الجمهورية الجزائرية الديموقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Minstère de l'Enseignement Supérieur et de la Recherche Scientifique

CENTRE UNIVERSITAIRE DE MILA
INSTITUT DES SCIENCES ET DE LA TECHNOLOGIE

Réf. ***/12***

Mémoire de fin d'étude
Présenté pour l'obtention du diplôme de

Licence Académique

Domaine : **Mathématiques et Informatique**
Filière : **Informatique**

Thème

Conception et développement d'une application de gestion d'une pharmacie

Présenté par :
1-Belmehboul Faycal.
2-Triha Zakaria.

Dirigé par :
-Khalfi Souheila.

Année universitaire 2011-2012

Remerciement

C'est avec l'aide de Dieu qu'a vu les jours ce présent travail.

Ensuite, il n'aurait pas pu être achevé sans le soutien, les conseils, les encouragements de certaines personnes aux quelles nous tenons ici à exprimer nos sincères remerciements.

En première lieu nous exprimons toute notre gratitude pour notre encadreur Melle khalfi souheila pour ses précieux conseils, ses disponibilités, la confiance qu'elle nous a toujours témoigné et la sollicitude dont elle nous a entouré, et ce tout au long de l'élaboration du présent travail.

Nous n'oublions pas la pharmacienne Mme ZEROUKI NORA pour tous les conseils et l'aide qu'il n'a cessé de nous prodiguer.

ET non plus nos enseignants qui tout au long du cycle d'étude au centre universitaire de Mila, nous ont transmis leur savoir.

Nous tenons enfin à remercier tous ceux qui ont collaborés de prés ou de loin à l'élaboration de ce travail. Qu'ils acceptent nos humbles remerciements.

ZAKARIA & FAYCAL

Dédicaces

Je dédie ce modeste travail à :

Mes très chers parents, pour leur patience, leur soutient et leur confiance.

Ma mère Nora qui m'est la plus chère au monde.

Mon père Tahar qui s'est sacrifié afin que rien n'entrave le déroulement de mes études.

A ma grande familles surtout : mon cher frère Ayoube, mon cher oncle Hassan.

A Mon binôme Fayçal et toute sa famille.

A toutes mes amis surtout : Dhayae , Nouri , Nassim ,Amine, Daas , Radouane , Fouad , Halim , Toufik , hamza , Moussa , Dia-elhak .

A tous mes amis informatiques surtout : Hayete, Imane, besma.

A tous mes amies en particulier : khawla, Mona, rokia, djanet, rababe, Sara.

A tous ceux que j'aime tant et que je n'ai pas cités ;

Je dédie ce mémoire …

Triha Zakaria

Dédicaces

Je dédie ce modeste travail à :

Mes très chers parents, pour leur patience, leur soutient et leur confiance.

Ma mère qui m'est la plus chère au monde.

Mon père qui s'est sacrifié afin que rien n'entrave le déroulement de mes études.

A tout mes frères et sœurs.

A tout ma famille.

A Mon binôme zaki et toute sa famille.

A toutes mes amis surtout : Amine, Nassim , abderrahmenne, Radouane , Fouad , Nouri ,Halim , Toufik , hamza , Moussa , Dia-elhak .

A tous ceux que j'aime tant et que je n'ai pas cités ;

Je dédie ce mémoire …

Fayçal Belmehboul.

Table des matières

Chapitre IV: Implémentation

Liste des figures

Liste des tableaux

1. Contexte de travail

La place déterminante de l'informatique et son rôle critique dans le fonctionnement des organisations n'est plus à démonter aujourd'hui. Elle se développe de jour en jour afin de nous faciliter la tâche dans tous les domaines d'activités. Les organisations souhaitent avoir des systèmes d'information plus efficaces, flexibles et adaptés pour traiter les problèmes informationnels et les assister dans leurs actions aussi bien quotidienne qu'a plus long terme.

L'administration d'une officine est l'une des responsabilités qui a besoin d'une meilleure gestion des différents traitements exigés par cette activité. C'est une lourde tâche pour les personnes qui en ont la charge. Son objectif principal est d'assurerez la sécurité et le bien-être des malades. Parmi ses activités essentielles se situe la gestion du stock. Le contrôle du stock doit être régulier pour permettre une bonne maîtrise de gestion, minimiser les pertes et situer les responsabilités en cas de problèmes. Les contrôles seront exercés tant sur la qualité des médicaments (dates de péremption, conditions de conservation, état des médicaments) que sur les quantités détenues en stock (disponibilité, suffisante et exactitude des stocks). Ce qui justifie la nécessité d'être assisté par un système d'information.

Un système de gestion de stock doit assurer une gestion rationnelle de stock au niveau d'un dépôt pharmaceutique. Il permet de gérer en temps réel le mouvement des produits.

2. Problématique et motivation

Pendant notre stage au niveau de la pharmacie, nous avons constaté que la gestion du stock est une tâche difficile pour le pharmacien du fait qu'elle est effectuée manuellement. Ce qui engendre un certain nombre de problèmes entre-autre :

- La difficulté de suivre le mouvement des produits pharmaceutiques stockés,
- La lenteur dans l'accès aux informations en temps réel parce que ces dernières sont stockées dans des documents,
- La vérification des produits en stock alerte est effectuéé manuellement,
- l'oublie de se débarrasser des produits périmés qui ont un impact direct sur le bien-être des patients.
- ... etc.

Dans le but d'alléger l'activité du pharmacien et d'éviter tous ces problèmes, nous allons construire un système d'information pour la gestion de la pharmacie.

3. Objectif de travail

Notre objectif consiste alors à la conception et la réalisation d'un système d'information de gestion d'une officine permettant de répondre aux besoins du pharmacien. Il prend en charge la gestion des produits dés leur réception depuis les fournisseurs à leur livraison aux clients. Précisément, il permet de :

- Connaître à chaque instant l'état du stock ;
- Connaître la consommation de la pharmacie pour chaque médicament ;
- Être au courant des produits périmés ;
- Éviter les ruptures de stock.

De plus, ce système doit être simple, utile, performant, Ergonomique et fiable.

Pour atteindre cet objectif, nous avons utilisé le langage UML (Unifier Modeling Language) dans la phase de conception. Pour l'implémentation, notre choix s'est porté sur l'environnement de développement DELPHI et la base de données est implémentée avec ACCESS.

4. Organisation du mémoire

Ce mémoire est structuré de la manière suivante :

- *Le **chapitre I*** s'intéresse à présenter le langage de modélisation UML et à décrire la démarche opté pour le développement de l'application.
- *Le **chapitre II*** est dédié à présenter la pharmacie dans lequel nous avons effectué notre stage. Il consiste aussi à étudier les différents postes de travail et les divers documents manipulés au sein de la pharmacie, pour y'arriver enfin à construire un cahier de charge. Ce dernier sert de base pour la conception et la réalisation du système.
- *Le **chapitre III*** est consacré à l'analyse de l'étude de cas.
- *Le **chapitre IV*** sert à présenter les outils dont nous nous sommes servis pour le développement de l'application ainsi qu'une brève description de quelques interfaces de cette dernière.

_ Ce modeste mémoire s'achève par **une conclusion générale** en présentant un récapitulatif de tout ce que nous avons réalisé tout en planifiant les perspectives que nous envisageons pour compléter ce travail.

Chapitre I
Le langage de modélisation unifie (UML)

1.1. Introduction

Ce chapitre présente les concepts de base du langage de modélisation UML, pour permettre une bonne lecture de ce qui va suivre et exprimer de manière uniforme l'analyse, la conception et la réalisation d'une application informatique. Par la suite nous abordons les notions d'**UML** (Unified Modeling Langages) avec ces diagrammes.

1.2. Le langage de modélisation unifié (UML)

UML est une notion graphique conçue pour représenter, spécifier, construire et documenter les systèmes logiciels. Ses deux principaux objectifs sont la modélisation de système utilisant les techniques de l'orienté objet, depuis la conception jusqu'à la maintenance, et la création du langage abstrait compréhensible par l'homme et intraitable par les machines. Il permet de construire plusieurs model d'un système, chacun d'eux met en valeur des aspects différentes : fonctionnels, statiques, dynamiques, organisationnels. UML est devenu un langage incontournable dans les projets de développement.

1.2.1. Définition et historique d'UML

UML (Unified Modeling Langage) est le langage de modalisation unifié qui est le résultat d'une opération d'unification d'un ensemble de concepts pris des méthodes oriontées objet dans le but de modéliser d'une façon claire et précise la structure et le comportement d'un système indépendamment de toute méthode et tout langage de programmation.

UML a été accepté comme standard industriel en novembre 1997 par le groupe OMG sous le patronat de trois méthodologues oriontées objet expérimentes : J.Raumbaugh, I.Jacobson et G.Booch. La modélisation orientées objet dans tous les domaines est aujourd'hui dominée par UML.

1.2.2. Les avantages de UML

- Il permet de représenter l'aspect traitement du système aussi bien que l'aspect données ;
- Il n'est nullement réservé à l'orientation objets, mais se prête également à l'analyse traditionnelle ou à une approche mixte; il permet éventuellement une transition en douceur vers l'orientation objets;
- Il remplace toutes les notations existantes, y compris Yourdon (Diagrammes de flux de données et entités - associations, etc.), Merise et autres;

- Il représente un standard de modélisation, une notation: il s'agit donc d'un outil et non d'une méthode; chacun peut utiliser la méthode la plus appropriée à ses besoins (processus, techniques, outils);
- Il s'utilise sur l'ensemble du cycle de développement de logiciels (analyse, conception, réalisation, tests, documentation);
- Il se prête à la construction de nouvelles applications, à la documentation d'applications existantes et à la définition de cahiers des charges pour la sélection de solutions du marché;

- Les principaux outils de modélisation permettent aujourd'hui de travailler avec cette notation;

1.3. Les Diagrammes UML

UML dans sa version 2 s'articule autour de treize diagrammes, chacun d'entre eux est appliqué à la représentation d'un système logiciel suivant un point de vue particulier. Ces diagrammes sont regroupés dans deux grands ensembles: les diagrammes structurels et les diagrammes de comportement.

➢ **Diagrammes structurels ou diagrammes statiques (UML Structure)**

_ diagramme de classes (Class diagram)
_ diagramme d'objets (Object diagram)
_ diagramme de composants (Component diagram)
_ diagramme de déploiement (Deploymentdiagram)
_ diagramme de paquetages (Package diagram)
_ diagramme de structures composites (Composite structure diagram)

➢ **Diagrammes comportementaux ou diagrammes dynamiques (UML Behavior)**

_ diagramme de cas d'utilisation (Use case diagram)
_ diagramme d'activités (Activitydiagram)
_ diagramme d'états-transitions (State machine diagram)

➢ **Diagrammes d'interaction (Interaction diagram)**

_ diagramme de séquence (Sequencediagram)
_ diagramme de communication (Communication diagram)
_ diagramme global d'interaction (Interaction overviewdiagram)

_ diagramme de temps (Timing diagram) maitrise

Ces diagrammes, d'une utilité variable selon les cas, ne sont pas nécessairement tous produits à l'occasion d'une modélisation. Les plus utiles pour la maîtrise d'ouvrage sont les diagrammes d'activités, de cas d'utilisation, de classes, d'objets, de séquence et d'états-transitions

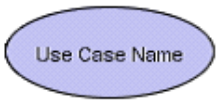
1.3.1 Le diagramme de cas d'utilisation :

Les diagrammes des cas d'utilisation identifient les fonctionnalités fournies par le système (cas d'utilisation), les utilisateurs qui interagissent avec le système (acteurs), et les interactions entre ces derniers, c'est en effet une vue statique de l'utilisation d'un système.

Les cas d'utilisation sont utilisés dans la phase d'analyse pour définir les besoins de "haut niveau" du système. Les objectifs principaux des diagrammes des cas d'utilisation sont:

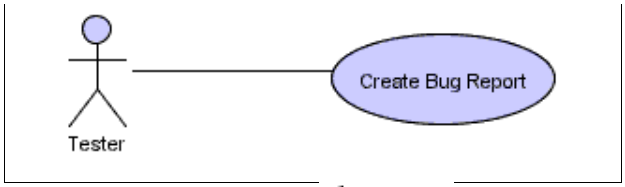
- fournir une vue de haut-niveau de ce que fait le système
- Identifier les utilisateurs ("acteurs") du système
- Déterminer des secteurs nécessitant des interfaces homme-machine(IHM).

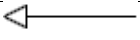
Acteur : 

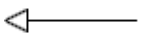
Cas d'utilisation : 

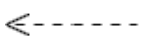
Association :

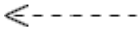
L'image suivante montre comment ces trois éléments de base collaborent pour former un diagramme de cas d'utilisation:



Généralisation : 

Héritage : 

Inclue : 

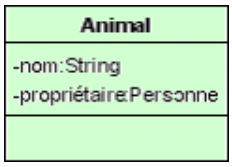
Extends : 

1.3.2 Le diagramme de classes :

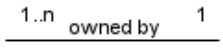
Le diagramme des classes identifie la structure de la classe d'un système, y compris les propriétés et les méthodes de chaque classe. Les diverses relations, telles que la relation d'héritage par exemple

Il représente un ensemble de classes, d'interfaces et de collaboration ainsi que leurs relations. Les diagrammes de classes représentent la vue statique d'un système.

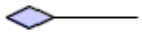
Classes :



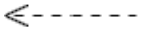
Association :



Composition :

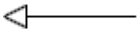


Dépendance :

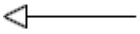


Agrégation :

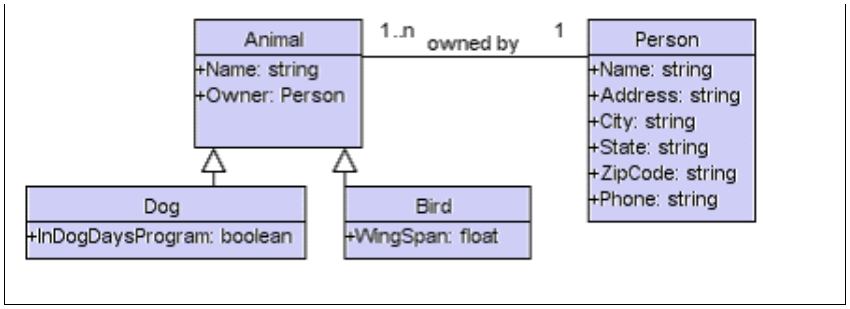
Généralisation :



Héritage :



Considérons l'exemple d'un système vétérinaire. Des animaux de compagnie, comme des chiens ou des oiseaux, sont suivis par leurs propriétaires. Le diagramme suivant modélise une solution potentielle. Considérant que les chiens comme les oiseaux sont "un genre" d'animal, nous utilisons une relation de généralisation.



1.3.3 Le diagramme de séquences:

Les diagrammes des séquences documentent les interactions à mettre en œuvre entre les classes pour réaliser un résultat, ces communications entre les classes sont reconnues comme des messages. Le diagramme des séquences énumère des objets horizontalement, et le temps verticalement. Il modélise l'exécution des différents messages en fonction du temps, donnant

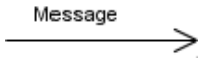
ainsi une vue dynamique sur le système. Les diagrammes de séquence mettent l'accent sur le classement chronologique des messages

Objet : 

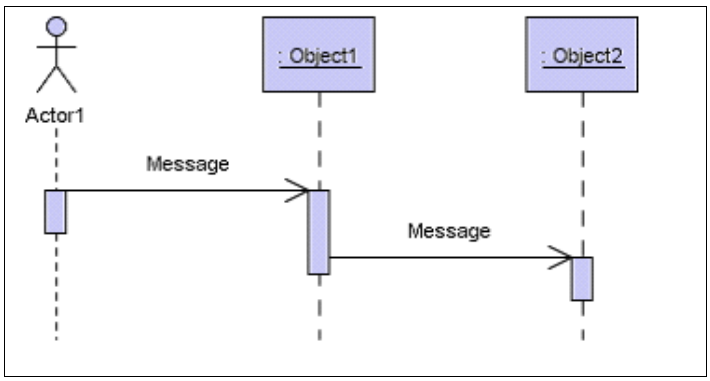
Acteur : 

Ligne de vie :

Activation : 

Message : 

L'exemple ci-dessous représente un diagramme des séquences qui utilise des objets par défaut (aucun nom n'est spécifié)



Les diagrammes de composants, de déploiement et de communication sont surtout utiles pour la maîtrise d'oeuvre à qui ils permettent de formaliser les contraintes de la réalisation et la solution technique.

1.3.4 Diagramme d'objets

Un diagramme d'objets représente des objets (*i.e.* instances de classes) et leurs liens (*i.e.*instances de relations) pour donner une vue figée de l'état d'un système à un instant donné. Un diagramme d'objets peut être utilisé pour :

• illustrer le modèle de classes en montrant un exemple qui explique le modèle ;

• préciser certains aspects du système en mettant en évidence des détails invisibles dans le

diagramme de classes ;

• exprimer une exception en modélisant des cas particuliers ou des connaissances non généralisables qui ne sont pas modélisés dans un diagramme de classe ; prendre une image (*snapshot*) d'un système à un moment donné.

1.3.5 Diagramme d'états-transitions

Les diagrammes d'états-transitions d'UML décrivent le comportement interne d'un objet à l'aide d'un automate à états finis. Ils présentent les séquences possibles d'états et d'actions qu'une instance de classe peut traiter au cours de son cycle de vie en réaction à des événements discrets.

Un diagramme d'états-transitions rassemble et organise les états et les transitions d'un classeur donné.

-État : Un état représente une période dans la vie d'un objet pendant laquelle ce dernier attend un événement ou accomplit une activité, se représente graphiquement dans un diagramme d'étatstransitions par un rectangles aux coins arrondis.

_ **État initial :** L'état initial est un pseudo état qui indique l'état de départ, par défaut, lorsque le diagramme d'états-transitions, ou l'état enveloppant, est invoqué. Lorsqu'un objet est créé, il entre dans l'état initial.

_ **État final :** L'état final est un pseudo état qui indique que le diagramme d'états. Transitions, ou l'état enveloppant, est terminé.

-**Événement :** Un événement de changement est généré par la satisfaction (i.e. passage de faux à vrai) d'une expression booléenne sur des valeurs d'attributs.

-**Transition :** Une transition définit la réponse d'un objet à l'occurrence d'un événement. Elle lie, généralement, deux états E1 et E2 et indique qu'un objet dans un état E1 peut entrer dans l'état E2 et exécuter certaines activités, si un événement déclencheur se produit et que la condition de garde est vérifiée.

-Condition de garde : La condition de garde est évaluée uniquement lorsque l'événement déclencheur se produit. Si l'expression est fausse à ce moment-là, la transition ne se déclenche pas, si elle est vraie, la transition se déclenche et ses effets se produisent.

1.3.6 Diagramme d'activités

Les diagrammes d'activités permettent de spécifier des traitements très proche des langages de programmation objet : spécifier des actions de base, structure de contrôle (condition, boucle), Ils sont donc bien adapter à la spécification détaillée enchainements d'actions de haut niveau, en particulier pour la description détaille des cas d'utilisation.

-Activité : Une activité définit un comportement décrit par un séquencement organisé d'unités dont les éléments simples sont les actions. Une activité est un comportement (behavior en anglais) et à ce titre peut être associée à des paramètres. De la gauche vers la droite, on trouve: le noeud représentant une action, un noeud objet, un noeud de décision ou de fusion, un noeud de bifurcation ou d'union, un noeud initial, un noeud final et un noeud final de flot.

-Transition : Le passage d'une activité vers une autre est matérialisé par une transition. Graphiquement les transitions sont représentées par des flèches en traits pleins qui connectent les activités entre elles. Les transitions spécifient l'enchaînement des traitements et définissent le flot de contrôle.

-La synchronisation Les flots de contrôle parallèles sont séparés ou réunis par des barres de synchronisation qui peuvent être des :

• **Débranchements:** les transitions qui partent d'un débranchement ont lieu en même temps.

• **Jonctions:** on ne franchit une jonction qu'après la réalisation de toutes les transitions qui s'y rattachent.

des traitements en phase de réalisation. On peut aussi de utiliser de façon plus informelle pour décrire des

1.3.7 Diagramme de composant

Les diagrammes de composant décrivent les composants et leurs dépendances dans l'environnement de réalisation, ils montrent le choix de réalisation. En général, ils ne sont utilisés que pour des systèmes complexes. Un composant est une vue physique qui représente une partie implantable d'un système. Un composant peut être un script, fichier de demande, un fichier de donnée, table d'un composant peut être réalisé par d'autre composants, etc... Les éléments utilises dans un diagramme de déploiement sont :

- Composant

Un composant représente une entité logicielle d'un système. Un composant doit fournir un service bien précis, les fonctionnalités qu'il encapsule doivent être cohérentes entre elle et générique.

-Dépendance

Une dépendance est utilisée pour modéliser la relation entre deux composants qui indiqué qu'un composant fait référence aux services offerts par autre composant.

1.3.8 Diagramme de déploiement

Un diagramme de déploiement décrit la disposition physique des ressources matérielles qui composent le système et montre la répartition des composants sur ces matériels. Chaque

ressource étant matérialisée par un noeud, le diagramme de déploiement précise comment les composants sont répartis sur les noeuds et quelles sont les connexions entre les composants ou les noeuds. Les diagrammes de déploiement existent sous deux formes : spécification et instance.

-Composant : Un composant représente une entité logicielle du système. Sur un diagramme de déploiement, les composants sont placés dans des noeuds pour identifier l'endroit de leur déploiement.

-Noeud : Un noeud représente un ensemble d'élément matériel du système. Cette entité est représentée par un cube tridimensionnel.

1.3.9 Le diagramme de la collaboration

Décrit les scénarios de chaque cas d'utilisation en mettant l'accent sur l'interaction des objets et les messages échangés.

1.4. Mise en oeuvre d'UML

UML n'est pas une méthode et ne propose pas une démarche de modélisation explicitant et encadrant toutes les étapes d'un projet. Il n'est qu'un langage de modélisation, ce qui justifie le besoin de faire appel à une méthode qui permet le passage de l'expression des besoins au code de l'application. Nous allons donc présenter une des méthodes proposées dans la littérature. Il s'agit d'une méthode simple et générique proposée par LAURENTAUDIBERT

Elle se situe à mi-chemin entre *UP* (*Unified Process*), qui constitue un cadre général très complet de processus de développement, et XP (*eXtremeProgramming*) qui est une approche minimaliste à la mode centrée sur le code.

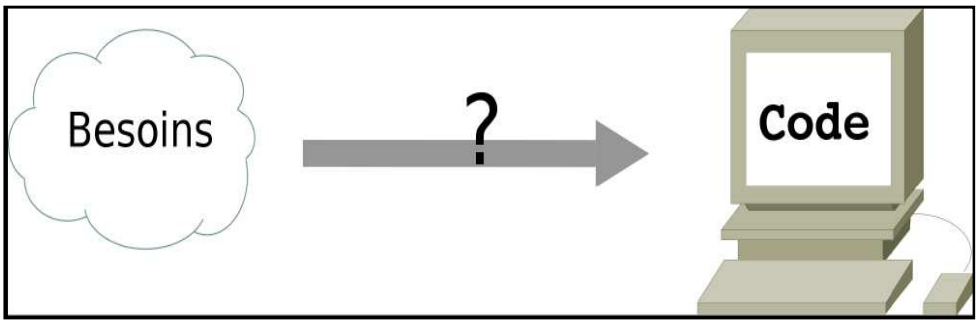


Figure 1.1: Quelle méthode pour passer de l'expression des besoins au code de l'application?

1.4.1 Identification des besoins et spécification des fonctionnalités

1.4.1.1 Identification et représentation des besoins diagramme de cas d'utilisation

Les cas d'utilisation sont utilisés tout au long du projet. Dans un premier temps, on les crée pour identifier et modéliser les besoins des utilisateurs (voir figure1.2).Ces besoins sont déterminés à partir des informations recueillies lors des rencontres entre informaticiens et utilisateurs. Durant cette étape, il faut déterminer les limites du système, identifier les acteurs et recenser les cas d'utilisation.

Les interactions entre les acteurs et le système (au sein des cas d'utilisation) seront explicitées sous forme textuelle et sous forme graphique au moyen de diagrammes de séquence.

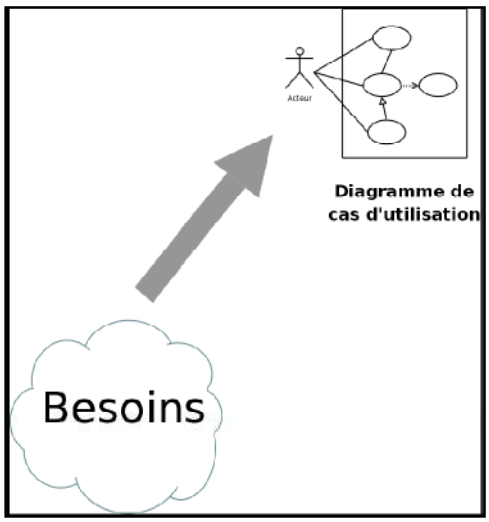


Figure 1.2 : Les besoins sont modélisés par un diagramme de cas d'utilisation.

1.4.1.2 Spécification détaillée des besoins Diagrammes de séquence système

Dans cette étape, on cherche à détailler la description des besoins par la description textuelle des cas d'utilisation, et la production de diagrammes de séquence système illustrant cette description textuelle (voir figure 1.3). Cette étape conduit souvent à mettre à jour le diagramme de cas d'utilisation puisque nous sommes toujours dans la spécification des besoins.

Les scénarios de la description textuelle des cas d'utilisation sont illustrés par des diagrammes de séquence système. Il faut, au minimum, représenter le scénario nominal de chacun des cas d'utilisation par un diagramme de séquence qui spécifier l'interaction entre l'acteur, ou les acteurs, et le système. Le système est ici considéré comme un tout et est représenté par une ligne de vie. Chaque acteur est également associé à une ligne de vie.

1.4.1.3 Maquette de l'IHM de l'application (non couvert par UML)

Une maquette d'IHM (Interface Homme-Machine) est un produit jetable permettant aux utilisateurs d'avoir une vue concrète mais non définitive de la future interface de l'application (voir figure 1.4). La maquette peut très bien consister en un ensemble de dessins produits par un logiciel de présentation ou de dessin. Par la suite, la maquette pourra intégrer des fonctionnalités de navigation permettant à l'utilisateur de tester l'enchaînement des écrans ou des menus, même si les fonctionnalités restent fictives. La maquette doit être développée rapidement afin de provoquer des retours de la part des utilisateurs.

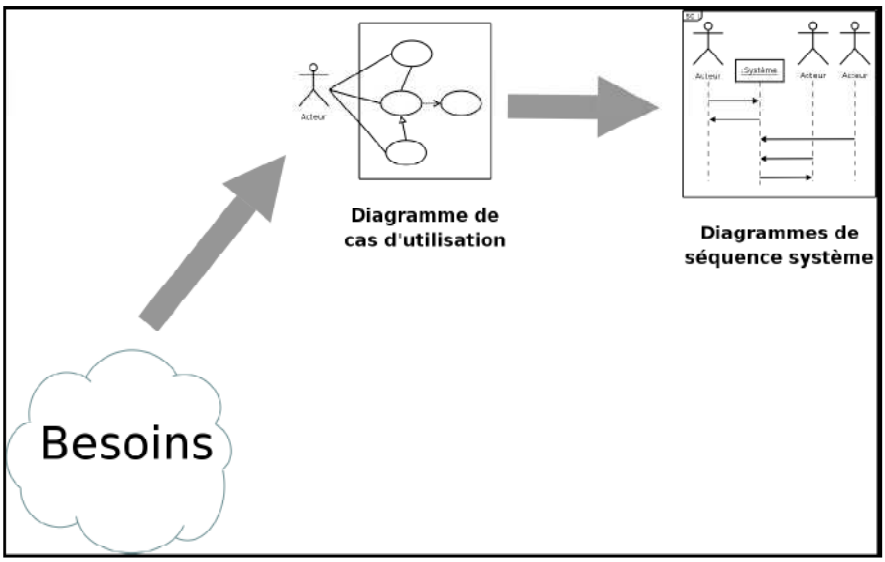


Figure 1.3: Les diagrammes de séquence système illustrent la description textuelle des cas d'utilisation.

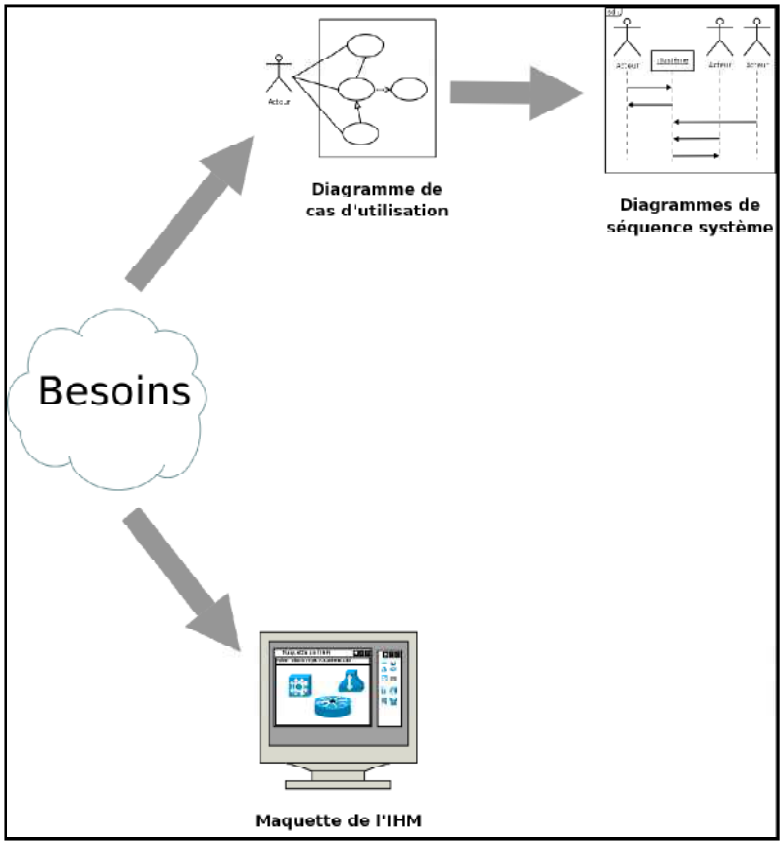


Figure 1.4 : Une maquette d'IHM facilite les discussions avec les futurs utilisateurs.

1.4.2 Phases d'analyse

1.4.2.1 Analyse du domaine : modèle du domaine

L'élaboration du modèle des classes du domaine permet d'opérer une transition vers une véritable modélisation objet. L'analyse du domaine est une étape totalement dissociée de l'analyse des besoins. Elle peut être menée avant, en parallèle ou après cette dernière.

La phase d'analyse du domaine permet d'élaborer la première version du diagramme de classes appelée modèle du domaine (voir figure 1.5).Ce modèle doit définir les classes qui modélisent les entités ou concepts présents dans le domaine de l'application. Il s'agit donc de produire un modèle des objets du monde réel dans un domaine donné. Ces entités ou concepts peuvent être identifiés directement à partir de la connaissance du domaine ou par des entretiens avec des experts du domaine.

Les classes du modèle du domaine ne doivent pas contenir d'opération, mais seulement les attributs.

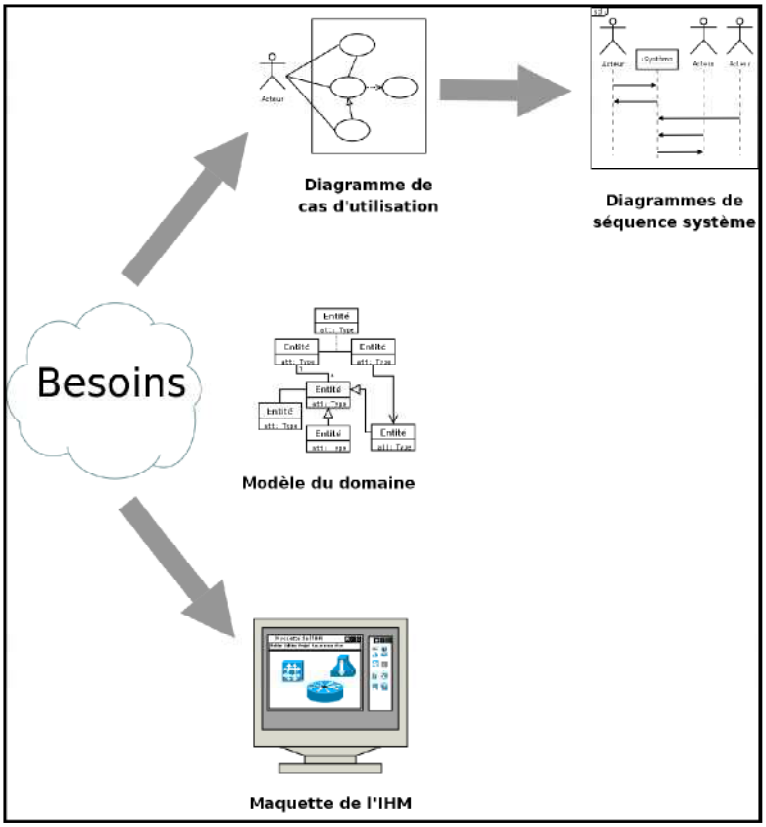


Figure 1.5 : La phase d'analyse du domaine permet d'élaborer la première version du diagramme de classes.

1.4.2.2 Diagramme de classes participantes

Le diagramme de classes participantes est particulièrement important puisqu'il effectue la jonction entre, d'une part, les cas d'utilisation, le modèle du domaine et la maquette, et d'autre part, les diagrammes de conception logicielle que sont les diagrammes d'interaction et le diagramme de classes de conception (voir figure 1.6). Le diagramme de classes participantes modélise trois types de classes d'analyse sont :

- **Les classes de dialogues :** Les classes qui permettent les interactions entre l'IHM et les utilisateurs sont qualifiées de *dialogues*. Ces classes sont directement issues de l'analyse de la maquette. Il y a au moins un dialogue pour chaque association entre un acteur et un cas d'utilisation. En général, les dialogues vivent seulement le temps du déroulement du cas d'utilisation concerné.
- **Les classes de contrôles :** Les classes qui modélisent la cinématique de l'application sont appelées *contrôles*. Elles font la jonction entre les dialogues et les classes métier en permettant aux différentes vues de l'application de manipuler des informations détenues par un ou plusieurs objets métier. Elles contiennent les règles applicatives et les isolent à la fois des dialogues et des entités.

- **Les classes entités :** Les classes métier, qui provient directement du modèle du domaine, sont qualifiées *d'entités*. Ces classes sont généralement persistantes, c'est-à-dire qu'elles survivent à l'exécution d'un cas d'utilisation particulier et qu'elles permettent à des données et des relations d'être stockées dans des fichiers ou des bases de données. Lors de l'implémentation, ces classes peuvent ne pas se concrétiser par des classes mais par des relations, au sens des bases de données relationnelles.

1.4.2.3 Diagrammes d'activités de navigation

Les IHM modernes facilitent la communication entre l'application et l'utilisateur en offrant toute une gamme de moyens d'action et de visualisation comme des menus déroulants ou contextuels, des palettes d'outils, des boîtes de dialogues, des fenêtres de visualisation, etc. Cette combinaison possible d'options d'affichage, d'interaction et de navigation aboutis aujourd'hui à des interfaces de plus en plus riches et puissantes.

UML offre la possibilité de représenter graphiquement les activités de navigation dans l'interface en produisant des diagrammes dynamiques. On appelle ces diagrammes des diagrammes de navigation (voir figure 1.7).

Les diagrammes d'activités de navigation sont à relier aux classes de dialogue du diagramme de classes participantes. Les différentes activités du diagramme de navigation peuvent être stéréotypées en fonction de leur nature : «fenêtre», «menu», «menu contextuel», «dialogue», etc.

La modélisation de la navigation à intérêt à être structurée par acteur.

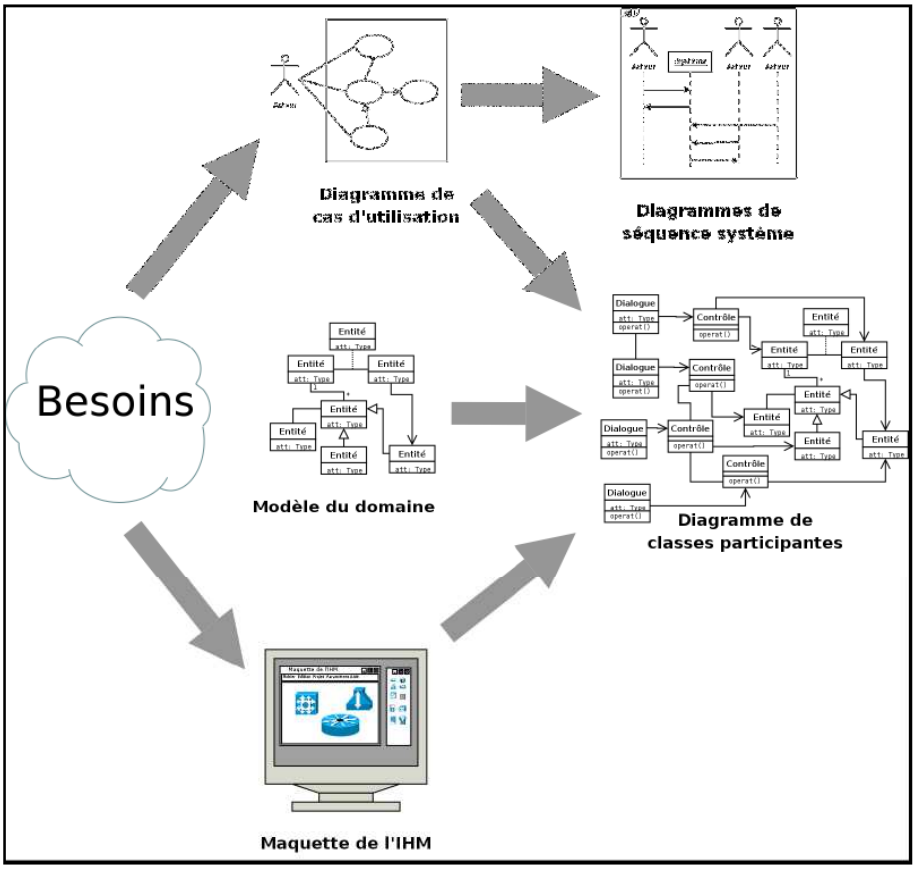


Figure 1.6 : Le diagramme de classes participantes effectue la jonction entre les cas d'utilisation, Le modèle du domaine et les diagrammes de conception logicielle.

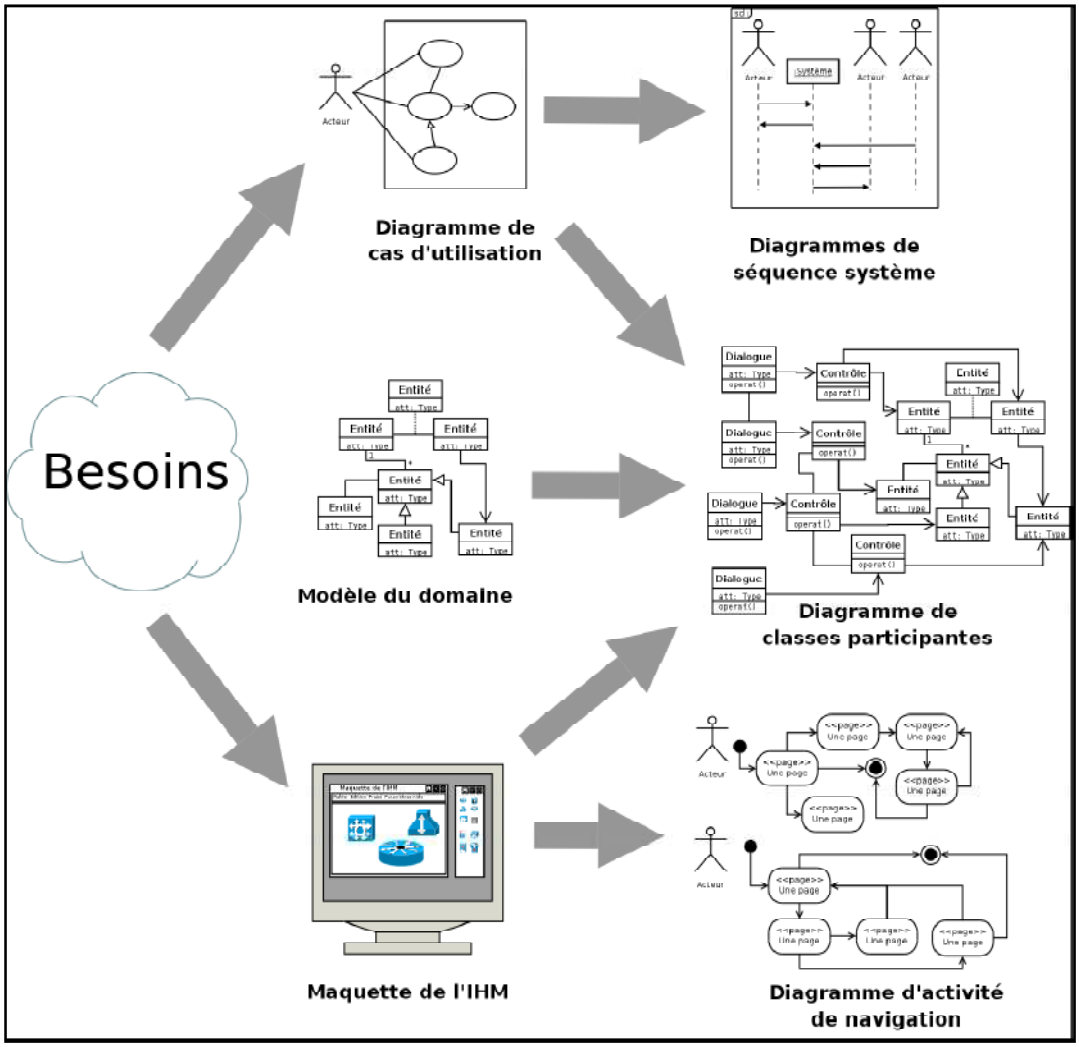


Figure 1.7 : Les diagrammes d'activités de navigation représentent graphiquement les activités de navigation dans l'interface.

1.4.3 Phase de conception

1.4.3.1 Diagrammes d'interaction

Maintenant, il faut attribuer précisément les responsabilités de comportement, dégagée par le diagramme de séquence système, aux classes d'analyse du diagramme de classes participantes. Les résultats de cette réflexion sont présentés sous la forme de diagrammes d'interaction UML (voir figure 1.8). Parallèlement, une première ébauche de la vue statique de conception, c'est-à-dire du diagramme de classes de conception, est construite et complétée.

Durant cette phase, l'ébauche du diagramme de classes de conception reste indépendante des choix technologiques qui seront faits ultérieurement.

Les diagrammes d'interactions sont utiles au concepteur pour décider quelle est la classe qui va contenir chaque service ou fonction. Dans les diagrammes d'interaction, les

objets communiquent en s'envoyant des messages qui invoquent des opérations sur les objets récepteurs. Il est ainsi possible de mettre en œuvre l'allocation des responsabilités à partir des diagrammes d'interaction.

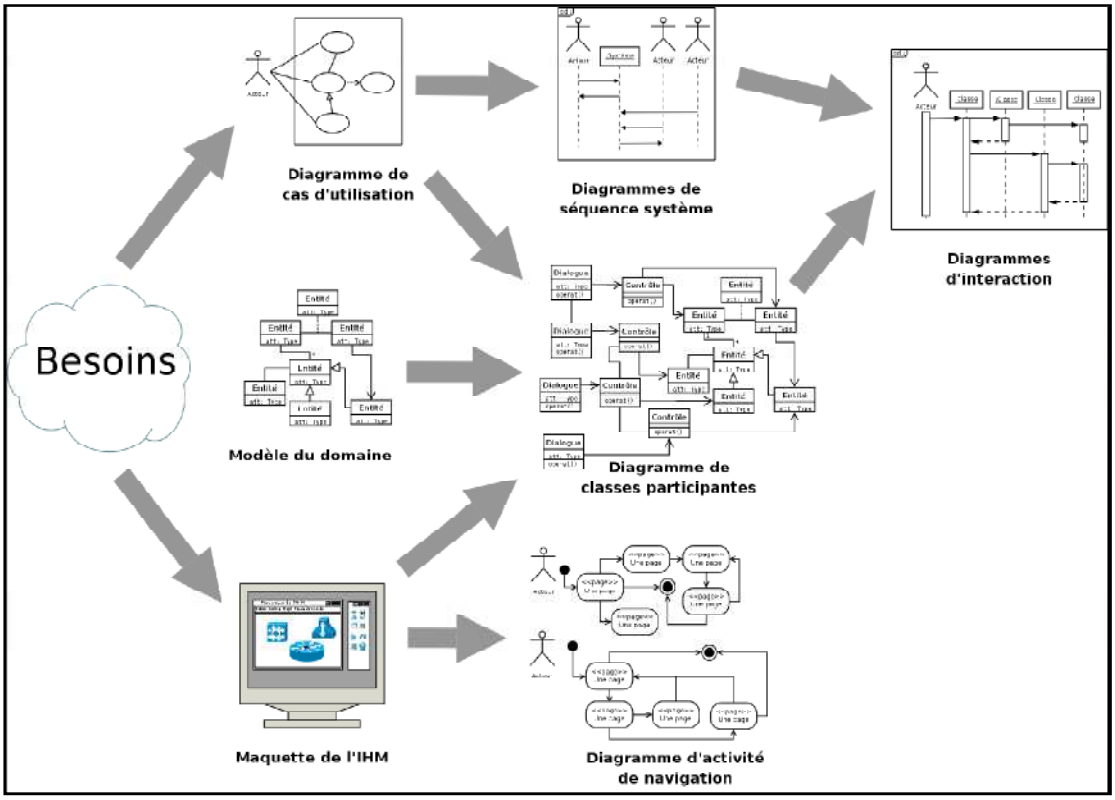


Figure 1.8 : Les diagrammes d'interaction permettent d'attribuer précisément les responsabilités de comportement aux classes d'analyse.

Par rapport aux diagrammes de séquences système, nous remplaçons ici le système, vu comme une boîte noire, par un ensemble d'objets en collaboration (voir figure 1.9). Ces objets sont des instances des trois types de classes d'analyse du diagramme de classes participantes, à savoir des dialogues, des contrôles et des entités.

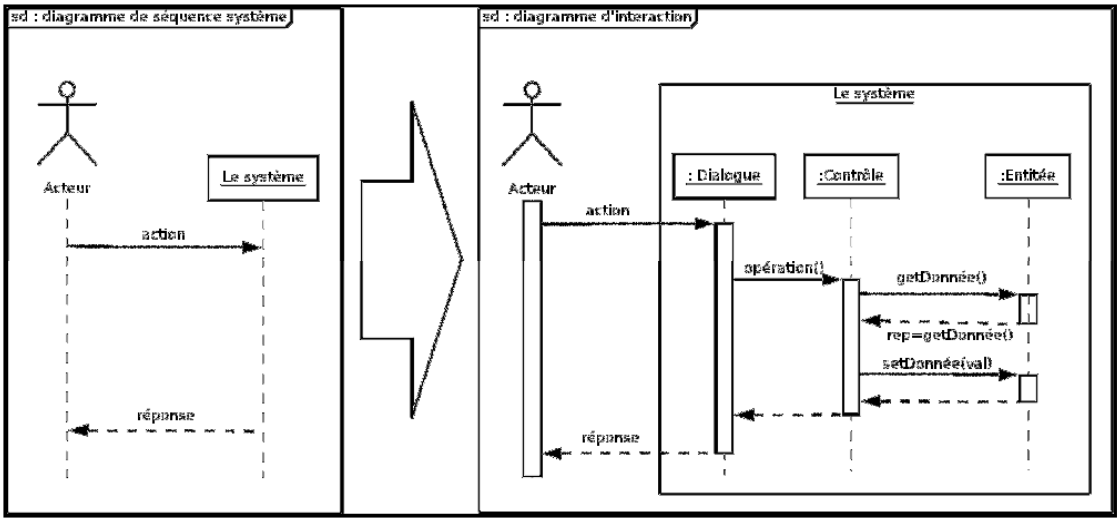


Figure 1.9 : Le système des diagrammes de séquences système, vu comme une boîte noire, est remplacé par un ensemble d'objets en collaboration.

1.4.3.2 Diagramme de classes de conception

L'objectif de cette étape est de produire le diagramme de classes qui servira pour l'implémentation (voir figure 1.10). Une première ébauche du diagramme de classes de conception a déjà été élaborée en parallèle du diagramme d'interaction. Il faut maintenant le compléter en précisant les opérations privées des différentes classes. Il faut prendre en comptes les choix techniques, comme le choix du langage de programmation.

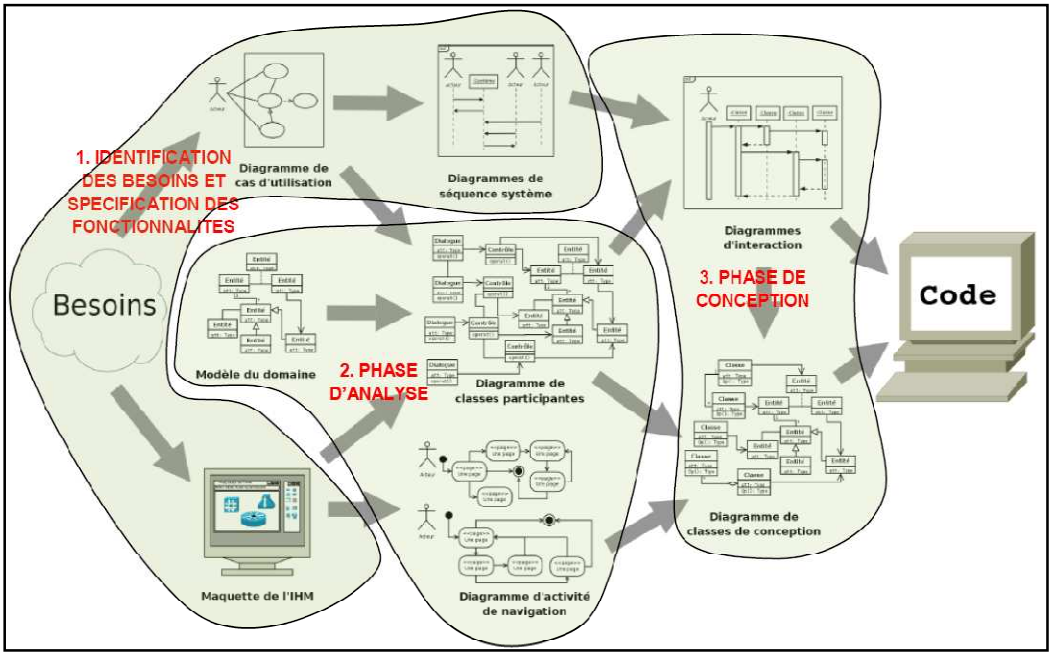


Figure 1.10 : Chaîne complète de la démarche de modélisation du besoin jusqu'au code.

1.5. Conclusion

Dans ce chapitre, on a présenté les notions essentielles d'UML et ses diagrammes qu'on va utiliser pour le développement de notre projet, puis on a vu les différentes étapes de notre démarche conception qu'on va suivre dans notre projet.

Chapitre II Etude de l'existant

2.1 Introduction

L'étude de l'existant est une étape indispensable dans tout projet informatique, elle consiste en un travail de terrain au terme duquel nous pouvons bien connaître la situation actuelle de l'organisation.

Ce chapitre représente une introduction au contexte du projet de fin d'étude intitulé conception d'un logiciel de gestion pharmacie. En premier lieu nous allons présenter les différentes activités de la pharmacie. Ensuite nous allons analyser et détailler les documents utilisés au niveau de la pharmacie.

2.2 Présentation de l'officine

Notre stage est effectué au niveau de la pharmacie de Zerrouki Nora qui se situe à willaya de Mila. La pharmacienne est la responsable de la gestion de l'activité globale de la pharmacie.

Nous avons remarqué que les opérations les plus importantes qui se déroulent au niveau du dépôt pharmaceutique sont :

- L'arrivé des produits livré par le fournisseur ;
- Le traitement des ordonnances des patients.

❖ La réception des produits

C'est le moment où les médicaments arrivent au dépôt. Le même contrôle qu'à la livraison est effectué par le gérant ou responsable du dépôt. Il :

- Vérifie s'il n'y a pas de discordance entre le bon de livraison délivré par le fournisseur et le bon de commande (quantités, formes et dosage).

- Vérifie la qualité physique et les dates de péremption des produits livrés.

- Rentre les différents items sur leurs fiches de stock respectives.

- Range les différents items dans leurs places respectives au niveau des étagères.

- Le gérant signe alors le bon de livraison et en conserve un exemplaire qu'il garde au dépôt.

❖ Le traitement des ordonnances

La pharmacienne gère les ventes et les opérations qui leurs sont liées.

2.3 Etude des postes de travail

Pour une compréhension du système existant, les postes de travail entrant dans notre champ d'étude doivent être étudiés et examinés minutieusement. Cela nous permettra d'identifier les besoins en informations des différents utilisateurs.

Dans cette étape, une fiche d'étude de poste sera élaborée pour chaque poste de travail. Dans notre cas, nous allons réalisés une seule fiche parce qu'il s'agit d'un seul poste :

<table>
<tr><th>Désignation</th><th colspan="2">Pharmacienne</th></tr>
<tr><th colspan="3">Taches faite à ce poste</th></tr>
<tr><td colspan="3">1. Etablissement des bons de commandes ;
2. Contrôle des factures ;
3. Vérifications des entrées et sorties ;
4. Mission pour l'achat des produits ;
5. Vente des produits commandés :
6. Vérification de la conformité entre le bon de commande et le bon de livraison puis préparation du « bon pour »
7. Mise à jour des fiche de ventilation.
8. Distribution des bons de commandes hebdomadaire et mensuelle
9. Obtention des prix unitaires de chaque produit depuis plusieurs fournisseurs pour sélectionner le bon choix</td></tr>
<tr><th>Documents remplis</th><th>Document entrées</th><th>Document sorties</th></tr>
<tr><td>-bon de commande
-fiche de ventilation
-liste des statistiques de consommation annuelle
- Rapports</td><td>-facture
-bon de livraison</td><td>-bon de commande (visé)
-facture (visé)
-liste des statistiques de consommation annuelle
- Rapports</td></tr>
</table>

Table 2.1 : Fiche d'étude du poste de travail

2.4 Etude des documents

Cette étude traite les documents véhiculés au sein de la pharmacie (entrée/sortie). Il y'a deux type de documents :

- Document entrée : provient par un organisme externe à la pharmacie.
- Document sortie : destiné à une organisation externe.

Les documents utilisés sont :

Document	Code
• **Bon de commande** • **Facture** • **Bon livraison** • **Fiche de stock** • **Fiche de ventilation** • **Listes des produits périmés dans l'année**	BC Fact BL FS FV LPP

Table 2.2: Documents manipulés.

1-Document N°01

Code : BC

Désignation : Bon de commande

Nature : sortie

Nombre d'exemplaire : 02

Remplie par : pharmacie

Destiné au : fournisseur

Rôle : détaille les opérations d'achats des produits.

Périodicité : chaque rupture de stock.

Information élémentaire	Code	type	taille	observation
En tête :				
Date bon de commande	Dat-BC	AN	10	JJ /MM/AAAA
Numéro bon de commande	N-BC	N	06	
Code fournisseur	Cod-Four	N	06	
Nom de fournisseur	Nom-four	AN	20	
Adresse	Adr	AN	20	
Code client	Cod-cl	N	03	
Tel de fournisseur	Tel-four	N	12	
Fax de fournisseur	Fax-four	N	12	
Corps :				
Code produit	Cod-prod	N	04	
Désignation de produit	Désig-prod	AN	40	
Quantité commandée	Qté-com	N	04	

pied : Signature de pharmacienne Signature de directeur Signature de fournisseur Date	 Sig-ph Sig-dirct Sig-four Dat	 10		 JJ/MM/AAAA

Table 2.3 : Fiche d'étude du document : Bon de commande

2- Document N°02

Code : fact

Désignation : facture

Nature : entrée

Nombre d'exemplaire : 04

Remplie par : fournisseur

Destiné au : pharmacie

Rôle : contient les informations détaillées sur les produits achetés tel que le type du produit, la quantité, le prix unitaire et le prix total.

Périodicité : après la livraison de chaque commande.

Information élémentaire	Code	type	taille	observation
En tête : Numéro de facture Date de facture Référence client Code client	 N-Fact D-Fact Réf Cod-Cl	 N AN N N	 5 10 03 03	 JJ /MM/AAAA
Corps : Code produit Désignation de produit Numéro de lot Date de péremption Prix unitaire Montant TVA HT Total UV	 Cod-prod Désig-prod N-lot Dat-pére PU Mont TVA HT T-UV	 N AN N AN N N N N N	 04 40 08 10 06 08 04 04 04	 JJ/MM/AAAA
Pied : Remise Net à payé Signature	 Rem Net-p Sig-four	 N N AN	 07 	

Table 2.4 : Fiche d'étude du document : facture.

3- Document N°03

Code : BL.

Désignation : bon de livraison.

Nature : entrée.

Nombre d'exemplaire : 01.

Remplie par : fournisseur.

Destiné au : pharmacie.

Rôle : confirme la livraison.

Périodicité : après la livraison de bon de commande.

Information élémentaire	Code	type	taille	observation
En tête :				
Numéro de bon de livraison	N-BL	N	05	
Nom de fournisseur	Nom-four	AN	20	
Corps :				
Nom client	Nom Cl	AN	20	
Numéro de commande	N-C	N	06	JJ/MM/AAAA
Nombre de colis	N-Colis	N	04	
Numéro de facture	N-Fact	N	05	
Numéro de lot	N-lot	N	06	
Pied :				
Nom du signature	Nom-sig	AN	15	
Date de livraison	Dat-liv	AN	10	JJ/MM/AAAA

Table 2.6 : Fiche d'étude du document : Bon de livraison.

4- Document N°04

Code : FS.

Désignation : fiche de stock.

Nombre d'exemplaire : aléatoire.

Remplie par : le pharmacien.

Destine au : pharmacien.

Rôle : permettre de conserver le mouvement du stock de chaque produit.

Périodicité : aléatoire

Information élémentaire	Code	type	taille	observation
En tête :				
Numéro de bon produit	N-prod	N	04	
Prix unitaire	PU	N	06	
Stock sécurité	Stock-S	N	04	
Stock alerte	Stock-Aler	N	03	
Stock maximale	Stock-Max	N	04	
DCI	DCI	AN	15	
Dénomination Commerciale	Denom	AN	15	
Forme	Form			
dosage	dosage	AN	10	
		AN	06	
Corps :				
Date d'entrer	Dat-entr	AN	10	
Date de sortie	Dat-sort	AN	10	
Origine	Origi	AN	15	
destination	Dest	AN	15	
Produit entré	Prod-entr	N	03	
Produit sortie	Prod-sort	N	03	
Stock du produit	Stock-prod	N	03	
Observation	obser	N	10	

Table 2.7: Fiche d'étude du document fiche stock.

5-Document N°05

Code : FV.

Désignation : fiche de ventilation.

Nombre d'exemplaire : aléatoire.

Remplie par : surveillant médical.

Destiné au : surveillant médical.

Information élémentaire	Code	type	taille	Observation
En tête :				
Numéro de fiche	N-fich	N	05	
Numéro de produit	N-prod	N	04	
Dénomination	Dénom	AN	20	
Conditionnement	Condi	AN	04	
Stock maximal	Stock-max	N	10	
Stock alert	Stock-aler	N	10	
Prix	PU	N	08	
Corps				
ENTRER :				
Date entrée	Dat-ent	AN	10	JJ/MM/AAAA
Référence	Ref	N	05	
Provenance	Proven	AN	10	
Quantité	Qté	N	02	

SORTIES :				
Date sortie	Dat-sort	AN	10	JJ/MM/AAAA
Destination	Destin	AN	10	
Numéro de bon de commande	N-B	N	06	
Quantité	Qté	N	04	
Date de stockage	Dat_stock	N	10	JJ/MM/AAAA

Table 2.8 : Fiche d'étude du document : fiche de ventilation.

6- Document N°06

Code : LPP.

Désignation : listes des produits qui seront périmés dans un délai..

Nombre d'exemplaire : 01.

Remplie par : pharmacien.

Destine au : pharmacie.

Rôle : contrôler les produits qui ont dépassé leurs date de péremption

Information élémentaire	Code	type	taille	Observation
En tête :				
Date	Dat	AN	10	JJ/MM/AAAA
Corps :				
Numéro	N°	N	03	
Code produit	Cod-prod	N	03	
Date de péremption	Dat-périm	AN	10	JJ/MM/AAAA
DCI	DCI	AN	40	
Forme	Form	AN	10	
Dosage	Dosage	AN	06	
Dénomination commerciale	Déno-comer	AN	40	
Prix unitaire	PU	N	08	
Montant	Mont	N	10	

Table 2.9 : Fiche d'étude du document : liste produits périmés.

2.5 Conclusion:

Durant l'analyse de l'existant nous avons pu recenser toutes les informations nécessaires et indispensables à notre projet à savoir la conception et la réalisation d'un système de gestion d'une pharmacie. Elle nous a permis la compréhension des besoins de l'organisation et ce à travers l'analyse détaillée des postes de travail et les documents manipulés.

Dans le prochain chapitre, nous aborderons l'étude conceptuelle de notre système.

Chapitre III
Etude de cas

3.1 Introduction

L'objectif de cette étape est de déterminer de façon détaillée et précise ce que le système devra faire, afin de répondre aux objectifs établis lors de l'étude de l'existant, tout en respectant les contraintes établies préalablement.

3.2 Identification des cas d'utilisations

Le tableau suivant englobe les différents CU de ce système :

No	Les cas d'utilisations	Les acteurs
01	Authentification	Pharmacienne
02	Réception produit	Pharmacienne
03	Vente produits	Pharmacienne
04	Produit périme	Pharmacienne
05	Ajouter fournisseur	Pharmacienne
06	Modifier fournisseur	Pharmacienne
07	Supprimer fournisseur	Pharmacienne
08	Ajouter produit	Pharmacienne
09	Modifier produit	Pharmacienne
10	Supprimer produit	Pharmacienne
11	MAJ STOCK	Pharmacienne
12	Fiche de stock	Pharmacienne

3.3 Description des cas d'utilisations

Nous allons maintenant donner une description de chaque cas d'utilisation

3.3.1 Cas d'utilisation « authentification »

Titre : Authentification

Finalité : Ce cas permet d'utiliser le système par le pharmacien.

Acteur : le pharmacien

Pré condition : le pharmacien possède un compte.

Enchaînements nominaux:

1. pharmacienne lance l'application
2. Le système affiche une fenêtre pour qu'il s'identifie.
3. le pharmacien saisie son login et son mot de passe.

4. Le système vérifie leur validité puis lance le menu principal de l'application.

Enchaînements alternatifs :

Login ou mot de passe erroné, l'authentification est demandée à nouveau au maximum trois fois.

Enchaînements exceptionnels:

Le pharmacien ne saisit pas le bon mot de passe. L'application se ferme.

Post conditions :

Le menu de l'application est accessible.

3.3.2 Cas d'utilisation « réception des produits »

Titre : réception des produits

Finalité : Ce cas permet de traiter les produits pharmaceutiques livrés.

Acteur principal : le pharmacien

Pré condition :

-le pharmacien s'est authentifié.

-La commande est reçue.

Enchaînements nominaux:

1. le pharmacien demande d'introduire les informations de la nouvelle livraison.
2. Le système lui affiche un formulaire.
3. le pharmacien remplis ce formulaire, s'il s'agit d'un nouvel produit et/ou d'un nouveau fournisseur, le pharmacien doit ajouter ces derniers.
4. Le système contrôle les informations saisies.
5. le pharmacien valide la saisie de formulaire
6. Le système enregistre les données de la livraison et mettre à jour la quantité en stock des produits pharmaceutiques réceptionné.

Enchaînements alternatifs :

Les informations sont incomplètes ou erronées.

Post conditions :

La nouvelle livraison a été enregistrée.

3.3.3 Cas d'utilisation « vente des produits »

Titre : vente des produits

Finalité : Ce cas permet de satisfaire les besoins des malades.

Acteur principal : le pharmacien

Pré condition :

-le pharmacien s'authentifie.

Enchaînements nominaux:

1. La pharmacienne demande de saisir les informations concernant la nouvelle vente.
2. Le système lui affiche le formulaire.
3. le pharmacien remplit ce formulaire par les données nécessaires (désignation produit, Quantité sortie, date, …etc.) et valide la saisie.
4. Le système contrôle la saisie puis enregistre l'opération et mettre à jour la quantité en stock des produits pharmaceutiques vendu.

3.3.4 Cas d'utilisation « produits périmés »

Titre : produit périmé

Finalité : Ce cas permet nettoyer le stock les produits périmes

Acteur principal : le pharmacien

Pré condition :

-pharmacienne s'est authentifié.

-Le contrôle des médicaments.

Enchaînements nominaux:

1. Le pharmacien demande de visualiser la liste des produits périmés.
2. Le système compare pour chaque médicament sa date de préemption avec la date système, si elle est dépassée, il l'affiche.
3. Le pharmacien lance la suppression de ces produits.
4. Le système supprimer touts les produits.

3.3.5 Cas d'utilisation « ajouter fournisseur »

Titre : service ajouter fournisseur.

Finalité : ce cas permet d'ajouter un nouveau fournisseur à la liste des fournisseurs.

Acteur principal : le pharmacien.

Pré condition :

- le pharmacien s'est authentifié.

Enchaînements nominaux:

1. le pharmacien demande d'ajouter un nouveau fournisseur.

2. Le système affiche la fenêtre pour saisis les informations de ce fournisseur.
3. La pharmacienne saisis les informations de fournisseur.
4. Le système enregistrer ce fournisseur dans la liste et lui donnez la main pour saisir un autre fournisseur.
5. En fin le pharmacien enregistrer l'opération.

Post conditions :

Le fournisseur a été ajouté à la liste des fournisseurs.

3.3.6 Cas d'utilisation « modifier fournisseur »

Titre : modifier fournisseur.

Finalité : ce cas permet de modifier les informations d'un fournisseur.

Acteur principal : pharmacienne.

Pré condition :

- pharmacienne s'est authentifié.

Enchaînements nominaux:

1. Pharmacienne demande la visualisation de la liste des fournisseurs.
2. Le système affiche la liste de tous les fournisseurs existant.
3. La pharmacienne sélectionner un fournisseur pour faire la modification.
4. Le système affiche les informations de fournisseur sélectionné.
5. La pharmacienne saisit les modifications.
6. Le système contrôle la saisie des informations modifié.
7. La pharmacienne valide la modification.
8. Le système enregistre la modification.

Post conditions

Les informations de fournisseur doit être modifier.

3.3.7 Cas d'utilisation « Supprimer fournisseur »

Titre : Supprimer fournisseur.

Finalité : Ce cas permet de supprimer un fournisseur existant.

Acteur : La pharmacienne.

Pré condition : la pharmacienne s'authentifie.

Le fournisseur concerné existe.

Enchaînements nominaux:

1. la pharmacienne demande la visualisation de la liste des fournisseurs.
2. Le système affiche la liste de tous les fournisseurs.
3. la pharmacienne sélectionne l'article qu'il veut supprimer.
4. Le système informe la pharmacienne s'il veut vraiment le supprimer.
5. la pharmacienne valide l'opération.

Post conditions :

Le fournisseur est supprimé.

3.3.8 Cas d'utilisation « ajouter produit »

Titre : Ajouter un produit.

Finalité : Ce cas permet d'ajouter un produit

Acteur : La pharmacienne.

Pré condition : La pharmacienne s'est authentifiée.

Enchaînements nominaux:

1. la pharmacienne demande l'ajout d'un nouveau produit.
2. Le système affiche le formulaire pour saisir les informations nécessaires.
3. la pharmacienne remplit ce formulaire.
4. Le système contrôle les informations saisies.
5. la pharmacienne valide la saisie du formulaire.
6. Le système enregistre les informations de le produit.

Enchaînements alternatifs :

Les informations sont incomplètes ou erronés.

Post conditions :

Le nouveau produit est ajouté.

3.3.9 Cas d'utilisation « modifier produit »

Titre : modifier un produit.

Finalité : Ce cas permet de modifier un produit.

Acteur : la pharmacienne.

Pré condition : la pharmacienne s'est authentifiée.

Enchaînements nominaux:

1. La pharmacienne demande la visualisation de la liste des produits.
2. Le système affiche la liste de tous les produits.
3. la pharmacienne sélectionne un produit pour modification
4. Le système affiche les informations du produit sélectionné.
5. la pharmacienne saisie les modifications.
6. Le système contrôle la saisie des informations modifiées.
7. la pharmacienne valide la modification.
8. Le système enregistre la modification.

Enchaînements alternatifs :

Les informations sont incomplètes ou erronés.

Post conditions :

Le produit est modifié.

3.3.10 Cas d'utilisation « Supprimer produit »

Titre : Supprimer un produit

Finalité : Ce cas permet de supprimer un produit existant

Acteur : La pharmacienne.

Pré condition : la pharmacienne s'est authentifiée.

Le produit concerné existe.

Enchaînements nominaux:

1. la pharmacienne demande la visualisation de la liste des produits existants.
2. Le système affiche la liste de tous les produits
3. la pharmacienne sélectionne le produit qu'il veut supprimer.
4. Le système informe la pharmacienne s'il veut vraiment la supprimer.
5. la pharmacienne valide l'opération.

Post conditions :

Le produit est supprimé.

3.3.11 Cas d'utilisation « fiche de stock »

Titre : fiche de stock

Finalité : Ce cas permet de consulter l'état de chaque produit

Acteur : La pharmacienne.

Pré condition : la pharmacienne s'est authentifiée.

Enchaînements nominaux:

1. Le pharmacien demande de visualiser la liste de tous les produits.
2. Le système affiche pour chaque produit sélectionné sa quantité entrée Globale, sa quantité sortie globale et sa quantité en stock

3.3.12 Cas d'utilisation « produits en stock alerte »

Titre : produits en stock alerte.

Finalité : Ce cas permet de vérifier et d'afficher les produits qui sont en stock alerte.

Acteur : le pharmacien.

Pré condition : le pharmacien s'est authentifié.

Enchaînements nominaux:

1. la pharmacienne demande au système d'afficher la liste des produits qui ont atteint le seuil minimum en stock.
2. le système affiche tous les articles dont la quantité en stock est inferieur ou égale au seuil minimal.
3. la pharmacienne lance l'impression de cette liste.

Post conditions :

-La liste est imprimée.

3.4 diagramme de cas d'utilisation

Figure 3.1 : diagramme de cas d'utilisation

3.5 Les diagrammes de séquences

3.5.1 Authentification

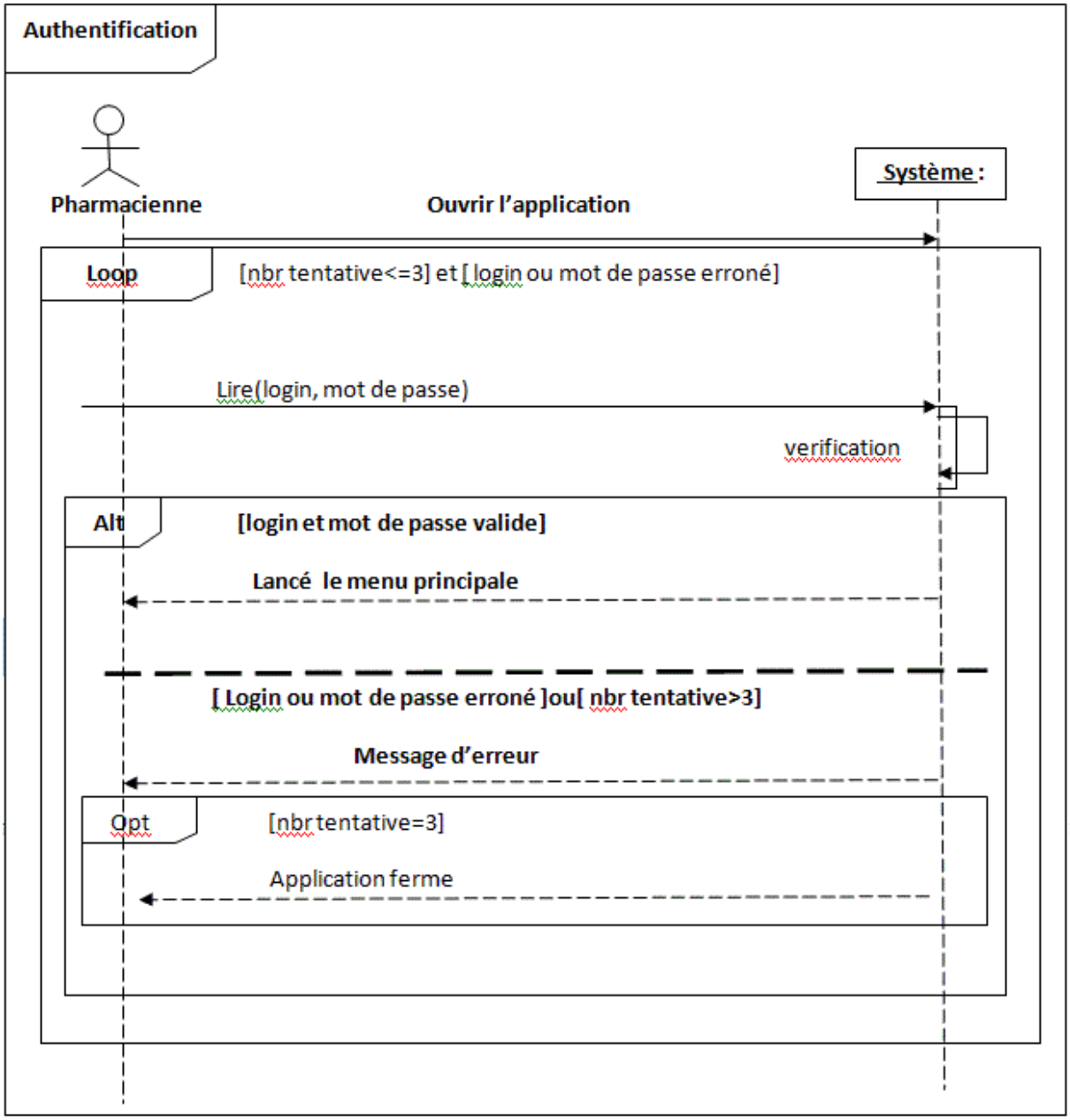


Figure 3.2 : diagramme de séquence du cas d'utilisation authentification

3.5.2 Réception des produits :

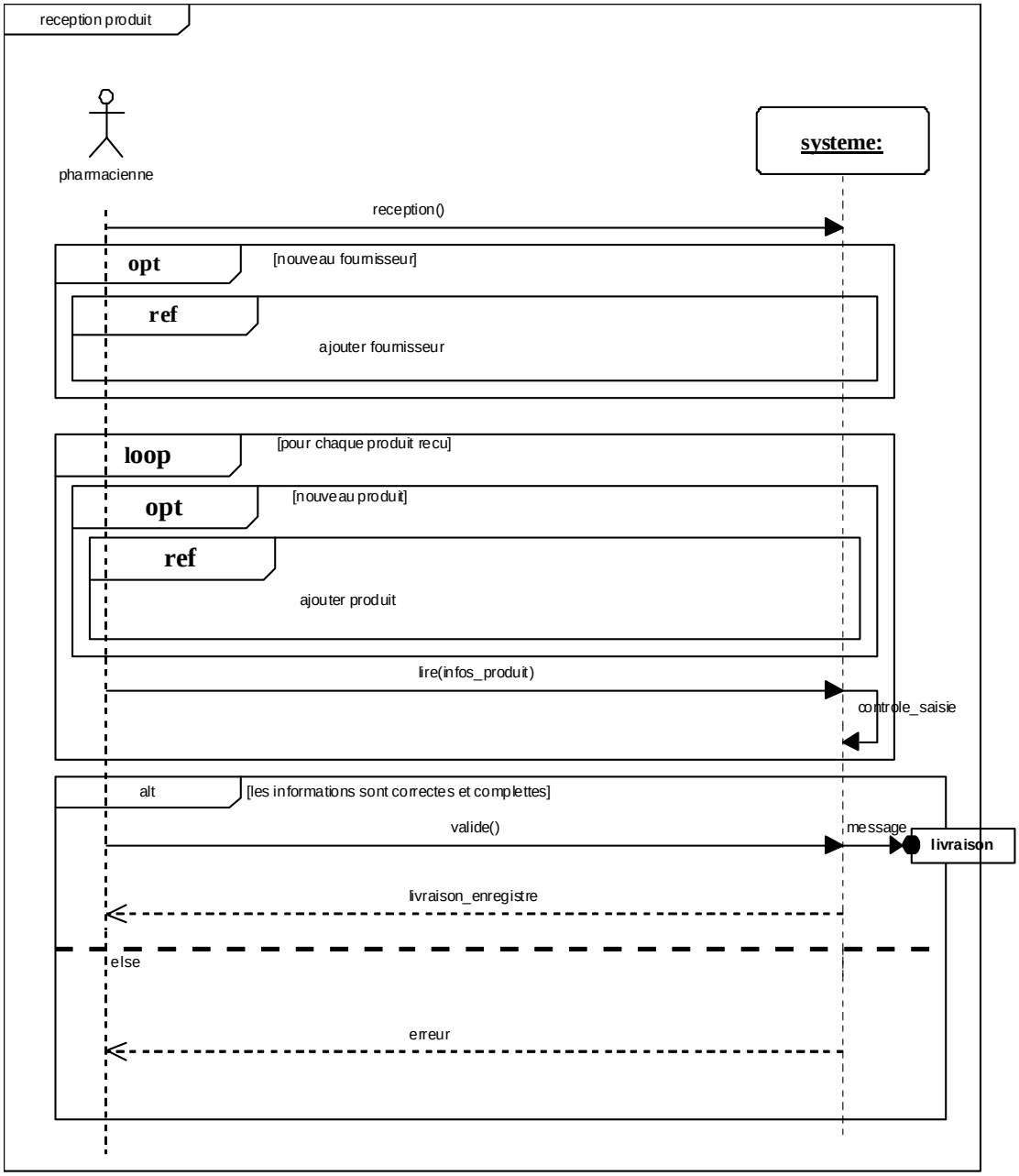


Figure 3.3 : diagramme de séquence du cas d'utilisation réception produit

3.5.3 Vente des produits

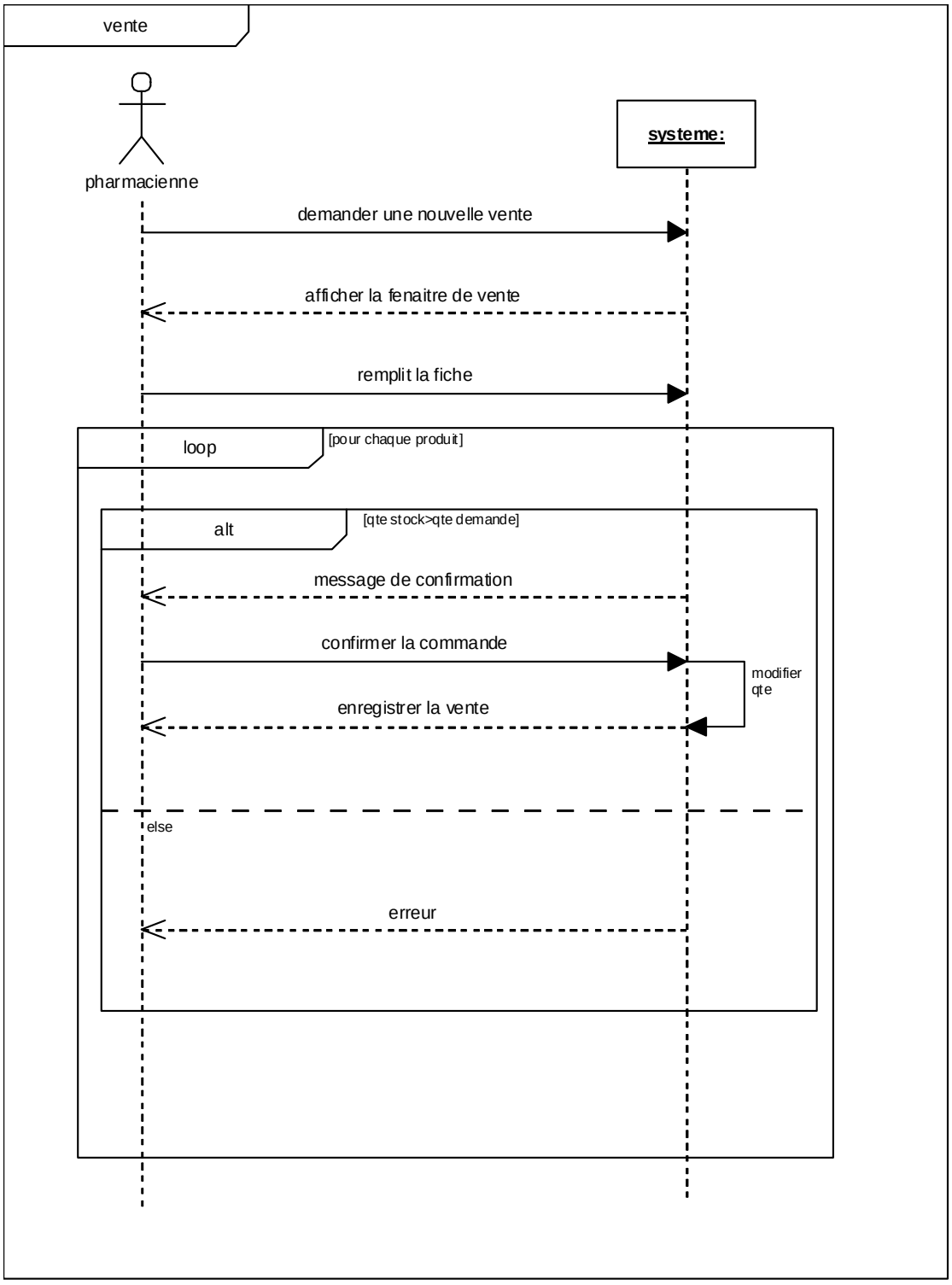


Figure 3.4 : diagramme de séquence du cas d'utilisation vente des produits.

3.5.4 Produit périmé

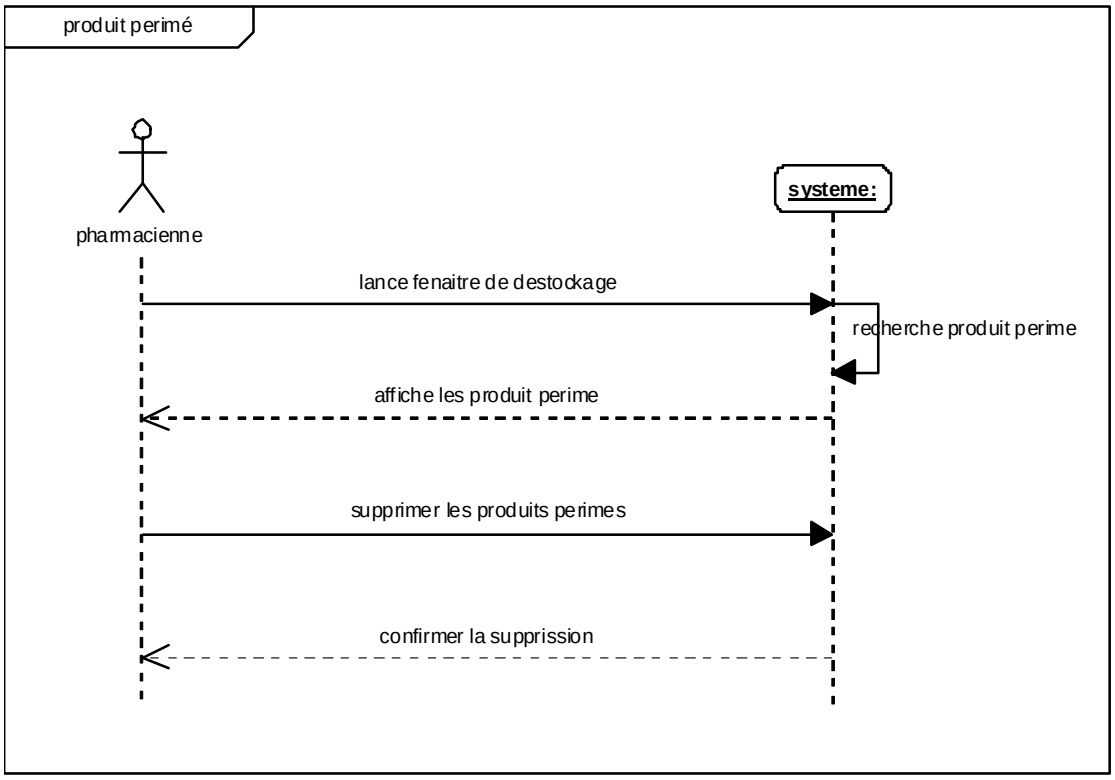


Figure 3.5 : diagramme de séquence du cas d'utilisation produit périmé

3.5.5 Fiche de stock

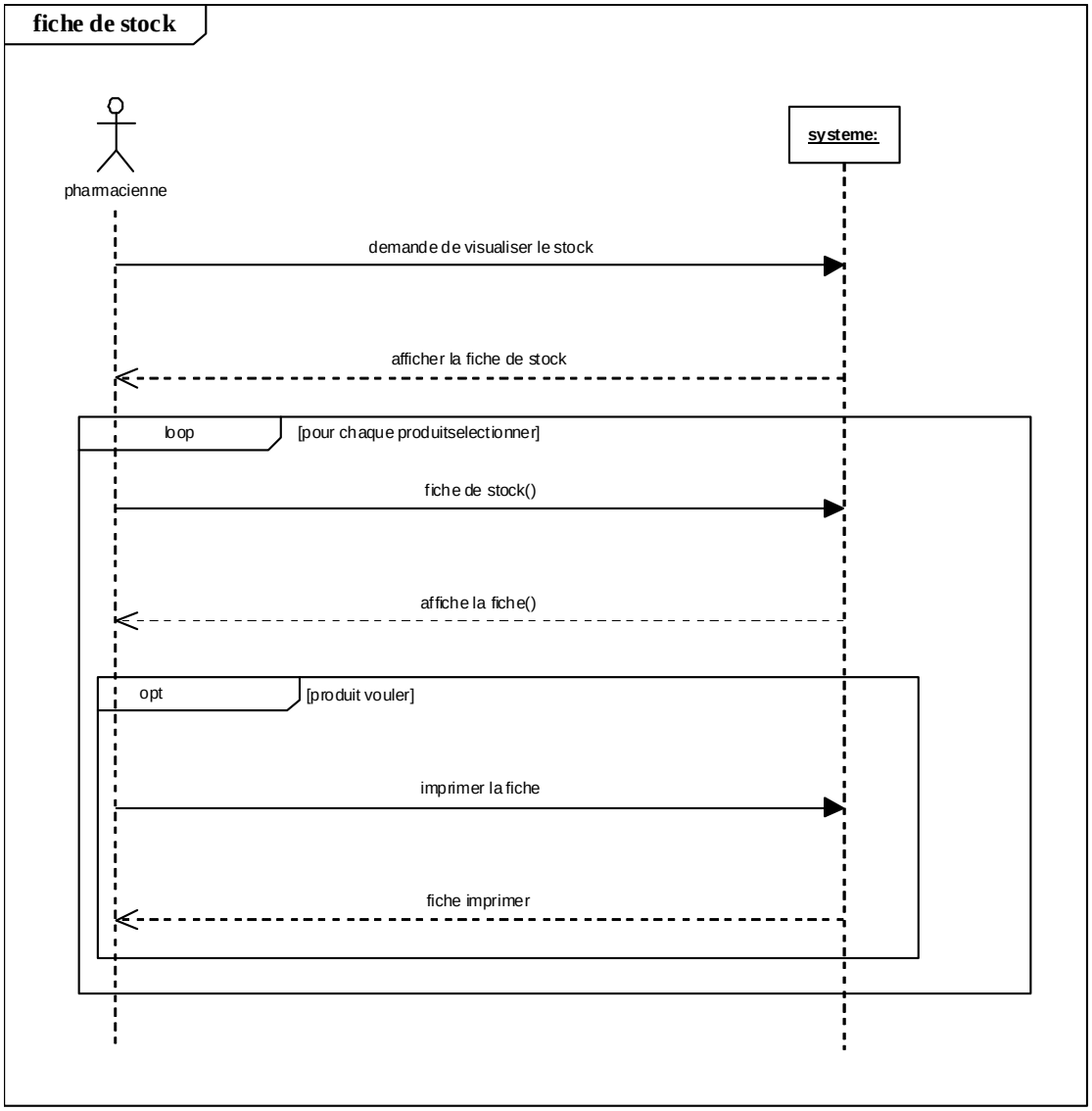


Figure 3.6 : diagramme de séquence du cas d'utilisation fiche stock.

3.5.7 Ajouter fournisseur

*

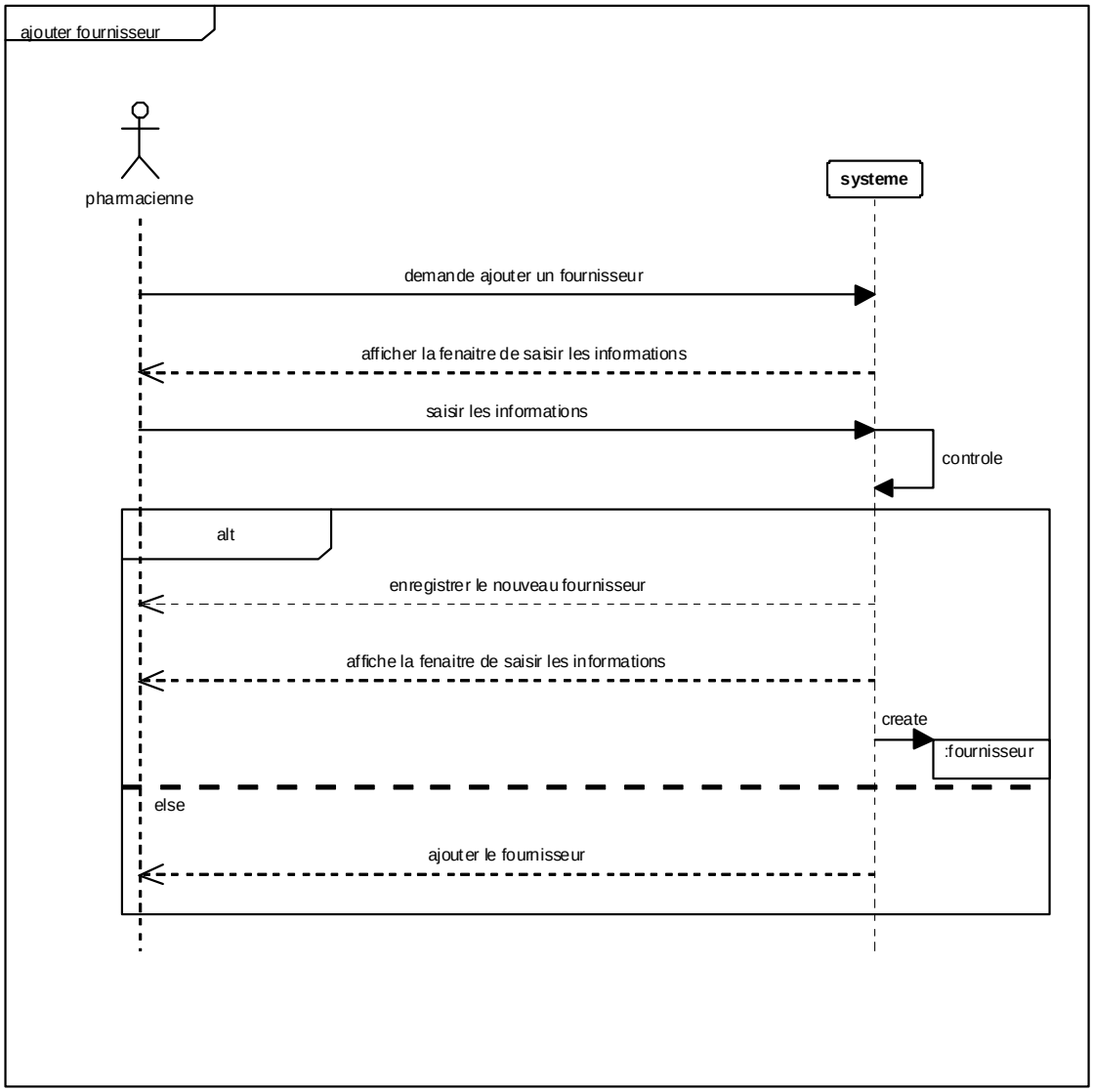


Figure 3.7: diagramme de séquence du cas d'utilisation ajouter fournisseur.

3.5.8 Modifier fournisseur

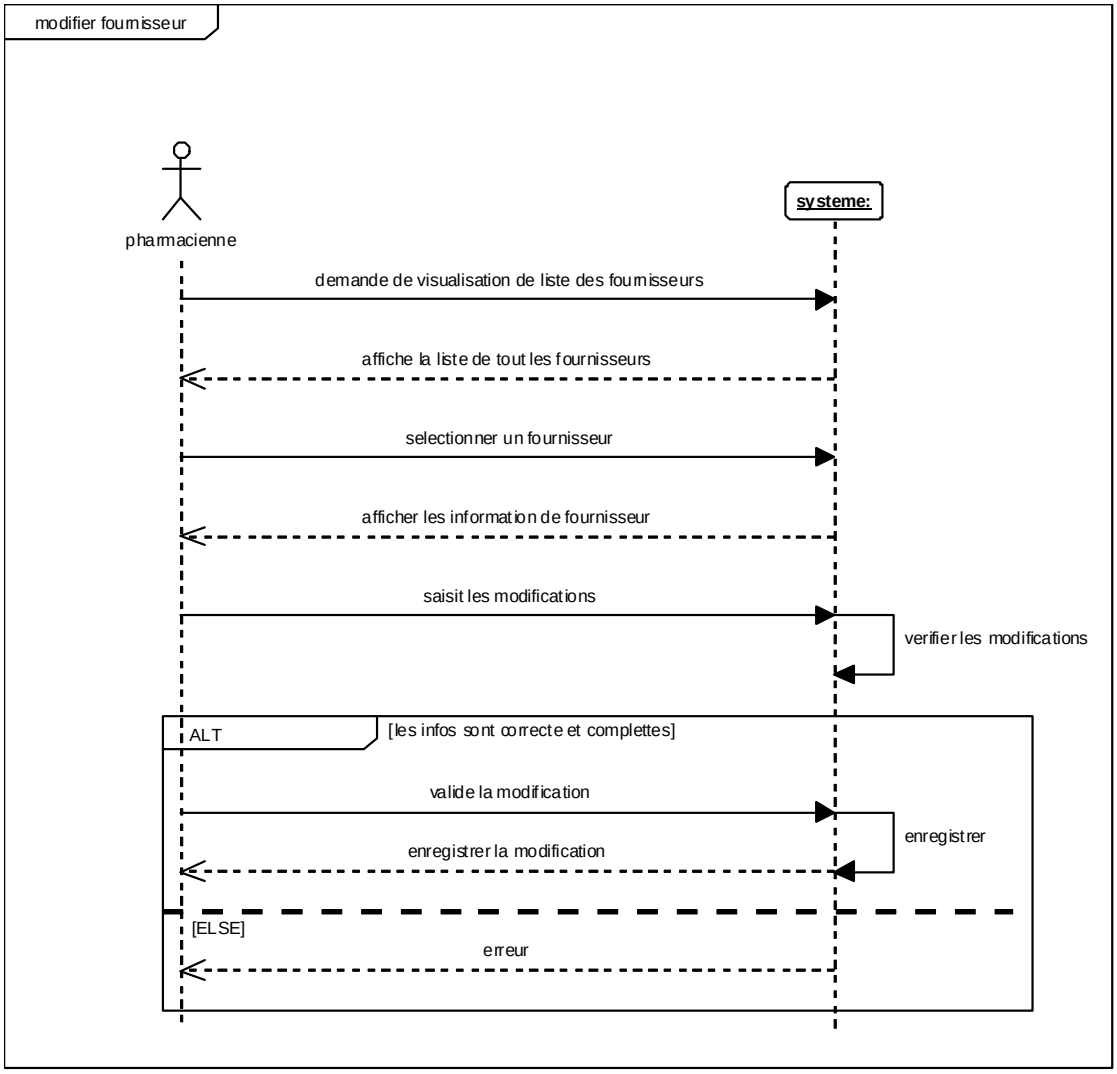


Figure 3.8 : diagramme de séquence du cas d'utilisation modifier fournisseur.

3.5.9 Supprimer fournisseur

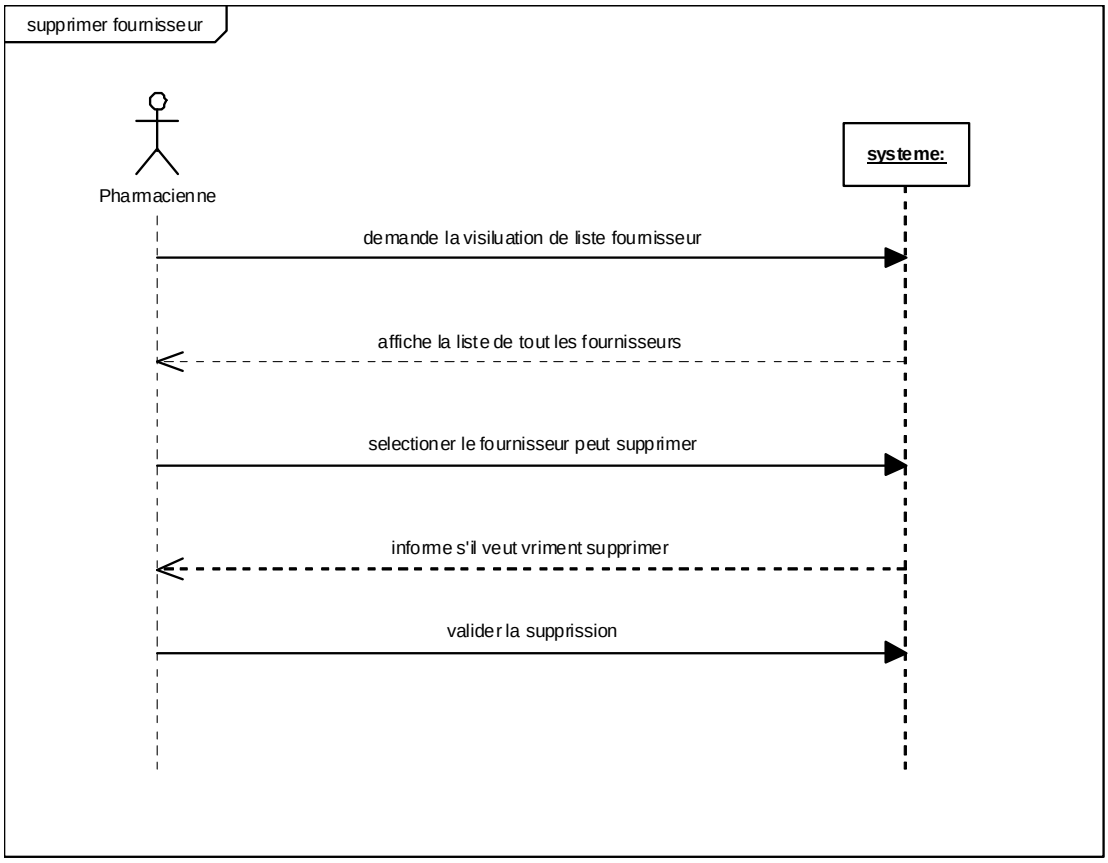


Figure 3.9 : diagramme de séquence du cas d'utilisation supprimer fournisseur.

3.5.10 Ajouter produit

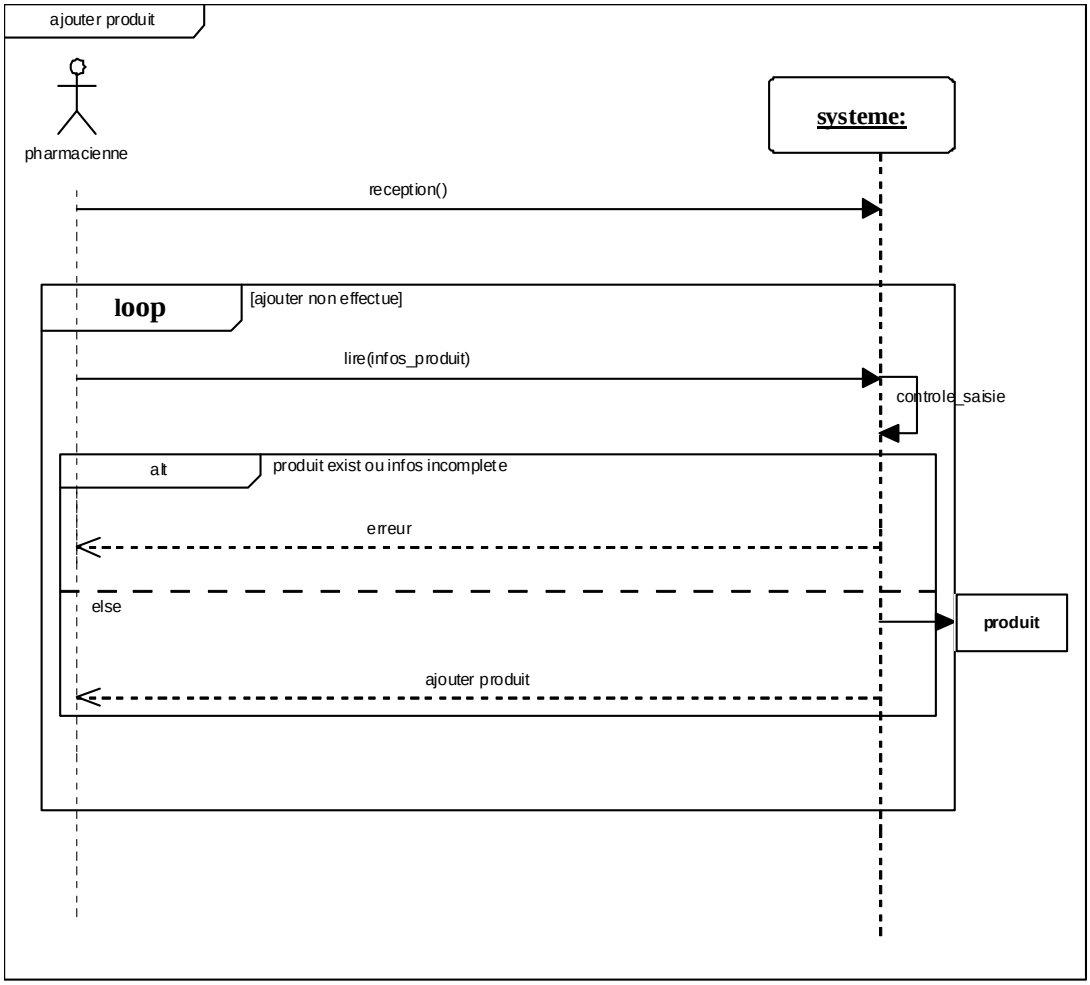


Figure 3.10 : diagramme de séquence du cas d'utilisation ajouter produit.

3.5.11 Modifier produit

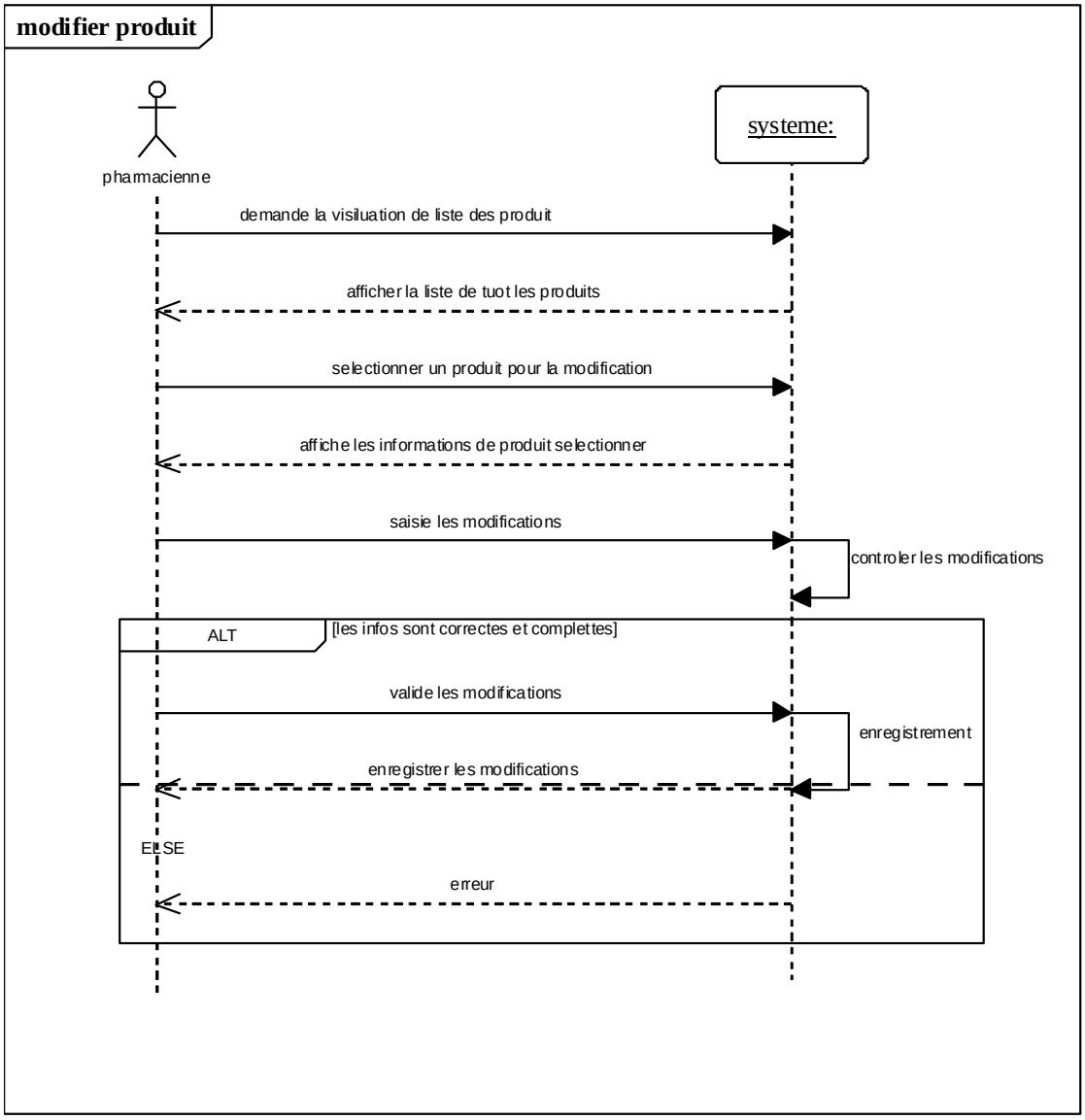


Figure 3.11 : diagramme de séquence du cas d'utilisation modifier produit.

3.5.12 Supprimer produit

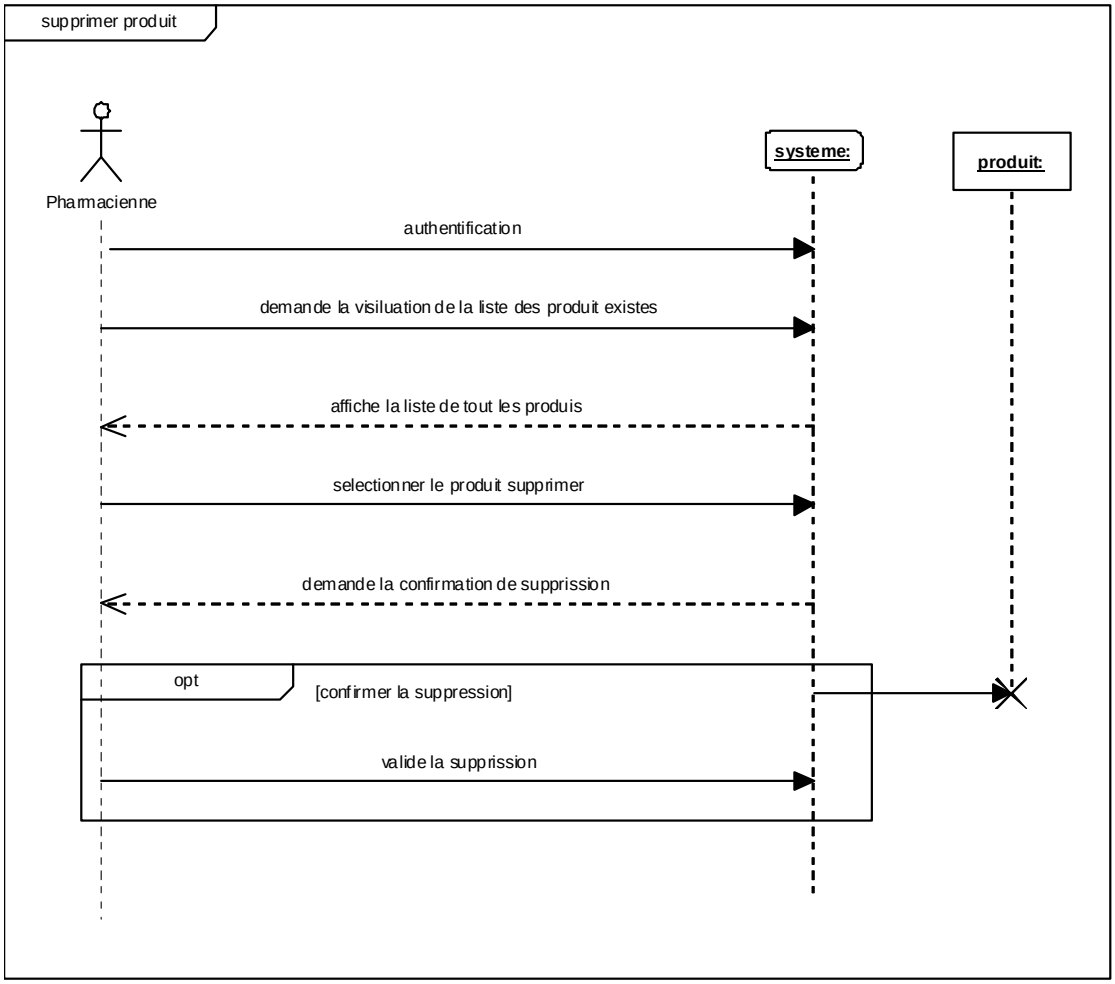


Figure 3.12 : diagramme de séquence du cas d'utilisation « supprimer produit ».

3.6 Diagramme de classe

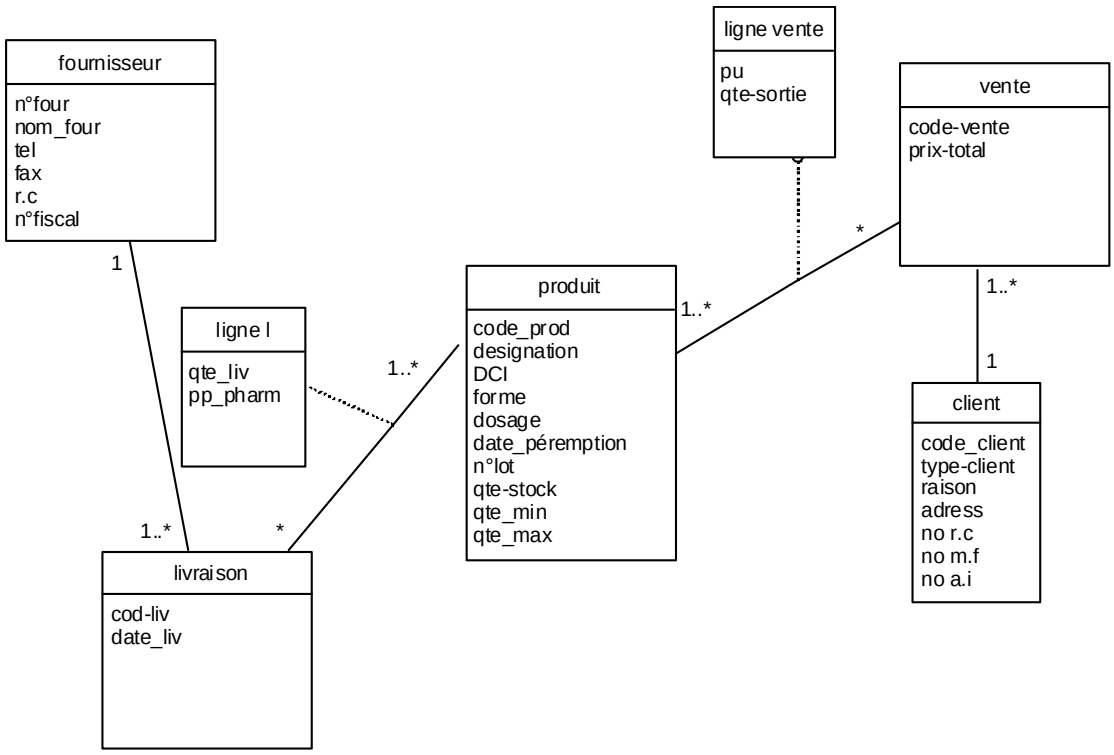


Figure 3.13 : diagramme de classe

3.7 Conclusion

Arrivant à cette étape, on peut dire que nous avons analysé le système qu'on veut construire.

D'autre part, nous signalons que la solution a été mise à jour tout au long de sa réalisation, car nous avons suivi une démarche itérative et incrémentale.

Enfin, la phase de conception abordé, permette de bien cerner la solution proposée et mieux comprendre le fonctionnement du système et ses différentes fonctionnalités, mais surtout permettent de préparer la phase de réalisation qui concrétisera tout ce qui a été présenté jusque-là.

Chapitre IV
Implémentation

4.1 Introduction

Après avoir terminé l'étude conceptuelle, nous pouvons alors passer à l'étape finale de ce mémoire à savoir l'implémentation. Elle a comme objectif d'aboutir à un produit final exploitable par les utilisateurs. Dans cette phase nous allons préparer notre base de données en convertissant le diagramme de classe que nous avons obtenue dans le chapitre précédant en une base de données relationnelle tout en appliquant un certain ensemble de règles que nous détaillant ci-après. Par la suite, nous présenterons les différents outils que nous avons utilisés pour le développement de notre application, puis nous décrivons quelques interfaces de l'application que nous avons extraites via des prises d'écrans (*screenprints*), afin d'illustrer les grandes et principales fonctionnalités réalisées.

4.2 Le passage du diagramme de classe au modèle relationnel

4.2.1 Règles de passage

Les classes entités et leurs associations seront convertit en une base de données relationnelle, qui sera sollicitée par l'application pour consultation et mise à jour. Pour ce faire nous nous sommes basées sur les quatre règles (de R1 à R4) le es plus opérationnelles :

1) Transformation des classes (R1):

la règle est assez simple : chaque classe devient une relation. Les attributs de la classe deviennent des attributs de la relation. Si la classe possède un identifiant, il devient la clé primaire de la relation, sinon, il faut ajouter une clé primaire arbitraire.

2) Transformation des associations

- Association 1..* :

la règle est la suivante :

R2 : il faut ajouter un attribut de type clé étrangère dans la relation fils de l'association. L'attribut porte le nom de la clé primaire de la relation père de l'association.

- Association *..* :

la règle est la suivante :

R3 : association/classe – association devient une relation. La clé primaire de cette relation est la concaténation des identifiants des identités connecté à l'association. Chaque attribut devient clé étrangère si entité/classe connectée dont il devient une relation en vertu de la règle R1. Les attributs d'association/classe – association doivent être ajoutés a la nouvelle relation.ces attributs ne sont ni clé primaire, ni clé étrangère

- Association 1..1 :

La règle est la suivante :

R4 : il faut ajouter un attribut de type clé étrangère dans la relation dérivée de l'entité ayant la cardinalité minimale égale à zéro. Dans le cas de diagramme UML il faut ajouter un attribut de type clé étrangère dans la relation dérivée de la classe ayant la multiplicité minimale égale à un.

L'attribut porte le nom de la clé primaire de la relation dérivée d'entité classe connectée à l'association. Si les deux cardinalités minimales égales à zéro, le choix est donné entre les deux relations dérivées de la R1.

Si les deux cardinalités minimales égales à un, il est préférable de fusionner les deux entités/classe en une seule .

4.2.2 Base de données

Pour gérer notre logiciel en implémenter la base de données par les tables suivantes :

Client (code_client, raison,adress,n°_r_c, no m_f,no a_i, type_client)

Fournisseur (n_four,nom_four,tel,fax,r_c, n°fiscal)

Produit(code_prod,désignation,date_péremption,n°_lot,qte_stock,dosage,forme, qte_max, qte_min,DCI)

Livraison(code_liv,date_arrivage,qte_liv,pp_harm)

4.3 Environnement de développement de l'application

4.3.1 Pourquoi Delphi ?

Delphi est un outil de développement visuel et rapide sous Windows (Rapid Application Development) qui permet de créer des applications fenêtrées, directement exécutables (.EXE) et redistribuables librement sous Windows ou DOS.

Delphi utilise le langage Pascal Orienté Objet (il est toutefois possible d'utiliser d'anciennes sources en Pascal standard grâce au compilateur en ligne de commande). Ce langage est facile à apprendre et beaucoup plus simple que le C++ traditionnel. Les objets utilisés ont des propriétés et des méthodes. Les propriétés sont les caractéristiques de l'objet (couleur, taille, ...) tandis que les méthodes sont les procédures (classiques ou événementielles) et fonctions qui y sont rattachées.

4.3.2 Les avantages de Delphi

L'environnement DELPHI a plusieurs avantages dont on peut citer :

- Il est bien structuré, d'une difficulté moyenne et il donne des applications rapides ;
- Il est disponibles en plusieurs version : version 1 pour windows 3.x, en version 2 et 3 pour windows 95 et NT ;
- Delphi est un environnement de développement de type rad (*rapide application développent*) basé sur le langage pascal. Il permet de réaliser rapidement et simplement des applications windows.
- Delphi propose un ensemble très complet de *composants visuels* prêts à l'emploi incluant la quasi-totalité des composants windows (boutons, boîtes de dialogue, menus, barres d'outils…) permettant de créer facilement divers types d'applications et de librairies ;
- Delphi est un outil moderne, qui fait appel à une conception objet.
- Il prend en charge le maintien automatique d'une partie du code source.
- Il permet de créer facilement de nouveaux composants qui peuvent être intégrés dans la palette des composants déjà existants ;
- Son compilateur intégré permet une application rapide et efficace car les erreurs éventuelles du code sont immédiatement détectées. L'utilisateur est alors informé précisément des erreurs de son programme ;
- Il permet également d'utiliser des formats images, textes, sons, grâce à certains composants ;
- Delphi n'est pas lié à un format de données spécifiques. Il peut en effet utiliser des tables, dbase, access, ou paradox et accéder à des bases de données sql serveur à travers sun odbc (open data base connectivity).

Il faut mentionner que la version utilisé pour le développement de ce projet est **DELPHI 7.**

4.3.3 Implémentation de la base de données sous Access :

Pour implémenter notre base de données nous avons utilisé Microsoft Access. C'est un logiciel qui sert à créer, et à gérer des bases de données relationnelles. Cette base de données a une extension en .mdb . Il s'agit de bases de données accessibles à travers le moteur qui porte le nom de JET.1-4-b utilisé nativement par Access.

4.4 Contrôle et sécurité

4.4.1 Contrôle

L'erreur peut naitre en de nombreux points du circuit de l'information, elle peut être provoquée par un facteur humain ou par une défaillance du matériel.

Il est important de détecter et rectifier l'erreur dés son apparition, car une information non correcte risque de rendre incorrectes de nombreuses autres informations. Par conséquent, différents contrôles s'imposent :

- Vérifier les informations à saisir ;
- Contrôler le type de l'information ;
- Contrôler la présence de l'information ;
- Utiliser les messages d'aides ;
- Des contrôles sont réalisés grâce à des relations de comparaisons.

4.4.2 Sécurité

Le concept de sécurité en informatique est très large, il s'agit de la sécurité de tous les dangers qui menacent tous les systèmes d'informations. Pour assurer le bon fonctionnement du système mis en place et pour éviter les destructions dues imprévues, il est indispensable de prévoir des mesures de sécurité pour protéger l'ensemble des informations contre toutes actions frauduleuses.

Pour cela nous avons distingué plusieurs mesures de sécurité :

- Il faut sauvegarder l'application sur des supports fiables.
- Utiliser les mots de passe.
- Utiliser les anti-virus.

4.5 Interfaces de l'application

4.5.1 Interface authentification

Au lancement de l'application, la fenêtre d'authentification s'affiche. L'utilisateur doit entrer son login et son mot de passe pour pouvoir accéder à son compte.

Figure 4.1 Interface authentification.

4.5.2 Interface du menu principale (Gestion de pharmacie)

Après l'authentification, la fenêtre du menu principale doit s'afficher. Elle permette d'avoir une vue globale sur l'application. Pour que l'utilisateur puisse accéder aux différents services et fonctionnalités offerts, il suffit de cliquer sur le bouton correspondant.

Figure 4.2 Interface menu principale (Gestion de pharmacie).

4.5.3 Interface client

Cette interface permet d'ajouter, supprimer et modifier un client. Ainsi, elle permet de visualiser la liste de tous les clients existants.

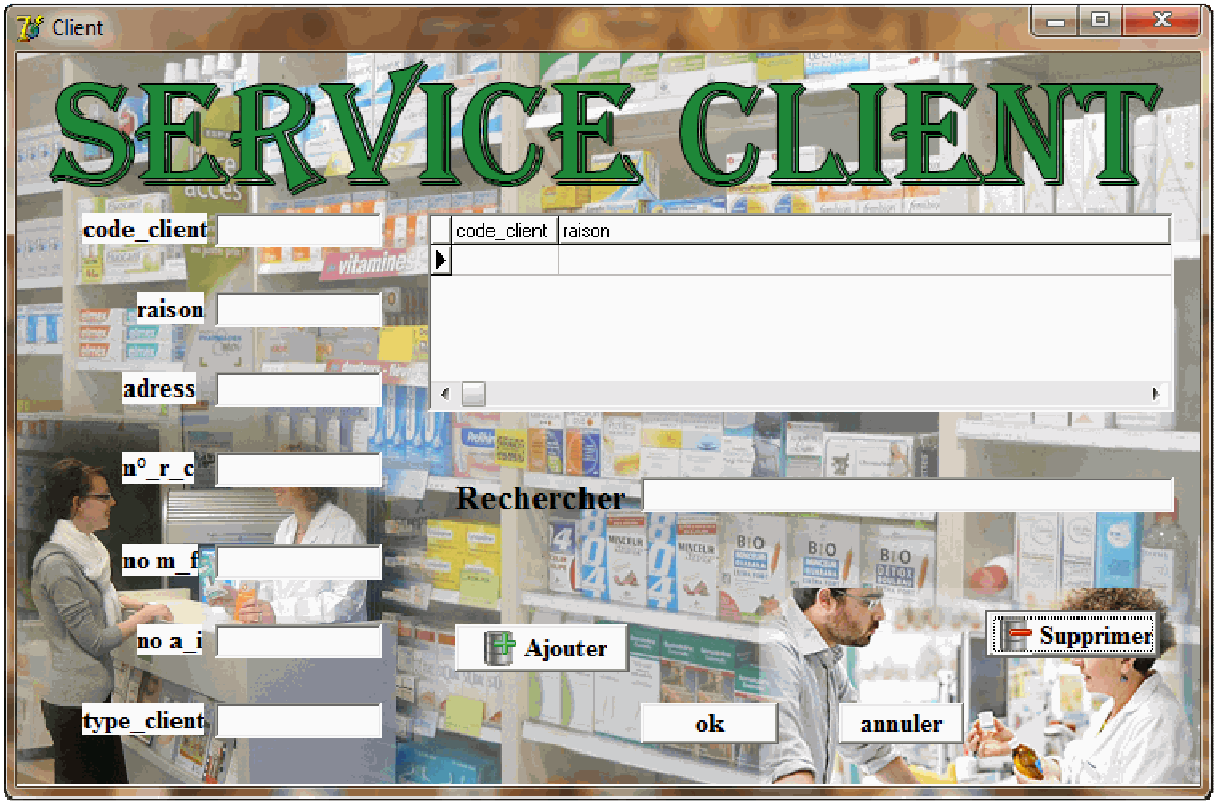


Figure 4.3 Interface client

4.5.4 Interface fournisseur

Cette interface permet d'ajouter, supprimer et modifier un fournisseur. Ainsi, elle permet de visualiser la liste de tous les fournisseurs existants.

Figure 4.4 Interface fournisseur.

4.5.5 Interface produit

Cette interface permet d'ajouter, supprimer et modifier un produit. Elle permet aussi de visualiser la liste de tous les produits existants.

Figure 4.5 Interface produit

4.5.6 Interface livraison

Cette interface permet d'ajouter, chercher, consulter et modifier une livraison, comme elle peut permettre la visualisation du détail de chaque livraison.

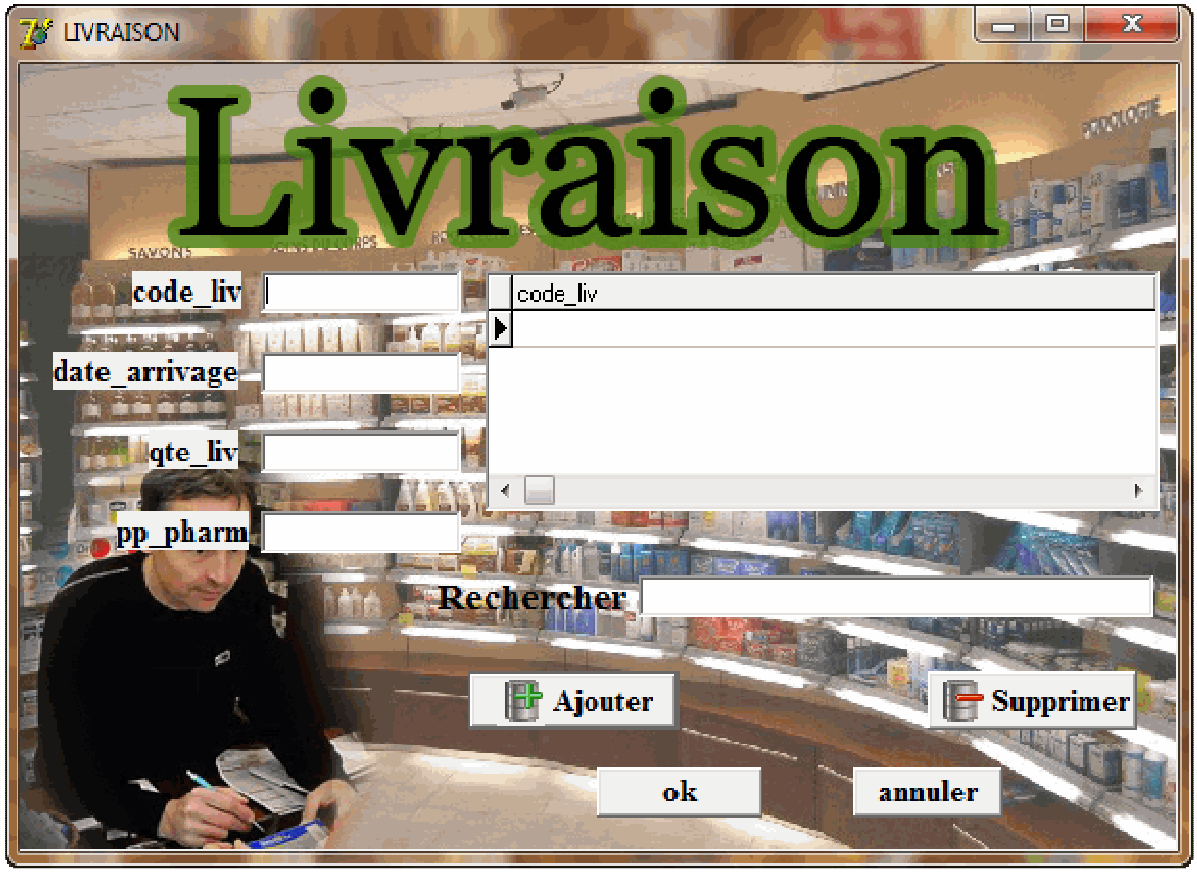


Figure 4.6 Interface livraison.

4.6 Conclusion

Dans ce chapitre, nous avons décrit brièvement la phase d'implémentation tout en donnant une vue globale sur les outils utilisés ainsi que l'application que nous avons réalisée. Nous avons présenté l'interface de l'application avec toutes les options qu'elle contient afin de faciliter son utilisation.

1. Bilan

Ce projet qui s'inscrit dans le cadre de la gestion du stock au niveau d'une pharmacie, était à l'objectif de répondre aux deux principaux processus de la gestion du stock, à savoir : la gestion des Entrées/Sorties de des produits. Pour ce faire, nous sommes passés par plusieurs phases, allant de la collecte d'informations acquises de l'étude de l'existant, passant par la suite par la phase d'analyse et de conception. Durant cette étape, nous avons eu recours au langage de modélisation UML tout en suivant la méthode proposée par LAURENT AUDIBERT. Pour finir, nous avons opté pour la mise en œuvre de l'application, l'environnement de développement DELPHI.

Au terme de notre travail, nous espérons avoir mis en place un système d'information qui offre une solution qui prend en charge les principaux besoins de la pharmacie même si des mises à jour doivent être apportées au système développé.

En fait, à la fin de la réalisation de ce mémoire, il est important de dire que ce projet nous a permis d'exploiter nos connaissances théoriques acquises pendant le cycle de notre formation. En outre, ce travail nous a donné un avant-goût du métier de développeur et il nous a permis de concevoir pour la première fois une vraie application pour un vrai client du fait que nous nous somme mieux rapprochés et familiarisés avec un environnement dynamique. Par conséquent, nous avons pu avoir une idée plus approfondie sur l'importance des systèmes d'informations dans les entreprises.

Enfin, nous pouvons dire que ce travail constitue une grande satisfaction personnelle et professionnelle qui signe le début d'aboutissement de notre formation.

2. Perspectives

Malgré les efforts consentis pour la mise en œuvre de ce système, il reste néanmoins primordial de lui appliquer de probables correctifs et améliorations nécessaires pour qu'il soit correctement exploitable.

Pour finir, nous prévoyons comme perspectives du travail réalisé dans ce mémoire, l'enrichissement du système actuel par l'implémentation des fonctionnalités suivantes :

- Le suivi de l'opération d'approvisionnement auprès du fournisseur;
- La gestion de la facturation et le suivi des paiements;

Et la gestion d'inventaire.

Bibliographie

[1]: *Benoît Charroux, Aomar Osmani, and Yann Thierry-Mieg. UML2. Pearson Education France, 2005*

[2]: *Pascal ROQUES, UML 2 par la pratique étude de cas et exercices corrigés, ÉDITIONS EYROLLES, Septembre 2006.*

[3]: *Laurent AUDIBERT, UML 2-de l'apprentissage à la pratique (cour et exercice)*