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

Université Mouloud MAMMERI de Tizi-Ouzou
Faculté de Génie Electrique et Informatique
Département Informatique

Mémoire

De fin d'études en vue de l'obtention du diplôme Master II en Informatique; option Système Informatique (SI).

Thème

Conception et réalisation d'une application web pour la gestion du parc informatique

Cas : Direction générale d'Algérie Télécom Mobile Mobilis (ATM) Sise à Alger

Dirigé par :

Mr. A. DIB

Réalisé par :

M^elle^ BENFERHAT Dihya.

Promotion : 2012 /2013.

Remerciements

Je tien à exprimer ma profonde gratitude à mon promoteur, Mr. A.DIB pour m'avoir encadré durant cette année.

Je remercie tous le personnel de l'entreprise « MOBILIS », en particulier Mr. Nadjib, Mme HABBES et Mr. Mohamed pour l'accueil chaleureux qu'ils m'ont réservés et le temps qu'ils m'ont consacrés le long de la période de mon stage.

Que les membres du jury trouvent ici mes plus vifs remerciements pour avoir accepter d'honorer mon travail par leur jugement.

Un grand merci aussi à tous ceux, et celles qui ont contribué de prés ou de loin pour l'accomplissement de ce modeste travail.

Dédicaces

Je dédie ce modeste travail :

À la mémoire de mon cher père que DIEU le garde dans son vaste paradis.

À ma très chère mère.

À mes frères et sœurs et leurs petites familles.

À tous mes amis(es).

À tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

♥*Dihya*

Table des Figures

La table des figures

Sommaire

Sommaire

Chapitre II : Présentation de l'organisme d'accueil.

Chapitre III : Le Framework Hibernate.

Chapitre IV : Analyse et conception.

Chapitre V : Implémentation et mise en œuvre.

Introduction Générale

Introduction générale

Au cours de ces dernières décennies, les réseaux d'entreprises ont connus une grande croissance. Ainsi, on assiste à une utilisation accrue d'applications sur le Web permettant d'accéder à l'information à tout moment et n'importe où, avec une connexion à l'Internet. Cela est rendu possible grâce aux progrès technologiques dans le domaine de l'informatique. C'est ainsi que l'Internet est venu révolutionner ce domaine. Cet outil indispensable est accessible au grand public. Aujourd'hui, vu l'intérêt croissant de vouloir gagner en temps, de conserver les données, de limiter le nombre d'employés et pas mal d'autres raisons, ont poussés petites, moyennes et grandes entreprises à chercher des solutions informatiques capables de répondre à leurs besoins.

Le Web comme premier en la matière de l'information et de communication est vite devenu le domaine de commerce et de publicité pour ces entreprises.

C'est dans ce cadre que s'inscrit mon projet de fin d'études qui consiste à réaliser une application web pour la gestion du parc informatique au sein de l'ATM Mobilis.

Pour ce faire, j'ai adopté la structure suivante :

- Le premier chapitre « Introduction aux technologies de l'information et de la communication » présente quelques notions de base concernant l'Internet, le web, le modèle Client/serveur et les applications web.
- Le deuxième chapitre « Présentation de l'organisme d'accueil », porte sur le domaine d'étude de mon application.
- Le troisième chapitre « Le Framework Hibernate », présente les différents concepts de ce Framework.
- Le quatrième chapitre « Analyse et conception », est consacré à l'analyse et à la conception de l'application proprement dite.
- En fin, le cinquième chapitre « Implémentation et mise en œuvre », porte sur la réalisation et l'implémentation de l'application ainsi que son fonctionnement.

Chapitre I :

Introduction aux technologies de l'information et de la communication

I.1. Introduction :

Depuis les 43 ans d'existence de l'Internet, les programmeurs ont tenté de faire sauter les barrières qui séparent les applications traditionnelles des applications Web. Le progrès des dernières années en matière de technologie, de vitesse de téléchargement ainsi qu'en outils de développement comble les lacunes que connaissaient ces applications.

Progressivement, les applications Web deviennent de plus en plus complètes en fonctionnalités, tout en étant plus simples d'utilisation.

Ce chapitre aura pour objectifs de donner, en premier lieu, un aperçu sur l'Internet et ses différents services, notamment le web, l'architecture client/serveur, puis présentera les applications web.

I.2. Internet :

I.2.1. Définition:

Internet signifie réseaux interconnectés (interconnected network). Constitue un réseau informatique, créé grâce à l'interconnexion de réseaux et d'ordinateurs entre eux, qui relie le monde entier en utilisant un protocole de transmission et de communication constituant un protocole commun permettant la connexion de toutes les machines, c'est le TCP/IP.

Pour pouvoir se raccorder à Internet quatre aspects doivent être pris en compte :

- Posséder un ordinateur suffisamment puissant.
- Un modem suffisamment rapide.
- Un compte ouvert auprès de fournisseur d'accès Internet.
- Des logiciels spécifiques d'accès à Internet (navigateur…).

I.2.2. Les services de l'Internet :

Internet fournit aujourd'hui une multitude de services à travers le monde tel que :

- **Le courrier électronique (E-mail) :** Ce service permet à un utilisateur de composer des notes et de les envoyer sous forme de messages à des individus ou à des groupes d'individus, ainsi que de lire les messages reçus.

- **Le transfert de fichier (FTP) :** En utilisant le programme de transfert de fichier, il est possible de copier d'un ordinateur à un autre de grandes bases de données contenant par exemple des images satellites, des programmes écrits en Pascal ou en C++ ou un dictionnaire français.
- **La connexion à distance (Telnet) :** Elle permet à un utilisateur assis devant un terminal ou un ordinateur de se connecter à un ordinateur distant et d'établir avec lui une session informatique. L'ordinateur distant donne l'illusion qu'il est directement relié à l'utilisateur.
- **Les forums électroniques (Usenet) :** Ce service permet d'accéder à des groupes de discutions, sur lesquels les utilisateurs peuvent se trouver pour échanger des idées sur différents thèmes.
- **Service de remise en mode non connecté :** Ce service est proposé par nombre de réseaux de transmission de données, il route les blocs de données indépendamment les uns des autres lors de leur échange entre un expéditeur et un destinataire. Il ne garantit ni leur remise fiable ni leur remise en séquence, et comme il effectue une correspondance directe avec le matériel sous-jacent, ce service est extrêmement efficace.
- **Service de transport fiable :** Il permet à une application d'un ordinateur local d'établir une connexion (une mise en relation) avec une application d'un ordinateur distant, puis d'envoyer des données sans restriction de volume sur cette connexion temporaire, comme s'il s'agissait d'une connexion permanente et directe.
- **Explorer le World Wide Web :** Qui permet d'utiliser tout ce qui précède et ajoute des liens vers d'autres ressources et des facilités multimédia (son, graphiques, vidéo).

I.2.3. IP (Internet Protocol) :

Il réalise le service de remise des datagrammes en mode non connecté (réseau) appelé protocole Internet. Trois définitions importantes sont attachées à ce protocole. La première définit le datagramme comme l'unité de données de base circulant sur un Internet TCP/IP, et spécifie le format des datagrammes. La seconde correspond à la fonction de routage qui consiste à déterminer le chemin le long duquel les datagrammes seront acheminés. La troisième, complémentaire des deux précédentes, spécifie un ensemble de règles concrétisant le concept de la remise non fiable des datagrammes. Ces règles précisent comment les ordinateurs et les routeurs doivent traiter les datagrammes, comment et quand les messages d'erreur doivent être générés et dans quelles conditions les datagrammes doivent être détruits.

Le protocole IP est une partie fondamentale de la conception d'un Internet TCP/IP, à tel point qu'on parle de technique fondée sur IP (IP-Based Technology).

Adressage IP :

Une adresse Internet contient l'identification du réseau et l'identificateur de la machine concernée sur ce réseau, cela permet un routage très efficace. Une propriété importante de l'adresse IP est de référencer le point d'accès au réseau et non la machine. Un ordinateur doté de plusieurs points d'accès dispose de plusieurs adresses IP. L'adresse réseau est placée sur les bits de poids fort, alors que l'adresse machine est calculée sur les bits de poids faible.

Une adresse IP est toujours représentée dans une notation décimale pointée constituée de 4 nombre (1 par octet) compris entre 0 et 255 et séparés par un point. Un serveur spécial appelé DNS (Domain Name System) se charge de traduire en noms mnémoniques les adresses IP.

I.2.4. TCP (Transmission Control Protocol) :

Le protocole de contrôle de transmission assure un service d'importance majeure : le service de remise fiable en mode connecté. TCP établit une connexion bidirectionnelle simultanée entre deux ordinateurs. Ceci permet à ces derniers d'échanger d'importants volumes de données de façon efficace.

Grâce au protocole de fenêtre glissante, TCP utilise efficacement le réseau sous-jacent sur lequel il ne fait que peu d'hypothèses. Il est donc suffisamment souple pour s'adapter à une grande variété de systèmes de remise. Il assure un contrôle de flux qui permet à des systèmes très différents de communiquer.

Les unités de transfert utilisées par TCP sont les segments. Ceux-ci permettent d'acheminer des informations de contrôle (afin de permettre aux logiciels TCP de deux ordinateurs d'établir une connexion ou de la libérer, par exemple) ou des données. Leur structure permet à un ordinateur de joindre les accusés de réception de données du flot de sens contraire.

TCP réalise le contrôle de flux en permettant au récepteur d'indiquer la quantité de données qu'il peut encore accepter. TCP supporte également des messages hors bande en utilisant une fonction de remise de données urgentes et la remise forcée à l'aide du mécanisme Push.

I.2.5. L'Internet, l'Intranet et l'Extranet :

S'ils sont très proches dans leur prononciation et leur orthographe, ces trois termes ont des significations bien distinctes.

Internet est le réseau mondial que tout le monde connait. Composé de sites web hébergés sur des serveurs partout dans le monde et accessible à tous via un simple navigateur et une connexion Internet.

A la différence d'Internet, l'Intranet est un réseau privé qui n'est accessible qu'en interne d'où le préfixe « Intra ». Il s'agit typiquement des sites de l'entreprise et qui ne sont accessible qu'à ses employés quand ces derniers sont au travail.

L'Extranet moins répondu, s'agit ni plus ni moins de l'extension de l'Intranet aux clients ou fournisseurs à l'extérieur de l'entrepris.

I.3. Le web:

I.3.1. Définition:

Le World Wide Web ou WWW, littéralement la « toile d'araignée mondiale », communément appelé le Web. C'est une source phénoménale d'information. Il s'agit d'un système d'information hypermédia. Il est constitué de documents MULTIMEDIAS (pages de textes enrichies de sons, graphiques, images fixes et animées, vidéos,…) qui sont reliés entre eux par des liens HYPERTEXTE.

Ainsi dans les pages qui composent un site web, chaque mot souligné voire certaines images, est un lien hypertexte que vous cliquez pour afficher un autre document. Ce document pouvant être localisé sur n'importe quel ordinateur du réseau. Ces hypertextes rendent la lecture plus dynamique. Des documents contiennent ainsi des références sur d'autres documents, créant une toile d'araignée de documents recouvrant le monde.

- ✓ **Hypertexte :** Il désigne un mode d'organisation des documents textuels informatisés caractérisé par l'existence de liens dynamiques entre ses différentes sections. Dans le web, des mots ou des expressions soulignées ou encore des images indiquent les liens sur lesquels on clique à l'aide de la souris. L'hypertexte laisse le lecteur décider de son cheminement dans le document en fonction de ses besoins et de ses intérêts.
- ✓ **Navigateur web :** Un navigateur web est un logiciel conçu pour consulter le World Wide Web, l'ensemble des sites web. Techniquement, c'est un client http dans une architecture client/serveur. Il peut être graphique ou texte.
- ✓ **Serveur web :** Ensemble matériel et logiciel hébergeant des documents HTML accessibles sur le World Wide Web.
- ✓ **HTTP (Hyper Text Transfert Protocol) :** Protocole utilisé pour transporter des pages HTML du WWW sur le réseau. L'accès aux services web se fait en donnant une adresse de type:
  http: //nom-de-domaine/répertoire/.

- ✓ **HTML (Hyper Text Markup Language) :** Langage de description des pages web dérivé du SGML. Il est composé d'une suite de signe ASCII, dans laquelle sont inclues les commandes spéciales concernant le formatage des pages, la police de caractères et les multimédia.
- ✓ **URL :** Adresse Internet exploitée par les navigateurs (Internet Explorer ou Navigator, par exemple). C'est l'adressage standard de n'importe quel document, sur n'importe quel ordinateur en local ou sur Internet. Structure de base d'une URL :

  protocole://serveur/répertoire/document.extension
- ✓ **Hébergeur :** Est une entité ayant pour vocation de mettre à disposition des internautes des sites web conçus et gérés par des tiers.

I.3.2. Différence entre web et Internet:

La définition d'Internet en tant que système global d'information permet alors de comprendre que le web n'est en fin de compte qu'une des applications ou usages d'Internet au même titre que peuvent l'être le courrier électronique, la messagerie instantanée…

Plus précisément le Web est le service qui permet de consulter des informations à partir d'Internet sous la forme de pages mises en ligne sur des sites et consultables à l'aide d'un navigateur web.

I.3.3. Page web :

La page Web est l'unité de consultation du World Wide Web. Ce terme a une signification pratique ; il n'a pas de définition technique formelle. C'est un document informatique qui peut contenir du texte, des images, des formulaires à remplir et divers autres éléments multimédia et interactifs.

Les pages Web de divisent en deux types :

- **Pages web statiques :** Ce sont des pages web écrites en HTML invariantes, proposées à l'avance, autrement dit, à chaque fois qu'un client affiche une page, cette dernière se présente de la même manière tout au moins tant que le code HTML correspondant n'a pas été mis à jour par l'auteur.
- **Pages web dynamiques :** Dans ce cas, les pages ne sont pas fabriquées en avance. Cette composante dynamique peut recouvrir des aspects tels que la diffusion d'informations évoluant dans le temps ou la récupération des renseignements saisis par l'utilisateur pour décider de déclencher telle ou telle autre action.

I.3.4. Site web :

Un site web est composé d'un ensemble de documents structurés (pages web), stockés sur un ordinateur (serveur) connecté au réseau mondial (Internet).

On distingue deux types de sites web : statiques et dynamiques.

- **Site web statique :** Site web constitué de pages HTML, dont le contenu change peu. On parle de site construit « en dur », donc plus rigide. Ses avantages sont sa souplesse, liberté dans le design et il est moins cher au départ.
- **Site web dynamique :** Site web relié à une base de données, dont les pages se construisent à la demande de l'Internaute. La mise à jour est plus facile, grâce à une interface d'administration.
  Les blogs, les forums, les sites marchands avec les catalogues, les sites en plusieurs langues, sont des sites web dynamiques.

I.3.5. L'adresse web :

Pour afficher le contenu d'une page web désirée, il faut préalablement la localiser. Pour cela, chaque page existante a une adresse électronique, l'URL (Uniform Resource Locator ou localisateur uniforme de ressource) fournie lors de sa création.

Cette adresse est constituée de trois parties :

1. Protocole utilisé (protocole de communication, exemple : http://)
2. Nom de serveur (nom de domaine, exemple : www.calis.ch/).
3. Chemin d'accès au fichier désiré (exemple : module/index.html).

Pour trouver un site d'entreprise, de société ou encore gouvernemental, dont on ne connait pas l'adresse, on peut intuitivement taper le nom de l'entreprise suivi d'un suffixe approprié (Exemple : www.mobilis.dz).

I.3.6. Les caractéristiques du web :

Le web comporte un certain nombre de caractéristiques qui y sont exclusives, elles sont décrites ci-dessous :

- ✓ Un système hypertexte : Il permet de naviguer.
- ✓ Un système multimédia : Grâce aux explorateurs graphiques, le web est devenu multimédia (un document web peut contenir : un texte ordinaire, des caractères spéciaux, des images, des séquences audio et vidéo).

- ✓ Un système de traitement distribué : Dans le web, les documents peuvent être stockés dans toutes les mémoires reliées au réseau ; cela signifie que les documents qui y sont mémorisés peuvent être distribués dans tout le réseau.
- ✓ Assure l'interface avec les systèmes de base de données : Une des fonctions les plus puissantes des explorateurs du web est leur capacité d'agir comme interface avec des systèmes de base de données reliée à Internet.

I.4. L'architecture client/serveur :

I.4.1. Définition :

C'est la description du fonctionnement coopératif entre le serveur et le client. Les services Internet sont conçus selon cette architecture. Ainsi, chaque application est composée de logiciel serveur et logiciel client. A un logiciel serveur, peut correspondre plusieurs logiciels clients développés dans différents environnements : Unix, Mac, PC,… La seule obligation est le respect du protocole entre les deux processus communicants (ils doivent utiliser le même protocole de communication).

I.4.2. Les notions de base :

- **Client :** Un programme qui s'exécute devient un client lorsqu'il émet une demande vers un serveur et qu'il attend une réponse.

Les caractéristiques d'un client :

- Il établit la connexion au serveur à destination d'un ou plusieurs ports réseaux.
- Lorsque la connexion est acceptée par le serveur, il communique comme le prévoit la couche applicative de modèle OSI.

- **Serveur :** Le terme Serveur s'applique à tout programme qui offre un service que l'on peut atteindre à travers un réseau. Le serveur accepte des demandes issues du réseau, les traite et renvoie le résultat au demandeur. Pour les services les plus simples, les demandes sont véhiculées par un seul datagramme IP et le serveur répond en retournant un seul datagramme IP.

Les caractéristiques d'un serveur :

- Il attend une connexion entrante sur un ou plusieurs ports réseaux.
- A la connexion d'un client sur le port en écoute, il ouvre un socket local au système d'exploitation.

- Suite à la connexion, le processus serveur communique avec le client suivant le protocole prévu par la couche application du modèle OSI.

➢ **Middleware :** C'est un ensemble des services logiciels. Il permet la communication entre des clients et des serveurs ayant des structures et une implémentation différentes. Il permet aussi l'échange d'informations dans tous les cas et pour toutes les architectures. Enfin, le middleware doit fournir un moyen aux clients de trouver leurs serveurs, aux serveurs de trouver leurs clients et en général de trouver n'importe quel objet atteignable.

I.4.3. Le fonctionnement de client/serveur :

Un système client/serveur fonctionne selon le schéma suivant :

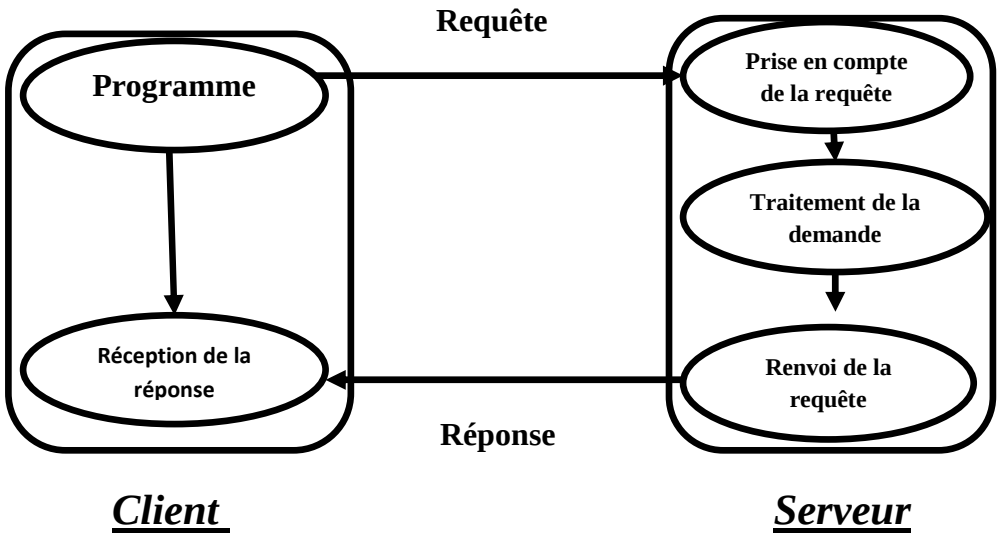


Figure I.1. Architecture client/serveur.

- Le client envoie une requête au serveur grâce à son adresse IP et le port, qui désigne un service particulier de serveur.
- Le serveur reçoit la demande et répond à l'aide de l'adresse de la machine cliente et son port.

I.4.4. Classification des architectures client/serveur :

- **Architecture à 2 niveaux :**

Dans une architecture deux tiers, encore appelée client/serveur de première génération ou client/serveur de données, le poste client se contente de déléguer la gestion des données à un service spécialisé. Le cas typique de cette architecture est une application de gestion fonctionnant sous Windows ou Linux et exploitant un SGBD centralisé. Ce type d'application permet de tirer partie de la puissance des

ordinateurs déployés en réseau pour fournir à l'utilisateur une interface riche, tout en garantissant la cohérence des données, qui restent gérées de façon centralisée.

La gestion des données est prise en charge par un SGBD centralisé, s'exécutant le plus souvent sur un serveur dédié. Ce dernier est interrogé en utilisant un langage de requête qui, plus souvent, est SQL. Le dialogue entre client et serveur se résume donc à l'envoi de requêtes et au retour des données correspondant aux requêtes.

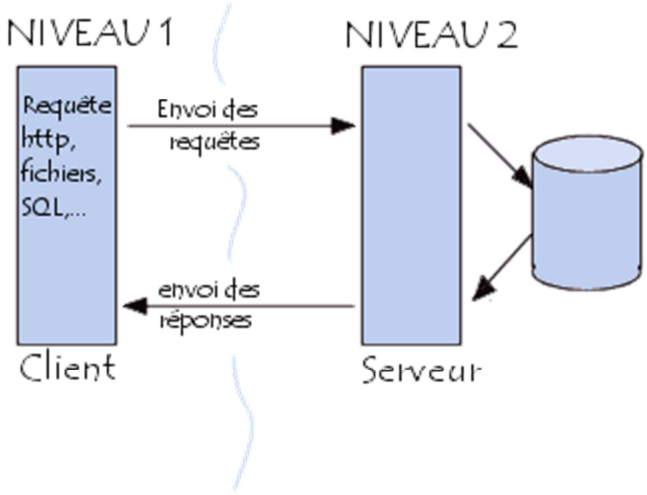


Figure I.2. L'architecture client/serveur à 2 niveaux.

- **Architecture à 3 niveaux :**

Cette architecture trois tiers, également appelée client/serveur de deuxième génération ou client/serveur Distribué, sépare l'application en 3 niveaux de services distincts, conformes au principe précédent :

- ✓ 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.

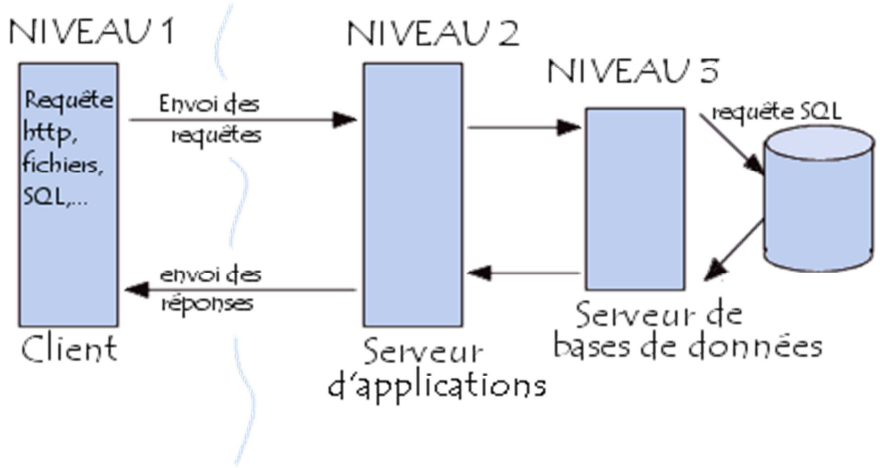


Figure I.3. L'architecture client/serveur à 3 niveaux.

- **Architecture à multi niveaux :**

Dans l'architecture à trois niveaux, chaque serveur (niveau 2 et 3) effectue une tâche (un service) spécialisée. Un serveur peut donc utiliser les services d'un ou plusieurs autres serveurs afin de fournir son propre service. Par conséquent, l'architecture à trois niveaux est partiellement une architecture à **n** niveaux.

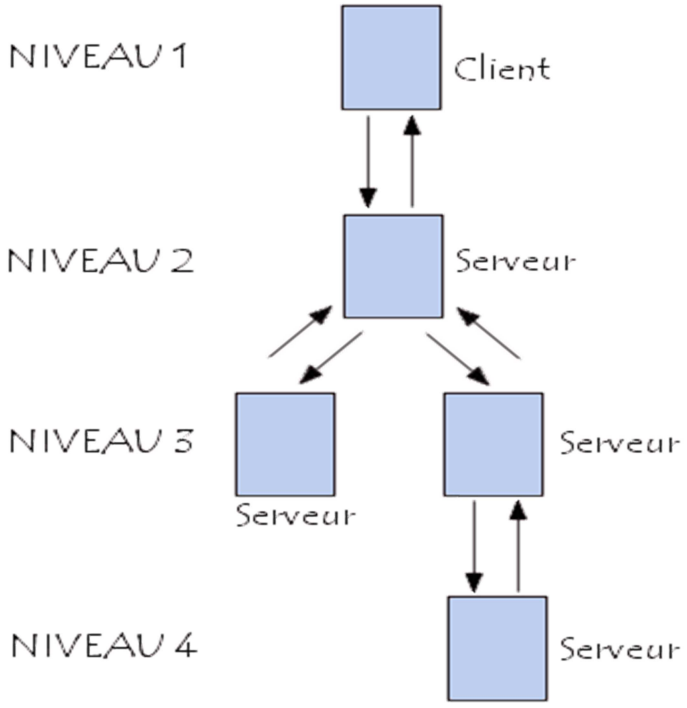


Figure I.4. L'architecture client/serveur à multi niveaux.

I.4.5. Les avantages et les inconvénients de l'architecture client/serveur :

- **Les avantages :**

L'architecture client/serveur est fortement recommandée pour les réseaux nécessitants un bon niveau de fiabilité, elle présente beaucoup de points forts, on cite entre autres :

- ✓ Des ressources centralisées : Le serveur constitue le centre du réseau et gère toutes les ressources communes et partagées pour l'usage des clients.
- ✓ Mise à l'échelle (scalability) : Il est facile de mettre à l'échelle le système selon les dimensions de problème à résoudre.
- ✓ Exploitation (usability) : Il est facile d'utiliser (préparer des données, comprendre les sorties) un système ou un composant.
- ✓ Flexibilité : Il est facile de modifier un système ou un composant pour l'utiliser dans des applications ou des environnements autres que ceux pour lesquels il a été conçu.
- ✓ Interopérabilité : Les systèmes ou les composants peuvent échanger et utiliser plus facilement les informations échangées.
- ✓ Une administration au niveau serveur (gestion des accès).
- ✓ Une meilleure sécurité.
- ✓ Une facilité de maintenance et de mise à jour des applications.
- ✓ Un réseau évolutif (facilité d'implémentation de nouvelles machines clientes ou de nouveaux applicatifs).

- **Les inconvénients :**

L'architecture client/serveur manifeste par ailleurs quelques points faibles à savoir :

- ✓ Coût élevé : Le serveur doit d'être une machine très performante car elle assume l'essentiel du travail.
- ✓ Maillon faible : Le serveur est le seul maillon faible de cette architecture, vu que toutes les machines clientes sont architecturées autour de lui. Ceci dit, le serveur doit posséder une grande tolérance aux pannes.

I.4.6. Exemples de modèle client/serveur :

- Les serveurs de fichiers : Qui mettent des documents à disposition sur le réseau.
- Les serveurs de Base de données : Qui mettent une base de données centralisée à disposition sur le réseau.

- Les serveurs de messagerie : Qui permettent l'accès et l'utilisation de la messagerie électronique.
- Les serveurs d'applications : Qui mettent des programmes sur le réseau.
- Les serveurs d'accès : Qui contrôlent les accès au réseau.
- Les serveurs d'impression : Qui mettent une imprimante à disposition des clients.

I.5. Application web :

I.5.1. Définition :

En informatique, une application Web (aussi appelée WebApp) est un logiciel applicatif manipulable grâce à un navigateur Web. De la même manière que les sites Web, une WebApp est généralement placée sur un serveur et se manipule en actionnant des widgets à l'aide d'un navigateur Web, via un réseau informatique (Internet, intranet, réseau local, etc.).

Les messageries web, les systèmes de gestion de contenu, les blogs sont des applications Web.

Les moteurs de recherches, les logiciels de commerce électronique, les jeux en ligne, les logiciels de forum peuvent être sous forme d'application Web.

Des appareils réseau tels que les routeurs sont parfois équipés d'une application Web dans leur micro-logiciel.

Les applications Web font partie de l'évolution des usages et de la technologie du Web.

I.5.2. Histoire des applications web :

Le concept même d'application web n'est pas récent. En effet, un des premiers langages de programmation pour le développement d'application web était le « Perl ». Il fut inventé par Larry Wall en 1987 avant qu'Internet ne devienne accessible au grand public. Mais c'est en 1995, lorsque le programmeur Rasmus Lerdorf rendit le langage PHP disponible à tous que le développement d'applications web prit vraiment son envol. De nos jours, encore plusieurs de ces applications sont développées en PHP dont Google, Facebook et Wikipédia.

Quelques mois plus tard, Netscape, ancien et populaire navigateur web, annonçait une nouvelle technologie, le JavaScript, permettant aux programmeurs de modifier dynamiquement le contenu d'une page web qui était, jusqu'à ce jour, du texte statique. Cette technologie permit une toute nouvelle approche dans le développement des applications web, lesquelles étaient désormais, et encore aujourd'hui, beaucoup plus interactives pour les usagers. A titre d'exemple, la fonctionnalité Google Instant, permettant d'afficher des

résultats de recherche au moment même où les lettres du mot sont tapées, fait un usage intensif du JavaScript. Le site web des mises à jour des produits Microsoft utilise également cette technologie.

- En 1996, deux développeurs, Sabber Bhatia et Jack Smith, lançaient Hotmail.
- En 1997, vint la réputée plateforme Flash, servant à ajouter du contenu interactif aux sites web, sous le nom de « Shockwave Flash ». Plus tard, après avoir été acquise par Macromédia puis par Adobe, Flash devint une plateforme de développement d'applications web interactives.
- Le 17 janvier 1998, le site Internet Drudge Report annonçait pour la première fois une nouvelle journalistique sur la toile avant même qu'elle ne soit diffusée dans les médias télévisuels et imprimés traditionnels. Celle-ci faisait état du scandale Clinton/Lewinski. Cet événement fut l'élément déclencheur du journalisme en ligne tel qu'on le connait aujourd'hui. Avant cette date, Internet n'avait jamais été considéré comme un média de premier plan.
- La même année, la compagnie Google mit en ligne son premier moteur de recherche qui, par sa nouvelle façon d'indexer les pages web, facilita grandement la recherche d'informations sur Internet. Google continua d'innover et devint l'une des entreprises les plus prolifiques en termes d'applications web, en répertoriant les très populaires Google Maps, Google Docs, Gmail,…
- Au début de l'année 2001, peu après l'explosion de la bulle Internet, Wikipédia fut lancé comme sous-projet de Nupedia, une encyclopédie traditionnelle en ligne. Pour développer sa plateforme, Wikipédia utilisa un type d'application web nommée « wiki » permettant à tout internaute d'ajouter du contenu. Les contributions vinrent rapidement, et à la fin de la première année d'opération, Wikipédia comptait déjà 20 000 pages en 18 langues. Aujourd'hui, ce sont près de 21 millions d'articles en 285 langues qui composent le sixième site web le plus visité au monde, le premier étant Google.
- En 2003, MySpace fut fondé et plus tard, en 2005 à 2008, le site devenait le média social le plus visité. MySpace fut une plateforme de lancement pour plusieurs autres applications web connues telles que YouTube, Slide.com et RockYou!, qui ont toutes débuté comme modules supplémentaires pour les utilisateurs MySpace avant de devenir eux-mêmes des sites web à part entière.
- En 2004, lors d'une conférence Web 2.0 donnée par John Batelle et Tim O'Reilly, le concept du « web en tant que plateforme » fut mentionné pur la première fois.
- En cette même année, un événement très important se produit, le lancement de Facebook, qui était alors à ses balbutiements, ouvert uniquement aux étudiants. Comptant un million d'abonnés à la fin de 2004, Facebook est devenu aujourd'hui le média social le plus utilisé avec plus de 900 millions d'utilisateurs. C'est le deuxième site web le plus visité sur la planète et celui qui possède le plus de photos partagées par les utilisateurs avec un total de 500 milliards de photos téléchargées sur la plateforme.

- En 2005, YouTube fut officiellement lancé, permettant aux internautes de partager des vidéos en ligne.
- Twitter, quant à lui, fut lancé en 2006. Avec les années, la popularité de Twitter n'a fait qu'augmenter, passant de 1.6 million de « tweets » en 2007 à l'impressionnant chiffre de 340 millions par jour en mars 2012 (ce qui équivaut à près de 4 000 « tweets » par seconde !).
- L'année 2007, fut remarquée par l'apparition de l'iPhone, avènement qui fut très certainement responsable du nouvel engouement pour les plateformes mobiles et les applications web. Elles étaient désormais accessibles par téléphone intelligent.
- Au début 2011, la compagnie Kichstarter, qui facilite le financement participatif de projets en ligne, a atteint les 4 000 projets avec plus de 30 millions de dollars en donation. D'ailleurs, près de 44% des projets ont été démarrés avec succès à partir de cette plateforme.

Tel qu'il est remarquable, l'impact des applications web sur la façon d'opérer en affaires, de transmettre et de recevoir de l'information, et même sur la vie des gens, est considérable. Les applications web constituent une opportunité de connecter les internautes entre eux et les entreprises avec leur clientèle. Bref, les développeurs d'applications web façonnent aujourd'hui le futur numérique de demain.

I.5.3. La technique de base :

Dans la technologie client/serveur, utilisée pour le World Wide Web, le navigateur web envoie au serveur des requêtes relatives à des pages web. Le serveur répond aux demandes en envoyant les pages au navigateur web. Le navigateur affiche alors les pages à l'utilisateur.

Les applications web utilisent cette technique pour mettre en œuvre leur interface graphique. Celle-ci est composée de pages créées de toutes pièces par le logiciel lors de chaque requête. Chaque hyperlien contenu dans la page provoque l'envoi d'une nouvelle requête, qui donnera en résultat une nouvelle page. A la différence d'un site web statique où les pages sont des fichiers préalablement enregistrés.

Les pages web contiennent divers widgets tels des boutons poussoirs, des icônes et des zones de texte, permettant la manipulation de l'application. Chaque manipulation d'un bouton poussoir provoque l'envoi d'une nouvelle requête. Les pages web peuvent contenir des applets.

Contrairement à d'autres logiciels, une application web mise en place sur un serveur est immédiatement utilisable par le consommateur sans procédure d'achat et d'installation sur son propre ordinateur, du moment que la machine du consommateur est équipé d'un navigateur web et d'une connexion réseau. Ceci évite des interventions des administrateurs système, interventions qui sont souvent plus coûteuses que le logiciel lui-même.

L'application web est souvent mise à disposition du consommateur par l'éditeur du logiciel sur ses propres serveurs (technique appelée Software as a Service).

L'usage du navigateur web comme partie client (un logiciel qui est disponible sur de nombreux systèmes d'exploitation) assure la portabilité d'une application web.

I.5.4. Différence entre site web et application web :

Un site web est ce que l'on retrouve lorsqu'on entre une adresse et qu'on atterrit sur une page. Le site correspond à cette page et toute page reliée qui est gérée par la même entité (souvent sous le même nom de domaine). Le rôle principal d'un site web est de fournir et présenter de l'information aux visiteurs. Un blogue, un site de nouvelles ou un site d'information sur un produit ou une compagnie sont de bons exemples de sites web.

Une application web est tout site web qui permet à ses utilisateurs d'accomplir des tâches spécifiques. Une application gère donc généralement des utilisateurs et toutes sortes de données selon les requis spécifiques au projet.

Pour simplifier les choses, un site web c'est le fait de visualiser une page web. Une application web, c'est le fait d'interagir avec une page web, la modifier dans son contenu, pouvoir faire certaines choses avec.

I.5.5. Utilisation :

De nos jours les applications web sont omniprésentes, elles font partie de l'évolution des technologies correspondant à ce que l'on appelle « le Web 2.0 ».

La technologie des applications web permet de nombreux usages. Les usages populaires sont les moteurs de recherche, le webmail, le commerce électronique et les jeux en ligne.

- Un moteur de recherche est une application web qui recherche des documents.
- Un webmail est une application web pour recevoir et envoyer du courrier électronique. La présentation des documents est similaire à celle d'un site web, cependant les documents sont générés par le logiciel lors de chaque demande. Le système de gestion de contenu effectue les traitements nécessaires à la mise en forme et la présentation des documents.
    - Un weblog est un système de gestion de contenu où des éléments de contenu sont présentés dans l'ordre chronologique de leur date de création.

- Un wiki (de l'hawaïen wikiwiki qui signifie vite) est un système de gestion de contenu qui vise à simplifier la création collaborative des documents. Il autorise plusieurs personnes à effectuer des modifications simultanées, et est équipé d'espaces de discussion.
- Un site web marchand est un système de gestion de contenu, où le contenu est des annonces concernant des produits. Il est utilisé pour la vente par correspondance. Les visites et les opérations d'achat sont enregistrées à des fins de marketing. Les sites web marchands sont utilisés aussi bien pour la vente des produits d'une société que pour des ventes entre particuliers ou des ventes aux enchères.

- Un jeu par navigateur est un jeu vidéo réalisé sous la forme d'une application web.
- Un logiciel de forum permet des discussions ouvertes entre des utilisateurs : un utilisateur écrit un message, et ce message peut être lu par tous les autres utilisateurs. Les logiciels de forums sont parfois réalisés sous forme d'application web.
- La messagerie instantanée (an anglais « chat ») permet l'échange instantané de messages texte entre différents utilisateurs. Les logiciels de messagerie instantanée sont parfois réalisés sous forme d'application web. Les messages peuvent être transmis à un autre utilisateur du logiciel, ou un téléphone portable via le short message service (abrégé SMS).
- « Google Maps » est une application web qui permet de consulter des cartes géographiques du monde entier.
- « Facebook »est une application web qui permet à chaque utilisateur de se constituer un réseau social (amis, associés, personnes qui partagent les mêmes centres d'intérêt).

I.5.6. Technologie :

Dans la technologie la plus courante, l'application web s'oriente autour d'un serveur web sur lequel est branché le logiciel applicatif, le tout parfois accompagné d'un serveur de base de données. L'ensemble est appelé « serveur d'application ».

Le code source du logiciel applicatif est placé directement dans des pages web. Ces pages sont stockées par le serveur. Lorsque le client demande une page, le serveur web va rechercher la page, puis exécute les instructions qu'elle contient. Ces instructions peuvent faire appel au serveur de base de données. Le serveur web transmet la page avec le résultat de l'exécution au client.

La transmission des informations entre le client et le serveur se fait selon le protocole HTTP, protocole également utilisé pour les sites web. Ce qui permet d'utiliser le même logiciel client- un navigateur web.

Les applications web font souvent usage du mécanisme des cookies : en réponse à une requête, le serveur envoie une information de repérage au client (le cookie). Puis le client va lui renvoyer cette information lors de la prochaine requête. Le mécanisme est utilisé pour identifier le client et suivre les manipulations.

Les pages web peuvent en outre contenir des applets. Ce sont des morceaux de code source qui seront exécutés par le navigateur web après transmission de la page, contrairement à la majorité du code source qui est exécuté par le serveur web avant la transmission. ActiveX, Java, AdobeFlash et Silverlight sont des technologies utilisées pour les applets.

Le logiciel client (le navigateur web) est identique à celui utilisé pour consulter un site web. Les logiciels nécessaires pour exécuter les applets sont inclus dans les navigateurs, parfois sous forme de complément (plugin).

I.5.6.1. Les plateformes :

- .NET : c'est une plateforme proposée par Microsoft, respecte les standards actuels sur le web (HTTP, XML, SOAP, UDDI, WSDL,…), développé dans l'environnement Visual Studio.net (commercial) et Matrix (gratuit), et exécuté sous l'environnement Common Language Runtime (CLR). Elle intègre la base de données ADO.net et repose sur le Framework .NET. Les applications sous cette plateforme peuvent être écrites dans tous langage compatible avec le CLR, y compris Visual Basic et C#.
- J2EE : proposée par SUN, ce terme signifie Java Entreprise Edition, et était anciennement raccourci en « J2EE ». Il fait quant à lui référence à une extension de la plateforme standard. Autrement dit, la plateforme Java EE est construite sur le langage Java et la plateforme Java SE, et elle y ajoute un grand nombre de bibliothèques remplissant tout un tas de fonctionnalités que la plateforme standard ne remplie pas d'origine. L'objectif majeur de Java EE est de faciliter le développement d'applications web robustes et distribuées, déployées et exécutées sur un serveur d'applications.

Le tableau suivant recense les principaux points communs et différences des deux principales plateformes :

	Java (SE EE)	**.NET**
Environnement d'exécution	Java Virtual Machine (JVM)	Commun Language Runtime (CLR)
Format de la compilation	Bytecode	Microsoft Intermediate Language (MSIL)
Langage	Mono langage : Java uniquement Quelques solutions open source permettent de générer du bytecode (JPython,…)	Multi-langage dont : C#, C++, Visual Basic, .Net, JScript, Delphi, Cobol…
Mode d'exécution	Mode interprété ou compilateur JIT	Compilateur JIT
Système cible	Support pour tous les systèmes proposant un JVM	Plateforme Windows uniquement Quelques solutions open source permettent l'exécution sur d'autres plateformes (Mono sous Linux)
Mode de diffusion	Fournie sous la forme de spécifications avec une implémentation de référence Chaque spécification peut être implémentée par un tiers.	Fournie sous la forme de produit
IDE	Nombreux outils commerciaux et open source (Eclipsse, NetBeans)	Microsoft Visual Studio
Evolutions	Gérées par le JCP (Java Community Process composé de membres de nombreuses sociétés ou d'individus)	Uniquement à l'initiative de Microsoft notamment pour les bibliothèques.
Gestion de la mémoire	Gorbage collector	Gorbage collector
Packaging	Archive jar, war ou ear selon le type de projet	Assembly (.dll) ou exécutable (.exe)
Serveur d'applications	Nombreux serveurs d'applications J2EE (Websphere, Weblogic, OAS, JBoss, Jonas,…)	IIS uniquement

I.5.6.2. Serveurs d'applications :

Le serveur d'application constitue le noyau des architectures multi niveaux. Son rôle est d'assurer la logique métier des applications en découplant celle-ci des présentations et des accès aux données et de permettre une évolution des niveaux indépendamment des autres. En voici quelques exemples de serveurs d'application :

- Borland appServer (BES).
- ColdFusion d'Allaire (Macromédia).
- iPlanet Application Server (Com one) de Sun.
- .Net Server de Microsoft.
- Oracle 9i Application Server d'Oracle.
- WebLogic Server de BEA Systems.
- WebSphere Application Server d'IBM.
- JBoss (logiciel libre).
- JRun de Allaire.
- SilverStream Application Server de SilverStream.
- Tomcat d'Apache.

I.5.6.3. L'accès aux bases de données :

Le Framework est utilisé pour pourvoir accéder aux bases de données. C'est un ensemble de composants qui servent à créer l'architecture et les grandes lignes d'une application. On peut le voir comme une boîte à outils géante, conçue par un plusieurs développeurs et mise à disposition d'autres développeurs, afin de faciliter leur travail. Il existe des Frameworks dans beaucoup de langages et plateforme, ce n'est pas un concept propre au développement web en particulier. Parmi les Framework les plus connu, on peut citer :

- .NET Framework : il peut être utilisé par un système d'exploitation Microsoft Windows et Microsoft Windows Mobile depuis la version 5. Une version légère et limitée fournie avec un moteur d'exécution fonctionnant à l'intérieur d'un navigateur ou d'un périphérique mobile est disponible sous le nom de Silverlight. La version 3.0 du Framework est intégrée à Windows Vista et à la plupart des versions de Windows Server 2008 ou téléchargeable depuis le site de l'éditeur Microsoft. Ce Framework s'appuie sur la norme Common Language Infrastructure (CLI) qui est indépendante du langage de programmation utilisé. Ainsi tous les langages compatibles respectant la norme CLI ont accès à toutes les bibliothèques installées dans l'environnement d'exécution. Ce Framework est un sous ensemble de la technologie Microsoft .NET, il propose une approche unifiée à la conception d'applications Windows ou Web, tout en introduisant des facilités pour le développement, le déploiement et la maintenance d'applications.
- Hibernate : c'est un Framework open source gérant la persistance des objets en base de données relationnelle. Il est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un développement web léger de type Apache Tomcat ou dans un environnement J2EE complet : WebSphere, JBoss Application Server et Oracle WebLogic Server. Il apporte une solution aux

problèmes d'adaptation entre le paradigme objet et les SGBD en remplaçant les accès à la base de données par des appels à des méthodes objet de haut niveau.

- Struts : est un Framework libre servant au développement d'applications web Java EE. Il utilise et étend l'API Servlet Java afin d'encourager les développeurs à adopter l'architecture MVC. Struts a été créé par Craig McClanahan et donné à la fondation Apache en mai 2000. Il a fait partie du projet Jakarta de mai 2000 jusqu'en mars 2004. Cette infrastructure permet la conception et l'implémentation d'applications web de taille importante par différents groupes de personnes. En d'autres termes, les designers, développeurs de composants logiciels peuvent gérer leur propre part du projet de manière découplée. Struts est un logiciel libre distribué selon les termes de la licence Apache.

Il existe beaucoup d'autres Frameworks tels que MFC, Spring, JPOX,…

I.5.6.4. Les langages de programmation :

Plusieurs langages de programmation sont utilisés dans le développement d'application web.

- HTML (Hyper-Text Markup Language) : c'est tout simplement le langage dans lequel sont écrites des pages web. Ce langage repose sur un système de balises, comme par exemple <head>. Mais nul besoin de les connaître pour réaliser une page web car il existe de nombreux éditeurs de pages web qui permettent de créer des pages web comme on crée un document Word (Mozilla NVU, Macromedia Dreamweaver, etc). A noter qu'on ne parle pas de langage de programmation pour HTML mais de langage de présentation.
- CSS (Caqcading Style Sheet = feille de Style en cascade) : est utilisé pour alléger le code HTML. Pour cela, on crée généralement un fichier *.css dans lequel on entre toutes les instructions codant le style de la page web (couleur d'arrière-plan, type de blocs, de bordures, etc.). En faisant appel à ce fichier dans la page HTML, on peut désigner un style lourd et complexe avec une simple petite balise. De plus si la CSS est employée dans un site entier (comme c'est toujours le cas), il suffit de mettre à jour la CSS pour refaire le design du site en entier.
- SQL (Structured Query Language) : est un langage d'interrogation de base de données très populaire. Il consiste aujourd'hui une norme implémentée par de nombreux SGBDs comme Oracle, PostgreSQL et MySQL.
- XML : langage à balise strict permettant de stocker n'importe quelles données.
- JavaScript : initialement appelé LiveScript, est un langage de programmation pour les applets, développé par Netscape. Les applets écrites dans langage sont exécutées par un interprète inclus dans le navigateur web. De nombreux navigateurs web ont un interprète JavaScript.

- Java : est un langage de programmation développé par Sun Microsystems, qui peut être utilisé pour les applets. Les applets écrites dans ce langage sont préalablement compilées, et exécutées par un logiciel branché au navigateur web, le plug-in Java (traduction littéral : qui se branche dessus).
- Visual Basic (VB) : est un langage de programmation événementielle de troisième génération ainsi qu'un environnement de développement intégré, créé par Microsoft pour son modèle de programmation COM. Visual Basic est directement dérivé du BASIC et permet le développement rapide d'applications, la création d'interfaces utilisateur graphiques, l'accès aux bases de données en utilisant les technologies DAO, ADO et RDO, ainsi que la création de contrôle ou objets ActiveX. Les langages de script tels que Visual Basic for Application et VBScript sont syntaxiquement proches de Visual Basic, mais s'utilisent et se comportent de façon sensiblement différente. Visual Basic est un des langages les plus utilisés pour l'écriture d'applications commerciales. Il a également été très utilisés dans le monde de l'ingénierie et de la recherche appliquée en raison de sa capacité à permettre des développements très rapides et très efficients permettant ainsi aux scientifiques de se consacrer davantage à l'algorithmique et moins aux aspects formels du codage. Bill Gates y était particulièrement attaché, probablement parce que son premier succès avait été un programme écrit en BASIC pour l'Altair, premier ordinateur grand public.
- C# : c'est un langage proposé par Microsoft, orienté objet, dont la syntaxe est très proche de C (et donc de Java).
- Python : c'est un langage de programmation objet, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée et orienté objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions, il est ainsi similaire à Perl, Ruby, Schema, Smaltalk et Tcl. Il est placé sous une licence libre et fonctionne sur la plupart des plateformes informatiques, des supercalculateurs aux ordinateurs centraux, de Windows à Unix en passant par Linux, Mac OS, ou encore Android, iOS et aussi avec Java ou encore .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser. Il est utilisé principalement comme langage de scripts.
- PHP (Hypertext Preprocessor) : plus connu sous son sigle PHP, est un langage de programmation compilé à la volée libre, principalement utilisé pour produire des pages web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprète de façon locale. PHP est un langage impératif disposant depuis la version 5 de fonctionnalités de modèle objet complètes.
- Ruby : est un langage de programmation libre, il est interprété, orienté objet et multi-paradigme. Le langage a été standardisé au Japon en 2011, et en 2012 par l'organisation internationale de normalisation (ISO).

Ci-après, un tableau montrant un sondage fait sur le net pour les langages serveur préférés par les développeurs web :

Langage	Nombre d'opinions	Pourcentage
PHP	811	60,39%
ASP	40	2,98%
VB.NET (ASP.NET)	41	3,05%
C# (ASP.NET)	102	7,59%
Java (JSP, Servlets)	232	17,27%
Perl	14	1,04%
Python	19	1,41%
Rebol	2	0,15%
Coldfusion	6	0,45%
CGI / C	8	0,60%
Delphi	21	1,56
Webdev	9	0,67%
Autres	19	1,41%
Sans opinion	19	1,14%

Figure I.5. Résultat du sondage.

I.5.6.5. Autres technologies utilisés dans le développement d'applications web :

- WebObjects : développé par Apple, est un logiciel de serveur web qui exécute le code source contenu dans les pages, code source écrit an langage Java.
- ASP : est une technologie développée par Microsoft, composée d'un interprète, qui se branche sur le logiciel serveur web Internet Information Services. L'interprète exécute le code source contenu dans les pages, code source écrit en langage VBScript.
- ASP .Net : est une technologie développée par Microsoft en remplacement de ASP, composée d'un logiciel qui se branche su le logiciel de serveur web Internet Information Services et qui exécute le logiciel applicatif. Ce dernier est composé de pages contenant du code source écrit dans un langage qui utilise le Framework de programmation .Net (Visual Basic .NET ou C#), le logiciel applicatif est préalablement compilé.
- JSP : est une technologie développée par Sun Microsystems, composée d'un logiciel de serveur web auquel peut se brancher un logiciel applicatif écrit en langage Java. Le logiciel applicatif est composé de pages web contenant du code source préalablement compilé.

- LAMP (Linux – Apache –MySQL – PHP) : est une suite de logiciels open source qui, utilisés ensemble, permettent de réaliser des serveurs d'application. Par définition, cette suite est composée du système d'exploitation Linux, du logiciel serveur web Apache, de l'interprète PHP, et du SGBD MySQL.
- Wamp (Windows – Apache – MySQL – PHP) : est une suite de logiciels qui, utilisés ensemble, permettent de réaliser des serveurs d'application. Par définition, cette suite est composée du système d'exploitation Windows, du logiciel serveur web Apache, de l'interprète PHP, et du SGBD MySQL.
- Une applet : est un morceau de logiciel incorporé dans une page web, et exécuté par le navigateur web de l'ordinateur client. Lorsque l'utilisateur actionne un widget placé dans une page web, l'applet peut alors modifier la présentation de la page (technique appelée DHTML), afficher des messages ou envoyer des requêtes au serveur d'application.
- Le Document Object Model (abrégé DOM) : est un ensemble d'objets normalisé qui représente la page affichée, ainsi que le navigateur web. Le DOM est l'interface de programmation utilisée dans les applets pour effectuer des modifications sur la page.
- Ajax : cette méthode consiste à utiliser de manière conjointe des technologies telles que JavaScript, CSS, XML, le DOM et le XMLHttpRequest dans le but de réaliser des applications web qui offrent une maniabilité et un confort d'utilisation supérieur à ce qui se faisait jusqu'alors (les Rich Internet Application (abrégé RIA)).
- ActiveX : est une technologie développé par Microsoft ou des composants logiciel peuvent être inclus en tant qu'applet dans des pages web. Cette technologie nécessite un système d'exploitation Windows sur l'ordinateur client.
- Flash : est une technologie développée par Adobe. Un logiciel branché au navigateur, le plug-in Flash permet d'afficher des animations, des images vectorielles, des vidéo, et exécuter des applets. Il comporte un interprète pour des applets en langage de programmation ActionScript, un langage similaire à JavaScript. Cette technologie est d'usage courant pour les jeux vidéo en ligne.

I.6. Conclusion :

Dans ce chapitre, nous avons traité certaines notions des nouvelles technologies de l'information et de la communication, voir : l'Internet, le web, l'architecture client/serveur et les applications web.

Les différents concepts traités au cours de ce chapitre nous aiderons à mieux comprendre notre mode d'opération et les notions fondamentales pour mener à bien notre travail.

Chapitre II :

Présentation de l'organisme d'accueil

II.1. Introduction :

Nous consacrons se chapitre à la présentation de l'organisme d'accueil pour bien comprendre le fonctionnement de l'organisme et mettre en évidence les acteurs intervenants dans ce système ainsi que leurs besoins.

II.2. Historique :

Mobilis, filiale d'Algérie Télécom, est le premier opérateur de téléphonie mobile en Algérie. Devenu autonome en Août 2003, il subit une réorganisation en Juin 2004 et une deuxième réorganisation en 2006 où on voit la naissance des divisions.

L'organisation d'ATM Mobilis s'est renforcée avec la création du poste de divisionnaire principale en 2010.

En Décembre 2004, Mobilis a lancé le premier réseau d'UMTS (Universel Mobile Telecomunication System) en Algérie, intégrant ainsi le club des quarante opérateurs dans le monde qui maîtrisent cette technologie.

Devenu un véritable opérateur multimédia en Algérie, il propose à ses clients une large gamme de produits et de services innovants et de haute qualité « offre post et prépayées vers tous les opérateurs en Algérie et à l'étranger ».

Les produits Mobilis :

Cinq produits phares adaptés à tous les budgets et à tous les usages :

- Les offres prépayées « Mobilis carte, Gosto Mobilight ».
- Les offres postpayées « Mobiposte, Mobicontrole, Résidentiel et Forfait».
- Les offres pour les entreprises « Forfait groupe, Forfait select et Flotte »
- L'offre pour les étudiants « TAWFIK ».
- L'offre « Mobi+ » (GPRS-MMS).

Les services Mobilis :

Mobilis offre à ses clients post et prépayés une large gamme de services :

- Consultation de la facture sur Internet.
- 3G.
- MMS : envoi et réception de photos, fichiers musicaux.
- Un portail WAP.

- Notification gratuite du solde après chaque appel.
- SMS vers tous les opérateurs en Algérie et à l'étranger.
- Roaming à l'international.
- Messagerie vocale.
- Tarification des appels à la seconde après la première minute.
- Double appel.
- Les rechargements électroniques, Arsselli, Sellekni…

Les ambitions de Mobilis :

- Reprendre rapidement ses parts de marché.
- S'inscrire à l'avant-garde de l'innovation.
- Développer l'expertise et la performance.
- Etre constamment compétitif (qualité, prix, et services).
- Générer des profits et de la croissance.
- Participer au développement national.

II.3. Organigramme général de l'ATM Mobilis:

ATM Mobilis est dirigé par un Président Directeur Général (PDG) et entouré de Divisionnaires, Directeurs Centraux et Régionaux et de Conseillers. Son conseil d'administration est composé de membres issus d'horizons professionnels différents, qui valident les choix stratégiques de la direction.

ATM Mobilis se décompose en trois principales divisions : la Division des Affaires Internes, la Division Commerciale et Marketing, et la Division Réseau et Services. Ces trois divisions sont soutenues par des directions autonomes : la Direction des Finances et de la Comptabilité, la Direction Stratégie et Programmation et Performance, la Direction des Systèmes d'information, la Direction de la Communication et de la Marque, la Direction Ressources Humaines et la Direction Audit.

Un organe de conseil est mis à la disposition du PDG, entouré par divers conseillers spécialisés dans différents domaines d'activités (technique, ressources humaines, finance, juridique et affaires générales…).

Un découpage régionalisé permet également de distinguer 8 directions régionales gravitant autour de villes importantes : Alger, Constantine, Chlef, Setif, Bechar, Annaba, Oran, Ouargla.

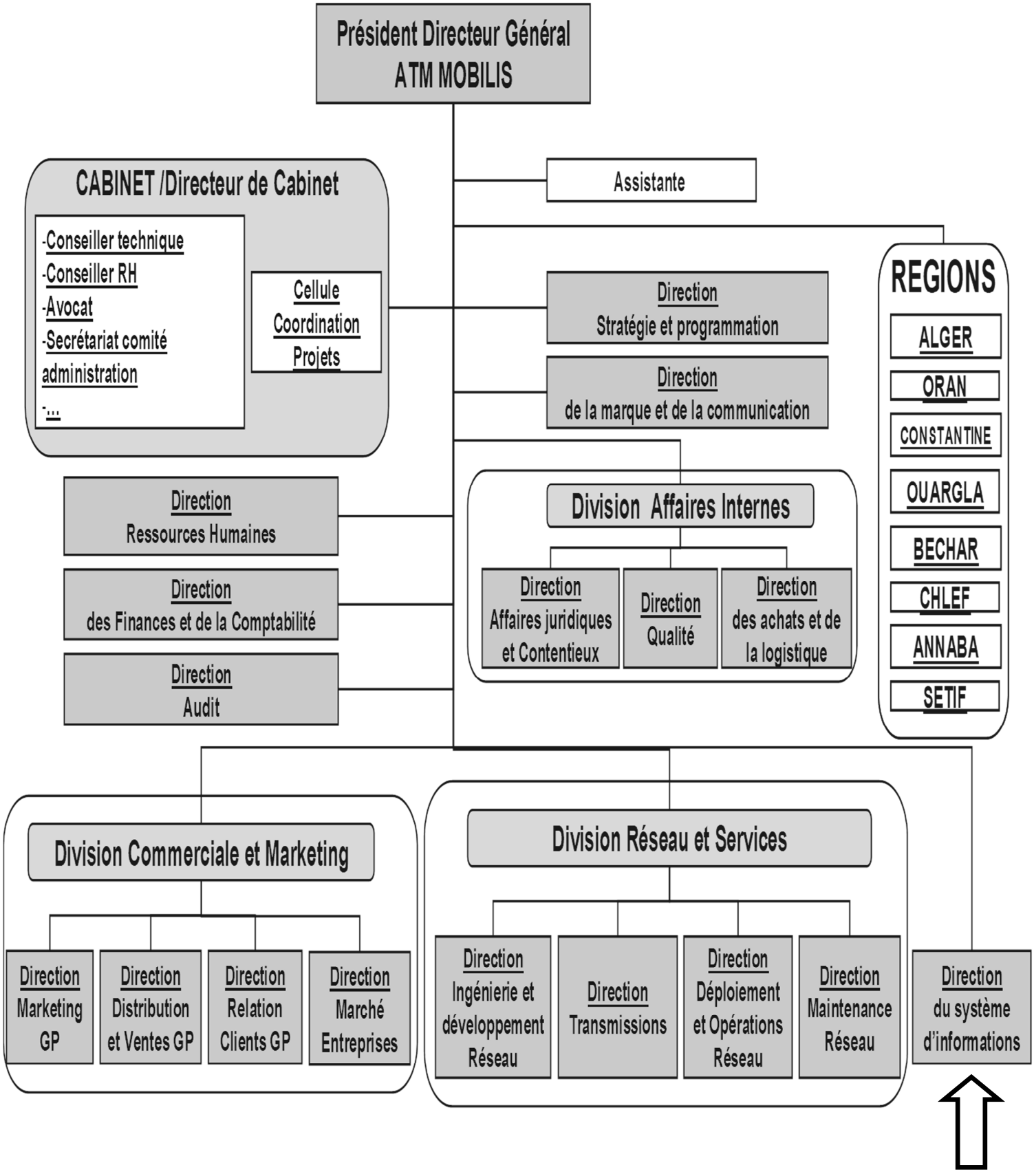


Figure II.1. Organigramme général de l'ATM Mobilis.

II.4. Structure de l'ATM Mobilis :

Nature	Classification	Structure Organisationnelle
Structures opérationnelles	*Structure centrale business*	• La Division Commerciale et Marketing regroupant: - La Direction du Marketing Grand Public - La Direction de la Distribution et des Ventes Grand Public - La Direction de la relation client Grand Public - La Direction du Marché Entreprises
	Structure centrale Business transverse	• La Direction de la Marque et de la Communication
		• La Direction de la Stratégie et de la Programmation
	Structure centrale technique	• La Division Réseau et Services regroupant: - La Direction de l'Ingénierie et du Développement Réseau - La Direction des Transmissions - La Direction du Déploiement et des Opérations Réseau - La Direction de la Maintenance Réseau **- La Direction du système d'information**
	Structure territoriale	• Les Directions Régionales
Structures fonctionnelles	*Structure centrale support*	• La Division des affaires internes regroupant: - La Direction des Affaires juridiques et Contentieux - La Direction de la Qualité - La Direction des achats et de la logistique
		• La Direction de l'Audit
		• La Direction des Finances et de la Comptabilité
		• La Direction des Ressources Humaines
Cabinet	*Structure centrale*	• Le Cabinet du Président regroupant : - Un Directeur de Cabinet qui assure également le secrétariat du COMEX - Un avocat conseil - Un Conseiller Technique - Un Conseiller en charge du développement des ressources humaines - Un assistant en charge du secrétariat technique - Une cellule de coordination des projets

II.5. Présentation de la direction de système d'information (DSI):

Pour une entreprise comme Algérie Télécom Mobilis qui a son actif des millions de clients, ses soucis sont de satisfaire la clientèle d'une part et d'autre part de conquérir de nouvelles parts de marché.

Donc pour qu'ATM Mobilis reste des leaders de la téléphonie mobile, il faut qu'elle ait un effectif très compétent, créatif et connaisseur de la valeur du temps.

Pour cela, il faut mettre en valeur la disponibilité du matériel informatique pour permettre la résolution des différents problèmes en un temps réduit, d'où l'augmentation de la productivité de chaque employé Mobilis en le motivant à résoudre tous les types de problèmes sans aucun retard « utilisateur de l'outil informatique ».

Tout cela a emmené ATM Mobilis à se doter d'un département de gestion du parc informatique.

II.5.1. Organigramme de la DSI :

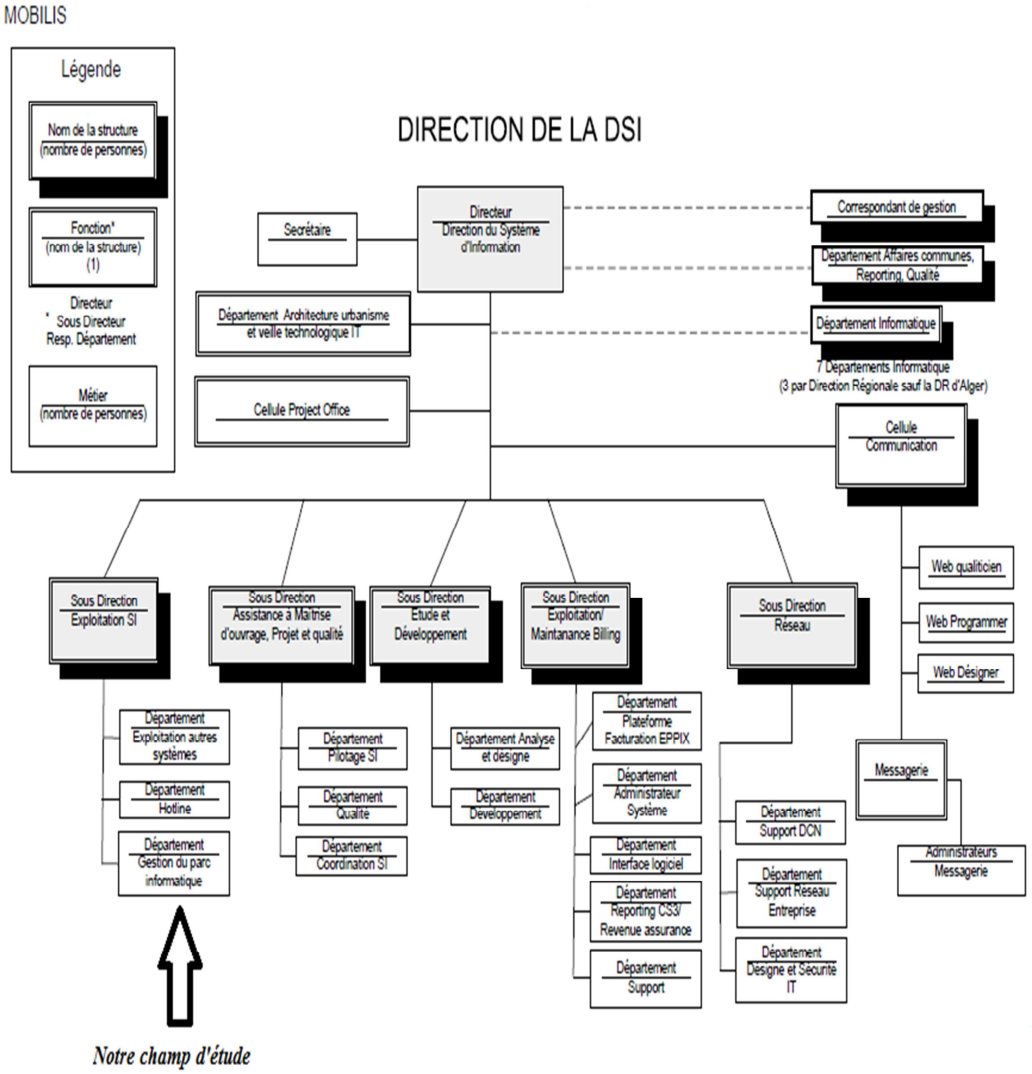


Figure II.2. Organigramme de la DSI.

II.5.2. Département Gestion du Parc Informatique (Champ d'étude) :

Le département dispose de deux effectifs, d'un chef de département et d'un gestionnaire de stock, qui gèrent le parc.

Les activités de ce département sont :

- Gestion de stock.
- Achat, acquisition du matériel (pour la DSI).

- Suivi des contrats (avec les prestataires).
- Réparation du matériel défectueux (contrat avec les sous-traitants et demande de réparation pour la DAL).
- Faire un inventaire de stock chaque année (physique avec valorisation) pour le matériel réseau.

Analyse des postes de travail :

Le poste de travail peut être considéré comme unité d'action dans une organisation, d'où la nécessité de déterminer tous les postes de travail concernant notre domaine d'étude.

Notre domaine d'étude comporte un poste de travail dont :

- Gestionnaire du Parc Informatique.

Pour ce poste de travail, on spécifiera les points suivants :

- ✓ Les responsabilités du poste.
- ✓ Les tâches accomplies par le poste.
- ✓ Les documents entrants au poste.
- ✓ Les documents sortants du poste.

Etude des postes de travail :

Une fiche d'analyse nous permettra de recueillir toutes les informations concernant le poste de travail cité précédemment.

Gestionnaire du Parc Informatiques :

Désignation du poste : Gestion du Parc Informatique.
Structure de rattachement : Sous Direction d'Exploitation et Autres systèmes.
Effectif : 02.
Responsabilités : Assurer la gestion du département Gestion du Parc Informatique.

Tâches :

- Réaliser les achats auprès des fournisseurs (appel d'offre, bon de commande, réception, demande de paiement).
- Suivi des dossiers d'achats.
- Suivi des contrats.
- Gestion des stocks (réception, livraison, inventaire, matériels sous garantie défectueux à changer).
- Gestion des rayonnages d'archives des dossiers.
- Etablir les états statistiques.
- Transmettre les besoins en matériels à la DAL (Direction Achat et Logistique).
- Prise en charge de la maintenance du matériel endommagé des utilisateurs avec la DAL.

Les documents entrants au poste :

Désignation	***Nombre d'exemplaire***	***Fréquence***	***Provenance***
Bon de commande interne	01	Aléatoire	Utilisateur Mobilis/ Directions régionales
Bon de sortie magasin	02	A la demande	DAL
Décharge réparation	02	A la demande	DAL

Les documents sortant du poste :

Désignation	***Nombre d'exemplaire***	***Fréquence***	***Destination***
Décharge	02	A la demande	Utilisateur Mobilis/ Directions régionales
Bon de retour interne	02	A la demande	Utilisateur Mobilis/ Directions régionales
Bon de sortie	02	A la demande	Directions régionales
Bon de commande interne Matériel	02	A la demande	DAL
Bon de retour externe	02	A la demande	Fournisseur
Demande de réparation	02	A la demande	DAL
Fiche d'inventaire	02	Annuel	Sous commission d'inventaire

II.6. Description des besoins de l'entreprise :

Afin de répondre aux besoins des utilisateurs de Mobilis en matière du matériel informatique, la direction système d'information de Mobilis nous a proposée de concevoir et de réaliser un système pour « la gestion du parc informatique ».

Pour cela, on aura à suivre les opérations suivantes :

- Gestion des commandes et des réceptions.
- Suivi des décharges.
- Suivi des statistiques.

L'outil doit être implémenté dans une plateforme client/serveur.

Objectifs :

Le système projeté devra réduire les tâches routinières et permettra un bon suivi de la gestion du parc informatique, ceci réalisé par :

- Avoir un traitement automatique des différentes opérations liées à la gestion du parc informatique.
- Faciliter l'accès à l'information utile en temps voulu.
- Assurer une meilleure fiabilité et confidentialité des informations.
- Alléger les tâches administratives et routinières ainsi les procédures de gestion.
- Accélérer les échanges d'information.

II.7. Conclusion :

ATM Mobilis est une entreprise nationale instaurée pour créer la richesse et générer du progrès, protéger et défendre les intérêts du consommateur en Algérie, assurer la qualité du service, la transparence, l'esprit d'équipe, le respect des engagements, l'éthique, l'innovation, l'excellence, le travail bien fait, le mérite et l'honnêteté.

Nous avons vu dans ce chapitre l'organisation générale de l'entreprise d'accueil ainsi que la structure de notre champ d'étude qui est le département gestion du parc informatique de la direction de système d'information et ses besoins.

Chapitre III :

Le Framework Hibernate

III.1. Introduction :

Travailler dans les deux univers que sont l'orienté objet et la base de données relationnelle peut être lourd et consommateur en temps dans le monde de l'entreprise d'aujourd'hui. Un outil de mapping objet/relationnel pour le monde de Java a donc été mise en place : Hibernate. Le terme mapping objet/relationnel (ORM) décrit la technique consistant à faire le lien entre la représentation objet des données et sa représentation relationnelle basé sur un schéma SQL.

Suite à cela, de nombreuses versions d'Hibernate ont été mises à disposition des utilisateurs afin de le faire évoluer. Hibernate se base sur l'architecture Modèle-Vue-Contrôleur qui fut introduite comme partie du SmallTalk80 afin de limiter les efforts de programmation liés à l'élaboration de système.

III.2.Introduction aux Frameworks :

III.2.1. Définition :

Le Framework désigne le cadre dans lequel va s'insérer une application. En programmation orientée objet, il désigne l'infrastructure logicielle qui facilite la conception des applications par l'utilisateur de bibliothèques de classes ou de générateurs de programmes.

C'est une bibliothèque de classes fournissant une ossature générale pour le développement d'une application dans un domaine particulier. Ces composants sont organisés afin d'être utilisés en interaction les uns avec les autres et sont spécifiques généralement à un type d'application.

Les Frameworks facilitent ainsi le travail du développement en fournissant un squelette d'application qu'il suffit de remplir pour l'adapter à ses besoins. La contrepartie est qu'un Framework représente un sur ensemble de tous les besoins génériques, ce qui conduit à supporter un grand nombre de choses même si souvent, une toute petite partie est utile pour le cas à réaliser. Ceci complique également la tâche d'apprentissage et d'assimilation de l'environnement par le développeur.

Un Framework est un ensemble de classes abstraites qui, dans le but de faciliter la création d'une partie d'un système logiciel, collaborent entre elles. I l partage le domaine visé en classes abstraites et permet de définir les responsabilités de chacune par rapport aux autres et les rapports entre elles.

III.2.2. Différents exemples de Frameworks :

MFC :

Les MFC de Microsoft est un exemple de Framework qui permet de développer une application en C++ basées sur une architecture Fenêtre Cadre-Document-Vue.

MFC est donc une bibliothèque de classes C++ fournissant un cadre général pour la programmation sous Windows. MFC encapsule une grande partie des fonctions API Windows et améliore la logique de la création des interfaces.

L'objet principal du MFC est donc de fournir un cadre général pour développer facilement des interfaces graphiques. Pour cela, il fournit une architecture de programme basée sur la notion de document et vue. Pour les MFC, une interface graphique est un moyen parmi d'autres de visualiser des données (document) ; cette visualisation (vue) s'effectuant dans une fenêtre cadre elle-même pilotée par un programme principal.

Jarka Struts :

Struts est un exemple de Framework open source, basé sur l'architecture MVC (Modèle vue Contrôleur). C'est un projet d'un Framework faisant partie de « Apache Jarka Project ». In sert à développer des applications pour le web.

Le cœur du Framework Struts est une couche contrôleur basée sur les technologies les plus acceptées comme la JSP, le JavaBeans, et le XML. Struts encourage les architectures basées sur l'approche Model 2, qui est variante du modèle classique MVC. Struts fournit son propre composant contrôleur et intègre d'autres technologies d'accès aux données comme les EJB, le JDBC. Pour la vue, Struts fonctionne bien avec les JSP, les Velocity Templates et d'autres systèmes de présentation.

Spring :

Spring est un conteneur dit « léger », c'est-à-dire une infrastructure similaire à un serveur d'application J2EE. Il prend donc en charge la création d'objets et leurs mises en relation par l'intermédiaire d'un fichier de configuration qui décrit les objets à fabriquer et les relations de dépendances entre ceux-ci.

Le gros avantage par rapport aux serveurs d'application est qu'avec Spring, vos classes n'ont pas besoin d'implémenter une quelconque interface pour être prises en charge par le Framework (au contraire des serveurs d'applications J2EE et des EJBs). C'est en ce sens que Spring est qualifié de conteneur « léger ».

Outre cette espèce de fabrication d'objets, Spring propose tout un ensemble d'abstractions permettant de gérer entre autres :

- ✓ Le mode transactionnel.
- ✓ L'appel d'EJB.
- ✓ La création d'EJB.
- ✓ La persistance d'objets.
- ✓ La création d'une interface Web.
- ✓ L'appel et la création de WebServices.

Pour réaliser tout ceci, Spring s'appuie sur les principes du design pattern IoC et sur la programmation par aspects (AOP).

III.3. Framework Hibernate :

III.3.1. Définition :

Hibernate est un logiciel écrit sous la responsabilité de Gavin King, qui fait entre autre de l'équipe de développement de JBoss.

Hibernate est un Framework en open source, gérant la persistance des objets (qui peuvent être défini par les propriétés, les méthodes ou les évènements qu'il est susceptible de déclencher) dans une base de données relationnelle. La persistance des objets représente la possibilité d'enregistrement de l'état d'un objet, par exemple dans une base de données, afin de pouvoir le recréer plus tard. Quant à la base de données relationnelle, elle contient de nombreuses tables et l'information est organisée par différentes relations entre tables. Pour information, on appelle SGBDR un logiciel mettant en œuvre une telle base de données.

L'ensemble des données nécessaires au fonctionnement de l'application est sauvegardé dans une base de données. La manipulation des données peut se faire de différentes manières : par l'accès directement à la base en écrivant les requêtes SQL adéquates, utiliser un outil d'ORM (Object Relationnal Mapping) permettant de manipuler facilement les données et d'assurer leur persistance : c'est le cas d'Hibernate.

Pourquoi ajouter une couche entre l'application et la base de données ? L'objectif est de réduire le temps de développement de l'application en éliminant une grande partie du code SQL à écrire pour interagir avec la base de données et en encapsulant le code SQL résiduel. Les développeurs manipulent les classes dont les données doivent être persistantes comme des classes Java normales. Seule une initialisation correcte d'Hibernate doit être effectuée, et quelques règles respectées lors de l'écriture et de la manipulation des classes persistantes.

Hibernate se place à un autre niveau que le Framework Struts qui lui gère l'interface homme-machine et se base sur l'architecture MVC. Hibernate est agnostique en terme d'architecture, il peut donc être utilisé aussi bien dans un développement client lourd que dans un environnement Web léger de type Tomcat (Apache) ou dans un environnement J2EE complet (Weblogic, Websphere, JBoss).

Non seulement, Hibernate s'occupe du transfert des classes Java dans les tables de la base de données (et des types de données Java dans les types de données SQL), mais il permet de faire des requêtes sur les données et propose des moyens de les récupérer. Il peut donc réduire de manière significative le temps de développement qui aurait été dépensé autrement dans une manipulation manuelle des données via SQL et JDBC. Le but d'Hibernate est de libérer le développeur de 95% des tâches de programmation liées à la

persistance des données communes. Hibernate n'est probablement pas la meilleurs solution pour les applications centrées sur les données qui n'utilisent que les procédures stockées pour implémenter la logique métier dans la base de données, il est plus utile dans les modèles métier orientés objets dont la logique métier est implémentée dans la couche Java dite intermédiaire. Cependant, Hibernate aide à supprimer ou à encapsuler le code SQL spécifique à la base de données et aide sur la tâche commune qu'est la transformation des données d'une représentation tabulaire à une représentation sous forme de graphe d'objets.

III.3.2. Architecture d'Hibernate :

Ci-après, une vue haut niveau d'architecture d'Hibernate.

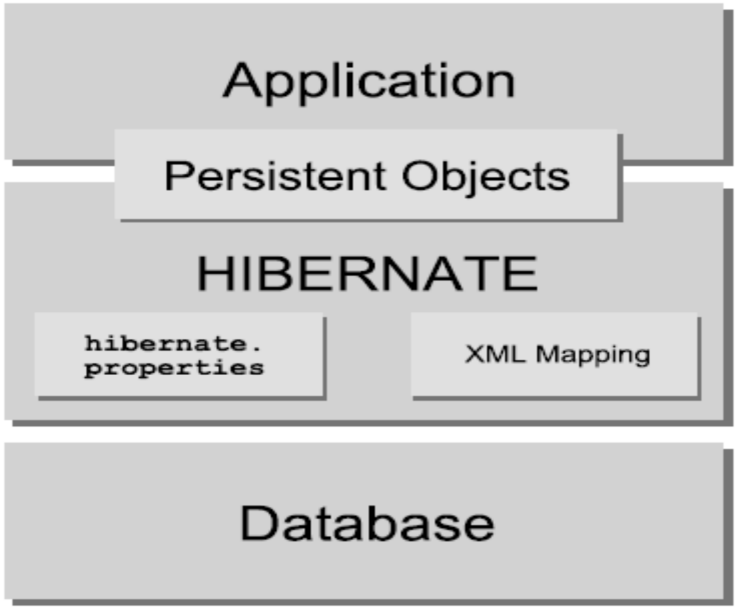


Figure III.1. Vue haut niveau d'architecture Hibernate.

Ce diagramme montre Hibernate utilisant une base de données et des données de configuration pour fournir un service de persistance (et des objets persistants) à l'application. Hibernate est flexible et supporte différentes approches. En voici deux approches très extrêmes :

- L'architecture légère laisse l'application fournir ses propres connexions JDBC et gérer ses propres transactions. Cette approche utilise le minimum des API Hibernate :

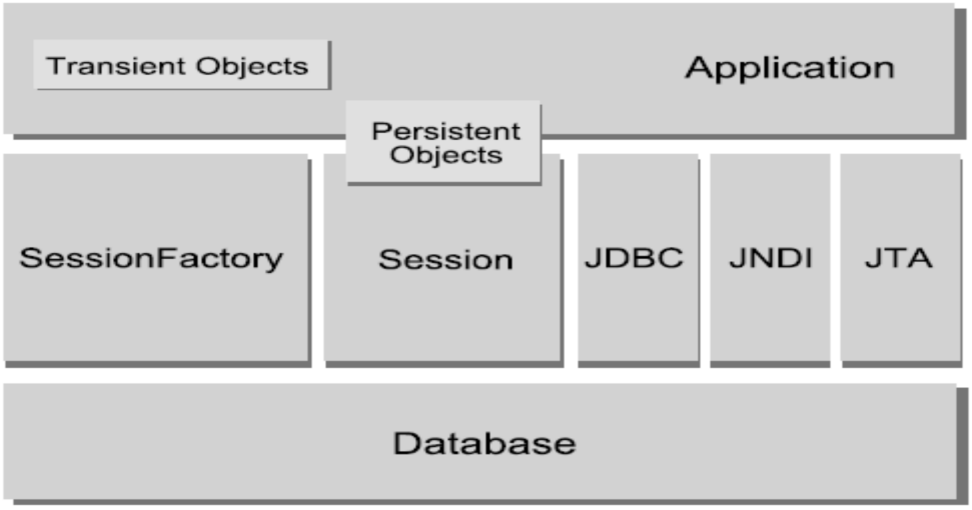


Figure III.2. Architecture légère d'Hibernate.

- L'architecture la plus complète fait abstraction de l'application des API JDBC/JTA sous-jacentes et laisse Hibernate s'occuper des détails.

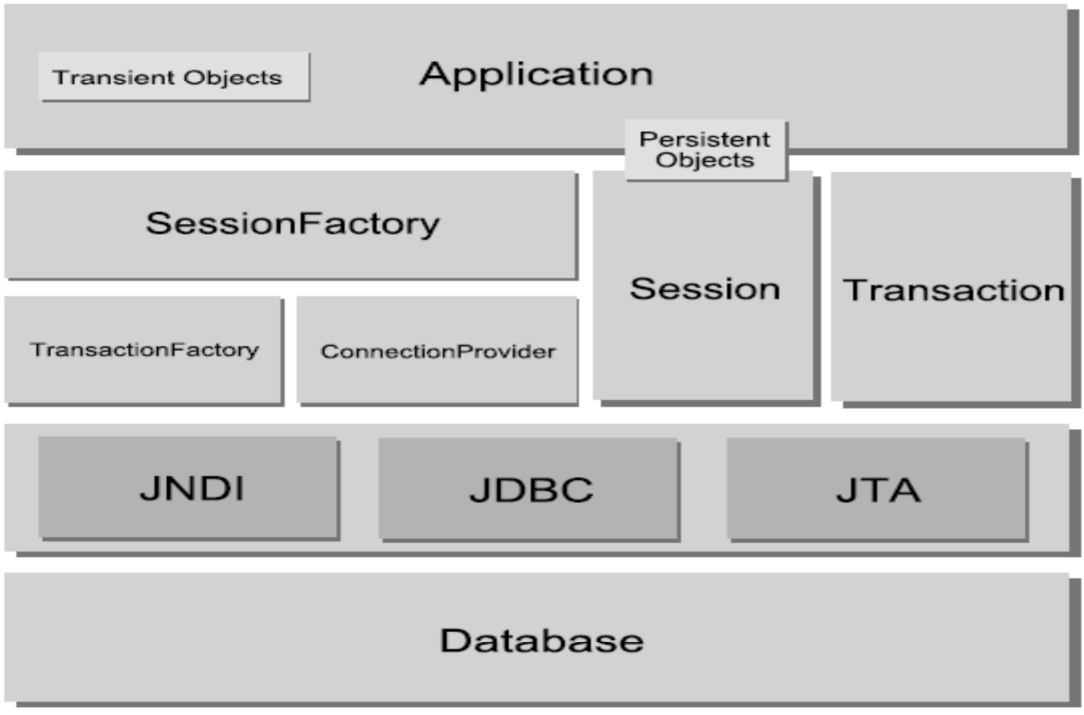


Figure III.3. Architecture complète d'Hibernate.

Nous allons maintenant définir les objets des diagrammes :

- **SessionFactory :** Un cache (threadsafe) des mappings vers une base de données. Il peut contenir un cache optionnel de données (de second niveau) qui est réutilisable entre les différentes transactions que cela soit au niveau du processus ou au niveau du cluster.
- **Session :** Un objet mono-threadé, à durée de vie courte, qui représente une conversation entre l'application et l'entrepôt de persistance. Elle encapsule une connexion JDBC. Elle contient un cache (de premier niveau) des objets persistants, ce cache est obligatoire. Il est utilisé lors de la navigation dans le graphe d'objets ou lors de la récupération d'objets par leur identifiant.
- **Objets et collections persistants :** Objets mono-threadé à vie courte contenant l'état de persistance et la fonction métier. Ceux-ci sont en général les objets de type JavaBean (ou POJO) ; la seule particularité est qu'ils sont associés avec une session. Dès que la session est fermée, ils seront détachés et libre d'être utilisés par n'importe laquelle des couches de l'application, c'est-à-dire de et vers la présentation en tant que Data Transfer Objects (objets de transfert de données).
- **Objets et collections passagers :** Instances de classes persistantes qui ne sont actuellement pas associées à une session. Elles ont pu être mises en instance par l'application, et ne pas avoir (encore) été persistées ou par une session fermée.
- **Transaction :** Objet mono-threadé à vie courte utilisé par l'application pour définir une unité de travail atomique et qui abstrait l'application des transactions sous-jacentes qu'elles soient JDBC, JTA, CORBA. Une session peut fournir plusieurs transactions dans certain cas.
- **ConnectionProvider :** Lieu de fabrication de connexion JDBC. Elle abstrait l'application de la source de données ou du manager sous-jacent de pilotes. Elle n'est pas exposée à l'application, mais peut être étendue et implémentée par le développeur.
- **TransactionFactory :** Fabrique d'instances de Transaction. Non exposée à l'application, mais peut être étendue et implémentée par le développeur.

Pour information, dans une architecture légère, l'application n'utilisera pas les APIs Transaction et TransactionFactory et/ou les APIs ConnectionProvider pour utiliser JTA ou JDBC.

III.3.3. Comparatif avec Hibernate :

Les recherches ont fait ressortir trois Frameworks de persistance dont les fonctionnalités sont intéressantes : Hibernate, JPOX et OJB. Nous avons alors établi un tableau récapitulatif des avantages et des inconvénients de chacun.

Framework	Avantages	Inconvénients
Hibernate	- Propose une API performance et robuste. - Est mature et dispose d'un support fiable et d'une documentation abondante. - Supporte la gestion des transactions. - S'intègre facilement à Eclipse et Spring. - Appropriation rapide.	- Ne se base pas sur les spécifications et standards JDO ou ODMG. - Nécessite un module pour la connectivité aux bases Oracle. - Possibilité de perte de vitesse après l'acceptation des spécifications JDO 2.0.
JPOX	- Une implémentation de référence des JSR JDO 1.0 et 2.0. - En gain de crédibilité. - Projet promu par SUN.	- Framework en version alpha. - Documentation insuffisante.
OJB	- Pas de RoadMap pour l'implémentation des spécifications JDO 2.0.	- Implémentation des spécifications JDO 1.0.

JPOX reste plus complet qu'OJB en terme de gestion transactionnelle et de conformité aux dernières JDO. Il a été prouvé que le Framework de persistance le plus intéressant reste malgré tout le JPOX.

Hibernate est beaucoup plus utilisé que JDO, une norme qui gère la persistance des objets. Par contre JDO ne se limite pas seulement aux bases de données relationnelles ; il peut gérer notamment le XML.

Hibernate ne respecte pas les specs JDO. Il ne le fera d'ailleurs jamais, car Hibernate constitue l'implémentation de référence qui a servi à monter les specs EJB 3.0.

Hibernate est beaucoup moins lourd à développer et à administrer que les EJB entité dans les versions 2. La norme EJB 3 en phase d'élaboration, reprend les expériences et le modèle de programmation d'Hibernate ou de Toplink. Hibernate dans sa version 3 supporte la norme EJB 3 et serait désigné comme LE standard de l'avenir par SUN.

III.3.4. Configuration d'Hibernate :

Hibernate est conçu pour pouvoir être utilisé dans différents types d'architectures d'application. Pour cela, chaque application doit indiquer à Hibernate comment celui-ci peut accéder et manipuler la source de données dans un fichier de configuration nommé « hibernate.cfg.xml ». Les principaux éléments à paramétrer sont les suivants :

- ✓ Le SGBD utilisé. Chaque SGBD propose une implémentation du langage SQL qui diffère souvent de la norme SQL. Hibernate doit connaitre le type de SQL qu'il doit générer.
- ✓ La connexion à la base de données. Si la connexion à la base de données se fait en utilisant JDBC, il faut indiquer à Hibernate, le driver JDBC, l'url de connexion ainsi qu'un nom d'utilisateur et un mot de passe permettant de se connecter à la base de données. Les connexions peuvent également être gérées par un serveur d'application. Dans ce cas, il faut indiquer à Hibernate comment il peut accéder aux connexions créées par ce serveur (Annuaire JNDI).
- ✓ Les Services tiers. Hibernate a besoin de gérer un ensemble (pool) de connexion à la base de données et un cache de données. Pour cela, Hibernate propose une implémentation rudimentaire de ces services mais peut aussi utiliser des services tiers plus performants.

Pour résumer, le paramétrage de Hibernate nécessite :

- ✓ La définition du modèle de classes exploitant la base de données.
- ✓ Une correspondance (mapping) entre le modèle de classes et la base de données.
- ✓ Une configuration au niveau système de l'accès.

III.3.5. La notion de mapping relationnel-objet :

Cette problématique est résolue par la notion de mapping relationnel-objet (également appelé ORM pour Object Relational Mapping), qui permet de rendre transparent le pont entre le monde orienté objet du Java et le monde relationnel des SGBD (Systèmes de Gestion de Base de données). Le mapping relationnel-objet permet ainsi de faire automatiquement les transcriptions suivantes :

Monde objet	Monde relationnel
Classe	Table
Attribut	Colonne
Instance	Enregistrement
Association	Clé étrangère

On parle alors de :

- ✓ Classe persistance : classe dont les instances peuvent être rendues persistantes, c'est-à-dire enregistrées en base de données.
- ✓ Instance persistante : instance d'une classe persistante dont les données sont présentes dans la base de données.

- ✓ Instance transiente ou volatile : instance d'une classe persistante dont les données ne sont pas présentes dans la base.

La création d'un fichier de correspondance :

Pour assurer le mapping, Hibernate a besoin d'un fichier de correspondance (mapping file) au format XML qui va contenir des informations sur la correspondance entre la classe définie et la table de la base de données.

Même si cela est possible, il n'est pas recommandé de définir un fichier de mapping pour plusieurs classes. Le plus simple est de définir un fichier de mapping par classe, nommé du nom de la classe suivi par « .hbm.xml ». Ce fichier doit être situé dans le même répertoire que la classe correspondante ou dans la même archive pour les applications packagées.

Différents éléments sont précisés dans ce document XML :

- ✓ La classe qui va encapsuler les données.
- ✓ L'identifiant dans la base de données et son mode de génération.
- ✓ Le mapping entre les propriétés de classe et les champs de la base de données.
- ✓ Les relations…

Exemple de fichier de mapping :

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
	"-//Hibernate/Hibernate Mapping DTD//EN"
	"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd" >

<hibernate-mapping package="com.mobilis.hibernate">
	<class name="Administrateur" table="administrateur">
		<composite-id>
			<key-property
				column="Password"
				name="Password"
				type="string"
			 />
			<key-property
				column="Login"
				name="Login"
				type="string"
			 />
		</composite-id>
	</class>
</hibernate-mapping>
```

III.3.6. Le langage de requête HQL :

Pour offrir un langage d'interrogation commun à toutes les bases de données, Hibernate propose son propre langage nommé HQL (Hibernate Querry Language).

L'intérêt de HQL est d'être indépendant de la base de données sous-jacente : la requête SQL sera générée par Hibernate à partir du HQL en fonction de la base de données précisée via un dialect.

Hibernate Query Language (HQL) est un langage de requête orienté objects qui permet de représenter des requêtes SQL : Les entités utilisées dans les requêtes HQL sont objets et des propriétés. La syntaxe de HQL et ses fonctionnalités de base sont très similaire à SQL.

Il est aussi possible d'utiliser l'API Criteria qui va en interne exécuter une requête HQL. Le point d'entrée est d'obtenir une instance de type Criteria en invoquant la méthode createCriterie() de la session Hibernate courante : elle attend en paramètre la classe des objects attendus en résultat. L'API permet de préciser les différents critères qui seront utilisés pour générer la requête. L'API Criteria utilise HQL en sous-jacent.

Hibernate propose aussi un support pour exécuter des requêtes natives. Ceci permet d'utiliser des fonctionnalités de la base de données sous-jacente qui ne soient pas supportées par HQL. Cependant dans ce cas, le support multi-base de données offert par HQL sera probablement compromis.

Le langage HQL est proche de SQL avec une utilisation sous forme d'objets des noms de certaines entités : il n'y a aucune référence aux tables ou aux champs car ceux-ci sont référencés respectivement par leur classe et leurs propriétés.

C'est Hibernate qui se charge de générer la requête SQL à partir de la requête HQL en tenant compte du contexte (type de base de données utilisée défini dans le fichier de configuration et la configuration du mapping).

La syntaxe de HQL :

HQL possède une syntaxe similaire de celle de SQL : la différence majeure est que HQL utilise des objets et leurs propriétés alors que SQL utilise des tables et leurs colonnes.

Exception faite des noms de classes et de variables, les requêtes HQL ne sont pas sensibles à la casse. Généralement, les mots clé HQL sont en minuscule pour faciliter leur lecture.

Une requête HQL peut être composée :

- ✓ De clauses.
- ✓ De fonctions d'agrégation.

- ✓ De sous requêtes.

Les clauses sont les mots clés HQL qui sont utilisés pour définir la requête.

Clause	Description	Syntaxe	Exemple
from	Précise la classe d'objets dont les occurrences doivent être retrouvées. Il est possible de définir un alias pour un objet en utilisant le mot clé alias.	from object [as objectalias]	from Personne as pers (retourne toutes les occurrences de type Personne).
select	Précise les propriétés à renvoyer. Doit être utilisé avec une clause from.		select pers.nom from Personne as pers (retourne le nom de toutes les personnes).
where	Précise une condition qui permet de filtrer les occurrences retournées. Doit être utilisé avec une clause select et/ou from.	where condition	from Personne ar pers where pers.nom= "Dupond" (retourne toutes les personnes don't le nom est Dupond).
order by	Précise un ordre de tri sur une ou plusieurs propriétés. L'ordre par défaut est ascendant.	order by propriété [asc|desc] [, propriété]… ;	select pres.nom, pers.prenom from Personne as pers order by pers.nom asc, pers.prenom desc
group by	Précise un critère de regroupement pour les résultats retournés. Doit être utilisé avec une clause select et/ou from.	group by propriété [, propriété]…	

Les fonctions d'agrégation HQL ont un rôle similaire à celles de SQL : elles permettent de calculer des valeurs agrégeant des valeurs de propriétés issues du résultat de la requête.

Fonction	Syntaxe
count	count([distinct|all|*] object | object.property)
sum	Sum([distinct|all] object.property)
avg	Avg([distinct|all] object.property)
max	Max([distinct|all] object.property)
min	Min([distinct|all] object.property)

Les sous requêtes sont des requêtes imbriquées dans une autre requête.

L'utilisation de sous requêtes dans HQL est conditionné par le support des sous requêtes par la base de données sous-jacente. Les sous requêtes sont entourés par des parenthèses : elles sont exécutées avant la requête principale puisque celle-ci a besoin des résultats pour son exécution.

La mise en œuvre de HQL :

La mise en œuvre de HQL peut se faire de plusieurs manières.

Le plus courant est d'obtenir une instance de la classe Query en invoquant la méthode createQury() de la session Hibernate courante : elle attend en paramètre la requête HQL qui devra être exécutée.

Exemple :

```
Session session ;
Query query= session.createQuery("from Personne as pers");
List result= query.list();
```

La méthode list() de la classe Query permet d'obtenir une collection qui contient les résultats de l'exécution de la requête. Il est également possible de définir des requêtes utilisant des paramètres nommés grâce à un objet implémentant l'interface Query. Dans ces requêtes, les paramètres sont précisés avec un caractère « : » suivi d'un nom unique.

L'interface Query propose de nombreuses méthodes serXXX() pour associer à chaque paramètre une valeur en fonction du type de la valeur (XXX représente le type). Chacune de ces méthodes possède deux surcharges permettant de préciser le paramètre (à partir de son nom ou de son index dans la requête) et sa valeur.

Pour parcourir la collection des occurrences trouvées, l'interface Query propose la méthode list() qui renvoie une collection de type List ou la méthode iterate() qui renvoie un itérateur sur la collection.

III.3.7. L'API Criteria :

En plus du langage HQL pour réaliser des requêtes d'extraction de données, Hibernate propose une API qui permet de construire des requêtes pour interroger la base de données. L'API Criteria d'Hibernate propose donc une alternative à HQL sous la forme d'une API.

Le HQL possède une syntaxe dérivée de celle de SQL dans laquelle les notions relationnelles sont remplacées par des notions objets. Ceci oblige les développeurs à utiliser une syntaxe proche de celle du SQL.

L'API Criteria Query propose des objets pour définir les critères d'une requête, ce qui permet aux développeurs de les définir d'une manière orientée objet plutôt que d'utiliser le HQL.

L'API Criteria propose donc d'avoir une approche orientée objet pour définir des requêtes et obtenir des données. L'utilisation de cette API permet d'avoir un meilleur contrôle grâce à la compilation.

Cette API permet de facilement combiner de nombreux critères optionnels pour créer une requête : elle est particulièrement adaptée pour créer dynamiquement des requêtes à la volée comme c'est le cas par exemple pour des requêtes effectuant des recherches multicritères à partir d'informations fournies par l'utilisateur.

Elle offre, pour la plupart des fonctionnalités, une approche bidirectionnelle :

- ✓ Appliquer un critère en désignant la propriété sur laquelle il s'applique.
- ✓ Appliquer un critère sur une propriété.

Elle propose des classes et des interfaces qui encapsulent les fonctionnalités de SQL dont les principales sont :

- ✓ Criteria.
- ✓ Criterion.
- ✓ Restrictions.
- ✓ Projection.
- ✓ Order.

L'interface org.hibernate.criterion.Criteria est le point d'entrée pour utiliser l'API Criteria. Elle permet de définir une requête à partir de critères pour retrouver des données.

La classe org.hibernate.criterion.Restrictions est une fabrique qui propose des méthodes statiques pour créer des instances de type Criterion.

Depuis la version 3.x d'Hibernate, il est préférable d'utiliser la classe Restrictions à sa classe fille Expressions.

L'interface org.hibernate.criterion.Projection encapsule un champ en réponse de la requête (un champ dans la clause "select" de la requête SQL).

La classe org.hibernate.criterion.Projections est une fabrique pour les instances de type Projection.

III.3.8. L'utilité de HQL et de l'API Criteria :

L'API Criteria permet de définir à la volée des recherches de données multicritères complexes. Un exemple typique d'utilisation où cette API est particulièrement utile est la recherche de données multicritères où elle va permettre de facilement construire dynamiquement les critères à appliquer en fonction de ceux précisés par l'utilisateur de l'application.

Ceci est d'autant plus vrai que le nombre de critères optionnels est important : la génération dynamique peut alors devenir particulièrement complexe surtout si elle implique des jointures conditionnées par les critères précisés.

L'approche traditionnelle consiste à construire dynamiquement la requête HQL par concaténation des chaînes de caractères qui correspondent à chaque critère renseigné avec généralement plusieurs problématiques à gérer :

- ✓ La gestion des opérateurs logiques notamment pour l'omettre dans le cas du premier critère.
- ✓ La gestion des jointures à utiliser (ajout des classes et des critères de jointures).
- ✓ L'utilisation directe des données dans la requête générée (pouvant par exemple conduire à des failles de sécurité dans les applications web).

III.3.9. Le choix entre HQL et l'API Criteria :

L'API Criteria est très puissante et elle est particulièrement bien adaptée pour certaines tâches (notamment la création de requêtes dynamiques à la volée impliquant de nombreux critères optionnels comme par exemple dans un formulaire de recherche multicritères). Dans ce cas, sa mise en œuvre peut permettre d'avoir un code plus propre, plus sûre et plus maintenable.

Cependant, son utilisation ne peut pas toujours être généralisée à tous les cas de figure car l'utilisation de HQL est parfois préférable, notamment si la requête n'est pas dynamique.

Il est par exemple préférable d'externaliser les requêtes HQL lorsque cela est possible ce qui présente plusieurs avantages :

- ✓ Faciliter de maintenance.
- ✓ Faciliter pour recenser et optimiser les requêtes.
- ✓ Faciliter pour mettre les requêtes en cache.

Il est donc nécessaire de bien choisir entre HQL et l'API Criteria en fonction des besoins et de leur adéquation avec ce que proposent les deux solutions qui se recouvrent mais sont aussi complémentaires.

III.3.10. Les relations :

Un des fondements du modèle de données relationnelles repose sur les relations qui peuvent intervenir entre une table et une ou plusieurs autres tables ou la table elle même. Les relations utilisables dans le monde relationnel et le monde objet sont cependant différentes.

Les relations du monde objets possèdent quelques caractéristiques :

- ✓ Elles sont réalisées via des références entre objets.
- ✓ Elles peuvent mettre en œuvre l'héritage et le polymorphisme.

Les relations du monde relationnel possèdent quelques caractéristiques :

- ✓ Elles sont gérées par des clés étrangères et des jointures entre tables

Les caractéristiques de ces deux modèles sont assez différentes : le but d'un outil de type ORM comme Hibernate est de permettre de manipuler des entités objets et de masquer au développeur le monde relationnel en assurant un mapping entre les deux mondes. Cependant Hibernate n'assure pas en automatique la gestion inverse des relations qui reste à la charge du développeur.

Hibernate propose de transcrire les relations du modèle relationnel dans le monde objet. Il supporte plusieurs types de relations :

- ✓ Relation de type 1-1 (one-to-one): Dans ce type de relation, deux entités sont liées de façon à n'avoir qu'une seule et unique occurrence l'une pour l'autre.
- ✓ Relation de type 1-n (one-to-many).
- ✓ Relation de type n-n (many-to-many).

Dans le fichier de mapping, il est nécessaire de définir les relations entre la table concernée et les tables avec lesquelles elle possède des relations.

Les relations peuvent aussi être définies avec des annotations.

III.3.11. Les caches d'Hibernate :

Un volume important d'échanges entre l'application et la base de données est fréquemment à l'origine des problèmes de performance d'une application. Le but d'un cache est de réduire les échanges entre l'application et la base de données en stockant en mémoire les entités déjà lues de la base de données. Une lecture dans la base de données n'est alors plus nécessaire que lorsque l'entité n'est déjà pas présente dans le cache (si l'entité n'est jamais lue ou si l'entité été retirée du cache). Le cache permet de stocker des données qui ont déjà été lues de la base de données, ce qui permet de réduire les échanges entre la base de données et l'application lorsque celle-ci a de nouveau besoin des données. L'accès à la mémoire est beaucoup plus rapide que l'accès à la base de données surtout si celui-ci nécessite un échange utilisant le réseau.

Hibernate propose une fonctionnalité de mise en cache des données qui peut permettre, si elle est correctement configurée et utilisée, d'améliorer les performances de l'application. Il est nécessaire de bien comprendre le mode de fonctionnement du cache pour l'utiliser à bon escient et le configurer en conséquence : dans le cas contraire, les performances peuvent se dégrader de façon sérieuse voire dramatique.

L'utilisation du cache peut donc être une solution à certaines problématiques de performance fréquemment rencontrée lorsqu'Hibernate n'est pas bien compris ou utilisé de manière non optimale. Même si ces deux critères sont parfaitement maîtrisés, le cache peut être utile pour améliorer les performances dans certaines circonstances notamment par exemple lors de la lecture de données statiques dans la base de données.

Hibernate propose d'utiliser un cache pour plusieurs types de fonctionnalités de manière à améliorer les performances en lecture et/ou écriture des entités dans la base de données :

- ✓ Cache de premier niveau : son utilisation est implicite car il est toujours actif et est utilisé par défaut. Implémenté dans la session, son champ d'actions est limité à la transaction courante.
- ✓ Cache de second niveau : son utilisation est optionnelle ; il doit être activé et configuré pour pouvoir être utilisé. Implémenté dans l'objet de type SessionFactory, son champs d'action est l'application : il est donc utilisable par toutes les transactions.
- ✓ Le cache de requêtes : son utilisation est optionnelle ; il doit être activé et configuré pour pouvoir être utilisé. Sa mise en œuvre utilise le cache de second niveau.

Une bonne connaissance de la façon dont ces caches fonctionnent et de la manière dont ils interagissent avec les autres API est essentielle pour obtenir les meilleurs résultats. Activer le cache et configurer les entités est facile mais cela ne garantie pas de pouvoir tirer le meilleur parti des fonctionnalités proposées par les caches d'Hibernate.

Un objet de type Session est une unité de travail, qui correspond à une transaction côté base de données : elle stocke l'état sur les entités suite à des lectures et/ou modifications et/ou suppressions. Ces modifications et suppressions sont transformées en requêtes SQL pour être reportées dans la base de données. Le stockage de cet état représente le premier niveau de cache d'Hibernate.

Dans le cache de premier niveau, implémenté dans la session, les entités sont directement stockées dans le cache. Si une entité est de nouveau récupérée de la session alors qu'elle a déjà été chargée, alors c'est le même objet qui encapsule l'entité qui est retourné. Ceci ne pose aucun problème puisque la portée de la session est la transaction courante. Ce mode de fonctionnement est utilisé jusqu'à ce que la transaction soit terminée ou que la méthode flush de la session soit invoquée.

Le cache de premier niveau est essentiellement utilisé pour limiter le nombre de requêtes SQL requises par la transaction : par exemple, si une entité est modifiée plusieurs fois durant la transaction, l'état final de l'entité est stocké dans la session qui ne générera qu'une seule requête SQL de type update, par défaut à la fin de la transaction.

Le cache de second niveau stock les objets lus de la base de données au niveau de l'objet SessionFactory : les entités sont donc partagées entre les transactions et utilisables au niveau de l'application. Dans ce cas,

lorsqu'une transaction a besoin de lire une entité, si celle-ci est déjà présente dans le cache alors l'exécution d'une ou plusieurs requêtes SQL est évitée. L'utilisation du cache de second niveau est optionnelle. La durée de vie ce cache est liée à celle de l'application. La durée de vie des entités contenues dans le cache est configurable selon le cache utilisé. Le cache de second niveau requiert l'utilisation d'une implémentation d'un cache par un tiers.

Par défaut, si le cache de second niveau est activé, la recherche d'une entité se fait prioritairement dans le cache de premier niveau, puis dans le cache de second niveau et enfin dans la base de données par l'exécution d'une requête SQL.

Hibernate propose une fonctionnalité qui permet de mettre en cache le résultat de requêtes SQL. Cette fonctionnalité possède des contraintes mais peut être intéressante dans certains cas de figure.

Hibernate propose deux types de cache, chacun ayant un but précis :

- ✓ Le cache de niveau 1 ou cache de premier niveau : il est toujours actif car il est implémenté dans la classe Session. La visibilité de ce cache est la transaction. Sa durée de vie est donc celle de la session. Le but est de réduire le nombre de requêtes SQL d'une même transaction.
- ✓ Le cache de niveau 2 ou cache de second niveau : il n'est pas actif par défaut. Il est implémenté dans la SessionFactory. La visibilité de ce cache est l'application. Une fois activé, sa durée de vie est celle de l'instance de type SessionFactory. Le rôle principal du cache de second niveau est de partager des données entre sessions.

III.3.12. Utilisation d'Hibernate :

L'utilisation de Hibernate se fait principalement au travers de la classe Session qu'il fournit. Un objet session offre les fonctionnalités suivantes :

- ✓ Rendre persistant un objet d'une classe : C'est la méthode « save » qui offre cette fonctionnalité. Elle prend en paramètre l'objet à rendre persistant.
- ✓ Charger un objet d'une classe à partir de la base de données : La méthode « load » est utilisée à cette fin. Elle prend en paramètre la classe de l'objet à charger ainsi que la valeur de l'identifiant (clé primaire) de cet objet.
- ✓ Modification d'un objet persistant : Il suffit pour cela de modifier la valeur des propriétés d'un objet puis d'appeler la méthode « flush » de l'objet session.
- ✓ Suppression d'un objet persistant : L'appel de la méthode « delete » avec en paramètre un objet persistant se charge d'effectuer la suppression dans la base de données.
- ✓ Rechercher des objets : Hibernate propose un langage de requête orienté objets nommé HQL dont la syntaxe est similaire au SQL et qui permet d'effectuer des requêtes sur le modèle objet.

III.4. Conclusion :

Les applications d'entreprises s'orientent de plus en plus vers des architectures n-tiers. La technologie J2EE et les Frameworks qui se sont créés autour offrent beaucoup d'outils pour répondre aux besoins modernes.

Dans cette partie, nous nous sommes intéressé au Framework de mapping objet-relationnel le plus populaire pour J2EE appelé Hibernate qui est à l'heure actuelle parmi les plus puissants, les plus complets, et surtout les plus utilisé dans le monde professionnel.

Chapitre IV :

Analyse et conception

IV.1. Introduction :

Nous présentons dans ce chapitre « analyse et conception » notre travail qui consiste à développer une application web pour la gestion du parc informatique de l'ATM Mobilis.

Nous commencerons par présenter les fonctionnalités de notre projet. Ensuite nous consacrerons une large place pour la modélisation de notre application en orienté objet à l'aide du langage UML (Unified Modeling Languge) qui permet de bien représenter les aspects statiques et dynamiques de notre projet par la série des diagrammes qu'il offre. Nous terminons par la conception de la base de données utile à notre application en spécifiant les tables à utiliser et leurs structures.

IV.2. Analyse :

IV.2.1. Objectifs de notre application :

Avant de réaliser l'application, il est nécessaire de préciser les objectifs de l'entreprise.

Le principal objectif de notre application est de faciliter la tâche au gestionnaire du parc informatique et lui permettre un gain de temps énorme en lui délivrant des interfaces claires et faciles à utiliser. Pour ce faire, notre système devra :

1. Offrir au gestionnaire du parc une interface simple et conviviale pour mieux utiliser l'application.
2. Permettre la gestion des différents articles informatiques dont le stock du parc dispose.
3. Permettre au gestionnaire du parc de gérer les employés et les fournisseurs.
4. Permettre au gestionnaire d'éditer des différents documents sortants du parc:
    - Les décharges.
    - Les bons de sortie.
    - Les bons de commande interne matériels.
    - Les demandes de réparation.
    - Les bons de retour interne.
    - Les bons de retour externe.
    - la Fiche d'inventaire.
5. Permettre au gestionnaire du parc d'enregistrer différents documents entrants au parc :
    - Les commandes internes.
    - Les bons de sortie magasin.
    - Les décharges de réparation.

IV.2.2. Définitions de base :

- **Acteur :** un acteur représente un rôle joué par une personne ou une chose qui interagit avec un système. Les acteurs se déterminent en observant les utilisateurs directs du système, ceux qui sont responsables de son exploitation ou de sa maintenance, ainsi que les autres systèmes qui interagissent avec le système en question.
- **Tâche :** c'est l'ensemble des fonctions qu'un acteur bien spécifié peut effectuer.
- **Scénario :** c'est une succession particulière d'enchainement, s'exécutant du début à la fin du cas d'utilisation.

IV.2.3. Identification des acteurs :

- **Administrateur :** dans notre cas, c'est le gestionnaire du département « gestion du parc informatique ». Il possède les droits d'accès à l'espace administrateur et gère l'application.

IV.2.4. Spécification des tâches :

Pour notre acteur nous spécifierons les tâches qu'il assure. Le tableau suivant résume ces tâches :

Acteur	Tâches
Administrateur	**T01 :** S'authentifier.
	T02 : Accéder à l'espace administrateur.
	T03 : Gérer les articles.
	T04 : Gérer les employés.
	T05 : Gérer les fournisseurs.
	T06 : Edition des décharges.
	T07 : Edition des bons de sortie.
	T08 : Edition des bons de commande interne matériels.
	T09 : Edition des demandes de réparation.
	T10 : Edition des bons de retour interne.
	T11 : Edition des bons de retour externe.
	T12 : Edition de la Fiche d'inventaire.
	T13 : Enregistrer les commandes internes.
	T14 : Enregistrer les bons de sortie magasin.
	T15 : Enregistrer les décharges de réparation.
	T16 : Changer le profil.
	T17 : Se déconnecter.

IV.2.5. Spécification des scénarios :

Les scénarios décrivant chacune des tâches définies auparavant sont représentés dans le tableau suivant :

Acteur	Tâche	Scénario
Administrateur	**T01 :** S'authentifier.	**S01 : S**aisir le login et le mot de passe dans la page d'accueil. **S02 :** Cliquer sur le bouton « Connexion».
	T02 : Accéder à l'espace administrateur.	**S03 :** Saisir le login et le mot de passe dans la page d'authentification qui s'affiche. **S04 :** Valider la saisie.
	T03: Gérer les articles.	**S03 :** Sélectionner le lien « Gestion du stock ». **S04 :** Ajouter ou supprimer les articles disponibles en stock.
	T04 : Gérer les employés.	**S05 :** Sélectionner le lien « Gestion des employés ». **S06 :** Ajouter ou supprimer des employés.
	T05 : Gérer les fournisseurs.	**S07 :** Sélectionner le lien « Gestion des fournisseurs ». **S08 :** Ajouter ou supprimer des fournisseurs.
	T06 : Edition des décharges.	**S09 :** Cliquer sur le lien « Décharge». **S10 :** Remplir les champs avec les données correspondantes. **S11 :** Valider la saisie.
	T07 : Edition des bons de sortie.	**S12 :** Cliquer sur le lien « Bon de sortie». **S13 :** Remplir les champs avec les données correspondantes. **S14 :** Valider la saisie.
	T08 : Edition des bons de commande interne matériels.	**S15 :** Cliquer sur le lien « Commande interne matériel». **S16 :** Remplir les champs avec les données correspondantes. **S17 :** Valider la saisie.
	T09 : Edition des demandes de réparation.	**S18:** Cliquer sur le lien « Demande de réparation». **S19 :** Remplir les champs avec les données

		correspondantes. **S20 :** Valider la saisie.
	T10 : Edition des bons de retour interne.	**S21 :** Cliquer sur le lien « Bon de retour interne». **S22 :** Remplir les champs avec les données correspondantes. **S23 :** Valider la saisie.
	T11 : Edition des bons de retour externe.	**S24 :** Cliquer sur le lien « Bon de retour externe». **S25 :** Remplir les champs avec les données correspondantes. **S26 :** Valider la saisie.
	T12 : Edition de la Fiche d'inventaire.	**S27 :** Cliquer sur le lien « Fiche d'inventaire». **S28 :** Remplir les champs avec les données correspondantes. **S29 :** Valider la saisie.
	T13 : Enregistrer les commandes internes.	**S30:** Cliquer sur le lien « Commande interne». **S31 :** Remplir les champs avec les données correspondantes. **S32 :** Cliquer sur « Enregistrer ».
	T14 : Enregistrer les bons de sortie magasin.	**S33:** Cliquer sur le lien « Bon de sortie Magasin». **S34 :** Remplir les champs avec les données correspondantes. **S35 :** Cliquer sur « Enregistrer ».
	T15 : Enregistrer les décharges de réparation.	**S36:** Cliquer sur le lien « Décharge réparation». **S37 :** Remplir les champs avec les données correspondantes. **S38 :** Cliquer sur « Enregistrer ».
	T16 : Changer le profil.	**S39 :** S'authentifier. **S40 :** Sélectionner le lien « Changer le profil ». **S41 :** Choisir l'information à changer puis remplir le champ correspondant. **S42 :** Mettre à jour le changement.
	T17 : Se déconnecter.	**S43 :** Cliquer sur le lien « Déconnexion ».

IV.2.6. Spécification des cas d'utilisation :

IV.2.6.1. Définition d'un cas d'utilisation :

Les cas d'utilisation (en anglais use cases) constituent une technique qui permet de déterminer les besoins des utilisateurs et de capturer les exigences fonctionnelles d'un système. En d'autres termes, ils décrivent le comportement d'un système du point de vue de ses utilisateurs. Ils décrivent les interactions entre les utilisateurs d'un système lui même.

IV.2.6.2. Les cas d'utilisation détaillés :

Les figures qui suivent représentent une description de certains cas d'utilisation de notre système :

- ✓ Cas d'utilisation « s'authentifier » :

Utilisation : S'authentifier.
Scénario : S01, S02.
Acteur : Administrateur.
Résumé : Cette fonctionnalité permet à l'administrateur d'accéder à son espace personnel.
Description :

1. Accéder à l'application.
2. Le système affiche la page d'accueil.
3. L'administrateur saisie le login et le mot de passe puis valide en cliquant sur le bouton « Connexion».
4. Le système vérifie les données, les compare avec celles de la base de données puis affiche l'interface correspondante, ou renvoie un message d'erreur si le login et/ou le mot de passe ne sont pas valides.

Figure IV.1. Cas d'utilisation « S'authentifier ».

- ✓ Cas d'utilisation « Enregistrer les commandes internes » :

Utilisation : Enregistrer une commande interne concernant un élément en stock.
Scénario : S30, S31, S32.
Acteur : Administrateur.
Résumé : Cette fonctionnalité permet à l'administrateur d'archiver les commandes internes passées par des employés.
Description :

1. L'administrateur accède à son espace.
2. Le système affiche la page d'accueil.
3. L'utilisateur clique sur le lien « Commande interne ».
4. Le système affiche la page des commandes internes.
5. L'administrateur remplie les champs avec les données correspondantes.
6. Puis il clique sur « Enregistrer ».
7. Le système affiche une page de confirmation.

Figure IV.2. Cas d'utilisation « Enregistrer les commandes internes».

- ✓ Cas d'utilisation « Changer le profil » :

Utilisation : Changer le profil.
Scénario : S39, S40, S41, S42.
Acteur : Administrateur.
Résumé : Cette fonctionnalité permet à l'utilisateur de changer son profil.
Description :

1. Authentification.
2. L'utilisateur clique sur le lien « Changer le profil ».
3. Le système affiche la page correspondante.
4. L'utilisateur choisit les informations à changer et remplit les champs correspondants, puis valide.
5. Le système met à jour les données ou renvoi un message d'erreur si les données saisies ne sont pas correctes.

Figure IV.3. Cas d'utilisation « Changer le profil ».

- ✓ Cas d'utilisation « Edition des décharges » :

Utilisation : Edition des décharges.
Scénario : S09, S10, S11.
Acteur : Administrateur.
Résumé : Cette fonctionnalité permet à l'administrateur d'établir une décharge.
Description :

1. Authentification.
2. L'utilisateur clique sur le lien « Décharge ».
3. L'utilisateur remplie les champs avec les données correspondantes puis il valide.
4. Le système lui renvoie un message de confirmation d'enregistrement de la décharge.
5. L'utilisateur peut imprimer le document en cliquant sur le bouton correspondant.

Figure IV.4. Cas d'utilisation « Edition des décharges ».

- ✓ Cas d'utilisation « Gérer les employés » :

Utilisation : Gérer les employés.
Scénario : S05, S06.
Acteur : Administrateur.
Résumé : Cette fonctionnalité permet à l'administrateur d'ajouter ou de supprimer des employés.
Description :

1. Authentification.
2. Le système affiche l'espace de l'administrateur.
3. L'administrateur clique sur le lien « Gestion des employés».
4. Le système affiche la page correspondante.
5. L'administrateur ajoute ou supprime les employés puis valide dans les coins correspondants.
6. Le système enregistre le changement et lui affiche un message de confirmation.

Figure IV.5. Cas d'utilisation « Gérer les employés ».

IV.2.6.3. Diagramme de cas d'utilisation :

Les diagrammes de cas d'utilisation représentent un ensemble de cas d'utilisation, et d'acteurs et les différentes relations entre eux. Ils représentent la vue statique des cas d'utilisation d'un système.

IV.2.6.3.1. Diagramme du cas d'utilisation général :

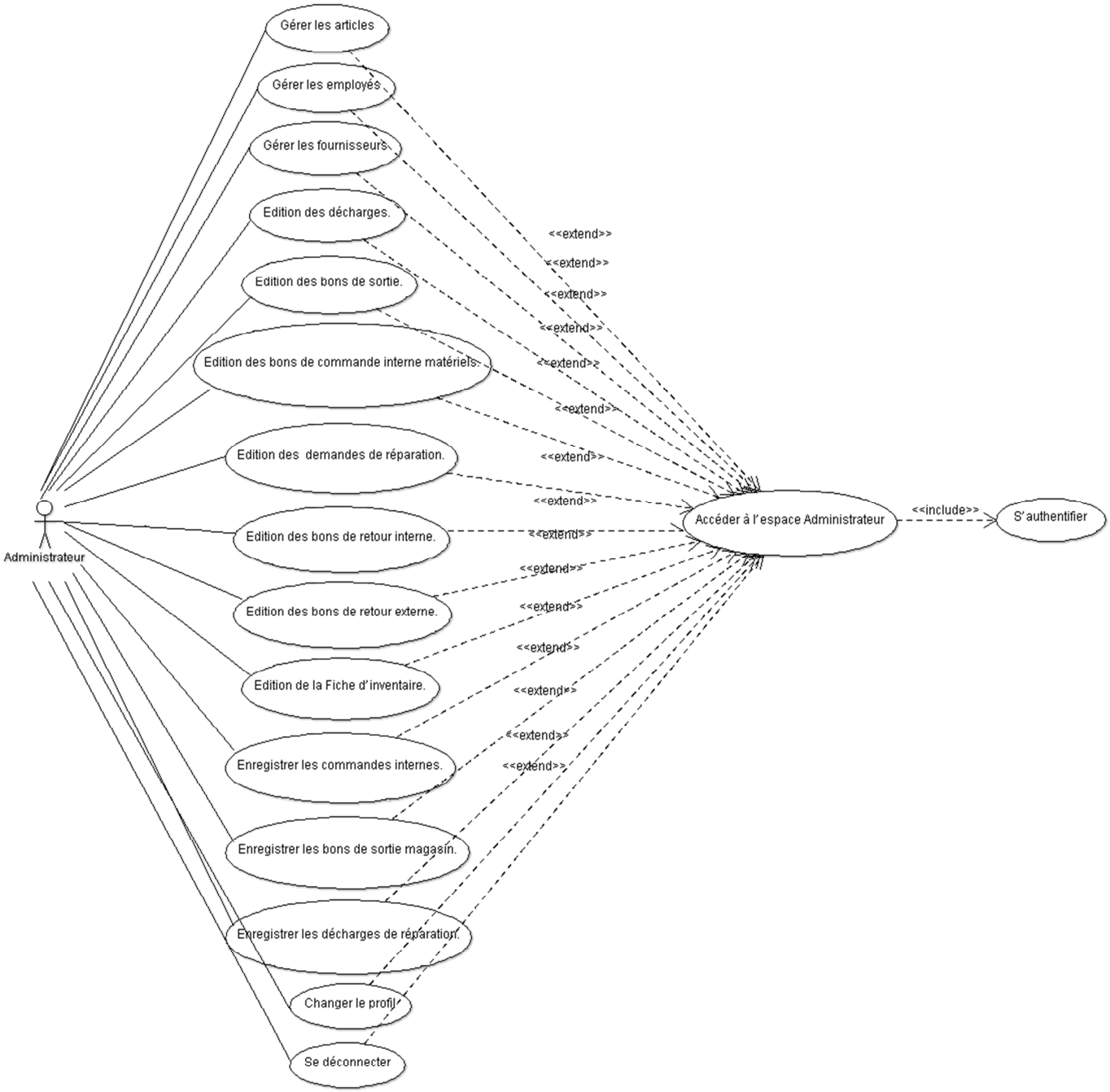


Figure IV.6. Diagramme du cas d'utilisation général.

IV.3. Conception :

Dans cette étape, une nouvelle vue du modèle fait son apparition. Cette vue exprime les modules et les exécutables physiques sans aller à la réalisation concrète du système.

Après avoir décrit textuellement les différents cas d'utilisation de notre système, nous allons les présenter formellement à l'aide des diagrammes de séquences.

IV.3.1. Diagramme de séquence :

Un diagramme de séquence fournit une représentation temporelle des interactions entre des objets (on parle aussi de rôles à la place d'objets). La formalisation des messages est une étape importante du passage à l'objet, puisqu'elle permet de spécifier les responsabilités et les interfaces des objets. C'est-à-dire ce que d'autres objets peuvent leur demander et quels types de réponse ils s'engagent à apporter aux demandes qu'on leur adresse.

IV.3.1.1. Diagramme de séquence du cas d'utilisation « S'authentifier » :

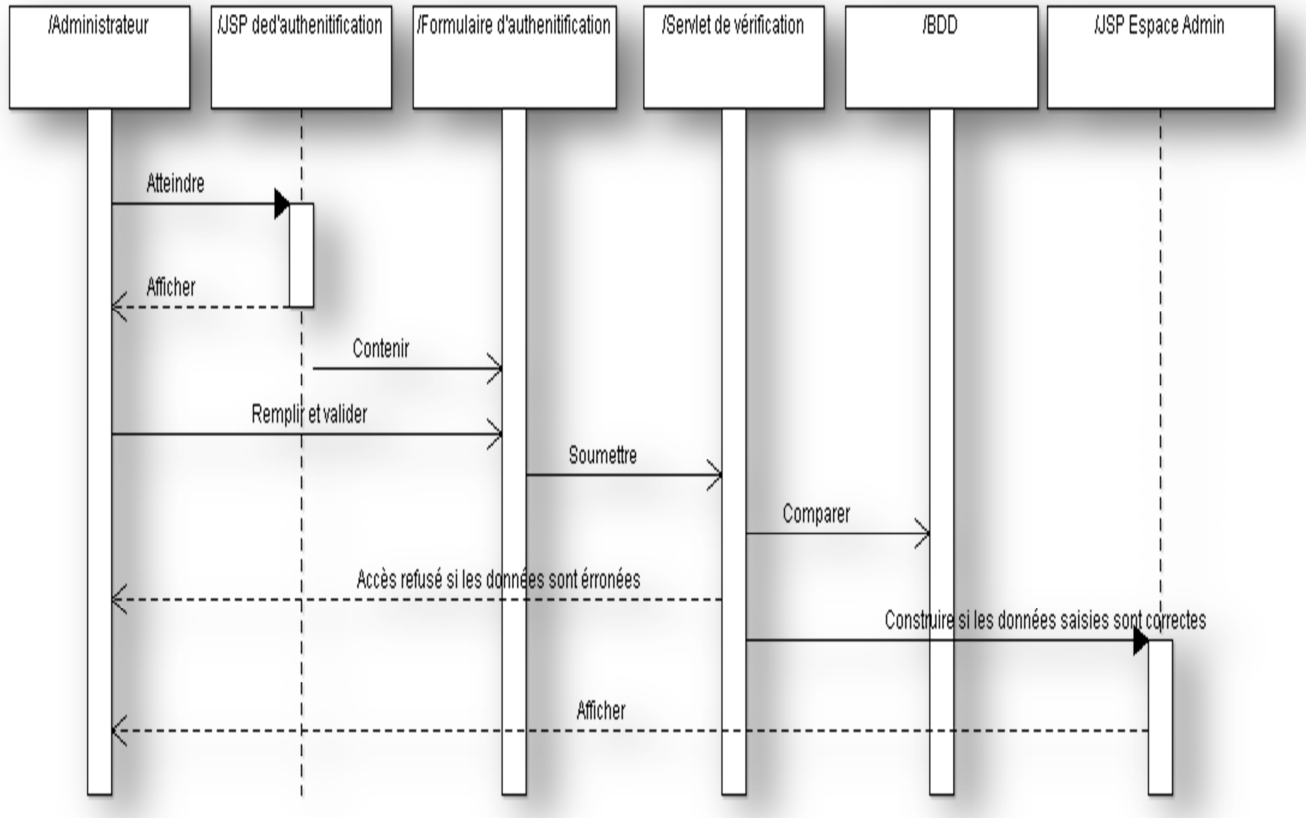


Figure IV.7. Diagramme de séquence du cas d'utilisation « S'authentifier ».

IV.3.1.2. Diagramme de séquence du cas d'utilisation « Changer le profil » :

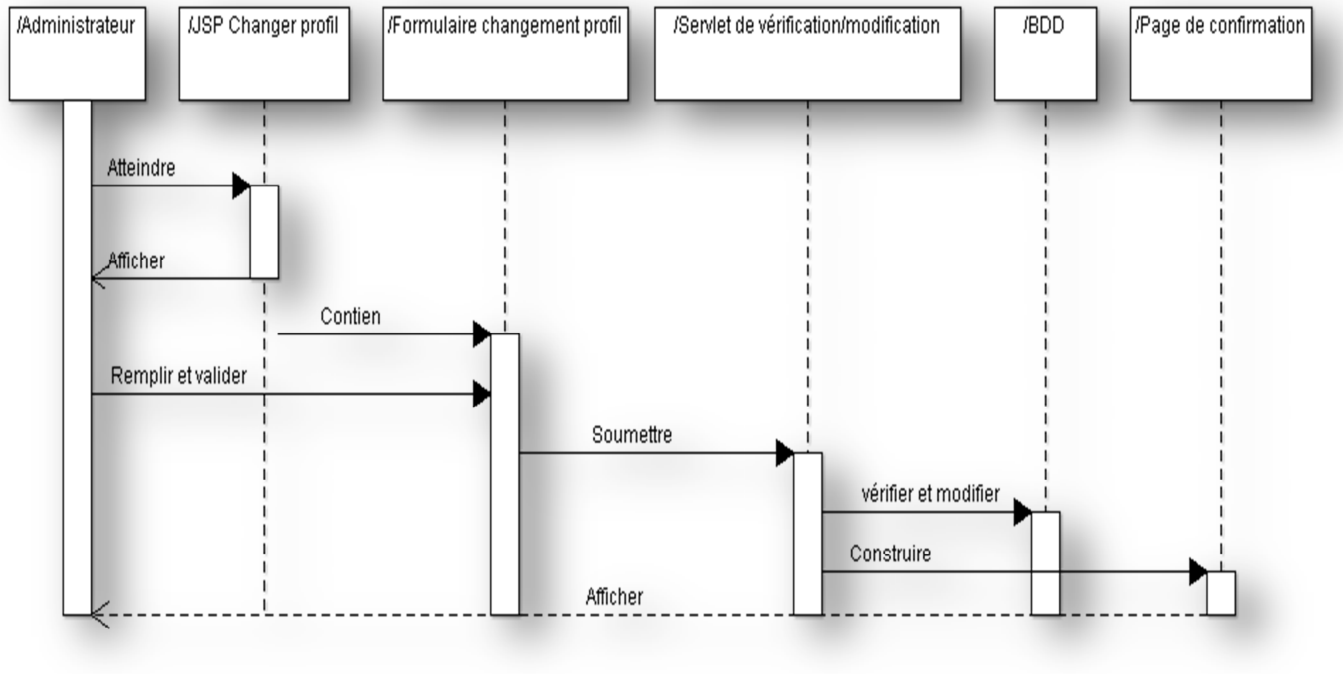


Figure IV.8. Diagramme de séquence du cas d'utilisation « Changer le profil ».

IV.3.1.3. Diagramme de séquence du cas d'utilisation « Edition des décharges» :

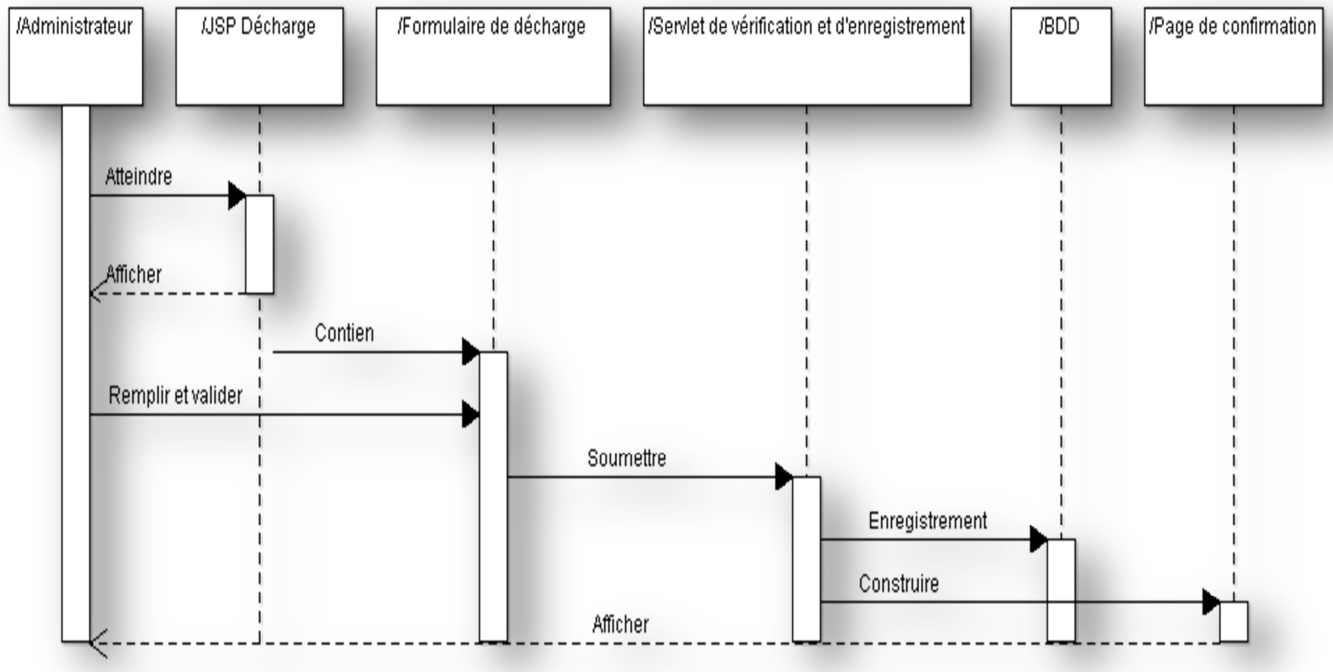


Figure IV.9. Diagramme de séquence du cas d'utilisation « Edition des décharges».

IV.3.1.4. Diagramme de séquence du cas d'utilisation « Gérer les employés» :

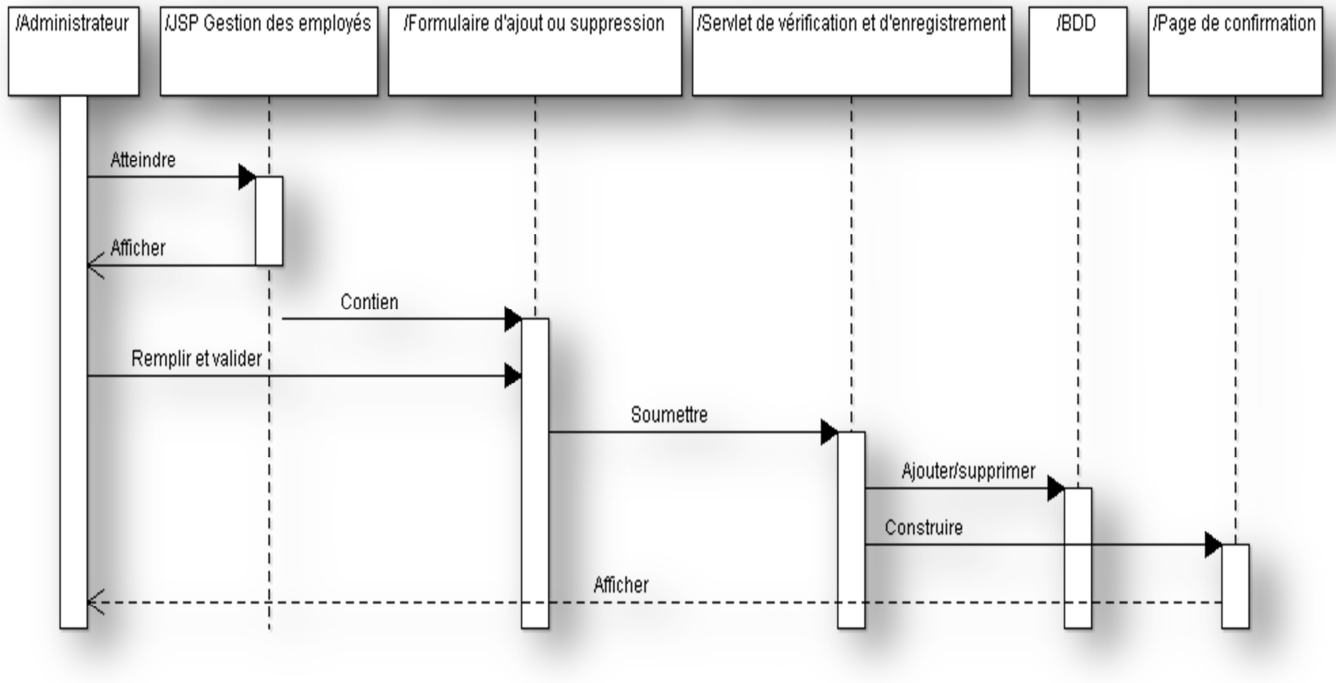


Figure IV.10. Diagramme de séquence du cas d'utilisation « Gérer les employés».

IV.3.2. Les diagrammes de classes :

Les diagrammes de classes sont sans doute les diagrammes les plus utilisés d'UML. Ils décrivent les types des objets qui composent un système et les différents types de relations statiques qui existent entre eux. Ils font abstraction du comportement du système.

Noun nous contentons de présenter quelques exemples seulement de diagrammes de classes.

IV.3.2.1. Diagramme de classes du cas d'utilisation « Edition des décharges» :

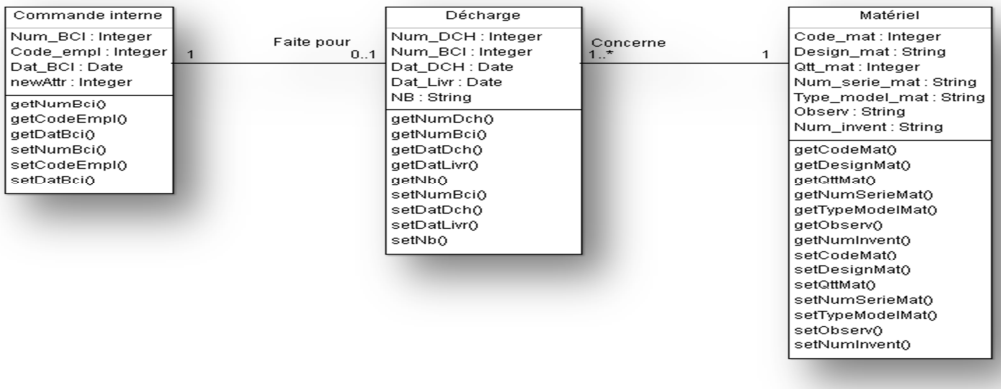


Figure IV.11. Diagramme de classes du cas d'utilisation « Edition des décharges».

IV.3.2.2. Diagramme de classes global :

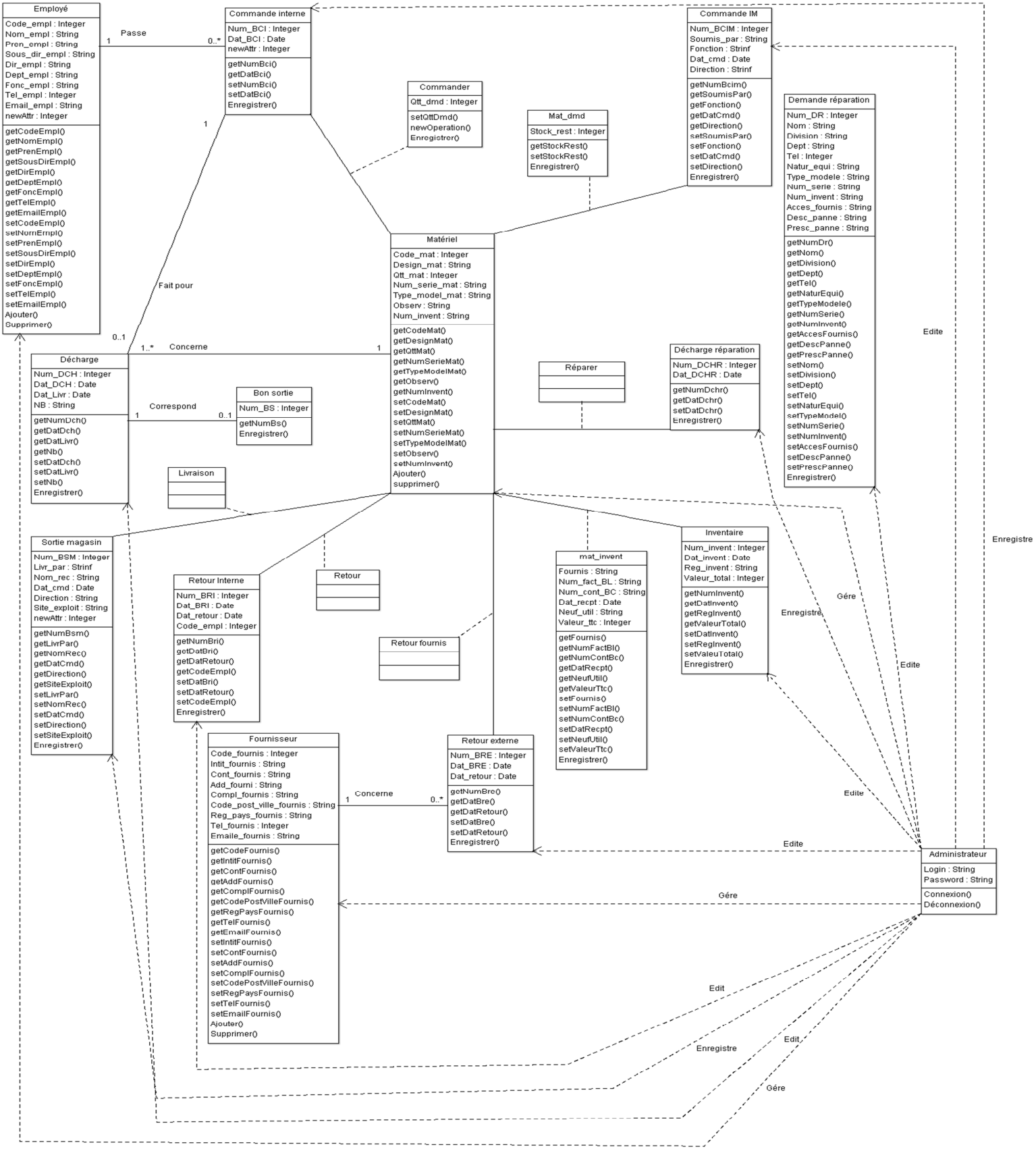


Figure IV.12. Diagramme de classes global.

IV.3.3. Conception de la base de données :

Après avoir modélisé notre application Web avec les différents diagrammes offerts par le langage de modélisation UML, une mise en œuvre d'une base de données sur un serveur Web est nécessaire car elle permet d'étendre les possibilités d'interactions avec l'utilisateur et de mettre des données à la disposition de ce dernier pour une consultation, une saisie ou une mise à jour tout en s'assurant des droits qui lui sont accordés.

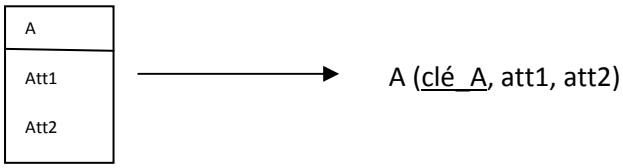
Pour l'implémentation de la base de données, on aura besoin d'élaborer un modèle relationnel de données.

➢ Règle de passage du diagramme de classes vers le modèle relationnel :

Un schéma relationnel est un ensemble de relations. Ce schéma peut être obtenu à partir d'un schéma conceptuel tel qu'un diagramme de classes.

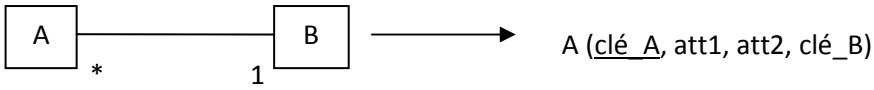
Pour cela, il faut :

- Traduire chaque classe par une relation à laquelle on ajoutera une clé.



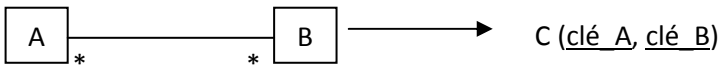
- Traduire les associations :

_Celles qui ont un bout avec une multiplicité maximum inférieure ou égale à 1 :



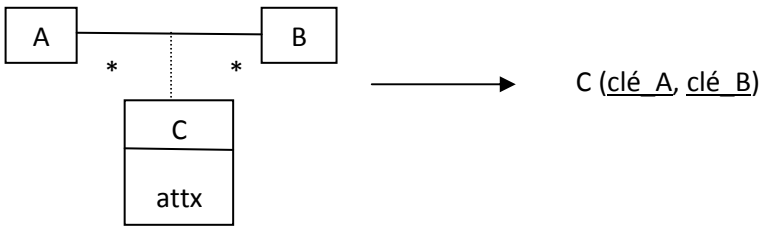
Peuvent être traduites en migrant la clé de B vers A en clé étrangère.

_Celles qui ont les bouts avec une multiplicité maximum * :



Nous ajoutons une relation C ayant une clé composée des clés de A et B.

_Classes associatives :



Nous ajoutons une relation C ayant une clé composée des clés de A et B, et pour attributs ceux de la classe associative.

- La traduction de la relation d'héritage se fait en propageant la clé de la classe mère vers les classes filles.

- **Les relations obtenues :**
  - Administrateur (Login, Password) ;
  - Bon_sortie (Num_BS, Num_DCH*) ;
  - Commande_im (Num_BCIM, Soumis_par, Fonction, Dat_comd, Direction) ;
  - Commande_interne (Num_BCI, Dat_BCI Code_empl*) ;
  - Commander (Num_BCI*, Code_mat*, Qtt_dmd) ;
  - Décharge (Num_DCH, Num_BCI*, Dat_DCH, Dat_livr, NB) ;
  - Décharge_réparation (Num_DCHR, Dat_DCHR) ;
  - Demande_réparation (Num_DR, Nom, Division, Direction, Dept, Tel, Natur_equi, Type_modele, Num_serie, Num_invent, Acces_fournis, Desc_panne, Presc_panne) ;
  - Employé (Code_empl, Nom_empl, Pren_empl, Sous_dir_empl, Dept_empl, Fonc_empl, Tel_empl, Email_empl) ;
  - Fournisseur (Code_fournis, Intit_fournis, Cont_fournis, Add_fournis, Compl_fournis, Code_post_ville_fournis, Reg_pays_fournis, Tel_fournis, Email_fournis) ;
  - Inventaire (Num_invent, Dat_invent, Reg_invent, Valeur_total) ;
  - Livraison (Num_BSM*, Code_mat*) ;
  - Mat_cmd (Num_BCIM*, Code_mat*, Stock_rest) ;
  - Mat_invent (Num_invent*, Code_mat*, Fournis, Num_fact_BL, Num_contr_BC, Dat_recpt, Neuf_util, Val_ttc) ;
  - Matériel (Code_mat, Design_mat, Qtt_mat, Num_serie_mat, Type_model_mat, Observ, Num_invent) ;
  - Réparer (Num_DCHR *, Code_mat*) ;
  - Retour (Num_BRI*, Code_mat*) ;
  - Retour_externe (Num_BRE, Dat_BRE, Dat_retour, Code_fournis*) ;
  - Retour_fournis (Num_BRE*, Code_mat*) ;
  - Retour_interne (Num_BRI, Dat_BRI, Dat_retour, Code_empl*) ;
  - Sortie_magasin (Num_BSM, Livr_par, Nom_rec, Dat_cmd, Directin, Site_exploit) ;

➢ Les tables de la base de données :

✓ Table Administrateur :

Nom du Champs	Type	Taille	Description	Clé
Login	Varchar	30	Login de l'administrateur	Primaire
Password	Varchar	30	Mot de passe de l'administrateur	Primaire

✓ Table Employé :

Nom du champ	Type	Taille	Description	Clé
Code_empl	Int	12	Code de l'employé	Primaire
Nom_empl	Varchar	30	Nom de l'employé	
Pren_empl	Varchar	30	Prénom de l'employé	
Sous_dir_empl	Varchar	30	Sous direction de l'employé	
Dir_empl	Varchar	30	Direction de l'employé	
Dept_empl	Varchar	30	Département de l'employé	
Fonc_empl	Varchar	20	Fonction de l'employé	
Tel_empl	Int	12	Téléphone de l'employé	
Email_empl	Varchar	40	Email de l'employé	

✓ Table Fournisseur :

Nom du champ	Type	Taille	Description	Clé
Code_fournis	Int	02	Code du fournisseur	Primaire
Intit_fournis	Varchar	30	Intitulé du fournisseur	
Cont_fournis	Varchar	30	Contact du fournisseur	
Add_fournis	Varchar	40	Adresse du fournisseur	
Compl_fournis	Varchar	40	Complément fournisseur	
Code_post_ville_fournis	Varchar	40	Code postal/ Ville fournisseur	
Reg_pays_fournis	Varchar	40	Région/ Pays du fournisseur	
Tel_fournis	Int	12	Numéro téléphone du fournisseur	
Email_fournis	Varchar	40	Email fournisseurs	

✓ Table Matériel :

Nom du champ	Type	Taille	Description	Clé
Code_mat	Int	11	Code matériel	Primaire
Design_mat	Varchar	30	Désignation du matériel	
Qtt_mat	Int	04	Quantité matériel	
Num_serie_mat	Varchar	30	Numéro de série du matériel	
Type_model_mat	Varchar	25	Type de modèle matériel	
Observ	Varchar	30	Observation	
Num_invent	Varchar	11	Numéro d'inventaire	

✓ Table Commande_interne :

Nom du champ	Type	Taille	Description	Clé
Num_BCI	Int	06	Numéro du bon de commande interne	Primaire
Dat_BCI	Date	10	Date du bon de commande interne	
Code_empl	Int	12	Code de l'employé	Etrangère

- ✓ **Table Sortie_magasin :**

Nom du champ	Type	Taille	Description	Clé
Num_BSM	Int	06	Numéro du bon de sortie magasin	Primaire
Livr_par	Varchar	30	Livraison faite par	
Nom_rec	Varchar	30	Nom recevant	
Dat_cmd	Date	10	Date de la commande	
Direction	Varchar	30	Direction de l'intéressé	
Site_exploit	Varchar	30	Site d'exploitation	

- ✓ **Table Décharge_reparation :**

Nom du champ	Type	Taille	Description	Clé
Num_DCHR	Int	06	Numéro de la décharge de réparation	Primaire
Dat_DCHR	Date	10	Date de la décharge de réparation	

- ✓ **Table Décharge:**

Nom du champ	Type	Taille	Description	Clé
Num_DCH	Int	06	Numéro de la décharge	Primaire
Num_BCI	Int	06	Numéro du bon de retour interne	
Dat_DCH	Date	10	Date de la décharge	
Dat_livr	Date	10	Date de livraison	
NB	Varchar	40	NB	

- ✓ **Table Bon_sortie :**

Nom du champ	Type	Taille	Description	Clé
Num_BS	Int	06	Numéro du Bon de Sortie	Primaire
Num_DCH	Int	06	Numéro de Décharge	Etrangère

- ✓ **Table Commande_im :**

Nom du champ	Type	Taille	Description	Clé
Num_BCIM	Int	06	Numéro de Bon de Commande Interne Matériel	Primaire
Soumis_par	Varchar	30	Soumission	
Fonction	Varchar	20	Fonction de l'intéressé	
Dat_cmd	Date	10	Date de la commande	
Direction	Varchar	30	Direction de l'intéressé	

- ✓ **Table Commander :**

Nom du champ	Type	Taille	Description	Clé
Num_BCI	Int	06	Numéro de Bon de Commande Interne	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère
Qtt_dmd	Int	04	Quantité du matériel demandée	

- ✓ **Table Inventaire :**

Nom du champ	Type	Taille	Description	Clé
Num_invent	Int	06	Numéro de l'inventaire	Primaire
Dat_invent	Date	10	Date de l'inventaire	
Reg_invent	Varchar	30	Région de l'inventaire	
Valeur_total	Int	20	Valeur totale des articles	

- ✓ **Table Demande_réparation :**

Nom du champ	Type	Taille	Description	Clé
Num_DR	Int	06	Numéro de Décharge de Réparation	Primaire
Nom	Varchar	30	Nom de demandeur	
Division	Varchar	30	Division du demandeur	
Direction	Varchar	30	Direction du demandeur	
Dept	Varchar	30	Département du demandeur	
Tel	Int	12	Téléphone du demandeur	
Natur_equi	Varchar	30	Nature de l'équipement	
Type_modele	Varchar	25	Type/modèle de l'équipement	
Num_serie	Varchar	30	Numéro de série équipement	
Num_invent	Varchar	8	Numéro de l'inventaire	
Acces_fournis	Varchar	25	Accessoires fournis	
Desc_panne	Varchar	40	Description de la panne	
Presc_panne	Varchar	40	Précision de la panne	

- ✓ **Table Livraison :**

Nom du champ	Type	Taille	Description	Clé
Num_BSM	Int	06	Numéro du Bon de Sortie Magasin	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère

- ✓ **Table Mat_cmd :**

Nom du champ	Type	Taille	Description	Clé
Num_BCIM	Int	06	Numéro du Bon de Commande Interne Matériel	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère
Stock_rest	Int	04	Stock restant	

- ✓ **Table Réparer :**

Nom du champ	Type	Taille	Description	Clé
Num_DCHR	Int	06	Numéro de Décharge de Réparation	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère

- ✓ **Table Mat_invent :**

Nom du champ	Type	Taille	Description	Clé
Num_invent	Int	06	Numéro de l'inventaire	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère
Fournis	Varchar	30	Fournisseur	
Num_fact_BL	Varchar	08	Numéro de facture ou Bon de Livraison	
Num_cont_BC	Varchar	08	Numéro de contrat ou Bon de Commande	
Dat_recpt	Date	10	Date de réception	
Neuf_util	Varchar	06	Neuf ou utilisé	
Valeut_ttc	Int	08	Valeur TTC de l'article	

- ✓ **Table Retour :**

Nom du champ	Type	Taille	Description	Clé
Num_BRI	Int	06	Numéro du Bon de Retour Interne	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère

- ✓ **Table Retour_externe :**

Nom du champ	Type	Taille	Description	Clé
Num_BRE	Int	06	Numéro du Bon de Retour Externe	Primaire
Dat_BRE	Date	10	Date du Bon de Retour Externe	
Dat_retour	Date	10	Date du retour	
Code_fournis	Int	02	Code du fournisseur	Etrangère

- ✓ **Table Retour_interne :**

Nom du champ	Type	Taille	Description	Clé
Num_BRI	Int	06	Numéro du Bon de Retour Interne	Primaire
Dat_BRI	Date	10	Date du Bon de Retour Interne	
Dat_retour	Date	10	Date du retour	
Code_empl	Int	12	Code de l'employé	Etrangère

- ✓ **Table Retour_fournis :**

Nom du champ	Type	Taille	Description	Clé
Nom_BRE	Int	06	Numéro du Bon de Retour Externe	Etrangère
Code_mat	Int	11	Code du matériel	Etrangère

IV.4. Conclusion :

Dans ce chapitre, nous nous sommes concentrés sur les aspects analytique et conceptuel de notre application ainsi que les bases de données qui interagissent avec elle.

Pour la phase analyse, nous avons définis les différents cas d'utilisation puis, nous les avons traduits à travers la construction des diagrammes de séquences et de classes, dans la phase de conception.

Enfin, nous avons définis les tables de la base de données utilisées.

Le chapitre suivant sera consacré à l'implémentation de l'application.

Chapitre V :

Implémentation et mise en œuvre

V.1. Introduction :

Ce chapitre traite les aspects techniques liés à l'implémentation et la mise en œuvre de notre application. Nous allons présenter d'abord nos choix des technologies et les outils adoptés qui nous ont servi d'appui pour la développer. Par la suite, nous présenterons ses principales interfaces et les fonctionnalités qu'elles regroupent.

V.2. L'environnement de développement:

V.2.1. La plateforme J2EE :

L'élaboration de notre application s'appuie sur la plateforme J2EE (Java 2 Enterprise Edition) qui est une norme proposée par la société SUN, portée par un consortium de sociétés internationales, visant à définir un standard de développement d'applications d'entreprises multi-niveaux, basées sur des composants.

On parle généralement de la « plate-forme J2EE » pour désigner l'ensemble constitué des services (API *Application Programming Interface*) offerts et de l'infrastructure d'exécution. J2EE comprend notamment:

- Les spécifications du serveur d'application, c'est-à-dire de l'environnement d'exécution : J2EE définit finement les rôles et les interfaces pour les applications ainsi que l'environnement dans lequel elles seront exécutées. Ces recommandations permettent ainsi à des entreprises tierces de développer des serveurs d'application conformes aux spécifications ainsi définies, sans avoir à redévelopper les principaux services.
- Des services, au travers d'API, c'est-à-dire des extensions Java indépendantes permettant d'offrir en standard un certain nombre de fonctionnalités. SUN fournit une implémentation minimale de ces API appelée J2EE SDK (J2EE Software Development Kit).
- Dans la mesure où JEE s'appuie entièrement sur le langage Java, il bénéficie des avantages de ce langage, en particulier une bonne portabilité et une maintenabilité du code

Ce choix est justifié par plusieurs facteurs à savoir :

- ✓ La maturité et la richesse de cette technologie.
- ✓ La possibilité de la réutilisation des différents composants qui en font partie.
- ✓ La séparation forte qu'offre la plupart des Frameworks relevant de cette architecture.

❖ **Le Framework:** C'est un ensemble de composants qui servent à créer l'architecture et les grandes lignes d'une application. On peut le voir comme une boîte à outils géante, conçue par

un ou plusieurs développeurs et mise à disposition d'autres développeurs, afin de faciliter leur travail. Il existe des Frameworks dans beaucoup de langages et plateformes, ce n'est pas un concept propre à Java EE ni au développement web en particulier.

En ce qui concerne Java EE, nous pouvons par exemple citer JSF, Spring, Strusts ou encore Hibernate. Toutes ces solutions sont des Frameworks que les développeurs sont libres d'utiliser ou non dans leurs projets.

V.2.2. L'architecture MVC:

Un modèle de conception, en anglais design pattern (ou encore patron de conception) est une simple bonne pratique, qui répond à un problème de conception d'une application. C'est en quelque sorte une ligne de conduite qui permet de décrire les grandes lignes d'une solution. De tels modèles sont issus de l'expérience des concepteurs et développeurs d'applications : c'est en effet uniquement après une certaine période d'utilisation que peuvent être mises en évidence des pratiques plus efficaces que d'autres, qui sont alors structurées en modèles et considérées comme standard.

Java EE permet plus ou moins de coder son application comme on le souhaite, en d'autres termes, on peut coder n'importe comment, et dans Java EE, il y'a « Entreprise », et le développement en entreprise implique entre autres :

- Que l'on puisse être amené à travailler à plusieurs contributeurs sur un même projet ou une même application (travail en équipe).
- Que l'on puisse être amené à faire évoluer une application que l'on n'a pas créée soi-même.
- Que l'on puisse être amené à maintenir et corriger une application que l'on n'a pas créée soi-même.

Pour toutes ces raisons, il est nécessaire d'adopter une architecture plus ou moins standard, que tout développeur peut reconnaître, c'est-à-dire dans laquelle tout développeur sait se repérer.

Il a été très vite remarqué qu'un modèle permettait de répondre à ces besoins, et qu'il s'appliquait particulièrement bien à la conception d'application Java EE : le modèle MVC (Modèle – Vue – Contrôleur). Il découpe littéralement l'application en couches distinctes, et de ce fait impacte très fortement l'organisation du code. Voici ce qu'impose MVC :

- ✓ Tout ce qui concerne le traitement, le stockage et la mise à jour des données de l'application doit être contenu dans la couche nommée « Modèle » (le M de MVC).
- ✓ Tout ce qui concerne l'interaction avec l'utilisateur et la présentation des données (mise en forme, affichage) doit être contenu dans la couche nommée « Vue » (le Vde MVC).
- ✓ Tout ce qui concerne le contrôle des actions de l'utilisateur et des données doit être contenu dans couche nommée « Contrôle » (le C de MVC).

Ce modèle peut être représenté par la figure suivante :

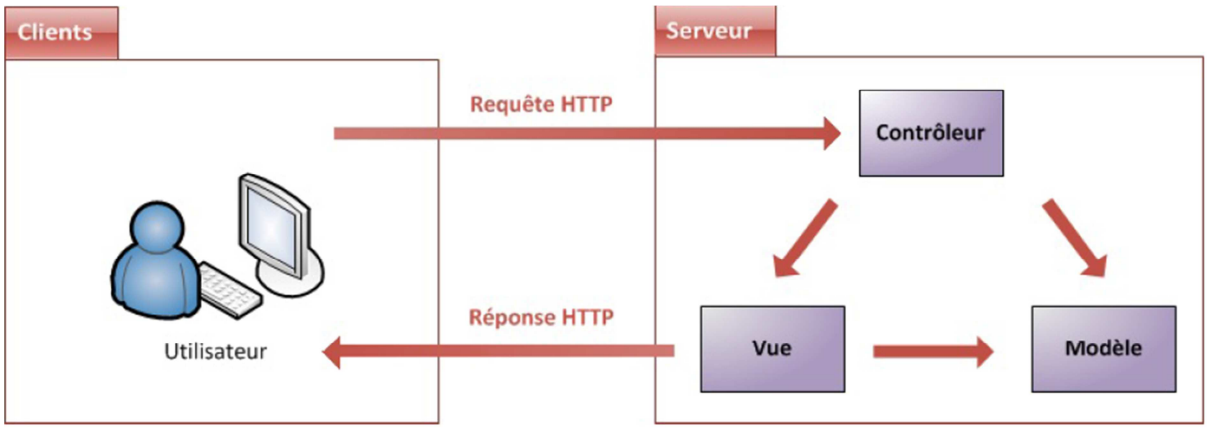


Figure V.1. MVC.

V.2.2.1. Modèle : des traitements et des données :

Dans le modèle, on trouve à la fois les données et les traitements à appliquer à ces données. Ce bloc contient donc des objets Java d'une part, qui peuvent contenir des attributs (données) et des méthodes (traitements) qui leur sont propres, et un système capable de stocker des données d'autre part. La complexité du code dépendra bien évidemment de la complexité des traitements à effectuer par l'application.

V.2.2.2. Vue : des pages JSP (Java Server Page):

Une page JSP est destinée à la vue. Elle est exécutée côté serveur et permet la génération de pages web dynamiques. Cette technologie permet de séparer la présentation sous forme de code HTML (CSS, JavaScript, XML, etc.) et les traitements sous formes de classes Java définissant un Bean ou une servlet. Ceci est d'autant plus facile que les JSP définissent une syntaxe particulière permettant d'appeler un Bean et d'insérer le résultat de don traitement dan la page HTML dynamiquement.

V.2.2.3. Contrôleur : des servlets :

Une servlet est un objet qui permet d'intercepter les requêtes faites par un client, et qui peut personnaliser une réponse en conséquence. Il fournit pour cela des méthodes permettant de scruter une requête HTTP. Cet objet n'agit jamais directement sur les données, il faut le voir comme un simple aiguilleur ; il intercepte une requête issue d'un client, appelle éventuellement des traitements effectués par le modèle, et ordonne en retour à la vue d'afficher le résultat au client.

V.2.3. Outils et environnement de développement:

V.2.3.1. Environnement de développement intégré : Eclipse:

Eclipse est un environnement de développement intégré libre extensible, universel et polyvalent, permettant de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

La spécificité d'Eclipse IDE vient du fait de son architecture totalement développée autour de la notion de plugin (en conformité avec la norme OSGI) : toutes les fonctionnalités de cet atelier logiciel sont développées en tant que plug-in.

Eclipse massivement utilisé en entreprise, est un outil puissant, gratuit, libre et multiplateforme. Les avantages d'un IDE dans le développement d'applications web Java EE sont multiples, et en voici quelque uns :

- Intégration des outils nécessaires au développement et au déploiement d'une application.
- Paramétrage aisé et centralisé des composants d'une application.
- Multiples moyens de visualisation de l'architecture d'une application.
- Génération automatique de portions de code.
- Assistance à la volée lors de l'écriture du code.
- Outils de débogage…

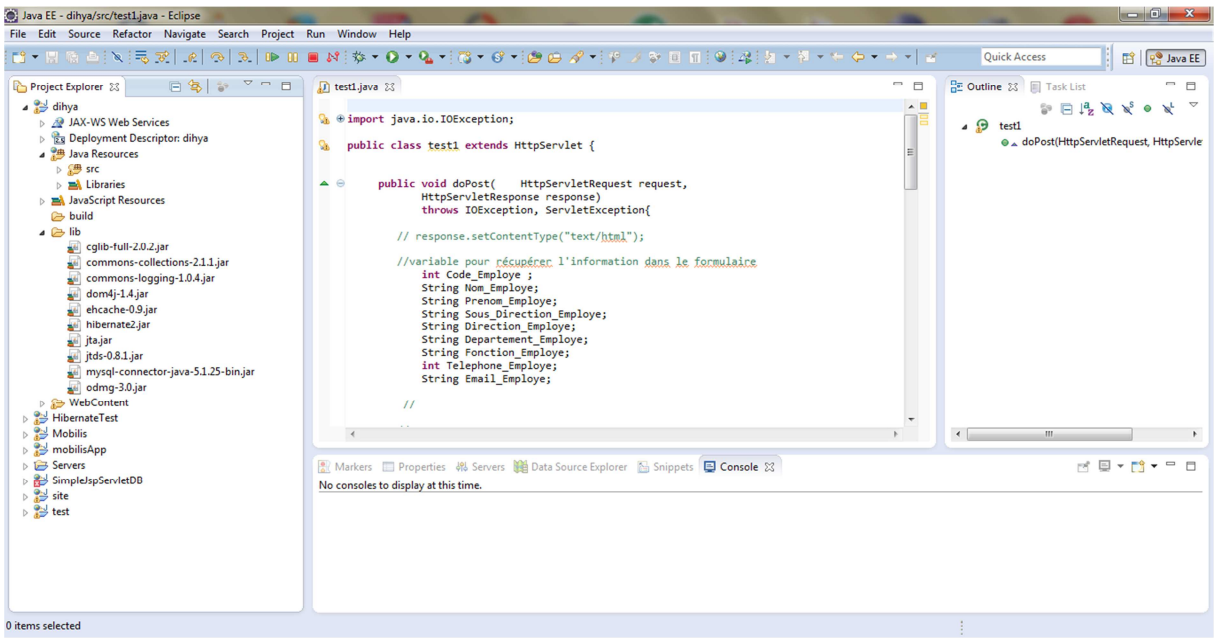


Figure V.2.Interface de l'IDE Eclipse.

V.2.3.2. Le serveur Tomcat:

Pour faire fonctionner une application web Java EE, nous avons besoin de mettre en place un serveur d'application. Il en existe beaucoup sur le marché, on a choisi d'utiliser Tomcat, car c'est un serveur léger, gratuit, libre, multiplateforme et assez complet pour ce que nous allons développer. On le rencontre d'ailleurs très souvent dans des projets en entreprise, en phase de développement comme en production.

V.2.3.3. Système de gestion de bases de données MySQL:

Le SGBD MySQL, avec sa configuration client/serveur, est l'un des plus connu dans le monde du web et du logiciel libre. C'est un véritable serveur de base de données SQL (Structured Query Language) qui est un langage de requêtes vers les bases de données exploitant le modèle relationnel.

MySQL est caractérisé par :

- ✓ Une implémentation libre et populaire.
- ✓ Facile à mettre en œuvre.
- ✓ Rapide à apprendre, robuste et convivial.
- ✓ Support multiplateforme.
- ✓ Fiable et rapide.

Afin de faciliter la manipulation de MySQL, nous avons utilisé MySQL Workbench (anciennement MySQL Administrator), qui est un outil de gestion et d'administration de BDD MySQL créé en 2004. Via une interface graphique intuitive, il permet entre autres la modélisation des données, le développement SQL et des outils d'administration complets pour la configuration des serveurs, l'administration des utilisateurs et davantage. MySQL Workbench est disponible sous Windows, Linux et Mac OS.

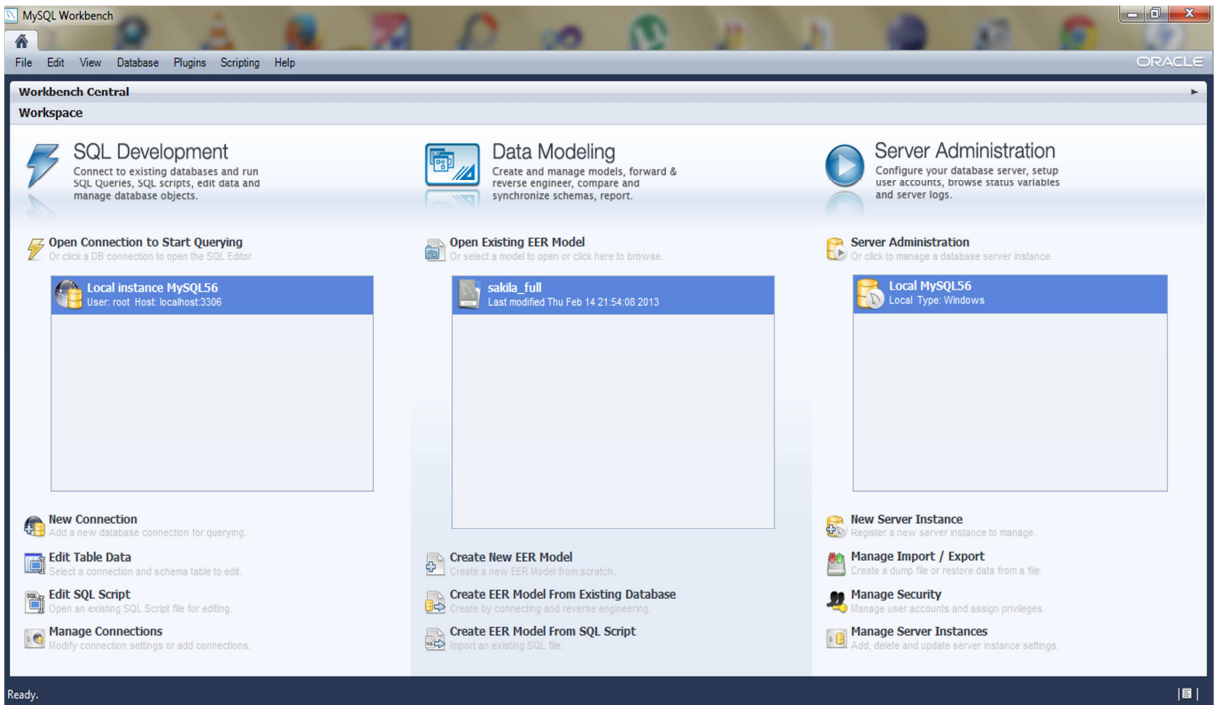


Figure V.3.Interface de MySQL Workbench.

Pour afficher le contenu de la base de données de notre projet, il faut cliquer sur le lien « Edit Table Data », et après la saisie du nm de l'utilisateur et le mot de passe il affichera toutes les bases de données, et en cliquant sur notre base de données « mobilis», toutes ses tables seront affichées.

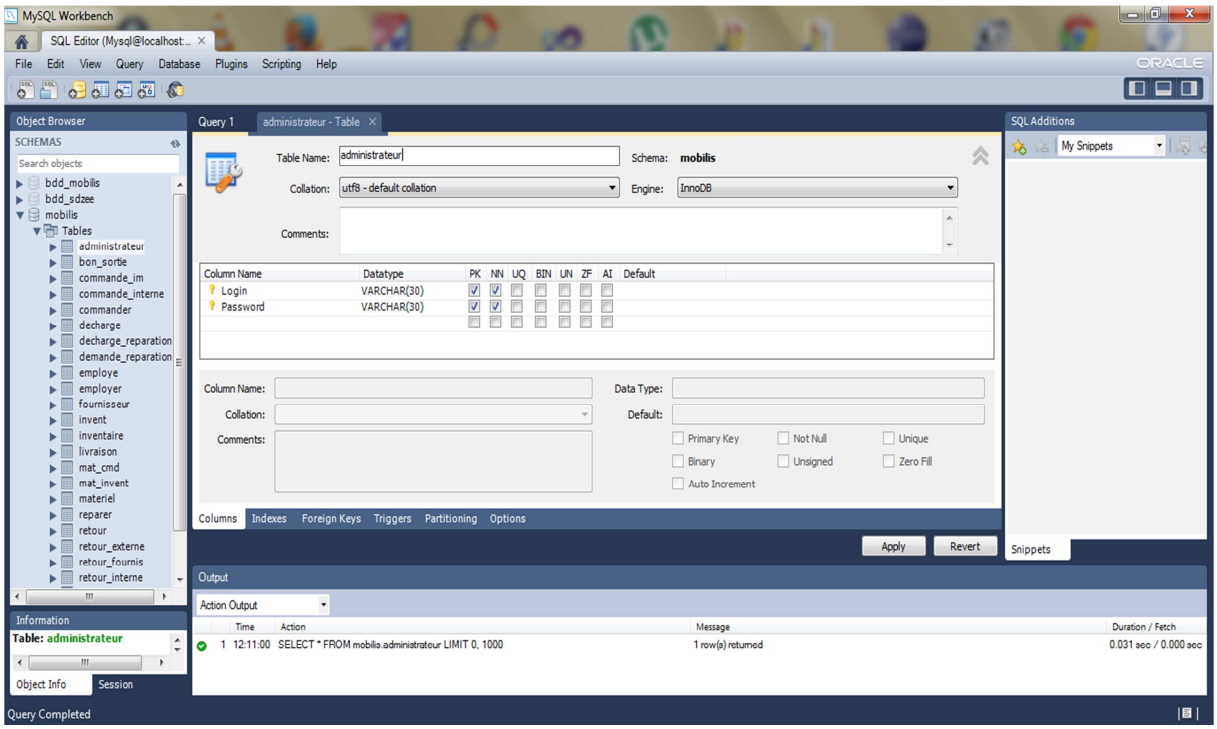


Figure V.4. Notre base de données « mobilis ».

V.2.3.4. Le Framework Hibernate:

Hibernate est une couche résidant dans la JVM permettant d'assurer le mapping des objets Java cachés dans la JVM au modèle relationnel ou modèle de données. Il assure aussi le transfert des classes Java dans les entités de données et donc des données des objets dans les entités et les tables. Hibernate présente aussi un langage de manipulation des objets mappés connus sous le nom HQL. Il s'agit de faire des Select, Update et Delete avec des opérations de somme, comptage (count), de calcul de moyenne, etc.

V.2.4. Langages de programmation :

Pour satisfaire les besoins de l'application à réaliser, nous avons utilisé les langages suivants :

- Le langage orienté objet Java pour la création des différentes classes Java.
- Des scripts Java pour les scriptlets dans le code HTML.
- HQL (Hibernate Query Language) pour l'élaboration des requêtes d'interrogation de la base de données.
- Le langage HTML (Hyper Text Mark-up Language) pour la création des JSP qui constitueront les interfaces de l'application.
- Le langage de requêtes MySQL pour la création de la base de données.
- CSS (Caqcading Style Sheet) pour la mise en forme des JSP et alléger le code HTML.
- XML pour la création de fichiers mapping relationnel-objet et pour la déclaration des servlets.

V.3. Présentation des interfaces de l'application :

L'objectif de cette partie du chapitre est de donner un aperçu de l'interface homme/machine de l'application.

- **La page d'authentification :** La page d'accès à l'application demande à l'administrateur de saisir son Login et son mot de passe. Si les informations entrées sont correctes, la connexion va se produire et l'administrateur va accéder directement à son espace Admin, sinon, il va être redirigé vers une page d'erreur lui demandant de vérifier les informations introduites.

Figure V.5. Page d'authentification.

- **La page d'accueil :** C'est la première page visualisé par l'administrateur après son authentification. Elle offre des liens vers les différentes pages que comporte l'application.

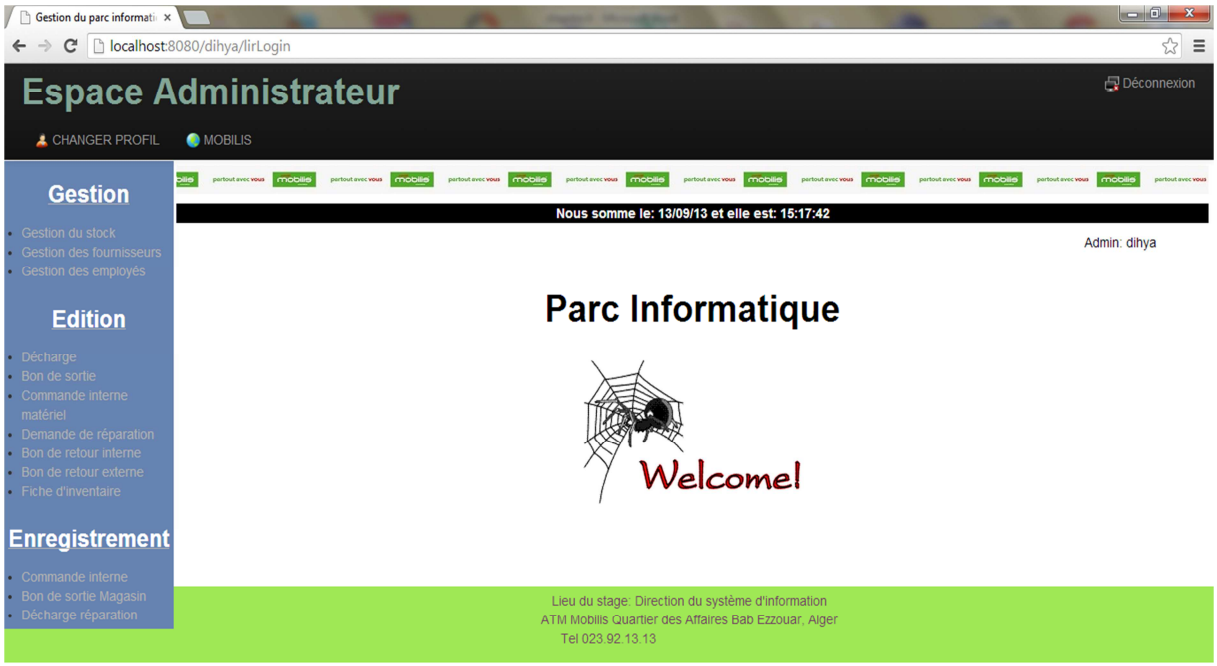


Figure V.6. Page d'accueil.

- **La page « Changer le profil » :** L'administrateur accède à cette page en cliquant sue le lien 'CHANGER PROFIL'. Elle contient deux formulaires, un pour changer le Login, et l'autre pour changer le mot de passe. L'admin, choisi ce qu'il veut changer et rempli le formulaire adéquat, puis valide le changement. Le système va lui affiché une page de confirmation.

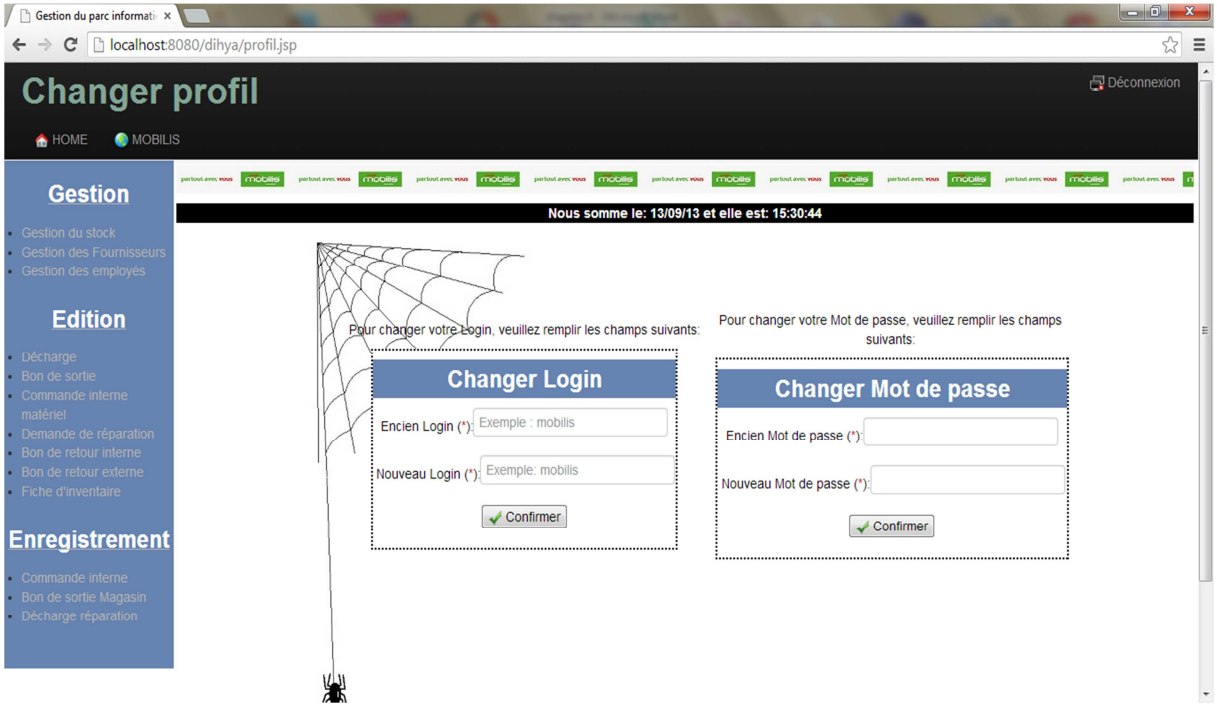


Figure V.7. Page de changement de profil.

- **La page « Gestion des Employés » :** L'administrateur accède à cette page en cliquant sue le lien 'Gestion des Employés'. Elle contient deux formulaires, un pour ajouter un nouvel employé, et l'autre pour la suppression d'un employé. L'admin, choisi ce qu'il veut faire et rempli le formulaire adéquat, puis le valide. Le système va lui affiché une page de confirmation.

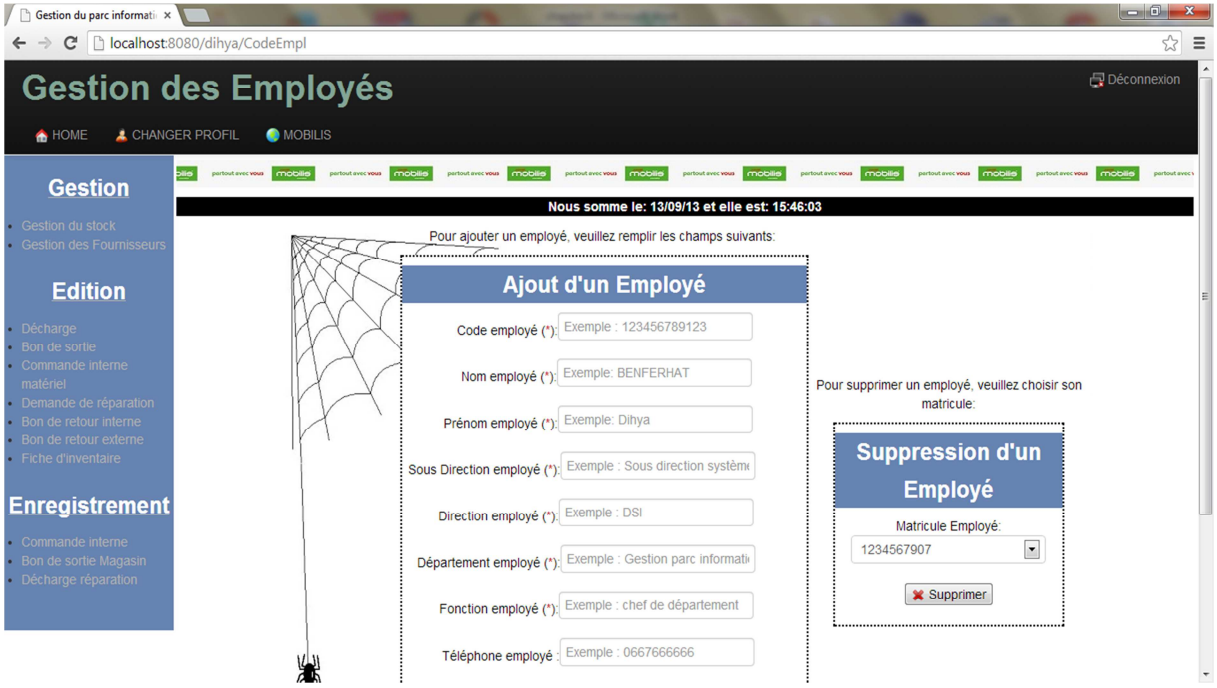


Figure V.8. Page de gestion des employés.

- **Page « Commande Interne » :** L'administrateur accède à cette page en cliquant sue le lien 'Commande interne'. Elle contient des champs à remplir, et un tableau où l'utilisateur choisi les articles à commander avec la quantité à commander, puis valide la saisie. Le système va lui affiché une page de confirmation.

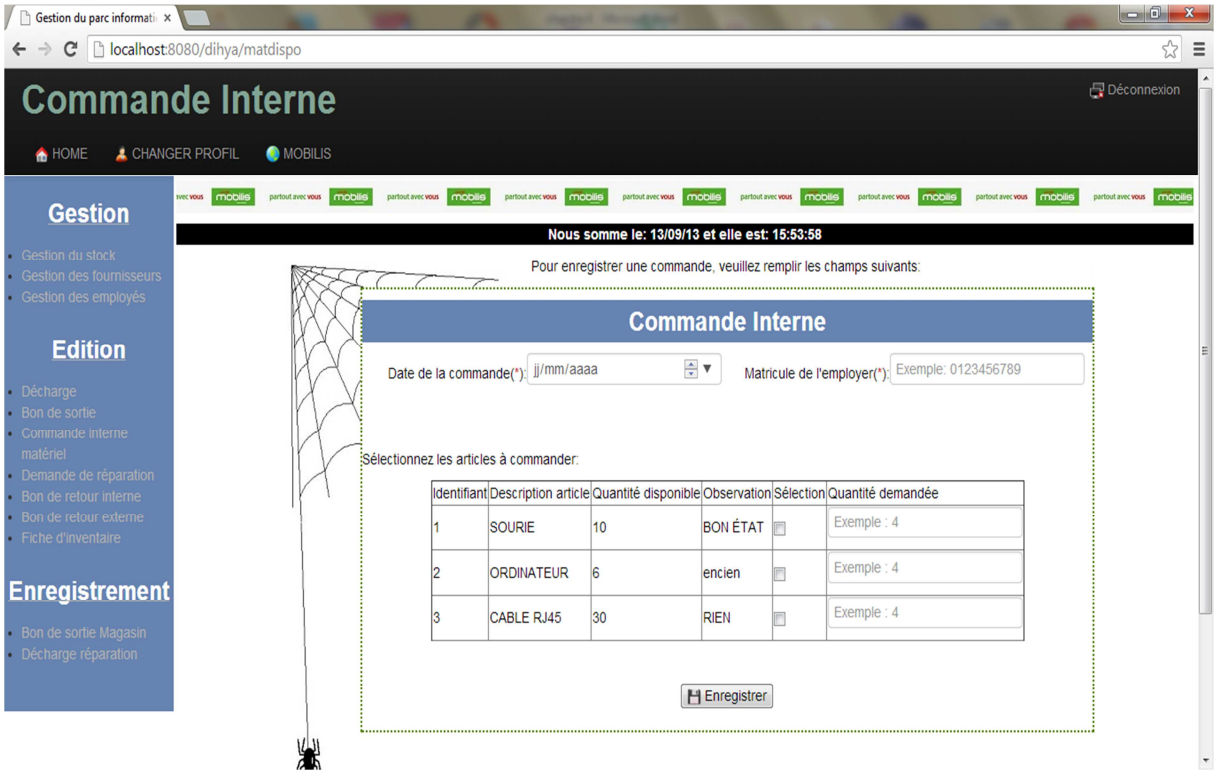


Figure V.9. Page d'enregistrement de commandes internes.

V.4. Conclusion :

Ce chapitre a décrit la dernière étape du développement qui est celle de la réalisation et la mise en œuvre du projet. Nous avons pu voir les exigences techniques du système avant de présenter l'environnement et les outils choisis. Par la suite, nous avons donné un aperçu de l'application réalisée.

Conclusion générale

Conclusion générale

L'objectif de mon projet été de créer une application Web, qui a pour fonction de gérer le parc informatique de l'ATM Mobilis.

Sur un plan plus technique, j'ai découvert le monde du développement web et j'ai appris les bases de la plateforme J2EE, notamment l'architecture MVC.

La réalisation de ce travail m'a donné l'occasion d'acquérir de nouvelles connaissances et d'en approfondir d'autres sur le développement des applications Web, je citerai le langage orienté objet JAVA précisément sous la plateforme J2EE, le langage de requêtes HQL, le CSS et le HTML. Ça m'a permis aussi de me familiariser avec un certain nombre d'outils informatiques de développement, à titre d'exemple, je cite l'IDE Eclipse, Tomcat et MySQL.

Hibernate quant à lui est beaucoup plus complexe à maitriser et nécessiterait un projet plus complet pour en maitriser toutes les facettes. Malgré cela, il m'est apparu comme étant un outil extrêmement puissant. Son efficacité est d'ailleurs reconnue à tel point que le Framework a été porté en C# pour être utilisé dans l'environnement .NET. C'est l'un des Frameworks incontournables dans le monde J2EE.

En fin, je prévois d'apporter d'éventuelles améliorations et j'espère que l'application réalisée sera adoptée par l'établissement d'accueil.

Bibliographie

Bibliographie

- Architecture lient/serveur par Eric Cariou [2010].
- Concevoir des applications Web avec UML par Jim Conallen. Edition Eyrolles [2000].
- UML 2, Analyse et conception par Joseph Gabay et David Gabay. Edition Dunod [2008].
- UML, outil du génie logiciel par N.Abdat et L.Mahdaoui. Edition Pages bleues [2007].
- UML 2, modéliser une application Web par Pascal Roques.
- Conception de bases de données avec UML par Gilles Roy.
- Le suivi de session sous J2EE par Jean-Noël Sorenti [2002/2003].
- Développement Web – Servlet par Jean-Michel Richer [2010].
- Cours web Formulaires par Lionel Seinturier de l'université Pierres & Marie Curie [2003].
- Servlet & JSP par Philippe Poulard. Edition INRIA [2003].
- UV Serveur d'Applications par Deruelle Laurent [2002].
- Application Web et J2EE par Pierre Gambarotto [2011].
- Développons en Java par Jean-Michel DOUDOUX.
- Introduction à SQL et MySQL par Sébastien Namèche [2003].
- Java2 Enterprise Edition par Pierre-Yves Gibello [2004].

Webiographie

- http://www.google.fr/
- http://www.sitedeziro.com/
- http://www.wikipedia.org/
- http://www.hibernate.org/
- http://www.eclipse.org/
- http://www.developpez.com/
- http://www.MySQL.com/
- http://www.tomcat.apache.org/