Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire

Université Abou Bakr Belkaid– Tlemcen

Faculté de Technologie

Département d'Informatique

Mémoire de fin d'études

En vue de l'obtention du Diplôme de Licence en Informatique

Thème :

Développement d'une application de gestion de location de véhicules avec Java RMI

Réalisé par :

- **BABA AHMED Farah.**
- **BELHAMRA Sarah.**

Encadré par :

- **Mr. BENMAMMAR Badr.**

Co-encadré par :

- **Mr.Bendaoud Fayssal.**

Présenté le 10 Juin 2014 devant la commission d'examination composée de MM.

- *Mr .Chouiti.S (Examinateur)*
- *Mme .Kazi.A (Examinateur)*

Année universitaire: 2013-2014

Remerciement

En préambule à ce modeste mémoire nous remercions ALLAH de nous avoir aidé

et donner la patience et le courage durant ces langues années d'étude.

Nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

Ces remerciements vont tout d'abord au corps professoral et administratif de la Faculté d'Abou Bakr Belkaid des Sciences, département d'informatique, pour la richesse et la qualité de leur enseignement et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

Nous tenons à remercier sincèrement et respectivement nos encadreurs Monsieurs, Benmammar Badr et Bendaoud Fayssal, qui se sont toujours montrés à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'ils ont bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu

le jour.

Sans oublier bien sûr nos chers parents pour leur contribution, leur soutien et leur

patience.

Enfin, nous adressons nos plus sincères remerciements à tous nos proches et amis de notre promotion sans oublier ceux du master 1 RSD et master 2 SIC, qui nous ont aider et toujours encouragés au cours de la réalisation de ce mémoire.

Merci à tous et à toutes.

Dédicaces

Tout d'abord, je dédie ce modeste travail à mes parents les fans de facebook, je cite :

Mon père Belhamra Mohammed.Salah qui a été mon ombre durant toutes les années des études, et qui a veillé tout au long de ma vie à m'encourager, à me donner l'aide et à me protéger.

Ma mère Hassaine Nasséra, qui m'a donné la vie, le symbole de tendresse, qui s'est sacrifiée pour mon bonheur et ma réussite.

Que dieu les gardes et les protèges.

Je le dédie aussi à :

Mon grand-père Ahmed, que dieu lui procure santé, bonheur et longue vie.

Mes deux grands chers frères, Karim et Lotfi.

Mes deux grandes chères sœurs, Hannane et Tita.

Ma petite nièce Manel et sa maman Fadwa.

Toute ma famille à Constantine.

Tous mes amis qui me sont chères et que j'aime beaucoup et avec qui nous avons passé les plus belles années d'études qui resteront graver à jamais.

BELHAMRA SARAH

Dédicaces

Je dédie humblement ce mémoire avec fierté aux personnes formidables de ma vie :

Tout d'abord à celle qui m'a toujours encouragé et soutenue dans tout ce que j'ai entrepris ;
Celle qui a toujours été là dans mes moments de détresse et de joie, ma très chère mère.
A celui qui a toujours cru en moi qui c'est dévouée et sacrifiée tout au longs de ma scolarité, mon très cher père.
A ma moitié, celle qui ma aidé soutenue et conseillé, celle qui m'a aidée du mieux qu'elle pouvait, mon adorable jumelle Lamia.
A mes petits frères et sœurs que j'adore Mohammed, Neila, Ghizlene, Neila et Nassim.
A mes deux formidables meilleures amies mes sœurs de cœur qui ont embellie et marqué mes plus belles années d'études Amel et Sarah.
Sans oublié les personnes qui m'ont aidé et qui sans eux je ne serais pas arrivé là ou j'en suis aujourd'hui : Leyla, Meryem, Hidayet, Ayoub, Rida, Adil et Noor.

BABA AHMED FARAH

Tables des matières

Sommaire

Introduction générale

De nos jours, nous avons remarqué l'intérêt croissant de vouloir gagner en temps, de conserver les données, de diminuer le coût de l'information, de limiter le nombre d'employés et pleins d'autre raison qui ont poussé les magasins, les agences, les sociétés ou les entreprises à chercher des solutions informatiques capable de résoudre leurs problèmes et répondre à leurs besoins.

En effet, l'informatique s'est imposée d'une manière inimaginable au sein des entreprises comme étant un soutien à leurs différentes opérations logistiques, et cela grâce à la réduction progressive des coûts et le gain en temps ainsi qu'à sa relation fascinante avec la gestion des bases de données.

L'informatique est donc de plus en plus utilisée dans tous les domaines d'activités y compris celui de la gestion de l'information et des données.

Dans le cadre de la gestion de l'information, nous avons choisi la gestion du stock auquel nous rattacherons d'ailleurs notre projet de fin d'études, qui consiste à réaliser une application de gestion de stocks destinée à une agence de location de voitures.

Pour aboutir à notre but, nous y sommes rendus au sein d'une agence de location de voitures, où nous avons accumulé plusieurs informations sur la structure du système et sur son fonctionnement, par parenthèse, nous avons pu constater qu'il y avait pas mal de traitement qui se font manuellement ceci a confirmé notre problématique.

Une fois les informations et les indications ont été rassemblées, il nous a fallu organiser notre travaille pour la bonne conception et réalisation de notre PFE et pour cela nous avons entrepris notre étude ou autrement dit notre mémoire selon les 2 chapitres suivants :

- **Chapitre I :** inclus deux parties, la première est consacrée à la présentation de l'architecture client/serveur, dans la deuxième partie, nous parlerons des outils utilisés en définissant le mécanisme de java RMI.
- **Chapitre II :** comporte la phase de présentation et de conception de notre application.

Chapitre I:

Architecture Client - Serveur & Outils utilisé

I.1 Architecture client serveur

I.1.1 Introduction

Aux cours de ces dernières années les architectures de communication ont vu une évolution considérable, dans le but de répondre aux besoins de la sécurité et de la centralisation des informations, l'architecture client-serveur s'impose sur le marché et offre une amélioration dans la capacité de stockage de données et une rapidité considérable de traitement des postes de travaille, cette technologie se développera à un rythme des plus accélérés et cela essentiellement a la baisse des prix de l'informatique personnelle et le développement des réseaux informatiques.

I.1.2 Définition

Le modèle client/serveur est une des modalités des architectures informatiques distribuées au sein de cette architecture, les processus sont classés entre offreurs de services (serveurs) et consommateurs de services (clients), le terme serveur s'applique a tout programme qui offre un service que l'on peut atteindre a travers un réseau il accepte des demandes issues du réseau, les traite et renvoie le résultat au demandeur, quant au terme client, il s'applique a tout programme qui émet une demande a un serveur et qui attend une réponse. [1]

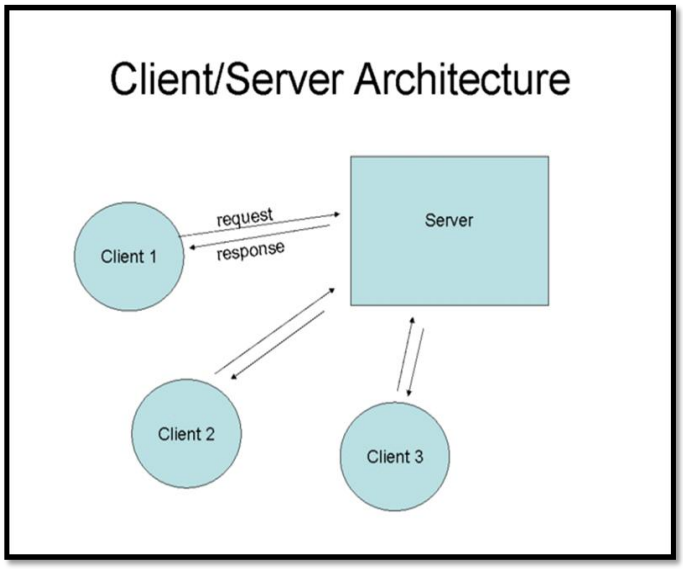


Figure I.1: Modèle client serveur

I.1.3 Dialogue Client/serveur

I.1.3.1 Client

1- Terminal sans puissance locale de traitement.
2- Les stations effectuent tous les traitements.

Le client envoie des requêtes au serveur et il attend et reçoit les réponses du serveur.

I.1.3.2 Serveur

3- Ordinateur central qui effectue tous les traitements.
4- Gère le réseau et stocke les bases de données.

Un serveur est initialement passif, il attend, il est à l'écoute, prêt à répondre aux requêtes envoyées par des clients. Dés qu'une requête lui parvient, il la traite et envoie une réponse.

I.1.3.3 Dialogue

Le client et le serveur doivent bien sûr utiliser le même protocole de communication.
Un serveur est généralement capable de servir plusieurs clients simultanément.

I.1.4 Les 4 principes de base du C/S

1- Rendre l'architecture matérielle transparente vis à vis des développeurs et des utilisateurs finaux.
2- Rendre le niveau physique (et logique dans une moindre mesure) des bases de données transparent pour les développeurs et les utilisateurs.
3- Utiliser au niveau de chaque station (cliente ou serveur) l'ensemble matériel/logiciel le plus adapté.
   - Chaque machine est adaptée à des besoins précis (implique l'hétérogénéité des matériels).
   - Optimisation de l'outil.
   - Diversité des services offerts à l'utilisateur.
   - Minimisation des coûts (le sophistiqué là où il est nécessaire).
4- Permettre une séparation physique entre les actions d'un programme liées à l'interaction avec les utilisateurs et les autres actions.
   - Gestion du dialogue par le client (interface).

- Gestion des données par le serveur.
- Il s'agit d'un modèle de traitement coopératif. [3]

I.1.5 Avantages et inconvénients du modèle client serveur

Quelle que soit l'implémentation utilisée (CORBA, DCOM, Java RMI, .Net ou encore la technologie des Web service) et quelle que soit l'organisation adaptée (les différentes typologies du pair-a-pair), le modèle client/serveur présente des avantages et des inconvénients que nous avons résumés dans le tableau ci-dessous : [4]

Avantages
- Sécurité très élevée. - Fiabilité très élevée. - Hébergement de bases de données de très grande taille (milliers de Go). - Gestion centralisée performante.
Inconvénients
- Serveur et système d'exploitation propriétaires (Ibm, Bull…), d'où une forte dépendance au fournisseur et un coût du système élevé. - Coût de développement des applications élevé. - Interface utilisateur sans souplesse. - Le terminal est un poste de travail sans intelligence dont l'usage est limité aux applications du serveur auquel il est relié. - Inadaptés dans le cas d'un grand nombre d'utilisateurs connectés. - La mise à jour des applications est lourde (les applications sont en effet à mettre à jour au niveau de chaque client).

I.1.6 Différents environnements clients –serveurs

Voici quelques architectures réseaux :

I.1.6.1 Architecture mainframe

Avant que n'apparaisse le mode client-serveur, les réseaux informatiques étaient configurés autour d'un ordinateur central appelé mainframe auquel étaient connectés des terminaux passifs (écran adjoint d'un clavier sans unité centrale). Tous les utilisateurs sont alors connectés sur la même unité centrale.

Le mainframe n'affiche que du texte à l'écran sans graphisme (pas de bouton, pas de fenêtre,…). Il est spécialisé dans la gestion d'informations de masse auquel il peut appliquer des instructions simples (addition, soustraction…) mais avec une grande vélocité. Ainsi, plusieurs milliers de personnes peuvent travailler sur cette unité centrale sans ralentissement.

Aujourd'hui, les anciens terminaux passifs ont été remplacés par des émulations logicielles installées sur les PC.

Pour pallier le manque de graphisme, différentes solutions existent dont l'intégration du mainframe dans une architecture 2, 3 ou N niveaux en laissant à d'autres la fourniture d'une interface homme-machine.

Cette architecture est déployée sur le MVS d'IBM mais aussi sur des serveurs sous UNIX, Linux,…

Avantages:

1. Gestion des données et des traitements centralisée.
2. Maintenance matériel minime.
3. Grande vélocité sur des grands volumes de données et de traitements.

Inconvénients:

1. Interface homme-machine minimaliste.
2. Utilisation de langages de programmation anciens.
3. Calcul scientifique complexe impossible.

I.1.6.2 Architecture à 2 niveaux

Ce type d'architecture (2-tier en anglais) caractérise les environnements client-serveur où le poste client demande une ressource au serveur qui la fournit à partir de ses propres ressources. [2]

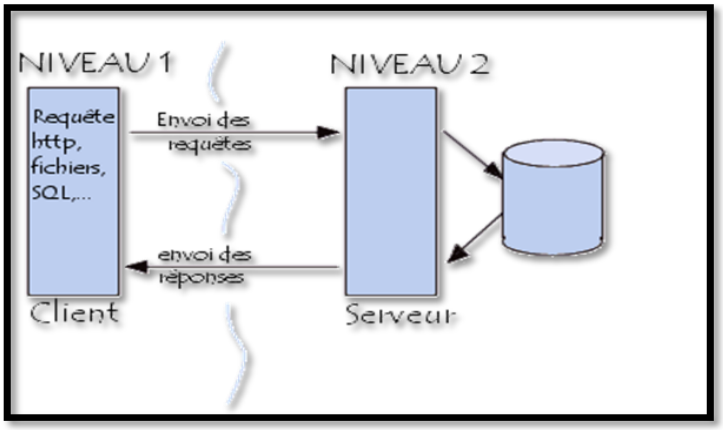


Figure I.2 : L'architecture à 2 niveaux

I.1.6.3 Architecture à 3 niveaux

Dans cette architecture (3-tier en anglais), aussi nommée trois tiers en français, un niveau supplémentaire est ajouté un client (l'ordinateur demandeur de ressources) équipé d'une interface utilisateur (généralement un navigateur web) chargée de la présentation. Un serveur d'application (appelé middleware) qui fournit la ressource, mais en faisant appel à un autre serveur. Un serveur de données qui fournit au serveur d'application les données requises pour répondre au client. [2]

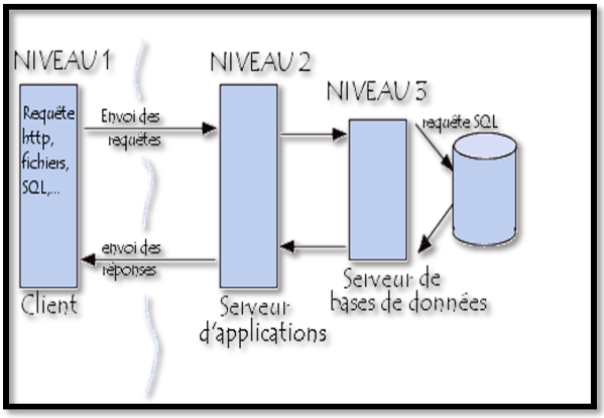


Figure I.3. L'architecture à 3 niveaux

I.1.6.4 Architecture à n niveaux

L'architecture 3 niveaux permet de spécialiser les serveurs dans une tache précise : avantage de flexibilité, de sécurité et de performance. L'architecture peut être étendue sur un nombre de niveaux plus important : on parle dans ce cas d'architecture à N niveaux (ou multi-tier). [2]

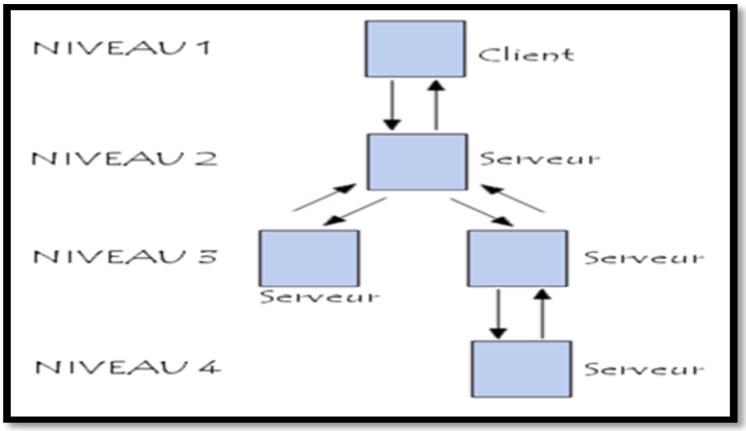


Figure I.4: L'architecture à n niveaux

I.1.7 Types de clients

I.1.7.1 Client léger

Le poste client accède à une application située sur un ordinateur dit « serveur » via une interface et un navigateur Web. L'application fonctionne entièrement sur le serveur, le poste client reçoit la réponse « toute faite » à la demande (requête) qu'il a formulée.

I.1.7.2 Client lourd

Le poste client doit comporter un système d'exploitation capable d'exécuter en local une partie des traitements. Le traitement de la réponse à la requête du client utilisateur va mettre en œuvre un travail combiné entre l'ordinateur serveur et le poste client.

I.1.7.3 Client riche

Une interface graphique plus évoluée permet de mettre en œuvre des fonctionnalités comparables à celles d'un client "lourd". Les traitements sont effectués majoritairement sur le serveur, la réponse "semi-finie" étant envoyée au poste client, où le client "riche" est capable de la finaliser et de la présenter.

I.2 Outils Utilisés

I.2.1 Introduction

Dans cette partie, nous allons présenter et déterminer les outils logiciels avec lesquels on a pu concevoir et réaliser notre application. Ici vient le rôle du génie logiciel, qui impose une bonne sélection des outils et méthodes pour construire une application conforme aux besoins et aux qualités exigées.

Parmi les différents outils logiciels on a opté pour la sélection suivante :

- Java.
- EasyPHP.
- PhpMyAdmin.
- NetBeans.
- JDBC (Java Data Base Connectivity).
- SQL.

I.2.2 Java

C'est un langage de programmation orienté objet, développé par Sun Microsystems. Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels.

Les applications Java peuvent être exécutées sur tous les systèmes d'exploitation pour lesquels a été développée une plate-forme Java, dont le nom technique est JRE (Java Runtime Environment - Environnement d'exécution Java). Cette dernière est constituée d'une JVM (Java Virtual Machine - Machine Virtuelle Java), le programme qui interprète le code Java et le convertit en code natif. Mais le JRE est surtout constitué d'une bibliothèque standard à partir de laquelle doivent être développés tous les programmes en Java. C'est la garantie de portabilité qui a fait la réussite de Java dans les

architectures client-serveur en facilitant la migration entre serveurs, très difficile pour les gros systèmes.

Enfin, ce langage peut-être utilisé sur internet pour des petites applications intégrées à la page web (applet) ou encore comme langage serveur (jsp).*

I.2.2.1 Caractéristiques de Java

Java possède un certain nombre de caractéristiques qui ont largement contribué à son énorme succès : interprété, indépendant de toute plate-forme, orienté objet, simple, fortement type, assure la gestion de la mémoire, sûre, économe, multitâche.[10]

I.2.2.1 RMI (Remote Method Invocation)

C'est une API Java permettant de manipuler des objets distants. En d'autre terme RMI est définit comme étant une technologie des objets distribués développée par Sun pour le langage de programmation Java. C'est un protocole pour pouvoir manipuler des objets à distance.

RMI a été introduite dans Java Development Kit (JDK) 1.1 et il est largement utilisé dans l'informatique distribuée objet. Il se base sur l'orienté objet l'équivalence d'appels de procédure à distance (RCP). Fonctionnalités du RMI sont dans un package de java.rmi et offrent une capacité d'objet distribué pour les applications basées sur Java.

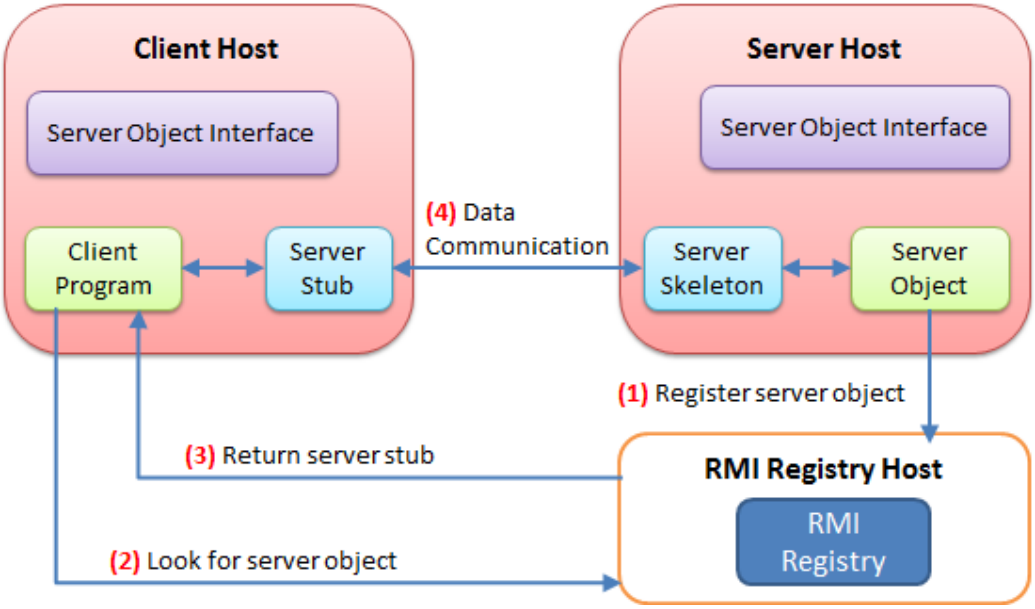


Figure I.5. : Architecture Java RMI [13]

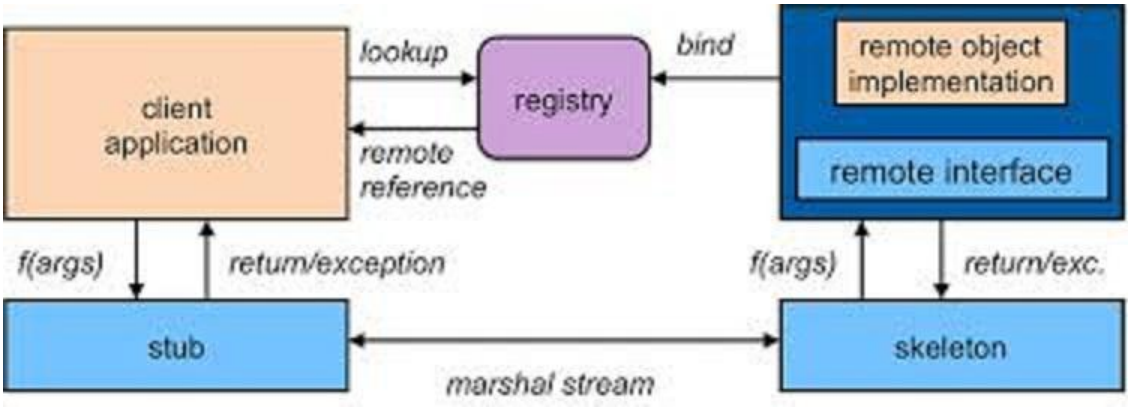


Figure I.6 : Architecture Java RMI [13]

Le but de RMI est de permettre l'appel, l'exécution et le renvoi du résultat d'une méthode exécutée dans une machine virtuelle différente de celle de l'objet l'appelant. Cette machine virtuelle peut être sur une machine différente pourvu qu'elle soit accessible par le réseau.

La machine sur laquelle s'exécute la méthode distante est appelée serveur.

L'appel coté client d'une telle méthode est un peu plus compliqué que l'appel d'une méthode d'un objet local mais il reste simple. Il consiste à obtenir une référence sur l'objet distant puis à simplement appeler la méthode à partir de cette référence.

La technologie RMI se charge de rendre transparente la localisation de l'objet distant, son appel et le renvoi du résultat. En fait, elle utilise deux classes particulières, le Stub et le Skeleton. Le Stub est une classe qui se situe côté client et le Skeleton est son homologue coté serveur. Ces deux classes se chargent d'assurer tous les mécanismes d'appel, de communication, d'exécution, de renvoi et de réception du résultat.

Le skeleton est devenu obsolète (dépassé). Une même classe skeleton générique est partagée par tous les objets distants. En plus, jusqu'à la version 5.0 du J2SE (2005), il fallait utiliser un compilateur de stub appelé RMIC (Java RMI Compiler) pour générer les stub/skeleton avant tout enregistrement sur le registre RMI. Désormais il est possible de les générer dynamiquement. [11]

Comment ça marche ?

Le développement coté serveur se compose de :

- La définition d'une interface qui contient les méthodes qui peuvent être appelées à distance.
- L'écriture d'une classe qui implémente cette interface.
- L'écriture d'une classe qui instanciera l'objet et l'enregistrera en lui affectant un nom dans le registre de noms RMI (RMI Registry).

Le développement côté client se compose de :

- L'obtention d'une référence sur l'objet distant à partir de son nom.
- L'appel à la méthode à partir de cette référence.

Enfin, il faut générer les classes stub et Skeleton en exécutant le programme rmic avec le fichier source de l'objet distant [11][12]

I.2.3 EasyPHP

Easy PHP est un logiciel qui a était conçue en 1999, pour simplifier et faciliter la manipulation et le travail sur des fichiers au format PHP (tel que son nom l'indique).

Il offre la possibilité de travailler dans un environnement serveur complet . L'application regroupe un serveur Apache, une base de données MySQL, une version du langage PHP ainsi que des outils facilitant le travail et le développement de nos sites ou nos application.

Dans notre application, nous avons utilisé ce dernier pour la création et la gestion de la base de données pour notre application. Les éttapes à suivre pour utiliser EasyPHP sont comme suit :

- Télécharger EasyPHP sur le site : http://www.manucorp.com/ ou http://www.easyphp.org/
- Lancer l'application « EasyPHP\ easyphp2.0b1_setup.exe » et suivre les instructions d'installation en choisissant la langue et le répertoire d'installation.
- Pour vérifier si EasyPHP fonctionne, il vous suffit de taper dans votre navigateur web préféré: http://localhost ou http://127.0.0.1

I.2.4 phpMyAdmin

Est une interface d'administration pour le SGBD MySQL. Il est écrit en langage PHP et s'appuie sur le serveur HTTP Apache.

Il permet d'administrer les éléments suivants :

- les bases de données
- les tables et leurs champs (ajout, suppression, définition du type)
- les index, les clés primaires et étrangères
- les utilisateurs de la base et leurs permissions
- exporter les données dans divers formats (CSV, XML, PDF, OpenDocument, Word, Excel et LaTeX)

Voici le lien pour le télécharger :

http://www.01net.com/telecharger/windows/Programmation/base_de_donne/fiches/41098.html

I.2.5 NetBeans

Netbeans est un Open source integrated development environment (IDE) (Environnement de Developpement Intégré) gratuit et multilingue, conçu par Sun en juin 200, pour la création et le devellopement en Java, C, C++ , Web (PHP, HTML, JavaScript, CSS) et d'autres languages de programmation.

Il est codé en Java et fonctionne sur la plupart des systèmes d'exploitation avec une machine virtuelle Java (JVM), y compris Solaris, Mac OS et Linux.

Ainsi,NetBeans constitue par ailleurs une plate forme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plate forme. Ce dernier s'enrichit à l'aide de plugins.

NetBeans comprend toutes les caractéristiques d'un IDE moderne, tel que : la coloration syntaxique, un projets multi-langage, refactoring et un éditeur graphique d'interfaces et de pages web.[7]

I.2.6 JDBC (Java Data Base Connectivity)

JDBC est une interface de programmation créée par Sun Microsystems un intermédiaire entre le programme java et le SGBD choisi, il permet un accès local ou à distance, à une base de données relationnelle.Il fonctionne selon un principe Client/Serveur, où le Client est le programme Java, le Serveur est la base de données. Il permet l'accès à des bases de données créées dans des SGBD variés, depuis un programme en Java. Il est fourni par le paquetage (package) java.sql. L'API JDBC est presque totalement indépendante des SGBD (quelques méthodes ne peuvent être utilisées qu'avec certains SGBD mais ne doivent être utilisées qu'en cas de nécessité impérieuse pour améliorer les performances). Le principe général est comme suit :

a) Le programme Java ouvre une connexion avec le SGBD.
b) Il envoie des requêtes SQL.
c) Il récupère les résultats.
d) Il met à jour la BD.
e) Il ferme la connexion.[6] et [7]

I.2.8 Language SQL (Structured Query Language)

SQL est un language de requetes structuré, en d'autre terme c'est un language informatique destiné à interroger ou piloter une base de donnéés. La version originale est appelée SEQUEL (structured English query language), elle a été conçu par un centre de recherche d'IBM en 1974 et 1975. SQL a donc été présenté comme un systémine de base de données commerciale en 1979 par Oracle Corporation. Le langage SQL n'est pas sensible à la casse (en anglais case sensitive), cela signifie que l'on peut aussi bien écrire les instructions en minuscules qu'en majuscule. Toutefois, cette insensibilité à la casse n'est que partielle dans la mesure où la différenciation entre minuscules et majuscules existe au niveau des identificateurs d'objets.[8] et [9]

I.3 Conclusion

Ce chapitre a exposé les deux parties principale pour la réalisation de notre mémoire, dans la premiere partie nous avons abordé une des modalités des architectures informatiques, intitulée l'architecture client-serveur et ses différents environement, alors que la deuxième a contenue les differents outils logiciels utilisé pou concevoir et finaliser notre projet.

Chapitre II:

Conception et implémentation de l'application

II.1 Introduction

Après avoir déterminé dans le chapitre précédent la méthodologie et les outils pour la conception et la réalisation de notre projet, nous allons, dans ce deuxième chapitre, aborder le fonctionnement de notre application de gestion de location de voiture et ceci à travers la présentation de ses différentes interfaces.

II.2 Tables de notre base de données

Nous avons créé une base donnés nommé 'PFE', elle contient 3 tables (Voir Figure II.1)

Figure II.1 : La base de données 'pfe'

Les 3 tables sont les suivant :

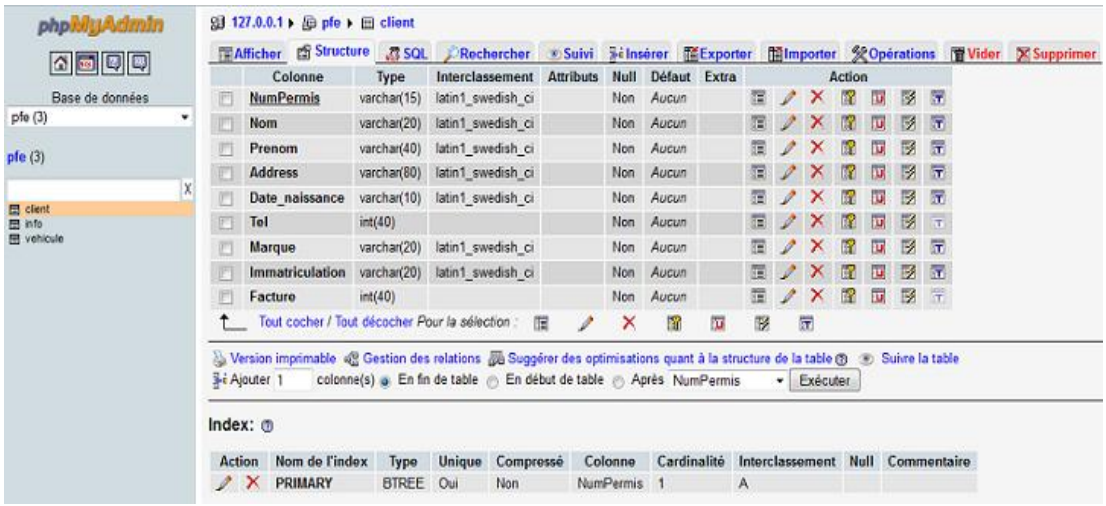


Figure II.2 : La table 'Client'

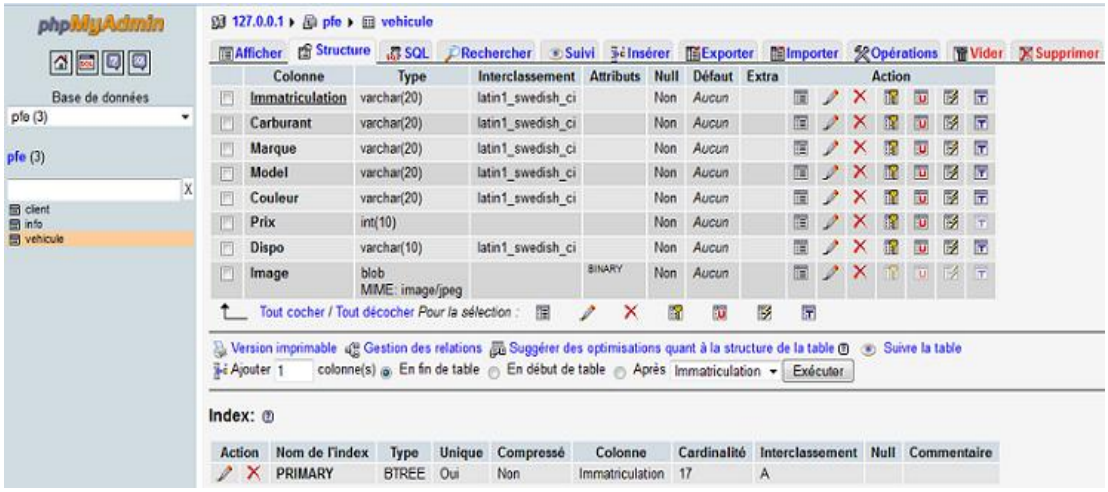


Figure II.3 : La table 'Véhicule'

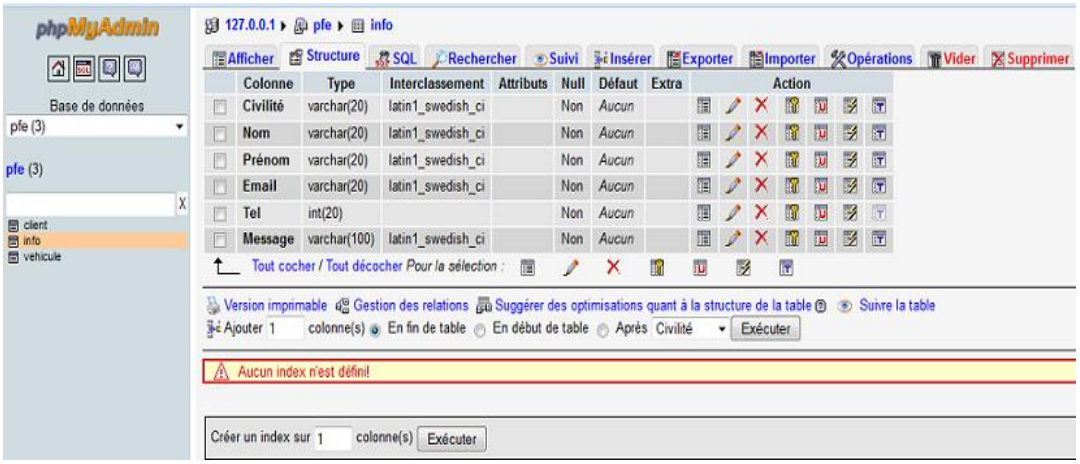


Figure II.4 : La table 'Info'

II.3 Règles de la gestion

Notre application marche selon un certain nombre de règles. Le protocole de réservation est le suivant :

- Un client consulte la liste des véhicules.
- Après la consultation, il peut réserver un véhicule disponible.
- Le client peut aussi annuler son réservation ou la modifiée.
- Un client a le droit de louer un seul véhicule.

II.4 Capture et présentation des interfaces

Notre application est composée de 6 interfaces que nous allons définir ci-dessous :

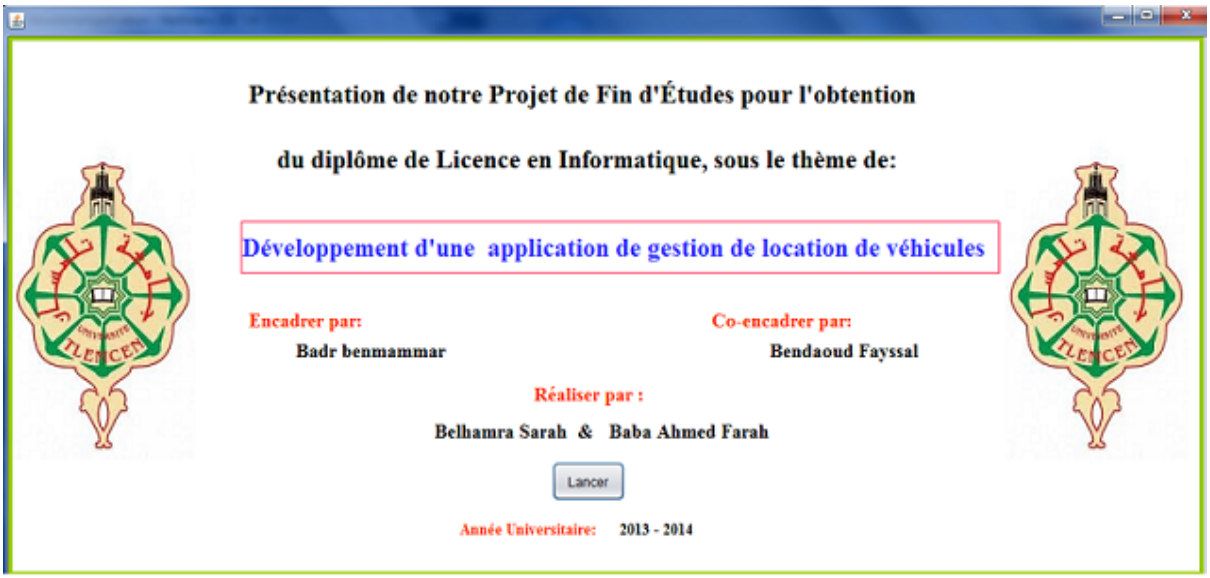


Figure II.5 : Page de présentation

La première interface représente la page d'accueil du client (Voir Figure II.6) où il y a une brève présentation de l'agence, et un menu.

Figure II.6 : Interface Principale

Nous avons 1 menu avec 5 boutons, chacun d'eux affiche une interface. Nous allons monter et décrire la fonctionnalité de ses derniers :

II.4.1 Accueil

Ce bouton représente la toute première interface, en d'autre terme la page d'accueil que nous venons de la présenter ci-dessus.

II.4.2 Recherche Voiture

En cliquant sur ce bouton, une seconde interface s'affiche (Voir Figure II.7), c'est celle ou le client fait son choix de voiture selon 3 critères :

- La marque.
- Le carburant.
- Le prix.

Le client peut donc, soit préciser un critère, deux, ou les trois en même temps, ou bien aucun de ces derniers. Une fois il clique sur le bouton « Rechercher », il va s'afficher dans le tableau toutes les voitures de l'agence. Pour choisir une voiture du tableau le client doit sélectionner une ligne.

Figure II.7 : Liste des voitures

Si la ligne sélectionner a le champ Dispo :'Non', un message s'affichera (Voir Figure II.8)

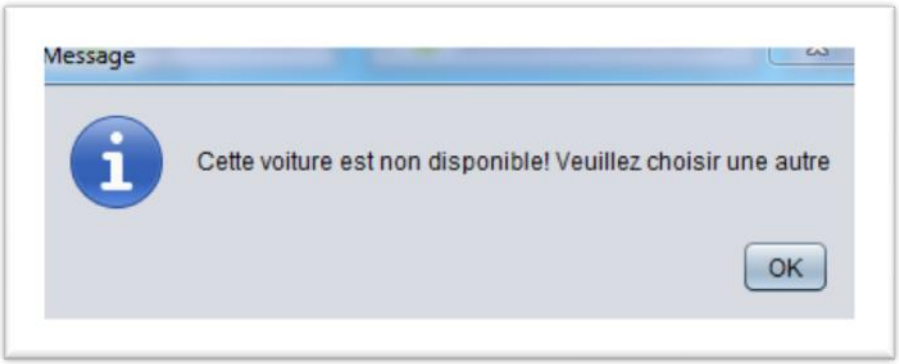


Figure II.8 : Message au client

Une fois le client clique sur « Réserver » une fenêtre de réservation s'ouvre (Voir Figure II.9). Elle contient un formulaire de coordonnée que doit remplir le client ainsi qu'une autre partie contenant des informations sur la voiture choisies et une facture.

Figure II.9 : Louer la voiture

II.4.3 Votre réservation

Ce bouton affiche une interface (Voir Figure II.10) où le client peut voir sa réservation tout en entrant son numéro de permis qui représente son identifiant. Le client peut ainsi soit supprimer sa réservation ou la modifier.

Figure II.10 : Votre réservation

Si le client clique sur « Modifier » l'interface de Recherche de voiture s'ouvre pour qu'il puisse choisir à nouveau une voiture et par suite la réserver. Et si il clique sur supprimer, la ligne contenant ses informations dans le tableau s'effacera et un message s'affichera au client (Voir Figure II.11).

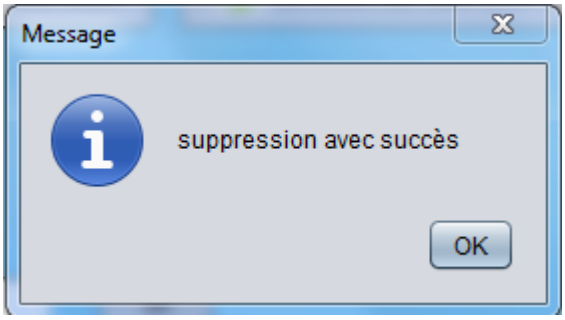


Figure II.11 : Message au client

II.4.4 Condition de réservation

C'est le bouton qui affiche l'interface des conditions de locations d'une voiture (Voir Figure II.12).

Figure II.12 : Conditions de réservation

Ici le client s'informe sur les différentes conditions de l'agence :

- Conditions de location.
- Assurance.
- Durée & Paiement.

II.4.5 Contact

Ce dernier bouton sert à afficher l'interface « Contact » (Voir Figure II.13). Dans cette dernière le client a la possibilité d'envoyer un message à l'agence et il y trouve aussi de coordonnée de l'agence.

Figure II.13 : Contact

II.5 Conclusion

A travers ce chapitre nous avons pu présenter les différentes interfaces de notre application que nous avons pu la réaliser avec les outils définis dans le chapitre précédant. Nous avons également présenté une partie de la conception de notre application.

Conclusion générale

Ce projet à contribuer à améliorer nos compétences dans divers domaines tout en améliorant nos connaissances en conception et en programmation.

Notre travail consiste à mettre au point une application client/serveur qui a pour but d'automatiser la gestion d'une agence de location de voiture.

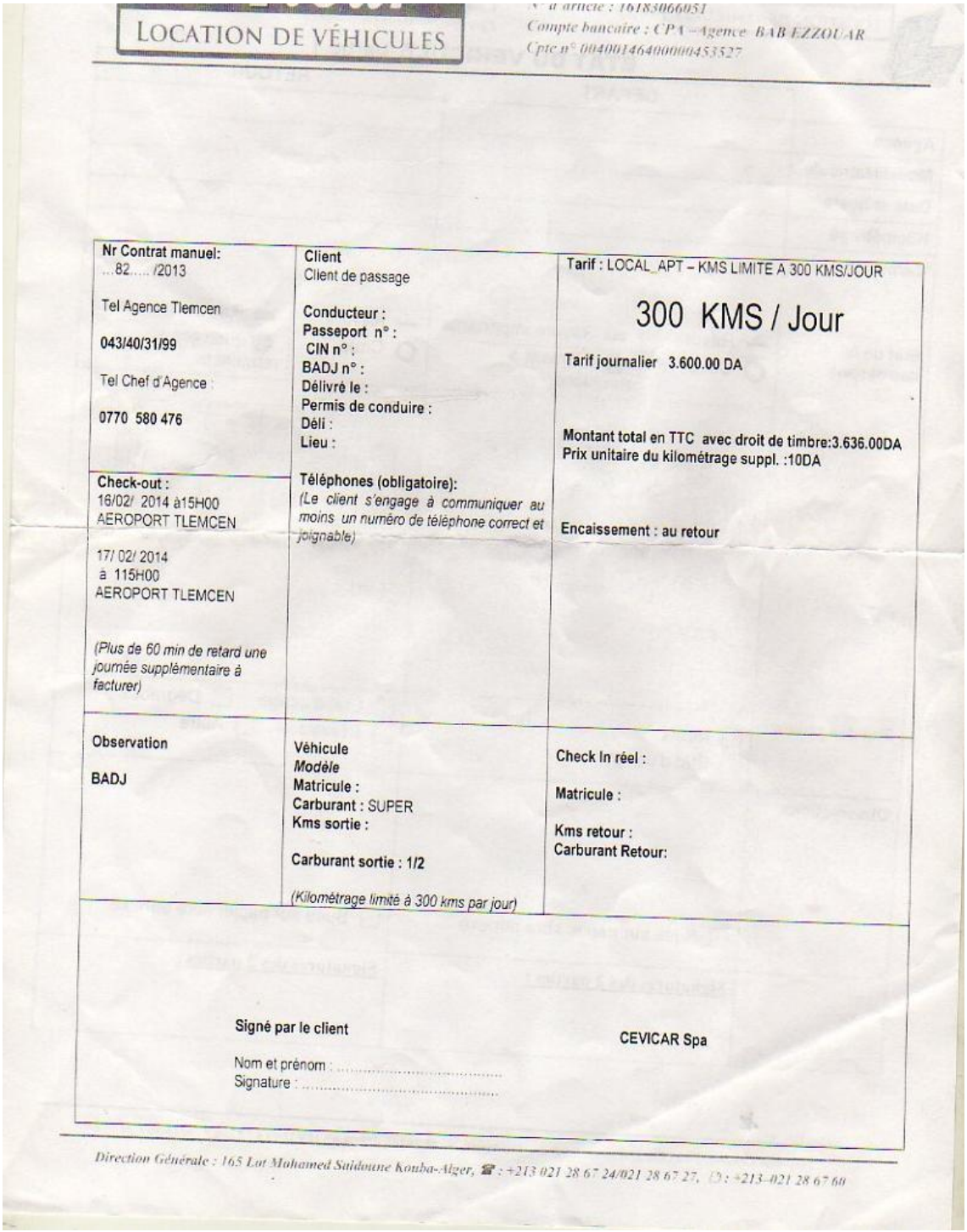
Pour développer notre application il nous a fallu métriser l'approche de l'architecture client/serveur avec la méthode Java RMI en intégrant une base de données.

Dans ce rapport, nous avons présenté au début les supports nécessaires au fonctionnement des communications, ensuite nous avons abordé les différentes étapes de la conception de notre application.

Au cours de la réalisation de notre projet, nous avons été astreints par quelque limite notamment, la contrainte du temps qui a été relativement un obstacle devant l'ajout de certaines fonctionnalités. Cependant il était une occasion pour mettre en évidence et déployer sur le plan pratique nos connaissances en informatique.

Le projet peut être améliorer, et ce, en lui ajoutant quelque perceptive comme lancer l'application sur un réseaux local, ajouter des fonctionnalités, agrandir la base de données et ajouter quelque interfaces pour mieux l'adopter aux besoins de l'utilisateur et pour qu'il soit toujours fiable et au niveau des progrès de l'agence.[22]

Annexe



LOCATION DE VÉHICULES

N° d'article : 16183066051
Compte bancaire : CPA – Agence BAB EZZOUAR
Cpte n° 00400146400000453527

Nr Contrat manuel: ...82..... /2013 Tel Agence Tlemcen **043/40/31/99** Tel Chef d'Agence : **0770 580 476**	**Client** Client de passage **Conducteur :** **Passeport n° :** **CIN n° :** **BADJ n° :** **Délivré le :** **Permis de conduire :** **Déli :** **Lieu :**	**Tarif :** LOCAL_APT – KMS LIMITE A 300 KMS/JOUR 300 KMS / Jour **Tarif journalier 3.600.00 DA** **Montant total en TTC avec droit de timbre:3.636.00DA** **Prix unitaire du kilométrage suppl. :10DA** **Encaissement : au retour**
Check-out : 16/02/ 2014 à15H00 AEROPORT TLEMCEN 17/ 02/ 2014 à 115H00 AEROPORT TLEMCEN *(Plus de 60 min de retard une journée supplémentaire à facturer)*	**Téléphones (obligatoire):** *(Le client s'engage à communiquer au moins un numéro de téléphone correct et joignable)*	
Observation **BADJ**	**Véhicule** ***Modèle*** **Matricule :** **Carburant :** SUPER **Kms sortie :** **Carburant sortie : 1/2** *(Kilométrage limité à 300 kms par jour)*	**Check In réel :** **Matricule :** **Kms retour :** **Carburant Retour:**

Signé par le client

Nom et prénom : .....
Signature : .....

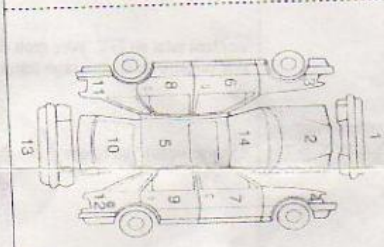
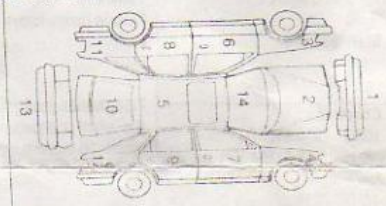
CEVICAR Spa

Direction Générale : 165 Lot Mohamed Saidoune Kouba-Alger, ☎ : +213 021 28 67 24/021 28 67 27, ⌂ : +213-021 28 67 60

LOCATION DE VÉHICULES

Compte bancaire : CPA – Agence
Cpte n° 00400146400000453527

ETAT DU VEHICULE partie 1

	DEPART	RETOUR
Agence		
Model/Matricule		
Date et heure		
Kilométrage		
Carburant		
Etat de la carrosserie	— Rayure = Rayure importante O Coup ⊗ Element à remplacer	— Rayure = Rayure importante O Coup ⊗ Element à remplacer
	 ☐ RAS	 ☐ RAS
Etat des pneus	☐ Neufs ☐ Etat d'usage	☐ Etat d'usage ☐ Dégradés ☐ Crevaison ☐ Autre .....
Observations	☐ Suite sur papier libre annexé	☐ Suite sur papier libre annexé
	Signatures des 2 parties :	Signatures des 2 parties :

Direction Générale : 165 Lot Mohamed Saideune Koubo-Alger. ☎ : +213 021 28 67 24/021 28 67 27, 🖷 : +213-021 28 67 60

Références bibliographiques

[1] Définition client serveur <http://www.commentcamarche.net/contents/222-environnement-client-serveur > consulté le 20/04/2014.

[2] Architecture <http://www.commentcamarche.net/contents/221-reseaux-architecture-client-serveur-a-niveaux>. Consulté le 22/04/2014.

[3] (principe client serveur) <http://www.iro.umontreal.ca/~ostrom/documents/client_serveur.pdf > consulté le 05/05/2014.

[4] Fiche Architecture client /serveur .support du cours en ligne .disponible sur le site <http://www.licp.fr/site/images/stories/pdf/BTS_cgo/fiche%2028%20%20architecture%20client%20serveur.pdf>.

[5] Netbeans.site web. Disponible sur le site <http://www.technoscience.net/?onglet=glossaire&definition=5346>.

[6]JDBC.Site web. Disponible sur <http://www.journaldunet.com/encyclopedie/definition/340/34/20/java_database_connectivity.shtml>.

[7]JDBC. Site web. Disponible sur <http://searchoracle.techtarget.com/definition/Java-Database-Connectivity>.

[8]SQL . site web .disponible sur http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/informatique-sql-2524/7.

[9] SQL <http://www.webopedia.com/TERM/S/SQL.html> visité le 26/04/2014.

[10] Definition Java.site web.disponible sur
< http://www.futura- sciences.com/magazines/high- tech/infos/dico/d/internet-java-485/> et sur <http://ipeti.forumpro.fr/t21-definition-de-langage-java-java-script>
[11]Java rmi <http://searchsoa.techtarget.com/definition/Remote-Method-Invocation>
http://www.commentcamarche.net/contents/1030-introduction-a-rmi-remote-method-invocation>
[12]<.http://infolab.stanford.edu/CHAIMS/Doc/Details/Protocols/rmi/rmi_description.html> et
<http://www.jmdoudoux.fr/java/dej/chap-rmi.htm#rmi>
[13]figure java rmi
< http://lycog.com/wp-content/uploads/2011/03/java-rmi-overview.png>
[1] Figure client serveur < http://exonous.typepad.com/mis/2007/10/clientserver-ar.html>

Liste des figures

Liste des abréviations

- IDE : (Integrateted Development Environment) Environnement de Développement intégré.
- SQL : (Stuctured Query Language) Langage de Requête structurée .
- SGBD : Systéme de gestion de base de données.
- Mysql: (My Structured Query Language) Mon Language de Requete Structuré.
- JRE : (Java Runtime Environment) Environnement d'exécution Java.
- JVM : (Java Virtual Machine) Machine Virtuelle Java.
- RCP: (Remote Procedure Call) Appel de procedure a Distance.
- UML : (Unified Modeling Language) Language de Modélisation Unifié.
- JDK : (Java Development Kit).

Résumé

Dans ce projet de fin d'études, nous avons réalisé une application de gestion de stock qui permet à une agence de location de voiture la facilité d'accès et de la gestion en temps réel du parc automobile. Cette application a été développé par la méthode Java RMI avec l'approche Client /Serveur et en utilisant l'environnement de développement NetBeans.

En terme de complexité, une implémentation orienté objet allégée est optimisée la mise au point, ceci nous a évité toute redondance des instructions qui sert à rendre l'application réutilisable.

Finalement, à travers ce projet nous avons acquis une expérience enrichissante qui nous a permis de développer et élargir nos compétences et réalisé l'importance du travail d'équipe.

MOTS CLES: JAVA, RMI, SWING, NETBEANTS, MYSQL.

Summary

In this Final Project Studies, we achieve a stock management application that allows an agency of rent cars easiness of access and real-time management of the park automobile. This application was developed by the Java RMI method with Client / Server approach and using NetBeans development environment. In terms of complexity, a lightweight object-oriented implementation is optimized and developed; this has saved us all redundant instructions which serve to make the application reusable.

Finally, with this project we have gained valuable experience that has allowed us to develop and expand our skills and realized the importance of teamwork.

KEYWORDS: JAVA, RMI, SWING, NETBEANTS, MYSQL.

ملخـص

من خلال مشروع نهاية الدراسة أنجزنا تطبيق لإدارة المخزون الذي يسمح لوكالة تأجير السيارات بسهولة و فعالية إدارة مخزونها من السيارات لقد تم تطوير هذا التطبيق من خلال طريقة 'جافا أر أم إي' مع إتباع منهج العميل/الخادم و باستخدام بيئة التطوير 'ناتبينز', فهي طريقة بسيطة التنفيذ لجعل التطبيق قابل لإعادة الاستخدام و التطوير.أخيرا من خلال هذا المشروع اكتسبنا خبرة قيمة، الأمر الذي سمح لنا بتطوير وتوسيع مهاراتنا وإدراك أهمية العمل الجماعي.

الكلمات المفتاحية : جافا، ر م ي، سوينغ، نات بينس، ماي اس كويل.