



Université de Caen  
Département informatique  
U.F.R. Sciences

Univers Informatique  
14700 Falaise



# CoolGestion

*CoolGestion*

RAPPORT DE STAGE DE LICENCE

Sous la direction de Monsieur Lionel Cosson  
Juin – août 2003

Enseignant tuteur :  
Monsieur Etienne Grandjean

Auteur :  
Nicolas Lefèvre

# COOLGESTION

## RAPPORT DE STAGE DE LICENCE

Sous la direction de Monsieur Lionel Cosson  
Juin – août 2003

Enseignant tuteur :  
Monsieur Etienne Grandjean

Auteur :  
Nicolas Lefèvre





## Remerciements

---

Je remercie tout particulièrement :

- Mon tuteur, Monsieur Lionel COSSON, pour m'avoir accueilli dans son entreprise et pour avoir été à mon écoute pendant toute la durée de ce stage.
- Les techniciens, Messieurs François DJABALI et Geoffrey DUVENT pour avoir été présents lors des phases clés de mon stage aussi bien lors de la conception que de la réalisation du logiciel.
- Mademoiselle Stéphanie LEFEBVRE pour m'avoir aiguillé en ce qui concerne l'intégration de la comptabilité dans le logiciel.
- Monsieur Etienne Grandjean pour son appui lors de la réalisation de ce rapport.



## Sommaire

---

Introduction .....	6
1. Présentation du cadre du stage .....	7
1.1. Présentation du magasin .....	7
1.2. Positionnement du stage .....	7
1.3. Cahier des charges.....	8
1.4. Analyse de l'existant .....	9
1.5. Structuration du problème .....	9
1.6. Outils informatiques utilisés .....	10
1.6.1. Le choix de Java.....	10
1.6.2. Le choix de MySQL .....	11
1.6.3. Connexion JAVA-MYSQL.....	12
1.6.4. Outils de documentation .....	12
2. Conception et réalisation du logiciel .....	13
2.1. La base de données .....	13
2.1.1. Critique du modèle .....	17
2.2. L'interface homme-machine .....	18
2.3. Implémentation du programme.....	21
2.3.1. Implémentation de la base de données.....	21
2.3.2. Programmation des classes intermédiaires.....	21
2.4. L'interface graphique .....	22
2.4.1. Explication de la mise en forme.....	24
2.4.2. Limitations de l'interface graphique .....	25
3. Avancement du projet.....	26
3.1. Problèmes rencontrés.....	26
3.2. Avancement du projet.....	27
3.3. Fonctionnalités ajoutées .....	28
3.4. Limitations .....	29
3.5. Extensions .....	30
Conclusion .....	32
Bibliographie .....	33
Annexes	



## Introduction

---

Dans le cadre de ma maîtrise informatique, j'ai effectué un stage, d'une durée de 9 semaines, au sein du magasin L'Univers Informatique à Falaise. Ce magasin est tout aussi spécialisé dans la vente de composants que dans la maintenance informatique.

Mon premier travail, en commun avec celui des équipes technique et comptable, a été de mettre en place le cahier des charges nécessaire à l'aboutissement d'un logiciel conforme aux besoins de l'entreprise. Dans un second temps, mon travail s'est focalisé sur une recherche de documentation dans le but de parfaire mes connaissances sur les sources de données permettant la connexion de programmes à des bases de données indépendantes.

Ayant toutes les clés en main, la troisième phase devait aboutir, au terme de ce stage, à la réalisation du logiciel en lui-même.

Ce rapport évoque, en détail, les différentes phases d'analyse et de développement nécessaires à la réalisation du logiciel. Celui-ci est complété par des informations concernant les problèmes rencontrés et les moyens informatiques d'y remédier.

Avant même le début de la phase de conception, il convient de découvrir les différents acteurs de ce stage à travers une présentation succincte de L'Univers Informatique. Nous évoquerons, par la suite, la structuration du sujet de ce stage et son positionnement dans l'entreprise, pour finir sur l'explication de nos choix en ce qui concerne le langage de programmation et le système d'exploitation.

Cette phase terminée, nous parlerons, ensuite, de notre réflexion conceptuelle autour du sujet dans une partie dédiée à la modélisation, dans un premier temps, de la base de données et, dans un second temps, du programme et de son interface graphique.

Cette partie nous conduira à la troisième et dernière phase, évoquant la réalisation proprement dite, de l'implémentation du noyau. Nous finirons, dans une troisième partie, par faire le point sur les résultats obtenus à la fin de ce stage puis nous conclurons sur son apport dans un cursus universitaire.



## 1. Présentation du cadre du stage

---

### **1.1. Présentation du magasin**

Domicilié au cœur de Falaise l'Univers Informatique est un magasin offrant un service des plus compétents tout aussi bien dans le domaine de la création de sites Internet que dans l'administration de systèmes et des réseaux. Cependant, leur principale activité s'oriente autour de la vente et de la maintenance de matériel informatique.

Sous la direction de Monsieur Cosson, deux techniciens offrent leurs compétences chacun dans des domaines spécialisés mais se recoupant concernant la maintenance de matériel informatique.

Les besoins de Monsieur Cosson et de ses techniciens auxquels on m'a demandé de répondre s'orientent autour de la création d'un logiciel orienté base de données. Après avoir présenté l'Univers Informatique et ses employés, il convient de définir le positionnement du stage dans l'entreprise.

### **1.2. Positionnement du stage**

Le stage a pour but de rendre le travail des employés plus efficace encore, par la création complète d'un logiciel de gestion du magasin. L'automatisation et, le cas échéant, la réduction du temps de mise en œuvre de nombreuses tâches, permettent la proposition d'un meilleur service.

Pour cela, la gestion du magasin se devra d'être précise, intuitive et surtout très efficace. Le logiciel est le fruit d'un réel besoin de toute l'équipe qui souhaite une centralisation des données en ce qui concerne la vente et la maintenance du matériel informatique. En effet, nous le verrons dans l'analyse de l'existant, tout processus est rendu laborieux car aucune trace informatique des transactions effectuées n'est créée. Cela implique, par conséquent, un long temps de recherche dans le cas, par exemple, du calcul du stock en fin d'année, processus qui pourrait être automatisé.

Pour bien représenter l'ensemble du travail demandé, il convient de définir les besoins et les contraintes dans le cahier des charges.



### **1.3. Cahier des charges**

Le stagiaire devra développer une application de gestion complète du magasin fonctionnant sous les systèmes Microsoft Windows. Pour cela, il devra lui-même choisir ses propres outils de développement, tant au point de vue du stockage des données que de la programmation de l'interaction entre l'interface graphique et ces mêmes données.

Le logiciel terminal devra être pourvu des fonctionnalités suivantes :

- La création, modification et consultation des données concernant un fournisseur ou un client.
- Le stockage de factures fournisseur permettant d'établir une trace des produits du magasin.
- La création et la consultation de factures client.
- La création, modification et consultation de devis client.
- La gestion du stock par un tableau des produits achetés, facturés, livrés.
- La création automatique d'un journal de caisse.
- L'envoi de la comptabilité du magasin par e-mail.
- L'envoi des factures client du jour au cabinet comptable.
- La création de bons de livraison et leurs consultations.
- La recherche rapide des prix des produits ainsi que leur quantité en stock.
- La création d'avoirs sur une facture client\*.
- Un tableau récapitulatif des avoirs créés\*.
- La création de bulletin d'intervention\*.
- Un tableau récapitulatif des achats et des devis d'un client.
- La génération de fichiers html imprimables pour les devis, les factures client, les bons de livraison et les avoirs.

Le logiciel se devra, en outre, d'utiliser le code barre des produits pour une plus grande rapidité quant à la création de factures client, d'une part, et la traçabilité des produits, d'autre part.

Après avoir schématisé l'ensemble de la base de données, celle-ci sera validée par l'équipe technique qui donnera son accord pour la phase de développement.

Après une courte étude graphique, le logiciel devra être pourvu d'une interface efficace permettant une meilleure intuitivité ainsi qu'un gain de temps dans l'exécution de processus.

Une phase de tests réels viendra conclure ce stage permettant de déceler les possibles problèmes.

---

\* Ces fonctionnalités ont été rajoutées au fur et à mesure du développement du logiciel.



#### **1.4. Analyse de l'existant**

A l'origine, une simple utilisation du logiciel Microsoft Excel permettait la création de factures, de devis et des journaux de caisse. Plusieurs problèmes se sont posés quant à cette méthode. En effet, il n'existait aucune automatisation des tâches, tout se faisait manuellement, ainsi, lors de la création d'une facture client, il fallait, en plus, écrire le montant de cette facture dans le journal de caisse. La gestion du stock était rendue quasi-impossible car il fallait reprendre l'ensemble des factures, qu'elles soient des fournisseurs ou pour les clients, et soustraire les quantités achetées à celles vendues.

Lors de la réception d'un colis provenant d'un fournisseur, les techniciens étaient obligés de « coller » une étiquette sur chaque produit pour permettre d'établir sa traçabilité dans le cas d'un retour en service après vente, d'où une grande perte de temps et donc de productivité.

Du point de vue comptabilité, le cabinet comptable était obligé d'envoyer un de ses représentants pour consulter le journal de caisse et les factures établies, là de même, il en résulte une grande perte de temps.

Nous le voyons à travers cette courte explication, aucun processus d'où la nécessité de créer un logiciel qui permettrait d'automatiser l'ensemble de ces tâches.

Le coût de revient d'un logiciel commercial, permettant une parfaite intégration avec le logiciel de comptabilité utilisé au cabinet comptable, étant trop élevé pour un magasin de la taille de l'Univers Informatique, les techniciens se sont vu poussés à établir un début de base de données sous Microsoft Access. Cependant, ayant d'autres occupations, ceux-ci n'ont pu mener à terme cette ébauche. De plus, désirant un coût minimal quant à la création du logiciel, l'utilisation d'Access n'était pas des plus sage.

J'ai d'abord étudié cette base de données ce qui m'a permis une première approche dans la conception du logiciel.

Après avoir analysé les différentes possibilités envisagées j'ai structuré le problème pour permettre le développement de la solution adéquate.

#### **1.5. Structuration du problème**

La structuration du problème est soumise, d'une part, au sujet, qui est la création d'un logiciel et, d'autre part, au cahier des charges lui-même qui dicte les phases de la réalisation.

Trois phases ont été définies lors de ce stage. La première consiste à réaliser une étude sur la conception de la base de données, plus connue sous le nom de méthode Merise. Cette méthode permet de créer une base de données conformément aux besoins de l'entreprise.



La seconde phase consiste, dans un premier temps, à implémenter la base de données en elle-même et, dans un second temps, de créer le noyau\* du programme permettant les interactions entre l'interface graphique et cette base. Le noyau se composera d'un ensemble de petits programmes simplifiant la réalisation de la troisième et dernière phase.

Cette dernière phase consiste à créer l'interface graphique. Dans un premier temps, une étude sera réalisée pour présenter une interface des plus efficace en établissant une charte graphique. Cette phase se terminera par la programmation de l'interface graphique en elle-même.

Pour mener à bien l'ensemble de ces phases et plus particulièrement les deux dernières, il convient de définir et d'expliquer nos choix quant à la sélection des outils informatiques.

## **1.6. Outils informatiques utilisés**

### *1.6.1. Le choix de Java*

Le langage de programmation Java met à notre disposition un ensemble de package graphique rendant simple la création d'interfaces utilisateurs en vue d'une interactivité avec le logiciel et une intuitivité des plus accrues. Un gestionnaire intégré à Java permet la création d'API permettant une reprise optimale du logiciel pour des modifications plus aisées.

De plus la syntaxe de Java est analogue à celle de C++, ce qui le rend économique et professionnel. Le fait de créer une autre version du C++ n'est cependant pas l'objet de ce langage. En effet le point clé de Java est le suivant: Il est beaucoup plus facile d'obtenir un code sans erreur avec Java qu'avec C++ car un certain nombre de points susceptibles d'apporter des erreurs tel que :

- L'allocation et la libération de mémoire manuelles ont été retirées. En effet la mémoire, dans Java, est allouée et libérée automatiquement.
- Java permet, entre autres, une vraie gestion des tableaux car les concepteurs ont retiré l'arithmétique des pointeurs. La notion de référence sur une zone mémoire remplace avantageusement celle de " pointeur ", car elle supprime la possibilité d'écraser toute zone mémoire à cause d'un compteur erroné.
- Les concepteurs ont supprimé l'héritage multiple en le remplaçant par une nouvelle notion d'interface dérivée d'Objective C. Une interface nous offre les mêmes possibilités que l'héritage multiple, sans la complexité de la gestion de hiérarchie d'héritage multiple.

---

\* **Noyau** : Programme minimal, dépourvu d'une quelconque interface graphique et de fonctionnalités. Les fonctionnalités viennent s'intégrer au noyau, conçu spécialement pour les recevoir. Dans le cas de notre logiciel l'interface graphique se sert du noyau pour communiquer avec la base de données.



Les objectifs de Java s'articulent autour de termes clés dont la simplicité, la portabilité, la fiabilité, et les performances élevées, le tout, qui plus est, gratuit.

### 1.6.2. Le choix de MySQL

De nombreux arguments nous ont conduits à choisir MySQL comme serveur\* de base de données plutôt que Microsoft Access. En effet, MySQL génère des fichiers de stockage beaucoup plus petits, ce qui rends l'ensemble de la base de données moins importante\* que sous Access.

Concernant la gestion Multi-Utilisateurs, là encore, MySQL propose un système beaucoup plus souple qu'Access. En effet, sous Access, chaque client désirant interagir avec la base de données conduit le système à la dupliquer, par conséquent, chaque client connecté dispose de sa propre base de données sur le serveur. Si le nombre de client dépasse 5, les échanges entre ceux-ci et le système surchargent le réseau ralentissant l'ensemble des échanges entre tous les ordinateurs. MySQL, quant à lui, prend en compte l'ensemble des interactions des clients avec la base de manière souple grâce à son démon\*. En effet, celui-ci prend en compte l'ensemble des requêtes des clients et les traite les unes après les autres sur une seule et unique base de données. Les échanges sur le réseau sont réduits à leur minimum.

D'autre part, MySQL nécessite une licence d'utilisation payante uniquement lorsque le programme est destiné à être vendu. La revente du logiciel n'étant pas envisagée, MySQL nous est donc gratuit, à la différence d'Access qui nécessite une licence payante et ce, pour une quelconque utilisation.

L'optique d'une utilisation sur Internet nous a confortés dans notre choix. En effet, le langage PHP, destiné à utiliser une base MySQL, est beaucoup plus conventionnel que le langage ASP, destiné, lui, à l'utilisation d'une base Access. La reprise d'une base de données MySQL pour une utilisation sur Internet est donc à la portée d'un plus grand nombre de développeurs.

MySQL a, cependant, aussi ses défauts, parmi eux, nous retiendrons là non-gestion des clés étrangères\*. Il en résulte une surcharge quant à la création des requêtes SQL. Cependant, toutes les requêtes nécessaires à l'utilisateur sont traduites dans le logiciel. Par conséquent, un utilisateur n'a pas besoin d'écrire ses propres requêtes. Ce défaut de MySQL est contourné, l'utilisateur n'en ayant aucune connaissance.

---

\* **Serveur de base de données** : Programme permettant la gestion d'un ensemble de données de manière optimale et professionnelle. MySQL est un serveur de base de données robuste, rapide et gratuit.

\* **Base de données moins importante** : Pour exemple la table contenant l'ensemble des codes postaux Français associés aux villes faisait quelque 6.5Mo sous Access contre 1Mo sous MySQL.

\* **Démon** : Un démon est un programme exécuté au lancement de la machine et fonctionnant en tâche de fond. L'utilisateur n'a pas forcément connaissance de l'existence d'un tel processus.

\* **Clé étrangère** : Pour toute valeur de la clé étrangère, elle réfère à une valeur identique de la clé primaire obligatoirement présente dans la table associée.



### 1.6.3. Connexion JAVA-MYSQL

L'utilisation de MySQL sous Java n'est opérationnelle que si un pilote\* est installé. Après une recherche de documentation sur Internet, notre choix s'est porté sur le pilote de Mark Mathews, le pilote MM. Il permet la connexion de Java à MySQL, utile pour gérer une base de données avec les outils offerts par Java tel qu'une interface graphique.

### 1.6.4. Outils de documentation

Notre application étant vouée à une utilisation sous Windows, nous avons décidé d'utiliser les outils que ce système d'exploitation met à notre disposition pour permettre une intégrité maximale avec celui-ci. En vue de cette intégrité, nous avons développé une aide, non pas sous le format PDF d'Adobe AcrobatReader mais sous le format CHM propre au système de Microsoft. Ainsi, l'utilisateur se voit en mesure d'utiliser cette aide dès son premier lancement, il n'a aucun besoin de se familiariser avec celle-ci, la normalisation étant faite pour être intuitive.

Aucun outil n'a été utilisé pour créer cette aide mise à part le compilateur CHM de Microsoft. En effet, il suffit de créer ses pages sous forme HTML sous un quelconque éditeur puis de les intégrer au fichier CHM pour pouvoir les utiliser.

Le choix de cet outil plutôt que de l'aide conventionnelle (fichier HLP) de Microsoft s'explique par sa simplicité et par son intuitivité quant à l'interaction avec l'utilisateur.

Après avoir défini le cahier des charges et conclu sur nos choix quant aux outils informatiques utilisés, il convient d'appliquer une bonne méthodologie de création ; entendons par-là, la conception de la base de données dans un premier temps et de l'interface homme-machine dans un second temps pour finir dans une partie concernant l'implémentation du programme qui n'est autre que l'écriture du logiciel.

---

\* **Pilote** : Programme permettant l'utilisation d'un premier programme dans un second si ces deux derniers sont indépendants l'un de l'autre et fonctionnels l'un sans l'autre. Il permet simplement la jonction entre deux programmes indépendants.



## 2. Conception et réalisation du logiciel

---

### 2.1. La base de données

A la base de la conception, le cahier des charges a nécessité une grande réflexion, permettant d'entrevoir toutes les possibilités que le logiciel devrait offrir. Les éléments soumis à une reprise ou à une consultation sont les points clé de la base de données. En effet, celle-ci stockera toutes les informations à tout point de vue, de la facture fournisseur jusqu'à la facture client.

Pour établir une base de données, non seulement conforme aux besoins de l'Univers Informatique mais aussi respectant les principales règles de gestion de la conception des bases de données, il convient de la concevoir méthodiquement à travers la méthode Merise.

En établissant les données d'un produit ainsi que sa vie dans le magasin, nous obtenons un grand nombre d'informations devant être implémentées dans la base de données ainsi :

« Un produit est vendu, à un certain prix et à une certaine date, au magasin par un fournisseur, ce produit a une désignation, un temps de garantie fournisseur. Pour être vendu, le magasin l'enregistre et fixe un prix de vente. Lors de sa vente, le magasin enregistre l'acheteur et la date de vente. »

Cette phrase semble cependant oublier de nombreux concepts, cependant, nous avons déjà une idée des informations minimales et obligatoires devant figurer dans la base de données. Pour être plus complets, nous nommons « fournisseur » un objet ayant de nombreux attributs que nous verrons dans le prochain schéma.

Pour être plus précis il ne faut pas réfléchir au niveau « produit » mais plutôt en essayant de s'adapter en fonction de la vie du produit ainsi, en ce qui concerne l'achat de produits chez des fournisseurs :

« Un **fournisseur** établit une **facture**, ayant une *date* et un *montant total*, concernant des **lots de produits** et leur *quantité*, ces lots de produits ont, tous, une *désignation*, un *prix d'achat* et de *vente* et un *temps de garantie*. »

Maintenant, que nos produits sont enregistrés, chacun d'entre eux peut être vendu à un client, ainsi :

« Un **client**, désirant acheter un certain **produit**, se voit remettre une **facture** à la *date du jour*, qu'il peut *payer de différentes manières*. Cette facture est établie par un *intervenant*. Le client doit donc payer un *montant total* regroupant les prix de tous les **lots de produits** qu'il désire acheter. »

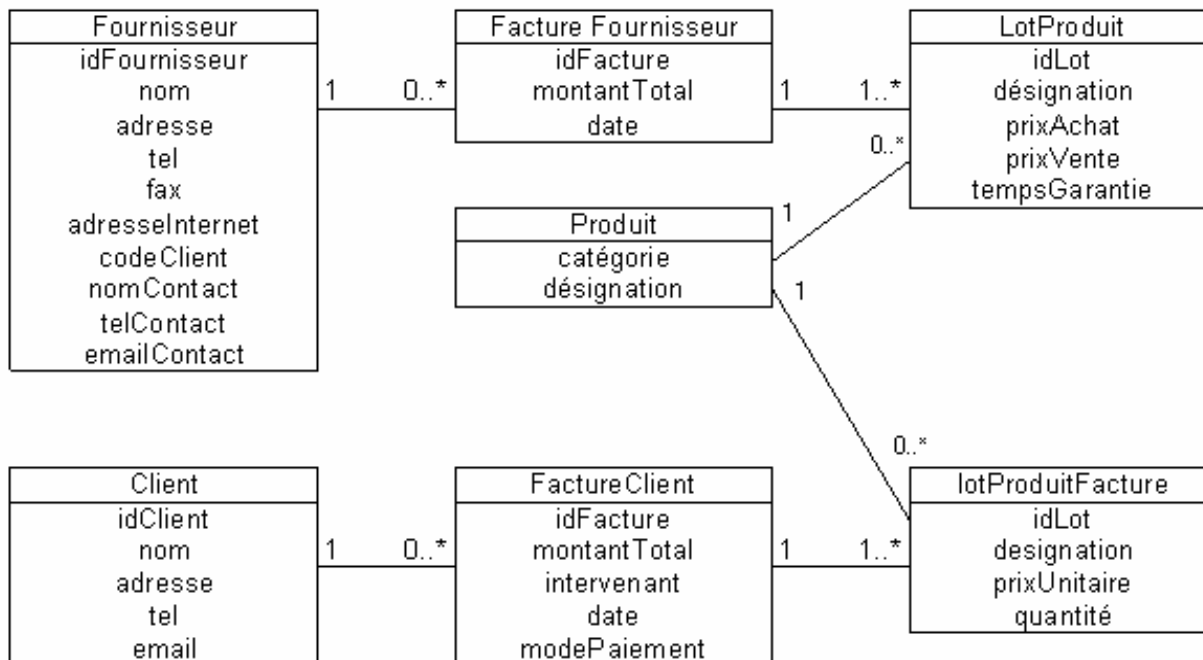


Dès lors, nous avons les informations minimales permettant de vendre des produits ayant préalablement été achetés. Nous en concluons donc la liste des tables suivante :

- Fournisseur
- Facture fournisseur
- Lots de produits
- Produit
- Client
- Facture client

Concernant ces tables nous avons aussi l'ensemble des attributs devant y figurer (En italique dans les phrases).

Voici donc le schéma minimum aussi appelé modèle conceptuel de données.



Modèle conceptuel de données minimal.

Nous avons maintenant l'ossature de notre base de données, de nombreux éléments vont s'y greffer en regardant de nouveau le cahier des charges. En effet, nous ne parlons pas encore de devis, de bons de livraison, d'avoirs, de comptabilité mais aussi est surtout d'utilisation de code barre.



Reprenons notre système pour y intégrer ces éléments :

« Lorsqu'un client est mécontent ou lorsqu'un des produits qu'il a achetés ne lui convient pas, le magasin peut établir un **avoir** sur la **facture** se référant à cet article. Un avoir se compose d'une *date*, d'un *montant* et est associé à une facture client. Il dispose en outre d'une *désignation*. »

« En vue d'un achat, un client peut demander au magasin d'établir un **devis**, pour cela, l'*intervenant*, conseille le client sur les différents choix d'achat disponibles dans l'ensemble des **produits**, en stock ou non, et ce, quelle que soit la *quantité* demandée. Le client se voit ensuite proposer un devis à un certain *montant*. Un devis se compose en outre d'une *date* et d'un *ensemble de lots de produits* et de leur *quantité*. »

« Il se peut qu'un client désire une quantité de matériel trop important et, n'ayant pas les moyens de tout ramener chez lui, demande au magasin de le livrer. L'intervenant établit donc un **bon de livraison** à chaque déplacement, rendant compte de la *quantité livrée* et de la *quantité due*. Un tel bon de livraison dispose d'une *date*, et d'un *ensemble de lots de produits* ayant chacun une *quantité livrée*. »

Pour permettre une utilisation des codes barre, pour rechercher les données d'un produit, il nous faut créer une nouvelle table qui, à chaque lot de produits acheté chez un fournisseur, associe un code barre. Il existe plusieurs comportements à adopter selon les catégories de produits. En effet, certaines catégories de produits n'ont qu'un code barre, identique pour tous les produits, c'est le cas des boîtes de disquettes ou de CD-R ou de tout autre sorte de consommable de manière générale.

D'autres catégories attribuent un code barre unique pour chaque produit. C'est le cas des disques durs ou tout autre produit disposant d'une garantie. Le troisième cas concerne les produits n'ayant tout simplement pas de code barre. C'est le cas des microprocesseurs et barrettes mémoires. Dans ce cas, il nous faut attribuer un numéro de lot, pour permettre de l'associer à une facture fournisseur.

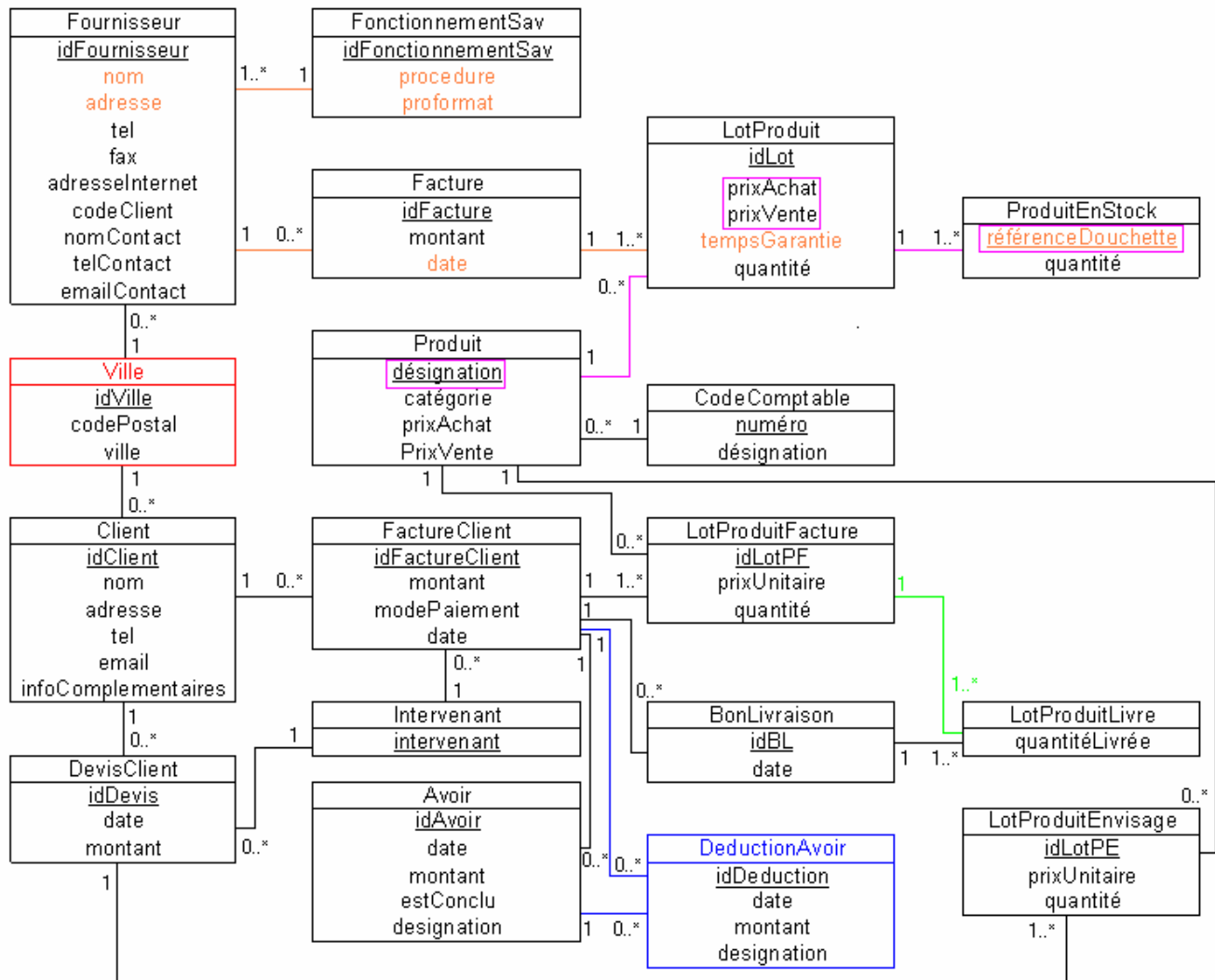
En ce qui concerne la comptabilité et son intégration dans le logiciel, il nous suffit de créer un système qui, pour chaque produit stocké ou non, associe le numéro du plan comptable.

Le modèle conceptuel suivant prend en compte tous ces besoins en essayant au mieux d'optimiser l'ensemble des données pour éviter la redondance d'informations, sans pour autant en perdre.

Ce modèle subira, par la suite, quelques ajouts. En effet, la mise en place rapide d'un jeu de tests et l'avancement du projet en milieu de stage nous a permis de développer quelques fonctionnalités utiles nécessitant toutefois la création de nouvelles tables. Ces fonctionnalités concernent, d'une part, le journal de caisse qui nécessite une trace des fonds de caisse d'une journée sur l'autre et d'autre part, un système de création de bulletins d'intervention permettant de garder une trace des travaux effectués sur les ordinateurs des clients du magasin.



Il est à noter que ces ajouts ne modifient en aucun cas le modèle suivant, ils sont même indépendants de celui-ci.



Modèle conceptuel de données final et ses cardinalités \*

Nous voyons dans ce modèle quelques entités (futurs tables dans la base de données) ajoutées, C'est le cas par exemple de la table ville, qui contient l'ensemble des codes postaux français (En rouge dans le schéma). Comme nous l'avons vu dans les paragraphes précédents, ce modèle est optimisé, toutes les informations que nous avons vues dans nos phrases logiques y figurent, cela nécessite donc des explications.

\* **Les cardinalités** : Entre chaque entité, nous voyons des doublons de chiffres ((0..\*), (1), (1,1..\*) etc.). Ces chiffres nous renseignent sur le nombre d'une première entité présente dans une seconde. Par exemple la cardinalité (1,0..\*) entre client et ville signifie que chaque client habite dans une seule et unique ville, tandis qu'une ville peut accueillir de zéro à un nombre indéfini de clients. De même un client peut établir autant de devis qu'il veut, chaque devis présent dans la base de données se réfère à un seul et unique client, ce qui explique la cardinalité (1,0..\*).



La mise en place d'une table « DeductionAvoir » (En bleu dans le modèle) nous permet de prendre tout ou partie d'un avoir précédemment créé. En effet, cette table stocke l'ensemble des déductions émises pour chaque avoir, ainsi, l'intervenant ne peut pas soustraire un montant supérieur au montant restant de l'avoir.

Lors de l'utilisation du code barre, la base de données nous permet de retrouver la désignation, le prix d'achat et le prix de vente du produit douché\* en consultant l'ensemble des codes barres (nommés « référenceDouchette ») de la table « ProduitEnStock » (En violet dans le modèle).

Ce code barre nous permet aussi, lors de l'utilisation de la traçabilité, de connaître le fournisseur ainsi que de nombreuses informations sur la facture liés ce produit. Le logiciel aiguille directement sur la procédure de retour en Service après vente selon le fournisseur (En orange dans le modèle).

En ce qui concerne les bons de livraison, et plus particulièrement les lots de produits livrés, ils sont intimement liés aux lots de produits facturés. En effet, chaque lot de produits livré a son équivalent dans la table des lots de produits facturés (En vert dans le modèle).

Le modèle propose une base de données prévue pour l'utilisation sur Internet. En effet, la mise en place de droits\* sur la table « produits » permettrait à tout visiteur du site hébergeant la base de données, d'établir son propre devis en sélectionnant ces produits dans cette table.

Ce modèle n'est cependant pas exempt de défauts, certains volontaires, d'autres révélés lors de la phase de test.

### 2.1.1. Critique du modèle

Ce modèle proposé, bien qu'adéquat pour une utilisation dans le magasin, n'est pas pour autant le meilleur modèle. En effet, celui-ci a ses limites.

Le cahier des charges stipule expressément qu'une gestion du stock doit être implémentée dans le logiciel. Cependant, nous ne voyons aucune table qui permettrait une gestion adéquate. Le modèle ne prend tout simplement pas en compte cette partie du cahier des charges. Pour pallier à ce problème, nous programmerons une méthode, en Java, permettant cette gestion. L'unique problème d'un tel programme est sa relative lenteur par rapport à un ensemble de requêtes MySQL, cependant, pour l'utilisateur, la différence est de l'ordre de la seconde.

---

\* **Produit douché** : Lecture du code barre du produit à l'aide d'un lecteur optique.

\* **Système de droit dans une base de données** : Les droits permettent de donner des privilèges à une certaine catégorie d'utilisateurs. Par exemple l'administrateur d'une base de données a tous les droits sur celle-ci (création, suppression, consultation des tables mais aussi administration des droits) et donne des droits aux autres utilisateurs. Pour une utilisation sécurisée sur Internet, l'administrateur doit donner le droit, aux clients, de consultation uniquement sur la table « Produit », les autres tables restant inaccessibles et invisibles aux yeux de ceux-ci.



En ce qui concerne le mode de paiement d'un client réglant une facture, le modèle est, là aussi, limité. En effet, il est impossible d'ajouter un mode de paiement. Ainsi, si un nouveau mode de paiement venait à voir le jour, il nous faudrait appliquer une petite modification dans le programme. Cependant, seul un développeur pourrait l'effectuer, à la différence d'une écriture des modes de paiement dans la base de données, qui pourrait se voir modifié par un utilisateur lambda.

La première phase de conception étant maintenant conclue, il convient à présent d'étudier la partie graphique du projet pour proposer une interface dont les mots clés sont convivialité, efficacité et intuitivité.

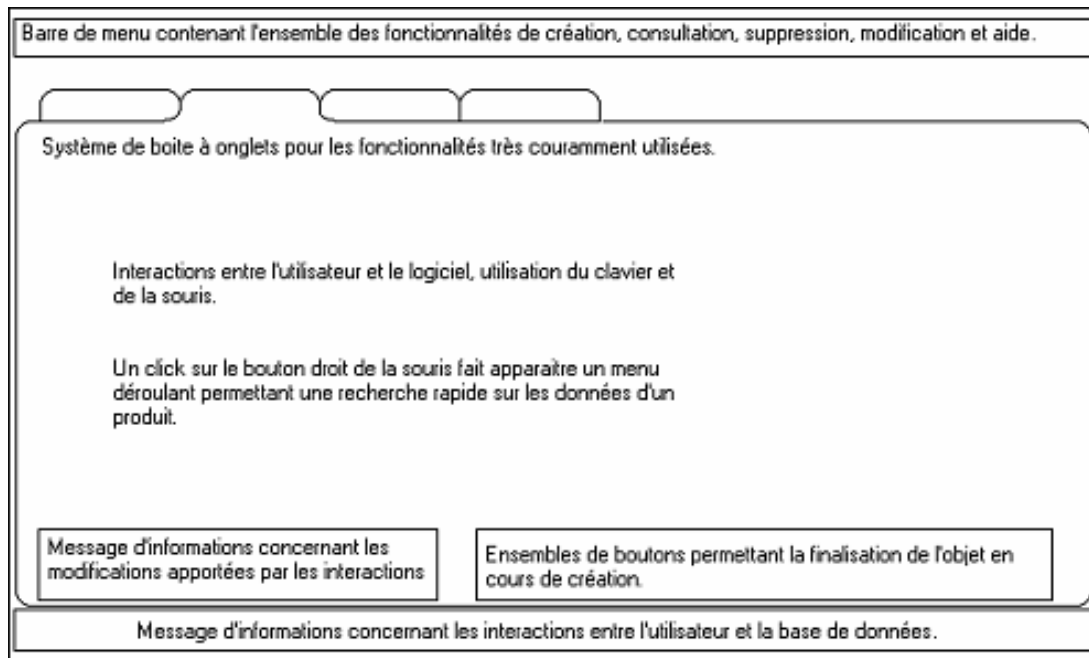
## **2.2. L'interface homme-machine**

La mise en place d'une charte graphique permet une cohérence au niveau interface, ainsi chaque élément trouve sa définition et son utilisation dans sa manière d'être représenté.

Notre langage de programmation met à notre disposition bon nombre d'outils permettant une plus grande simplicité d'utilisation, et il convient bien évidemment de les utiliser à bon escient. L'étude mise en place nous a conduits à appréhender les éléments graphiques suivants :

- La barre de menu, pour contrôler entièrement le logiciel.
- Les boutons standards, pour l'aiguillage vers une partie quelconque de l'interface graphique.
- Les boutons radios, pour une sélection unique dans une liste de choix.
- Les zones de textes, pour la saisie d'informations au clavier.
- Les onglets permettant un aiguillage rapide vers les fonctionnalités les plus courantes.
- Les boutons à cocher pour la sélection d'un ensemble de choix dans une liste.

Après avoir défini l'ensemble de besoins en ce qui concerne les outils graphiques, il convient de mettre en forme la fenêtre principale. Aux vues des besoins définis dans le cahier des charges, nous proposerons une interface graphique principale définie par trois zones d'informations ou d'interactions.



Proposition d'interface graphique.

La barre de menu, située en haut de la fenêtre principale, constitue l'élément majeur permettant de contrôler l'ensemble des fonctionnalités du logiciel. Ainsi, cette barre permet d'interagir avec la base de données en ce qui concerne les créations, suppressions, modifications de données.

La zone principale de l'interface graphique se compose d'une boîte d'onglets proposant l'ensemble des fonctionnalités les plus couramment utilisées, telle que la consultation du journal de caisse ou la création de factures. Cette boîte d'onglets permet une rapidité accrue quant à l'exécution des tâches qu'elle propose par rapport à un système de barre de menu.

Les onglets disposent tous de deux sous-parties. En bas à gauche se trouve une zone d'informations concernant les modifications apportées au contenu de l'onglet en lui-même. Par exemple, lors de la sélection d'un client, le nom de celui-ci se voit écrit dans cette zone. En bas à droite, se situe une zone contenant un ensemble de boutons, permettant de contrôler l'onglet et de le finaliser, c'est à dire d'écrire ses données dans la base.

La troisième et dernière zone contient les informations sur les interactions entre l'utilisateur et la base de données par l'intermédiaire de l'interface graphique. Ainsi, tous les messages de confirmations, mais aussi d'erreurs d'écriture se trouvent écrit ici. Cette zone est très importante dans la compréhension de l'état de la base de données à chaque instant.

Un menu reste cependant inaccessible à l'utilisateur tout pendant qu'il ne clique pas sur le bouton droit de la souris. En effet, un tel clic révèle un menu déroulant permettant la recherche d'informations au sujet des prix des produits ou sur un produit dont nous disposons du code barre ou du numéro de lot en vue de son



retour en service après vente. Ce menu, disponible dans chaque fenêtre du logiciel, permet à l'utilisateur de s'informer sur le stock et le prix d'un certain produit ou d'une catégorie de produits de manière très rapide. La mise en place d'un tel menu est le fruit d'une motivation en ce qui concerne le renseignement rapide d'un client qu'il soit au guichet ou au téléphone.

Le nombre d'interfaces étant assez restreint\*, un schéma d'enchaînement des interfaces nous semble le bienvenu pour conclure cette étude concernant la partie graphique de ce projet.

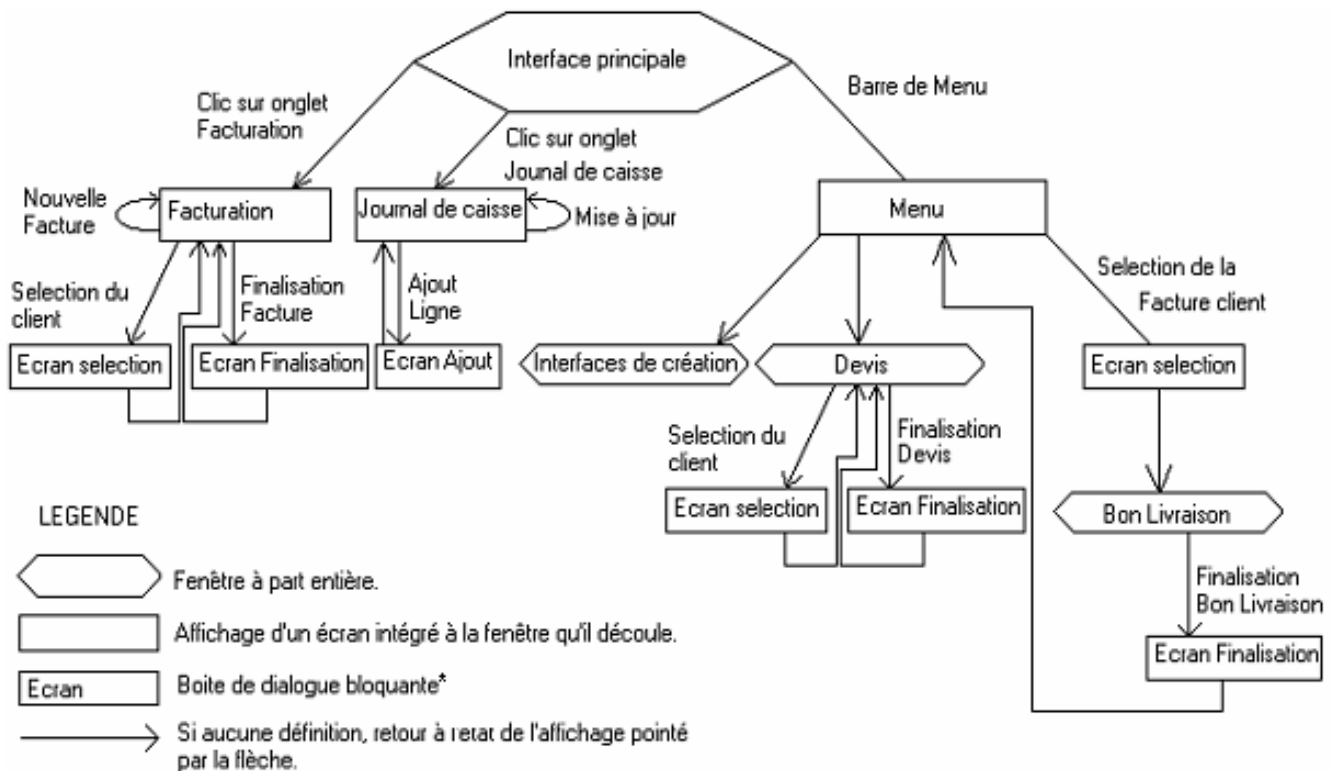


Schéma navigationnel d'interfaces.

Après avoir terminé la phase de conception, il convient maintenant d'appréhender l'étape de réalisation, qui consiste à utiliser notre étude conceptuelle ainsi que les outils informatiques pour créer un logiciel conforme aux besoins et contraintes de ses utilisateurs.

\* **Boîte de dialogue bloquante** : fenêtre obligeant l'utilisateur à une interaction avec elle-même. Les autres fenêtres du logiciel sont bloquées tout le temps que cette fenêtre n'a pas été fermée. Typiquement, ce genre de fenêtre est utilisé pour le questionnement, l'erreur, ou encore l'information.

\* En effet, mis à part l'interface principale, nous disposons d'un ensemble d'interfaces secondaires pour la création d'enregistrement dans la base de données.

\* Ce schéma manque toutefois de précisions, en effet, les interfaces de création n'ont pas été schématisées parce que celles-ci sont assez nombreuses, et bien souvent quasi-identiques. Pour ce qui est des interfaces de consultation et suppression, elles sont identiques, se composant d'une boîte de dialogue bloquantes et exécutant leur processus respectif sur l'objet sélectionné.



## 2.3. Implémentation du programme

### 2.3.1. Implémentation de la base de données

Comme nous l'avons vu dans la première partie, nous avons choisi MySQL comme gestionnaire de base de données. Cet outil met à notre disposition de nombreux éléments simplifiant la création des tables (Annexe I) ainsi que l'enregistrement de données. En effet, en voulant respecter les règles de gestion définies par la méthode Merise, il est nécessaire, pour chaque table créée d'implémenter une clé primaire\*. MySQL permet une gestion automatique des clés primaires, ainsi, celui-ci attribue automatiquement un nombre (identifiant) pour les occurrences de toutes les tables de la base de données sauf la table « FactureFournisseur » car l'identifiant est donné par le fournisseur qui attribue lui-même un numéro à sa facture.

Nous pouvons donc retrouver chaque donnée de la base par son numéro associé. Toutefois, l'utilité de l'identifiant ne réside pas dans la recherche des données le concernant, il permet, plutôt, de ne pas avoir de doublon dans la base de données. Il est beaucoup plus simple, et valorisant pour le client, de chercher ses données à partir de son nom plutôt que par son numéro.

Nous le verrons dans la prochaine partie, cette attribution automatique d'identifiant n'est pas sans poser de problème en ce qui concerne l'établissement des factures.

L'interface graphique ne peut utiliser la base de données directement, et ce, pour des raisons de souplesse de programmation, nous avons donc mis en place un système de classes intermédiaires.

### 2.3.2. Programmation des classes intermédiaires

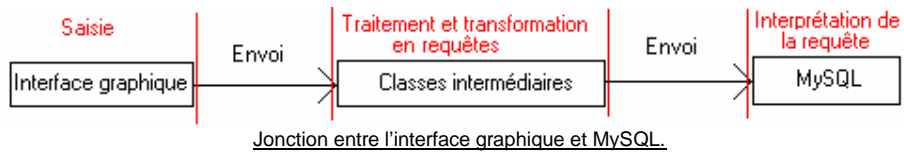
Le but du logiciel étant, d'un point de vue purement informatique, la gestion d'une base de données par une interface graphique, il convient de mettre en place une interface\* entre cette interface graphique et la base de données. Nous avons vu dans la première partie que le pilote MM se charge de la communication avec la base de données. Cette connexion établie, il nous faut enregistrer les données que saisit l'utilisateur. Pour cela, nous avons mis en place un système de classes\* intermédiaires permettant la jonction entre l'interface graphique et MySQL :

---

\* **Clé primaire** : Attribut ou champ particulière de la table telle que, pour chacune des valeurs de attribut, il en existe au maximum une occurrence dans cette table. La clé primaire de la table Client pourrait être, par exemple, le numéro de sécurité sociale, différent pour chaque individu.

\* Nous parlons ici d'une interface, d'un programme qui fait le lien entre la base de données et le logiciel. En aucun cas il ne s'agit d'une interface graphique.

\* **Classe** : Terme informatique, dédié à la programmation orientée objet, désignant un modèle d'objet. La classe « Client », par exemple, se charge de gérer la table « Client » de la base de données. Ainsi lorsque l'utilisateur crée un nouveau client, les données saisies dans l'interface graphique sont envoyées à la classe client qui les traite et les envoie sous forme de requêtes interprétables par MySQL.



Le but des classes intermédiaires, programmées en Java, tout comme l'interface graphique, cependant invisibles aux yeux de l'utilisateur, est de simplifier la programmation en l'allégeant. Les classes sont programmées en fonction de la structure de la base de données. Ainsi, pour chaque table de la base de données, une classe intermédiaire a été créée. Cette classe centralise les actions possibles pour une table. Lorsque l'utilisateur clique sur un des boutons de création, Client par exemple, il n'interagit pas avec la base de données directement. En effet le programme crée un objet\* ayant la structure d'un client, c'est à dire disposant d'un nom, d'une adresse etc. Cet objet est ensuite écrit dans la base de données.

D'une manière générale, le principe du logiciel est le suivant : En fonction du bouton de la barre de menu sur lequel l'utilisateur a cliqué, le programme cible les besoins de celui-ci en s'aiguillant automatiquement sur la partie concernée par le bouton. Par exemple, si l'utilisateur clique sur le bouton « Création de client », le logiciel se tourne automatiquement vers la classe « Client » et utilisera la fonction d'écriture intégrée à cette classe.

#### **2.4. L'interface graphique**

Comme nous l'avons vu dans la précédente partie, notre interface graphique devait ce composer de 3 zones, dont un menu, une zone d'interactions entre l'utilisateur et le logiciel, et une dernière concernant l'interaction entre l'utilisateur, et plus particulièrement le logiciel, et la base de données.

---

\* **Création d'un objet** : D'un point de vue purement informatique, les objets facilitent le passage par paramètres. En effet, plutôt que de passer le nom, l'adresse, le numéro de téléphone le code postal et la ville en paramètres, nous passons un objet de type Client. Ce type d'objet a une structure contenant l'ensemble de ces informations. Par conséquent, nous passons 1 seul paramètre au lieu de 5.





#### 2.4.1. Explication de la mise en forme

En ce qui concerne la barre de menu, nous avons décidé non pas de classer les boutons par actions, mais plutôt de classer les boutons par rapport aux objets de la base de données auxquels ils se rapportent. Ainsi, nous n'avons pas de menu « création » ou « suppression », mais un menu concernant les fournisseurs etc.

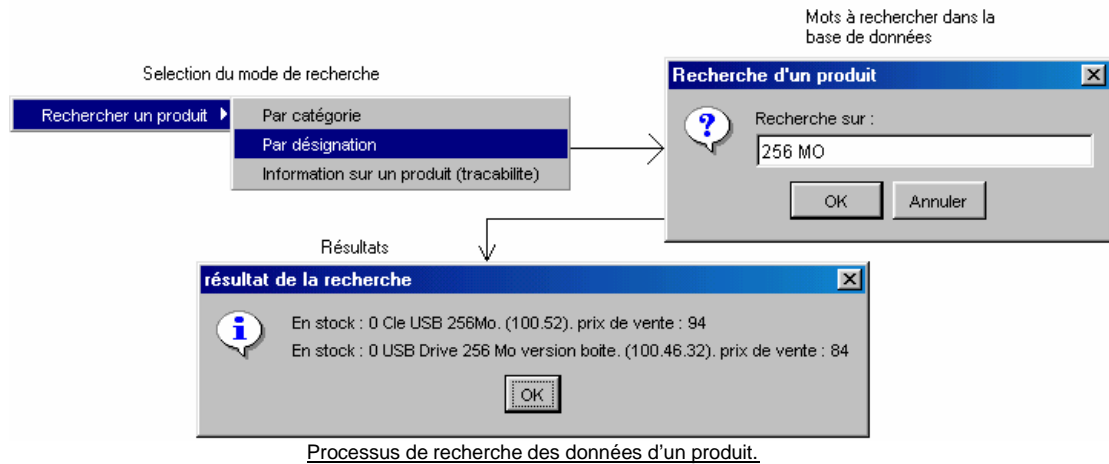
Par rapport au système de boîte à onglets, nous avons décidé d'y implanter les systèmes de facturation et de création de bulletin d'intervention ainsi que le gestionnaire de journal de caisse. N'y figurent pas les devis, qui selon nous, sont moins utilisés, les clients demandant rarement l'établissement de devis, préférant, une énumération des prix des produits désirés. En effet, les devis sont généralement établis pour un ensemble de produits en vue d'un très probable achat.

En ce qui concerne les autres possibilités telle que la création de bons de livraison ou encore d'avoirs, là aussi, les probabilités nous ont conduits à ne pas créer d'onglets à part entière.

Cependant, malgré l'inexistence de certains onglets, le lancement d'un possible processus associé n'est pas forcément plus long. En effet, la mise en place de raccourcis clavier vient palier à ce problème. Ainsi, les actions plus « rares » que celles présentes dans les onglets se voient dotées de raccourcis permettant, d'une simple saisie au clavier, leurs lancements. Nous avons donc 3 degrés d'utilité :

- Les onglets pour les opérations courantes.
- Les raccourcis clavier et la barre de menu pour les opérations de moindre utilité.
- La barre de menu pour les opérations plus rares.

Nous l'avons vu dans la partie précédente, la mise en place d'un menu déroulant permet une interrogation très rapide de la base de données. En effet, d'un simple clic, l'utilisateur peut rechercher les données d'un produit par la saisie complète ou partielle de sa désignation. Lors d'une saisie partielle, le logiciel affiche l'ensemble des produits concernés. Ce petit moteur de recherche accepte même la saisie de caractère d'espacement en vue d'une recherche sur plusieurs mots. L'idée de ce menu est rapidité et efficacité ainsi qu'utilité et intuitivité.



L'interface graphique autorise un bon nombre d'actions, mais en interdit ou n'en permet pas certaines autres.

#### 2.4.2. Limitations de l'interface graphique

L'interface graphique n'autorise pas certaines actions, interdites par la législation. Ainsi, l'utilisateur n'a pas la possibilité de modifier une facture établie. En effet, la loi française l'interdit formellement. Toutefois, celui-ci a la possibilité de l'effacer. Si une telle action se produit, la facture sera belle et bien effacée mais au niveau comptable, un « trou » dans les numéros de facture sera visible, la facture supprimée n'étant pas remplacée dans la base de données\*.

L'interface propose un nombre considérable d'opérations possibles sur la base de données, Nous l'avons cependant vu précédemment, toutes les opérations ne peuvent être implémentées car pour la plupart quasi-inutiles, seules les opérations courantes ont été implémentées. Nous pouvons donc regretter l'inexistence d'un module intégré au logiciel permettant la saisie de requête SQL.

Malgré les limitations de l'interface graphique, le logiciel fourni au terme du stage est fonctionnel. Cependant, au cours du stage, quelques problèmes nous sont apparus, nécessitant une plus grande réflexion. Nous les verrons dans la prochaine partie concernant les problèmes rencontrés ainsi que les limitations et extensions possibles du logiciel.

\* **Numéro de facture** : Prenons l'exemple d'une facture, que l'utilisateur supprime, ayant le numéro X. La prochaine facture aura le numéro X+1 malgré tout. Le cabinet comptable et aussi, pourquoi pas, le contrôleur fiscal aura une trace des factures numéro X-2, X-1 et X+1, et remarquera par conséquent qu'il manque la facture numéro X.



### 3. Avancement du projet

---

#### 3.1. Problèmes rencontrés

Comme nous l'avons vu dans la première partie, l'implémentation a été relativement courte. Nous avons donc pu mettre en place une phase de test assez rapidement ce qui a permis de corriger en heure et en temps les problèmes d'implémentation. Cependant, le principal problème lors de la réalisation d'un tel projet, et qui ne peut être résolu par une phase de test, est de se projeter dans le futur et d'imaginer le comportement du logiciel dans plusieurs mois. Malgré cette réflexion, nous ne pouvons imaginer que les problèmes les plus simples, laissant, du même coup, de côté des problèmes bien plus importants dont nous ignorons la nature et même l'existence.

Le plus gros problème rencontré concerne la centralisation des données. En effet, à l'origine, les potentialités du driver MM nous étaient méconnues. Cependant après avoir appréhendé ce pilote, il nous est apparu possible de connecter une machine à la base de données d'un « serveur ». La simplicité de mise en œuvre d'une application client-serveur décuplait les potentialités du logiciel en lui-même. En effet, les perspectives qu'offre une application client-serveur sont multiples. Plus besoin d'attendre la finalisation d'une facture pour en créer une nouvelle, si le logiciel est installé sur  $n$  postes,  $n$  opérations peuvent être effectuées en même temps sur la même base de données, pourvu que ces opérations soient atomiques\*.

Un problème a été soulevé par la mise en place de l'utilisation en réseau. En effet, si la réutilisation des informations écrites dans la base de données est simple, il n'en est rien pour les données générées sous forme de fichier, comme, tous les documents imprimables. En effet, un programme « client » génère son propre fichier sans pour autant que les autres clients ne puissent le consulter.

Prenons l'exemple d'une facture : Un poste « client » enregistre une facture. Lors de cet enregistrement, les données sont écrites dans la base de données et un fichier HTML représentant la facture est généré sur ce même poste. Imaginons qu'un autre poste « client » désire visionner cette facture. En consultant la base de données, il apprend qu'elle existe bel et bien, mais aucun fichier HTML ne se rapporte à la dite facture car les fichiers des postes « clients » sont inaccessibles.

Dans un premier temps, pour pallier à ce problème, l'interface était pourvue d'un bouton permettant de re-générer une facture à partir de son enregistrement dans la base de données. Nous avons, par la suite, préféré centraliser toutes les données que le logiciel pouvait générer. Finalement, chaque poste « client », générant une facture, crée un fichier HTML imprimable sur le « serveur », ce qui le rend consultable par les autres postes du réseau.

Tous les enregistrements étant maintenant regroupés sur un même poste, la sauvegarde de toutes les données est du même coup rendue plus aisée.

---

\* Une opération est dite **atomique** si elle ne peut être interrompue durant son exécution par une autre opération.



Dans un second temps, il a fallu appréhender le problème de la comptabilité. En effet, il était convenu dans le cahier des charges que le cabinet comptable devait recevoir sous forme de mails l'ensemble des transactions du magasin. Plusieurs solutions s'offraient à nous. La première consistait à utiliser un programme d'envoi de mails par ligne de commande qui serait lancé par le logiciel. Le problème de cette solution concerne la licence d'utilisation du client mail, qui, dans la plupart des cas, est payante. La seconde était d'utiliser un programme Java connu sous le nom de JavaMail. Nous avons retenu cette solution car, d'une part, ce programme est libre de droit et, d'autre part, son intégration au logiciel est parfaite par le fait que tous deux soient programmés en Java.

Après avoir mis en place un système d'envoi de mails, le problème suivant consistait à ajouter des « pièces jointes » aux mails. En effet, le cabinet comptable doit avoir une trace de chaque facture établie. Cependant, JavaMail permet d'envoyer un fichier joint par mail uniquement et la question d'envoyer un mail par facture établie ne se posait même pas. La solution a été de créer un fichier d'archives (connu sous le nom de fichier ZIP) quotidiennement et de le joindre à l'envoi du mail. Il nous a donc fallu appréhender la création de fichier « ZIP » en Java.

Lors de la phase de conception nous avons omis la gestion des avoirs pour une meilleure comptabilité. Cependant, aucune modification n'a été apportée sur le programme existant, seules deux tables ont été ajoutées dans la base de données et une interface graphique permettant l'enregistrement de données dans ces tables a été créée.

Si la résolution de tels problèmes nous a demandé un peu de temps, cela ne nous a pas empêché de fournir un logiciel fonctionnel avant même la fin du stage.

### **3.2. Avancement du projet**

Comme nous l'avons vu dans les paragraphes précédents, la phase d'implémentation ne nous a pas pris trop de temps malgré les problèmes cités. Nous avons pu fournir un logiciel conforme aux besoins signalés dans le cahier des charges. Cependant certains éléments agrémentant l'intuitivité n'ont pas été finalisés, c'est le cas par exemple du fichier d'aide.

L'écriture d'un fichier d'aide nécessite dans un premier temps la mise en place d'une petite charte graphique permettant la compréhension et la clarté des informations produites. Dans un second temps, il s'agit d'écrire une documentation claire et complète concernant l'utilisation et la configuration du logiciel. Vis à vis du temps restant lors du stage, nous n'avons pu fournir une documentation utilisateur et un fichier d'aide exhaustif.

L'installation du logiciel est rendue très difficile car aucun programme d'installation n'a été implémenté. Tout est donc manuel et, par la même occasion, très délicat car la moindre erreur de configuration rend le logiciel inexploitable. La phase de test ayant révélé des problèmes plus importants, la création d'un programme d'installation a été sans cesse reculée jusqu'à l'échéance du stage.



Au fil du test, il est apparu que le cahier des charges ne proposait pas forcément une liste exhaustive des fonctionnalités dont devrait disposer le logiciel. Celui-ci s'est donc vu rajouter bon nombre de fonctions permettant l'automatisation de certaines tâches d'une part, et d'autre part de rendre le logiciel plus adapté aux utilisateurs.

### 3.3. Fonctionnalités ajoutées

Les fonctionnalités sont le fruit d'une demande des utilisateurs au cours de la phase de test. Bien que moins utiles que les fonctionnalités décrites dans le cahier des charges, celles-ci ajoutent à l'utilité du logiciel en lui-même. L'ensemble de ces fonctionnalités agit en tout point du logiciel.

Parmi elles, l'ajout de l'onglet « Bulletin d'intervention » permet de gérer les interventions des techniciens sur les ordinateurs des clients du magasin. Plus qu'une optimisation en ce qui concerne le temps de création d'un bulletin, un tel onglet permet de garder une trace de toute intervention en vue d'une consultation future.

Création sav

Choix de l'intervenant

Geoffrey  Lionel  Comptoir  François

Problème

Problème de disque dur

Info bulletin

Intervention

Imprimante  Unite centrale

Portable  Ecran

Scanner  Modem

Cable  Autre

Materiel déposé

Panne sous garantie

Devis a faire avant depannage

Forfait depannage 40E

Autre

Client Nouveau Client

Nouveau Bulletin Generer le SAV

Selection du client par recherche

bernard

Client selectionné : BERNARD Olivier

Interface de création d'un bulletin d'intervention



Une routine, quant à elle, permet d'alerter l'utilisateur lorsque celui-ci vend un produit dont la quantité en stock est passé sous un seuil défini lors du paramétrage. La recherche ne se fait pas sur un seul produit mais sur toute une catégorie de produit. L'utilisation typique de cette alerte concerne l'ensemble des consommables. Ainsi, l'utilisateur se voit prévenu du manque de stock.

En plus de ces fonctionnalités, nous avons créé un programme autonome permettant la création et l'envoi de fichiers zip à la comptabilité précédemment citée. Ce programme se connecte à la même base de données, et créer un fichier archive en conséquence des factures créées la même journée.

Malgré ces ajouts, et même si le logiciel est bien adapté au besoin du magasin, certains points limitent l'efficacité du logiciel pour quelques tâches. Nous allons maintenant parler des limitations du logiciel.

### **3.4. Limitations**

A ce niveau de développement et même si le logiciel est fonctionnel, il n'offre pas une parfaite satisfaction. En effet, malgré une rédaction de cahier des charges en adéquation avec les besoins de ses futurs utilisateurs, le logiciel est tout de même limité, et ce, sur plusieurs points. Plusieurs raisons expliquent de telles limitations, la première concerne la durée du stage, un peu courte pour le développement d'un logiciel et la correction des problèmes décelés lors de sa phase de test. La seconde concerne les outils informatiques utilisés qui, bien qu'adéquats, automatisent certaines tâches, les rendant, du même coup, inaccessibles.

La première limitation concerne la base de données en elle-même. L'administration de celle-ci fait défaut car elle reste accessible à tous les utilisateurs. En effet, à ce jour, aucun mot de passe n'est demandé lors de la connexion à la base de données, par conséquent, tout utilisateur peut se connecter à cette base et la modifier à son propre gré. Un travail sur l'administration de MySQL permettrait simplement d'en sécuriser l'accès. En ce qui concerne le logiciel, aucun travail n'est à apporter, le problème ayant déjà été prévu. En effet, les noms et mots de passe sont modifiables dans le fichier de configuration. La connexion se fait donc indépendamment du programme compilé.

La mise en place d'un système de droits sur la base de données pourrait, à l'avenir, être très utile pour des extensions possibles du logiciel.

Nous l'avons vu dans l'introduction de cette partie, certains outils informatiques couramment utilisés lors du stage automatisent des tâches, c'est le cas de MySQL. Lors de la création des tables, la spécification de certains attributs permettent leur gestion automatique, c'est le cas, par exemple, de la clé primaire d'une table. Si celle-ci est spécifiée « AUTO\_INCREMENT », MySQL attribue une valeur automatiquement à chaque création de ligne. Un problème de gestion nous



est apparu lorsqu'un des utilisateurs a voulu supprimer\*, de la base de données, la dernière facture créée. Lors de la création d'une nouvelle facture, celle-ci ne se voyait pas attribuer le numéro de clé de la facture supprimée mais le numéro suivant. Au fil du temps et des suppressions, les numéros de factures ne se suivront plus, des « trous » apparaîtront.

Lors du bilan comptable de fin d'année ou lors d'un contrôle fiscal, le magasin aura le plus grand mal à se justifier vis à vis de ces numéros de factures inexistantes. A ce jour, le problème reste entier, aucune solution permettant la suppression de facture erronée, n'a été trouvée, mis à part, bien sur, la création d'avoirs ce qui allonge considérablement le temps de saisie d'une facture.

La comptabilité intégrée au logiciel est, de nouveau, concernée par un autre problème. Quotidiennement, celui-ci envoie un e-mail au cabinet comptable. Ce mail contient un fichier HTML, représentant le journal de caisse, et un fichier zip, contenant l'ensemble des factures du jour. Cet envoi de mail n'est pas sécurisé ce qui offre, à des personnes malveillantes, la possibilité de connaître l'ensemble des transactions du magasin. Une solution consisterait à envoyer les e-mails de manière sécurisée en utilisant un protocole adéquat.

Concernant les interventions sur la base de données, celles-ci nécessitent la connaissance des commandes MySQL. En effet, le logiciel ne propose qu'un ensemble de boutons traduisant certaines de ces commandes, néanmoins, cet ensemble est très restreint et se cantonne à des insertions ou des suppressions dans les tables. Certaines modifications sont possibles, c'est le cas d'un devis par exemple, d'autres impossibles et nécessitent l'intervention, en ligne de commandes, à partir d'une console MySQL.

Aux vues du nombre de possibilités de modifications, suppressions ou insertions, il nous était impossible de rendre l'ensemble de commandes pré interprétées exhaustif auquel cas il serait inapproprié.

### **3.5. Extensions**

L'utilisation de MySQL pour la gestion de la base de données ouvre les portes de l'utilisation sur Internet, par le biais de PHP. La mise en place de devis en ligne permettrait aux personnes connectées de connaître le prix des composants en temps réel et d'établir leur propre devis de chez eux.

Cependant, la mise en place d'un tel service soulève plusieurs problèmes. Le premier concerne la base de données en elle-même qui devrait d'être stockée sur un serveur, dans le magasin\*, ayant une adresse IP\* fixe pour pouvoir être consultée

---

\* **Suppression d'une facture** : Normalement interdit, le logiciel en offre tout de même la possibilité pour effacer une facture erronée. La solution adéquate est de créer un avoir du montant de la facture erronée et de le reporter sur la nouvelle facture.

\* Le serveur doit obligatoirement être dans le magasin pour permettre une connexion illimitée à la base de données. En effet, l'hébergement de la base pourrait conduire à une inactivité lors d'une panne d'ADSL par exemple, rendant la facturation ou toute autre opération impossible.



depuis Internet. De plus, celle-ci devrait se voir dotée d'un système de droit adéquat ne permettant que la consultation des données et non leurs modifications.

Une telle extension permettrait aussi de lancer le logiciel à travers Internet permettant aux administrateurs de gérer la base de données depuis n'importe quel poste connecté pourvu que le logiciel y soit installé.

Nous l'avons vu dans la partie précédente, l'envoi de données par e-mail au cabinet comptable n'est pas sécurisé. Une solution plus appropriée consisterait à étudier le logiciel de comptabilité utilisé et d'adapter notre logiciel pour que celui-ci envoie directement des données interprétables par le logiciel comptable. Cela automatiserait la tâche de saisie comptable et rendrait incompréhensible les données envoyées pour quiconque ne disposant pas de ce logiciel.

Une telle étude nécessiterait beaucoup de temps, mais rendrait les deux logiciels solidaires.

En ce qui concerne l'utilisation\* du logiciel dans un autre établissement, elle est tout à fait envisageable pourvu que ce magasin ait les mêmes besoins\* que l'Univers Informatique. En effet, ayant les mêmes besoins, la base de données et l'ensemble des tables seraient identiques. L'interface graphique n'étant qu'une couche permettant l'utilisation simplifiée de cette base de données, aucun élément ne serait sujet à modification.

Nous venons de voir les résultats obtenus à la suite de ce stage, à travers les parties concernant l'avancement, les fonctionnalités ajoutées, les limitations et les extensions du logiciel. Il nous reste à conclure sur l'apport d'un tel stage, tant au point de vue des connaissances qu'au point de vue humain, dans un cursus universitaire, à travers la conclusion.

---

\* **Adresse IP** : ensemble de 4 numéros à 3 chiffres permettant de désigner un ordinateur sur Internet.(Exemple : 192.168.0.55)

\* A propos de cette **utilisation**, Le logiciel ne peut se voir vendu, mais donné car sa vente nécessiterait l'achat d'une licence MySQL.

\* Lorsque nous parlons de **besoins**, nous ne parlons pas forcément d'un autre magasin d'informatique. En effet, les éléments contenus dans les tables peuvent ne pas être d'origine informatique. Ainsi, en reprenant la table Produit, on se rend compte que tout article disposant d'une catégorie, d'une désignation, d'un prix d'achat et d'un prix de vente peut se voir intégré dans la base de données. Or, généralement, tous les articles vendus dans un quelconque magasin dispose de ces renseignements. Par conséquent, la table Produit est adaptée, et donc utilisable dans toutes sortes de magasin. Le logiciel est donc utilisable dans un magasin si toutes ses tables sont adaptées.



## Conclusion

---

Après avoir présenté les auteurs du sujet de ce stage, Monsieur Cosson et ses techniciens, nous venons de voir comment, conceptuellement parlant ainsi que du point de vue du développement nous avons répondu à leurs attentes.

Les objectifs de tels stages sont multiples : Il faut tirer une double expérience (communication omniprésente avec les responsables du sujet et acquisition de nouvelles connaissances dans le domaine ciblé par le stage) et que nous puissions apporter à l'entreprise pour laquelle nous travaillons un bénéfice sous la forme de nouveaux logiciels.

La durée de ce stage, 9 semaines à temps plein, se révèle être une expérience dans le monde du travail très bénéfique. En ce qui me concerne, cela représente, à ce jour, ma plus longue insertion dans le monde du travail, et plus particulièrement du génie logiciel. J'ai réellement pu dépasser le stade de « formation » qui est une étape obligatoire lorsque l'on arrive dans une nouvelle entreprise. Ensuite, le stage devient une vraie expérience professionnelle puisque j'ai été amené à travailler de manière autonome.

Ce stage a répondu à mes attentes aussi bien au niveau professionnel qu'en terme de compétences relatives au savoir-faire nécessaire pour toute autre activité relationnelle. Il est mon premier pas vers un apprentissage de la réalisation complète d'une application pour une entreprise en ayant besoin et ce, grâce à la communication entre elle et nous.

---



## Bibliographie

---

Dans le cadre du stage de cursus licence – maîtrise, il m'a été demandé de développer une application permettant la gestion complète d'un magasin de vente de matériel informatique. Afin de mieux aborder ce sujet, j'ai dû faire appel à certains documents.

### Livres

Java 2 édition SDK 1.3

Livre traitant des principes du langage Java

ISBN 2-7464-0105-3 OSMAN Eyrolles multimédia

### Site Internet

Tutorial Java (consulté le 1<sup>er</sup> juillet 2003) [En ligne]

Adresse URL: <http://java.sun.com/docs/books/tutorials>

### Documentation électronique

Documentation MySQL 3.23. Manuel de référence.

Bruce Eckel. Thinking In Java

Live électronique traitant du langage Java

ISBN 0-13-659723-8



## Résumé

Un stage annuel entre dans le cadre du cursus licence - maîtrise informatique. Sur une demande du magasin l'Univers Informatique, celui-ci portait sur la création d'un logiciel permettant la gestion complète du magasin, de l'enregistrement de factures fournisseur à la création de factures client, en passant par une gestion du stock. Les outils utilisés sont Java, pour la programmation, et MySQL pour le stockage des données.

## Mots clés

Gestion complète de magasin  
Stage de Licence - maîtrise  
Univers Informatique  
Java  
MySQL