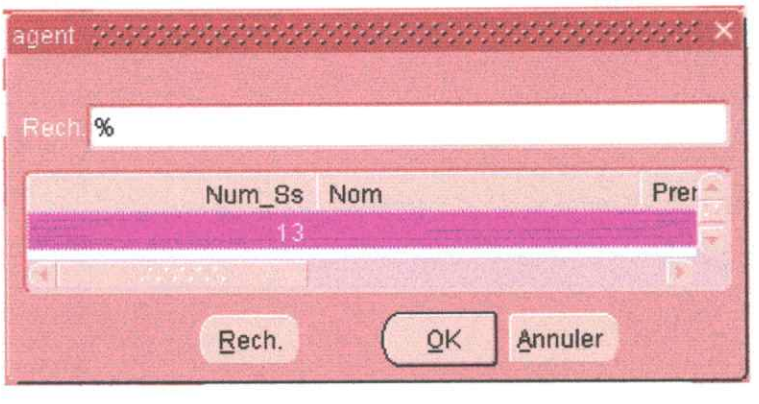
En détaillera cette page et son fonctionnement dans ce qui va suivre :



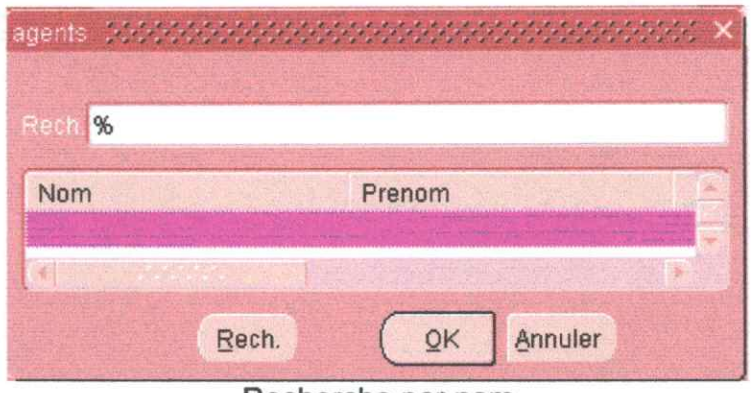
On choisissant l'anglet agent la figure suivante apparaît, et on pourra faire des recherche concernant les agents, comme recherche par matricule ou par numéro de sécurité sociale, etc..



Recherche par matricule.



Recherche par numéro de sécurité sociale.



Recherche par nom

3. conclusion :

Dans cette phase nous sommes arrivés à faire une application web pour le service prestations sociales qui fait le suivi et la gestion du personnelles de direction en appliquant les techniques d'accès aux bases de données ,et on utilisant les outils de développement tel que Oracle9i, Oracle IDS, JSP,etc....

Conclusion générale :

Notre travail ayant pour objectif la conception et la réalisation d'une base de données pour le service prestations sociales fonctionnant sur le réseau Internet et intranet, nous avons réalisé l'application en utilisant une architecture dite architecture trois-tiers, et une approche composant comme technique d'accès a la base de données a partir d'un serveur d'applications

A partir de ce serveur d'applications nous avons élaboré des composants JavaBeans qui effectuent des tâches bien spécifiques telle que charger, enregistrer des informations dans un serveur de données, faire une connection persistante a la base de données, etc.... puis nous avons entamé l'étape d'analyse qui nous a permis d'aboutir aux résultats attendus.

De l'approche méthodologique et du langage de modélisation adopté nous avons réalisés une base données au moyens de la méthode OMT et du langage UML avec le SGBD oracle 9i sous l'environnement Windows. Concernant la partie présentation nous avons utilisés les JSP et Oracle IDS

Grâce à la technique d'accès aux données a partir d'un serveur d'applications par l'utilisation des entités JavaBeans nous avons obtenu les résultats escomptés. Ces résultats ont été soumis à l'appréciation des responsables concernés de la direction générale de SONATRACH et nous espérons qu'ils constitueront des arguments valables pour qu'il soit intégré dans leur système.

Annexes

A. Internet

B. Le SQL

C. Présentation de Sonatrach

1 - INTRODUCTION

Contrairement aux idées reçues, Internet n'est pas un phénomène récent. C'est en 1969 que le gouvernement américain (plus précisément le DOD : Department Of Defense, soit le ministère de la défense) décide de mettre en place un système (Arpanet) pour relier les ordinateurs entre eux à des fins de défense nationale. En 1986, la NSF (National Science Foundation) et la NASA décident d'interconnecter les universités américaines. Très rapidement, ce système sera étendu à de nombreuses universités étrangères. Depuis 1992, Internet est devenu accessible au grand public.

Internet repose sur trois principaux types de services :

- La messagerie (e-mail ou mél en français) qui permet l'échange de documents et courriers électroniques. Un aspect intéressant de la messagerie est la possibilité de s'inscrire à une mailing list, c'est à dire émettre et recevoir des messages à une communauté de personnes ayant des centres d'intérêt communs.
- Les forums de discussions (newsgroups) qui permettent à plusieurs individus de se regrouper autour d'un même thème.
- Le Web permet d'accéder à des pages via un navigateur (browser). Ces pages contiennent du texte, des données, des fichiers, des images, du son, ...

Depuis son début, Internet connaît une croissance très forte. Le nombre de serveurs a presque doublé sur un an (source : http://www.nw.com/zone/host-count-history), passant de 19 à 38 millions en juillet 1998. Selon IDC, les dépenses liées au développement de sites passeront de 211 milliards de dollars cette année à un montant estimé de 954 Milliards de dollars en 2002. Les dépenses spécifiques au commerce en ligne passeront, toujours sur la même période, de 17 à 167 Milliards de dollars. Enfin, selon le groupe Forrester (http://www.forrester.com), le nombre de sites doublerait tous les 53 jours.

Toutes les activités informationnelles de la vie courante peuvent être réalisées sur Internet :

Consulter des bibliothèques, des rapports, des études, voir des expositions, visiter des écoles, contacter des experts, écouter la radio, voir la télévision, lire un journal ou un livre, "rencontrer" des gens et échanger des idées, ... L'information sur Internet est très, voire trop abondante.

2 - OUVERTURE A INTERNET

Internet, et son petit frère à l'échelle de l'entreprise intranet, deviennent aujourd'hui des éléments incontournables du système d'information. Le succès récent de cette technologie relativement ancienne est du à l'apparition du World Wide Web (WWW) en 1989, permettant à un utilisateur d'accéder au réseau Internet à travers un navigateur (browser). Ce dernier lui permet de visualiser les informations sous une forme ergonomique, sans avoir besoin de connaissances en informatique.
Les pages visualisées ne sont pas stockées sur le poste client mais sont envoyées, à la demande, par un serveur Web. Elles sont écrites en langage HTML (Hyper Text Markup Langage) et contiennent du texte formaté, des liens vers d'autres documents ou d'autres parties de la page présentée et des images.

2.1 - ACCES A DES PAGES STATIQUES

Affichage de rapports statiques. L'outil d'aide à la décision doit permettre de générer les rapports sous une forme HTML, afin de les visualiser depuis un navigateur et de les mettre à disposition des autres utilisateurs, par exemple sur un serveur d'entreprise, Intranet. Les pages définies sont alors des pages statiques, contenant à la fois la présentation et les données et non mises à jour dynamiquement. Afin de présenter aux utilisateurs des informations récentes, l'outil doit permettre de régénérer automatiquement les rapports à intervalles réguliers (par exemple tous les jours) ou après chaque chargement de données dans le Data Warehouse.
Automatisme du lien entre l'outil d'aide à la décision et le serveur Web. Le serveur Web devient ici un serveur d'entreprise, permettant simplement de stocker des documents et de les mettre à la disposition de toute personne possédant un navigateur Internet et ayant le droit d'accéder à ces pages. Pour que cette solution soit viable, il faut que l'outil soit capable de mettre automatiquement les pages HTML à disposition sur le serveur Web.
Même si elle apparaît sommaire, cette fonctionnalité peut être assez intéressante. Par exemple, dans le cadre d'une entreprise et d'un réseau intranet, les rapports élémentaires peuvent être mis à la disposition de l'ensemble des utilisateurs qui n'ont

ainsi pas à maîtriser et à employer un outil d'aide à la décision pour accéder aux informations qui les intéressent.
Mais, au delà de ces fonctionnalités statiques, il est également nécessaire d'accéder directement aux données à travers le navigateur et de générer des pages HTML dynamiquement, à la demande de l'utilisateur.

2.2 - ACCES A DES PAGES DYNAMIQUES

Au delà de l'envoi de pages statiques, le serveur Web est aujourd'hui capable de créer dynamiquement des pages, à la demande de l'utilisateur. Ceci peut se faire à travers des scripts CGI (Common Gateway Interface). Ils vont alors se charger d'interroger la base de données. Des interfaces plus évoluées sont proposées par Netscape avec NSAPI et par Microsoft avec ISAPI. Elles sont plus performantes et, contrairement à CGI, ne nécessitent pas la création d'un processus séparé à chaque exécution de script. Par exemple, ISAPI fait appel à des DLLs et non à des exécutables. Plus performantes, elles sont cependant moins fiables car un problème survenant lors de l'exécution d'un script risque de provoquer l'arrêt du processus qui est alors également celui du serveur Web (on parle de démon HTTP).
D'autre part, le langage Javascript permet de joindre des programmes à des pages HTML, afin de soulager le serveur et d'exécuter certains programmes au niveau client. Il peut s'agir, par exemple, de programmes permettant de ontrôler si l'utilisateur a bien renseigné tous les champs obligatoires dans un formulaire, avant de l'envoyer vers le serveur, afin d'éviter des aller et retours inutiles.
Enfin, le langage Java permet de créer de petites applications (appelées des applets) qui pourront être chargées directement sur le poste client et exécutées à partir du navigateur, il faut pour cela que ce dernier soit compatible Java.
Au delà du simple partage de rapports, la génération de pages dynamiques est une caractéristique indispensable. L'utilisateur doit pouvoir formuler ses requêtes et récupérer les résultats à travers son navigateur Internet.
D'autre part, il est nécessaire de lui laisser manipuler les données, par exemple dans le cas d'outils permettant d'effectuer de l'analyse multidimensionnelle, de naviguer dans les données.

Au delà d'applications " clé en main " mises à la disposition de l'utilisateur et lui permettant de manipuler les données dans le cadre qui lui a été imparti, ces outils devraient permettre à l'utilisateur de définir ses requêtes aussi librement qu'il le fait avec l'outil, de même pour la valorisation des résultats. Ceci permet alors de mettre à la disposition de l'ensemble des utilisateurs les données de l'entreprise, évite les coûts et les efforts d'installation et de mise à niveau des produits, l'application étant alors basée sur le serveur.

1. Présentation de SQL

SQL signifie Structured Query Language c'est-à-dire Langage d'interrogation structuré.

En fait SQL est un langage complet de gestion de bases de données relationnelles. Il a été conçu par IBM dans les années 70. Il est devenu le langage standard des systèmes de gestion de bases de données (SGBD) relationnelles (SGBDR).

C'est à la fois :

_ un langage d'interrogation de la base (ordre SELECT)

_ un langage de manipulation des données (LMD; ordres UPDATE, INSERT, DELETE)

_ un langage de définition des données (LDD ; ordres CREATE, ALTER, DROP),

_ un langage de contrôle de l'accès aux données (LCD ; ordres GRANT, REVOKE).

Le langage SQL est utilisé par les principaux SGBDR: DB2, Oracle, Informix, Ingres, RDB,... Chacun de ces SGBDR a cependant sa propre variante du langage. Ce support de cours présente un noyau de commandes disponibles sur l'ensemble de ces SGBDR, et leur implantation dans Oracle Version 7.

2. Normes SQL

SQL a éte normalisé dès 1986 mais les premières normes, trop incomplètes, ont été ignorées par les éditeurs de SGBD.

La norme actuelle SQL-2 (appelée aussi SQL-92) date de 1992. Elle est acceptée par tous mais les SGBD relationnels qui dominent actuellement le marché (en particulier Oracle) ne sont toujours pas totalement adaptés à cette norme.

SQL-2 définit trois niveaux :

_ Full SQL (ensemble de la norme)

_ Intermediate SQL

_ Entry Level (ensemble minimum à respecter pour se dire à la norme SQL-2)

3. Utilitaires associés

Comme tous les autres SGBD, Oracle comprend plusieurs utilitaires qui facilitent l'emploi du langage SQL et le développement d'applications de gestion s'appuyant sur une base de données relationnelle. En particulier SQL-FORMS facilite grandement la réalisation des traitements effectués pendant la saisie ou la modification des données en interactif par l'utilisateur. Il permet de dessiner les écrans de saisie et d'indiquer les traitements associés à cette saisie. D'autres utilitaires permettent de décrire les états de sorties imprimés, de sauvegarder les données, d'échanger des données avec d'autres logiciels, de travailler en réseau ou de constituer des bases de données réparties entre plusieurs sites.

Ce cours se limitant strictement à l'étude du langage SQL, nous n'étudierons pas tous ces utilitaires. Nous verrons les commandes essentielles d'un seul utilitaire, SQLPLUS, qui facilite l'utilisation interactive du langage SQL par un utilisateur. Ces commandes permettent de modifier les ordres SQL et de constituer des fichiers de commandes SQL que l'on peut réutiliser ensuite.

Les commandes du langage SQL peuvent être tapées directement au clavier par l'utilisateur ou elles peuvent être incluses dans un programme écrit dans un langage de troisième génération (Cobol, Langage C, Fortran, Ada,...) grâce à un précompilateur fourni par Oracle.

4. SGBD Oracle

Oracle est un SGBD (système de gestion de bases de données) édité par la société du même nom (Oracle Corporation - http://www.oracle.com/), leader mondial des bases de données.

La société *Oracle Corporation* a été créée en 1977 par Lawrence Ellison, Bob Miner, et Ed Oates. Elle s'appelle alors *Relational Software Incorporated (RSI)* et commercialise un Système de Gestion de Bases de données relationnelles (SGBDR ou RDBMS pour *Relational Database Management System*) nommé *Oracle.*

En 1979, le premier prototype (RDBMS - RSI1) intégrant la séparation des espaces d'adressage entre les programmes utilisateurs et le noyau Oracle est commercialisé. Cette version est entièrement développée en langage assembleur. La seconde version (RDBMS - RSI2) est un portage de l'application sur d'autres plates-formes.

En 1983 la troisième version apporte des améliorations au niveau des performances et une meilleure prise en charge du SQL. Cette version est entièrement codée en langage C. A la même époque RSI change de raison sociale et devient *Oracle*.

En 1984 la première version d'Oracle (Oracle 4) est commercialisée sur les machines IBM.

En 1985 Oracle 5 permet une utilisation client-serveur grâce au middleware *SQL*Net*.

En 1986 Oracle a été porté sur la plateforme 8086.

En 1988 Oracle 6 est disponible sur un grand nombre de plates-formes et apporte de nombreuses nouvelles fonctionnalités ainsi qu'une amélioration notable des performances.

En 1991, Oracle 6.1 propose une option *Parallel Server* (dans un premier temps sur la DEC VAX, puis rapidement sur de nombreuses autres plates-formes).

En 1992, Oracle 7 sort sur les plates-formes UNIX (elle ne sortira sur les plates-formes Windows qu'à partir de 1995). Cette version permet une meilleure gestion de la mémoire, du CPU et des entrées-sorties. La base de données est accompagnée d'outils d'administration (SQL*DBA) permettant une exploitation plus aisée de la base. En 1997, la version Oracle 7.3 (baptisée *Oracle Universal Server*) apparaît, suivie de la version 8 offrant des capacités objet à la base de données

Oracle est écrit en langage C et est disponible sur de nombreuses plates-formes matérielles (plus d'une centaine) dont :

- AIX (IBM)
- Solaris (Sun)
- HP/UX (Hewlett Packard)

- Windows NT (Microsoft)

Oracle depuis la version 8.0.5 est disponible sous Linux

Les versions d'Oracle

Oracle se décline en plusieurs versions

- Oracle Server **Standard**, une version comprenant les outils les plus courants de la solution Oracle. Il ne s'agit pas pour autant d'une version bridée...
- Oracle Server **Enterprise Edition**

Les fonctionnalités d'Oracle

Oracle est un SGBD permettant d'assurer :

- La définition et la manipulation des données
- La cohérence des données
- La confidentialité des données
- L'intégrité des données
- La sauvegarde et la restauration des données
- La gestion des accès concurrents
-

Les composants d'Oracle

Outre la base de données, la solution Oracle est un véritable environnement de travail constitué de nombreux logiciels permettant notamment une administration graphique d'Oracle, de s'interfacer avec des produits divers et d'assistants de création de bases de données et de configuration de celles-ci.

On peut classer les outils d'Oracle selon diverses catégories :

- Les outils d'administration
- Les outils de développement
- Les outils de communication
- Les outils de génie logiciel
- Les outils d'aide à la décision

Les outils d'administration d'Oracle

Oracle est fourni avec de nombreux outils permettant de simplifier l'administration de la base de données. Parmi ces outils, les plus connus sont :

- Oracle Manager (SQL*DBA)
- NetWork Manager
- Oracle Enterprise Manager
- Import/Export : un outil permettant d'échanger des données entre deux bases Oracle

Outils de développement d'Oracle

Oracle propose également de nombreux outils de développement permettant d'automatiser la création d'applications s'interfaçant avec la base de données. Ces outils de développement sont :

- Oracle Designer
- Oracle Developer
- SQL*Plus : une interface interactive permettant d'envoyer des requêtes SQL et PL/SQL à la base de données. SQL*Plus permet notamment de paramétrer l'environnement de travail (formatage des résultats, longueur d'une ligne, nombre de lignes par page, ...)
- Oracle Developper : il s'agit d'une suite de produits destinés à la conception et à la création d'applications client-serveur. Il est composé de 4 applications :
    - Oracle Forms (anciennement SQL*Forms) : un outil permettant d'interroger la base de données de façon graphique sans connaissances préalables du

langage SQL. SQL*Forms permet ainsi de développer des applications graphiques (fenêtres, formulaires, ...) permettant de sélectionner, modifier et supprimer des données dans la base.

- Oracle Reports (SQL*ReportWriter) : un outil permettant de réaliser des états
- Oracle Graphics : un outil de génération automatique de graphiques dynamiques pour présenter graphiquement des statistiques réalisées à partir des données de la base
- Procedure Builder : un outil permettant de développer des procédures, des fonctions et des packages

Outils de programmation

Oracle dispose d'un grand nombre d'interfaces (API) permettant à des programmes écrits dans divers langages de s'interfacer avec la base de données en envoyant des requêtes SQL. Ces interfaces (appelées précompilateurs) forment une famille dont le nom commence par *PRO** :

- Pro*C
- Pro*Cobol
- Pro*Fortran
- Pro*Pascal
- Pro*PLI
- ...

3. Les commandes SQL

En SQL, les commandes sont dites de format libre. Cela signifie qu'il n'existe aucune règle indiquant qu'un mot donné doit commencer à une position particulière de la ligne. Cette manière d'écriture est plus lisible.

Quelques commandes SQL :

CREATE Table : permet de créer une table,
DROP Table : pour supprimer une table,
INSERT : en ajoutant des lignes aux colonnes, on encadre les valeurs par des apostrophes,
SELECT : permet d'afficher les données sélectionnées,
UPDATE : pour modifier la valeur d'une donnée,
DELETE : pour supprimer une enregistrement,

Des conditions sont associées à ces commandes grâce à la clause **WHERE**, elle est utilisée pour retrouver les enregistrements qui répondent à une certaine condition. Cette condition est appelée simple. En connectant plusieurs conditions simples par l'utilisation des opérateurs : **AND**, **OR** et **NOT**, on aura les conditions combinées.
La clause **IN** est une manière concise de formuler certaines conditions.

L'opérateur **BETWEEN** n'est pas une fonctionnalité essentielle de SQL, mais il rend toutefois certaines commandes SELECT plus simples.
Dans la plupart des cas, les conditions nécessitent une concordance exacte. Pour cela, l'opérateur **LIKE** avec un caractère joker est utilisé.

L'ordre des lignes dans une table est sans importance dans un SGBD. D'un point de vue pratique, lors de l'interrogation d'une base de données relationnelle, le résultat s'affiche sans ordre prédéfini. Les lignes peuvent s'afficher dans l'ordre choisis spécifié à l'aide de la clause **ORDER BY**.

SQL dispose de fonctions pour calculer :
la somme : en utilisant la fonction **SUM**,
la moyenne : grâce à la fonction **AVG**,
ainsi pour compter, on utilise **COUNT**,
et pour trouver la valeur du maximum et de minimum, en introduisant les fonctions **MAX** et **MIN** respectivement.
L'opérateur **DISTINCT** n'est pas une fonction, mais il peut néanmoins se révéler utile dans certaines situations, couplé avec la fonction **COUNT**.

La clause GROUP BY permet de grouper des données dans un ordre particulier puis de calculer des statistiques si nécessaire.
La clause HAVING est utilisée pour les groupes, en effet, elle fait pour les groupes ce que la commande WHERE fait pour lignes.

En SQL, on réalise la jointure entre tables en incluant une condition dans la clause WHERE de façon à s'assurer que les colonnes en concordance contiennent des valeurs égales. Pour réaliser la jointure, on utilise des sous requêtes avec **IN** et **EXISTS**.

I. PRESENTATION GENERALE

I.1. Présentation de l'organisme d'accueil (SONATRACH)

I.1.1. Dénomination :

Société Nationale pour la recherche, la production, le Transport, la transformation et la Commercialisation des Hydrocarbures.

I.1.2. Forme juridique :

Entreprise publique et économique a sa création, elle est transformée, sans création d'une nouvelle personne morale, en une société par actions (SPA) par le décret présidentiel N°98-48 du 11 Février 1998.

I.1.3. Siége social :

Le siège social de SONATRACH est à Hydra ALGER, sis Djenane El Malik. Il peut être transféré en tout autre lieu du territoire national par délibération de l'assemblée générale.

I.1.4. Capital social :

SONATRACH dispose d'un capital social de **245.000.000.000 (deux cents quarante cinq milliards de dinars)** réparti en deux cent quarante cinq mille actions d'un million de dinars chacune, entièrement et exclusivement souscrit et libellé par l'état.

I.2. Historique de la SONATRACH

L'entreprise nationale SONATRACH (Société nationale de transport de transformation et de commercialisation des hydrocarbures) a été créée le **31/12/1963 (décret 63-491)** pour assurer la responsabilité de la production du transport et la commercialisation des hydrocarbures.

Les missions et les prérogatives de l'entreprise nationale SONATRACH ont été élargies le **22 septembre 1966 (décret 66-626)** ; ainsi ses missions qui se limitaient à l'origine au transport, la transformation et la commercialisation des hydrocarbures ont été élargies à tous les domaines de l'industrie pétrolière, à savoir la prospection, la recherche, la production, le transport, la transformation et la commercialisation des hydrocarbures.

Depuis le **24 février 1971**, date de nationalisation des hydrocarbures, l'entreprise a pris en charge l'ensemble du domaine minier et s'est vue confier le développement de toutes les branches de l'industrie pétrolière.

La SONATRACH est passée de 33 agents en 1964 à 103.000 vers la fin des années 80. Pour assurer une meilleure gestion et améliorer les performances dans le cadre de la politique nationale pour la réorganisation de l'économie du pays, elle entreprend sa restructuration pour donner naissance à 17 entreprises industrielles.

Actuellement, la SONATRACH compte un effectif de 36.000 agents environ et conserve pour sa part la charge des opérations de recherche, de production, de transport par canalisation, de traitement, conditionnement et liquéfaction des hydrocarbures liquides et gazeux.

Dans le cadre de la restructuration décidé en 1982, la SONATRACH a fait l'objet d'un découpage qui a donner naissance à treize entreprises parmi lesquelles, NAFTAL, NAFTEC, ENTP, ASMIDAL, ENSP, ...etc.

I.3. Missions principales de la SONATRACH

Sous l'autorité d'un Directeur Général, la SONATRACH a notamment pour missions essentielles :

- Le développement, la conservation et la valorisation des réseaux énergétiques sur tout le territoire national.

- La reconstitution et l'augmentation des réserves d'hydrocarbures.

- L'intensification des efforts d'exploitation et capitalisation des études réalisées dans ce domaine, pour une meilleure connaissance du sous-sol et la mise en évidence des réserves d'hydrocarbures.

- La diversification des marchés et des produits destinés a l'exportation.

- L'approvisionnement énergétique national à moyen terme, compte-rendu des réserves nationales.

- L'adaptation de l'outil commercial aux exigences du marché énergétique pour une meilleure maîtrise de ses mécanismes et des performances commerciales accrus.

- Le développement la maîtrise et la maintenance des complexes de production, de transport et de conditionnement des hydrocarbures.

- Le développement des techniques modernes de gestion nationale par le biais de la formation continue.

Figure 1-1 Organigramme de l'ensemble de l'entreprise de SONATRACH

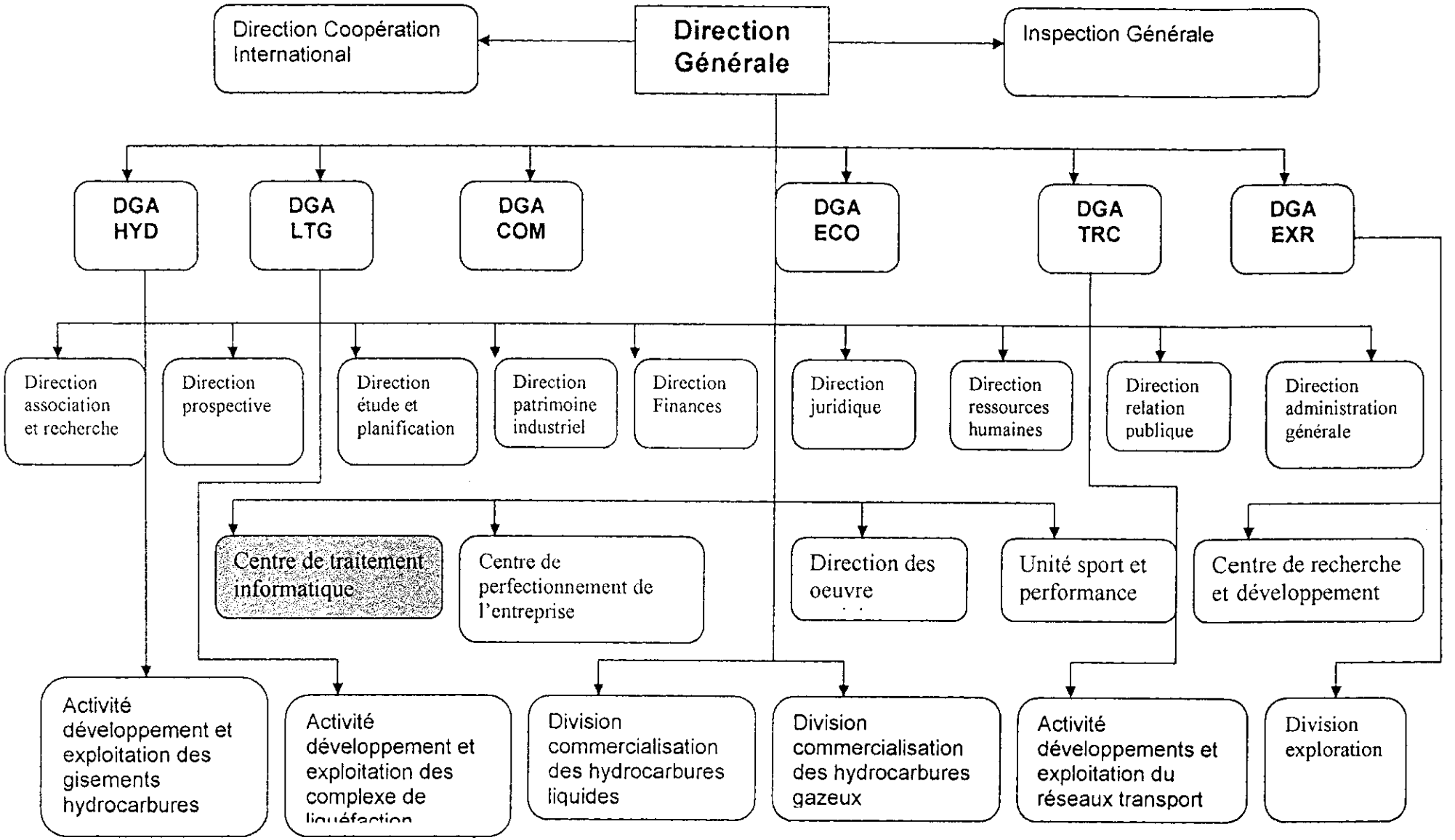
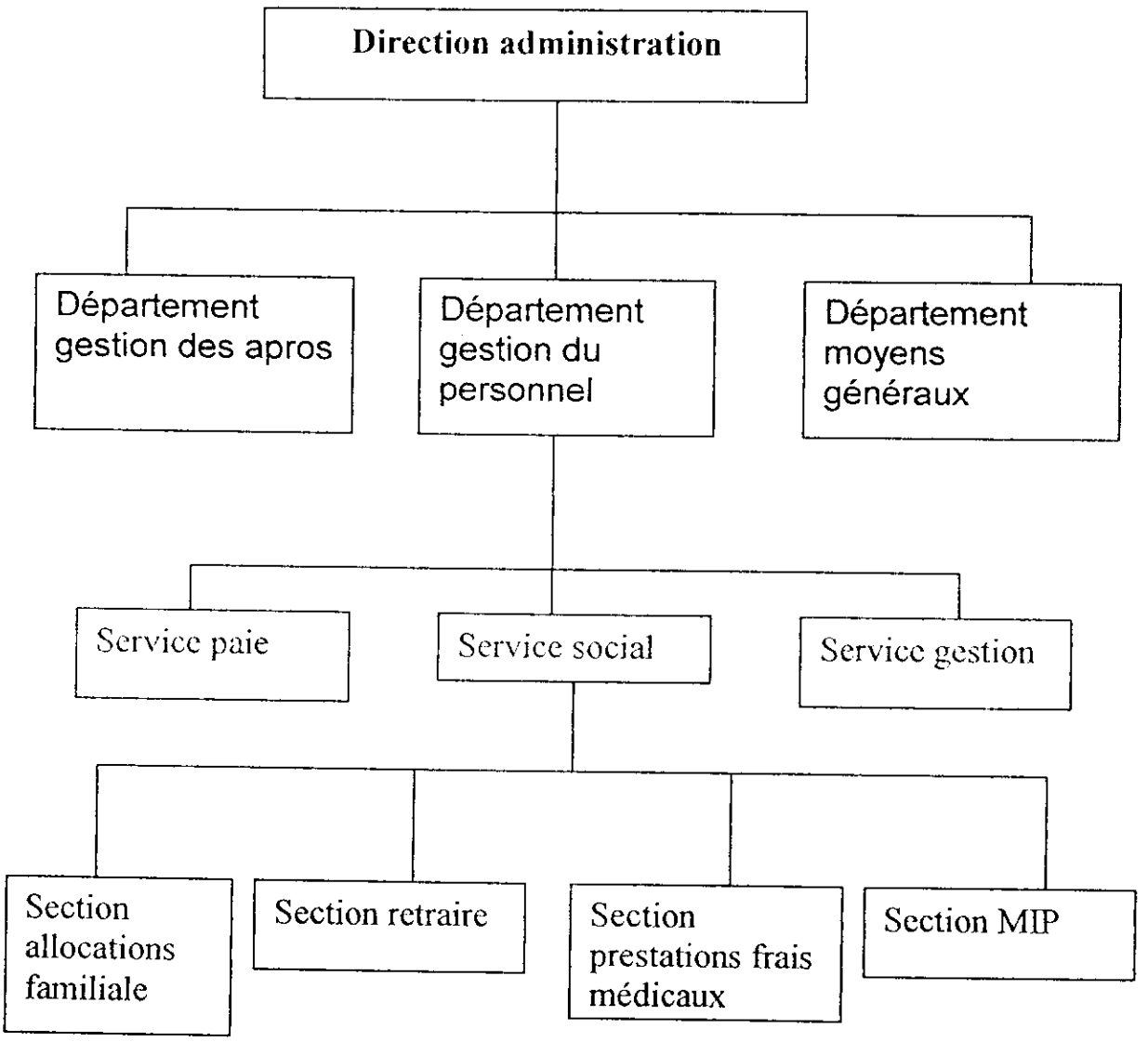


Figure 1-2 Organigramme de la structure d'étude



Note

Web[3.1]

WWW (World Wide Web): Encore appelé Web. En anglais, web signifie tissage ou toile d'araignée, en référence à la topologie du réseau Internet. Le CERN (Centre Européen de Recherches Nucléaires) fut l'acteur principal du WWW. Par la suite, le suivi du WWW fût pris en charge par le W3C (World Wide Web Consortium).

CGI[3.2]

CGI (Common Gateway Interface) : Nous verrons par la suite que cette interface qui présente l'avantage d'être standard et reconnue par tous, n'est pas très performante et tend à être progressivement remplacée.

SGML[3.3]

SGML (Standard General Markup Language) : Standard de description de document permettant de se concentrer sur le contenu plutôt que sur la forme.

TCP/IP[3.4]

HTTP n'est pas du tout lié au protocole TCP/IP et on pourrait envisager de faire fonctionner une version adaptée à un protocole comme IPX.

ISAPI[3.5]

ISAPI est l'API d'extension des serveurs Web de Microsoft.

NSAPI[3.6]

NSAPI est l'API d'extension des serveurs Web de Netscape.

contraignant[3.7]

Pas question, en effet, d'utiliser le click droit pour les applications devant fonctionner sur Macintosh, par exemple.

OAK[3.8]

OAK signifie chêne en anglais.

graphiques[3.9]

Les traditionnelles classes AWT, mais aussi, depuis la version 1.2 du JDK (Java Development Kit), les classes SWING permettant de réaliser des interfaces graphiques indépendantes du système hôte.

JDBC[3.10]

JDBC (Java Data Base Connectivity) : API fédérant la connexion des programmes Java à des bases de données.

RMI[3.11]

RMI (Remote Method Invocation) : Mécanisme d'appel de méthode distante utilisé par les programmes Java.

CORBA[3.12]

CORBA (Common Object Request Broker Architecture) : Architecture objet de l'OMG visant à l'intégration d'applications par la communication entre objets sur le réseau de l'entreprise.

Références bibliographiques :

[LEF97] Alain LEFEBURE. Eyrolles, 1997
"Intranet clent-serveur universel"

[ROO] Robert OGOR
"Modélisation avec UML".

[PRF03] Pascal Roques , Franck Vallée 2003
"UML en action, de l'analyse des besoins a la conception en java ".

[FAP00] Fabrice POPINEAU 2000
"Introduction a la conception orientée objet ".

[LAU96] Laurent HENOCQUE - Ecole supérieur d'ingénieur de luming, 1996
"Introduction a la méthode OMT "

[DKK01] Duane K. FIELDS, Mark A. KOLB 2001.
" JSP java server page "

[JMF97] Jean-Marie FAVRE 1997
"Java Beans"

[GER01] George Reese ,2001
" JDBC et JAVA Guide du programmeur "

[GIB03] Gilles Briard avec la collaboration de la société Digora, 2003
" Oracle 9i sous windows "

Les sites web:

www.sun.com

www.oracle.com

cedric.babault.free.fr

www.abcdoc.net

www.java-source.com

www.moteurprog.com

www.eunis.org

www.orsys.fr

www.eclipse.org

jakarta.apache.org

www.lesjsp.com