Dossier conception : Gestion Médicale

Réalisé par :

Sabah Bnouachir
Asma Jmari

Encadré par :

Mr LACHGAR Mohamed

Date de rédaction :
05/04/2013

Cadre réservé à l'encadrant :

Code d'identification du Candidat : ……

Nom des Validateurs	Commentaires :

Sommaire

Ce chapitre présente l'étude préliminaire du projet, qui consiste à effectuer un premier repérage des besoins fonctionnels et techniques du projet.

A cet effet, on a conçu un diagramme des cas d'utilisation du système, décrivant ses fonctionnalités depuis le cahier des charges.

Chapitre 1

Etude Préliminaire et fonctionnelle

Axes du Chapitre :

- Etude préliminaire
- Présentation du projet
- Etude fonctionnelle
- Capture des besoins fonctionnels
- Les fonctionnalités couvertes par le projet

I. Etude préliminaire et fonctionnelle

I. Étude préliminaire

L'étude préliminaire (ou Pré-étude) est la toute première étape du processus 2TUP. Elle consiste à effectuer un premier repérage des besoins fonctionnels et opérationnels, en utilisant principalement le texte, ou diagrammes très simples. Elle prépare les activités les plus formelles de capture des besoins fonctionnels et de capture techniques.

i. Présentation du projet

Ce projet de fin d'année a pour objectif de créer une application qui permettra d'établir la gestion d'une clinique. La page d'accueil est dédié à l'utilisateur (médecin, infermière, administrateur ou secrétaire) devra saisir ses identifiants. Apres si l'opération s'est bien déroulé l'utilisateur aura accès aux informations selon son droit d'accès, cependant s'il les identifiants ne sont pas correctes il sera aussi tôt averti avec un message d'erreur en lui demandant de ressayer à nouveau.

En effet, après la phase d'authentification, et selon le droit attribué, l'utilisateur pourra consulter les informations qui lui sont attribué, le gestionnaire sera chargé de l'information liée à l'activité d'organisation.

L'administrateur qui est chargé de la gestion des utilisateurs en premier lieu, lui seul aura le droit de traité les informations lies aux utilisateurs.

Afin de répondre à ces objectifs, le projet doit inclure les modules suivants :

Un module d'administration :

Le but de ce module est de définir les fonctionnalités suivantes :

- Gestion des employés.
- Gestion des chambres.

Un module organisation :

Le but de ce module est de définir les fonctionnalités suivantes :

- Gestion des patients.
- Gestion des rendez-vous.
- Gestion paiement.
- Gestion des employés.
- Gestion des mesures.

II. Étude fonctionnelle

i. Capture des besoins fonctionnels

Cette phase représente un point de vue « fonctionnel » de l'architecture système. Par le biais des cas d'utilisation, nous serons en contact permanent avec les acteurs du système en vue de définir les limites de ceux-ci, et ainsi éviter de trop s'éloigner des besoins réels des utilisateurs finaux.

ii. Les fonctionnalités couvertes par notre application

Les fonctionnalités couvertes par notre projet sont les suivantes :

- Gestion des employés.
- Gestion des patients (consultation et recherche des patients, ajout des Patients,

  modification des patients).
- Gestion des rendez-vous et des consultations.
- Gestion des chambres.
- Gestion des mesures (consultation et recherche des mesures, ajout des mesures, modification des mesures).
- Gestion de paiements.

Conclusion

Au terme de ce chapitre, nous avons présenté l'étude préliminaire du projet, qui sert à avoir un

premier regard sur les besoins fonctionnels et techniques du projet.

Nous avons présenté également l'étude fonctionnelle du projet réalisé en respectant le processus de développement 2TUP. Ainsi, nous avons traité la branche gauche du modèle, où nous prêtons attention à la capture des besoins fonctionnels.

Analyse et conception

Axes du chapitre :

- Modèle de conception
- Présentation UML
- Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme de séquence
- Diagramme d'état de transition
- Diagramme d'activité
- Diagramme de déploiement

A :Architecture couches pour l'application nationale

L'application nationale est divisée en cinq couches de fonctionnalités, totalement autonomes les unes des autres, et communiquant par un système de file : chaque couche ne dialogue qu'avec les couches voisines supérieure et inférieure. Toutes les couches doivent agir de
façon transparente les unes des autres. La séparation des couches est la suivante :

Couche Données : cette couche contient les données physiques stockées dans la base de données. Elle ne requiert pas d'implémentation Java particulière et fonctionne simplement commun espace de consultation massif.

Couche Mapping **:** cette couche contient l'implémentation des accès à la base de données afin de la masquer à la couche métier. Cette couche est entièrement gérée par Hibernate.

Couche Métier : cette couche contient les objets métiers de l'application. Il existe un objet par Fonctionnalité de l'application. Ces objets implémentent les fonctionnalités spécifiques relatives d'un clinique médicale, et font le lien entre la couche contrôleur et la couche mapping.

Couche Application : cette couche contient la partie fonctionnelle de l'application. Elle s'appuie sur les objets métier pour réaliser les actions sollicitées par l'utilisateur par l'intermédiaire de la couche présentation. Elle est en charge de vérifier la validité des requêtes de la couche présentation. Il existe un contrôleur par fonctionnalité de l'application.

II. Analyse et Conception

I. Modèles de conception

Avantages de l'utilisation des modèles de conception

Réutilisation : Ils fournissent une solution prête pouvant s'adapter à différents problèmes.
Ils sont expressifs : Ils fournissent un vocabulaire commun de solutions pouvant exprimer des solutions très larges.
Ils ont été prouvés : Les modèles de conception reflètent l'expérience, les connaissances et la perspicacité des développeurs qui ont réussi en utilisant ces derniers dans leurs travaux.

II. Présentation UML (langage de modélisation unifié)

UML (UnifiedModelingLanguage, que l'on peut traduire par "langage de modélisation unifié) est une notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion de plusieurs méthodes existant auparavant, et est devenu désormais la référence en terme de modélisation objet, à un tel point que sa connaissance est souvent nécessaire pour obtenir un poste de développeur objet.

Il est basé sur la programmation orientée objet, cette dernière consiste à modéliser informatiquement un ensemble d'éléments d'une partie du monde réel (que l'on appelle domaine) en un ensemble d'entités informatiques. Ces entités informatiques sont appelées objets. Il s'agit de données informatiques regroupant les principales caractéristiques des éléments du monde réel(taille, la couleur, ...).

La difficulté de cette modélisation réside dans la création d'une représentation abstraite, sous forme

d'objets, d'entités ayant une existence matérielle (chien, voiture, ampoule, ...) ou bien virtuelle

(sécurité sociale, temps, ...).

b. Diagramme des cas d'utilisations

I. Définition

Les diagrammes de cas d'utilisation permettent de décrire les grandes fonctionnalités du système du point de vue des utilisateurs. Un cas d'utilisation est un service rendu par le système. Les cas d'utilisation sont organisés en package.

II. Rôle du diagramme des cas utilisation

Donne une vue du système dans son environnement extérieur ; Définit la relation entre l'utilisateur et les éléments que le système met en œuvre.

III. Identification des acteurs

Un acteur représente l'abstraction d'un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système) qui interagissent directement avec le système étudié. Les principaux profils qui auront à utiliser le SI sont les suivants :

Acteur	**Description**
Secrétaire	Possède des droits sur la gestion des rendez vous, des patients
infermière	Possède des droits la modification d'une fiche médicale d'un patient
Médecin	Donne une prescription, fait le suivi d'un dossier patient

Description détaillée des cas d'utilisation

Les cas d'utilisation du système s'étalent sur plusieurs package qui détaillent les fonctionnalités
du système :

[1] Secrétaire ;
[2] médecin;
[3] infermière ;

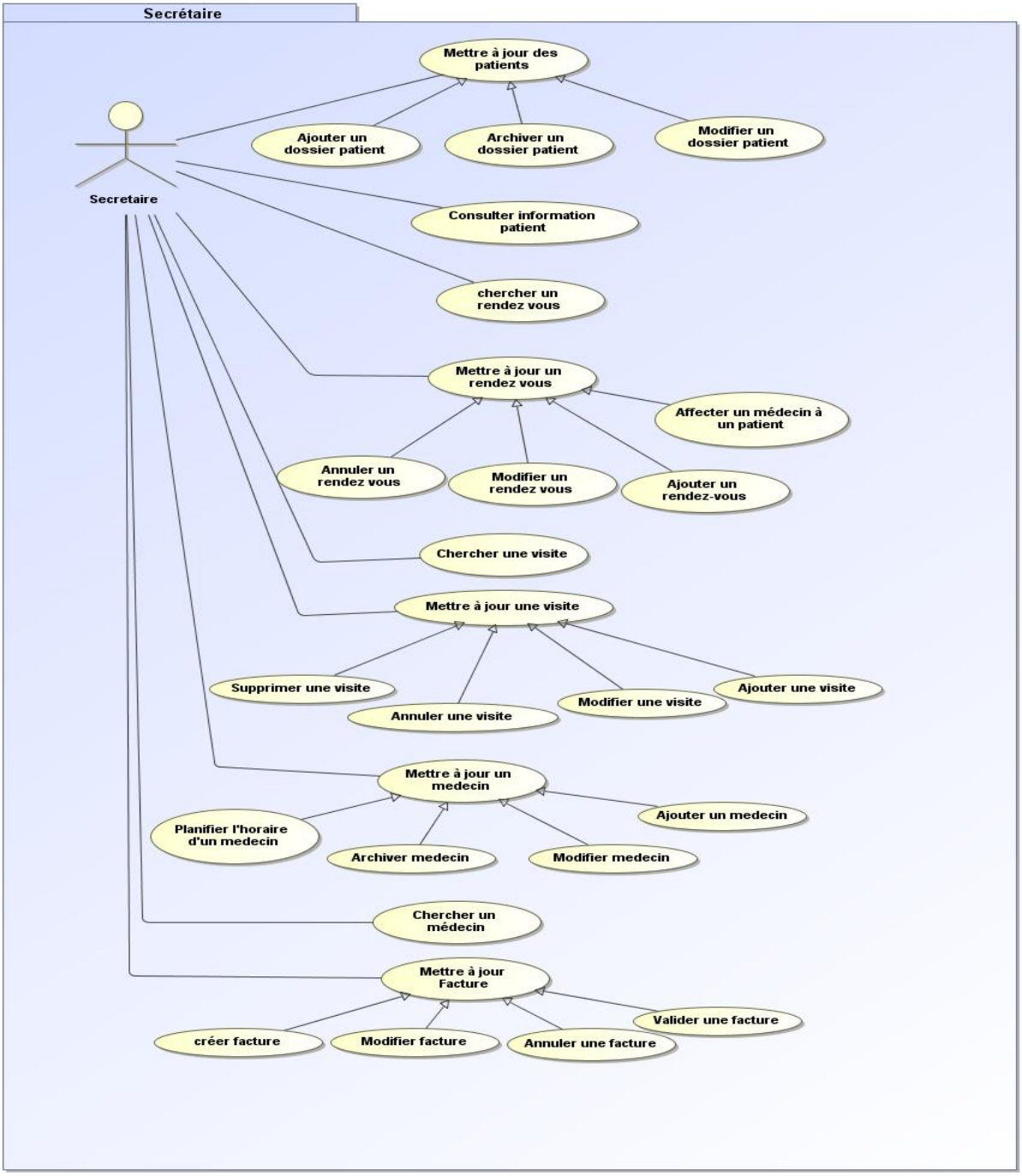


Figure 1 : Diagramme use case secrétaire

description de la figure : la **Figure 1** ci dessus montre ce qu'une secrétaire réalise comme tâches dans le système

Sommaire d'identification :

Titre : Gestion des Secrétaires
Objectifs :
Résumé : Cette fonctionnalité permet :
- À la secrétaire de Ajouter un dossier, de Modifier un dossier, Archiver un dossier du patient, Planifier l'horaire des médecins et des chambres, ajouter rendez vous, modifier rendez vous, supprimer rendez vous, ajouter une visite, modifier une visite, annuler une visite, Affecter un patient à un médecin.

Acteurs : Secrétaire

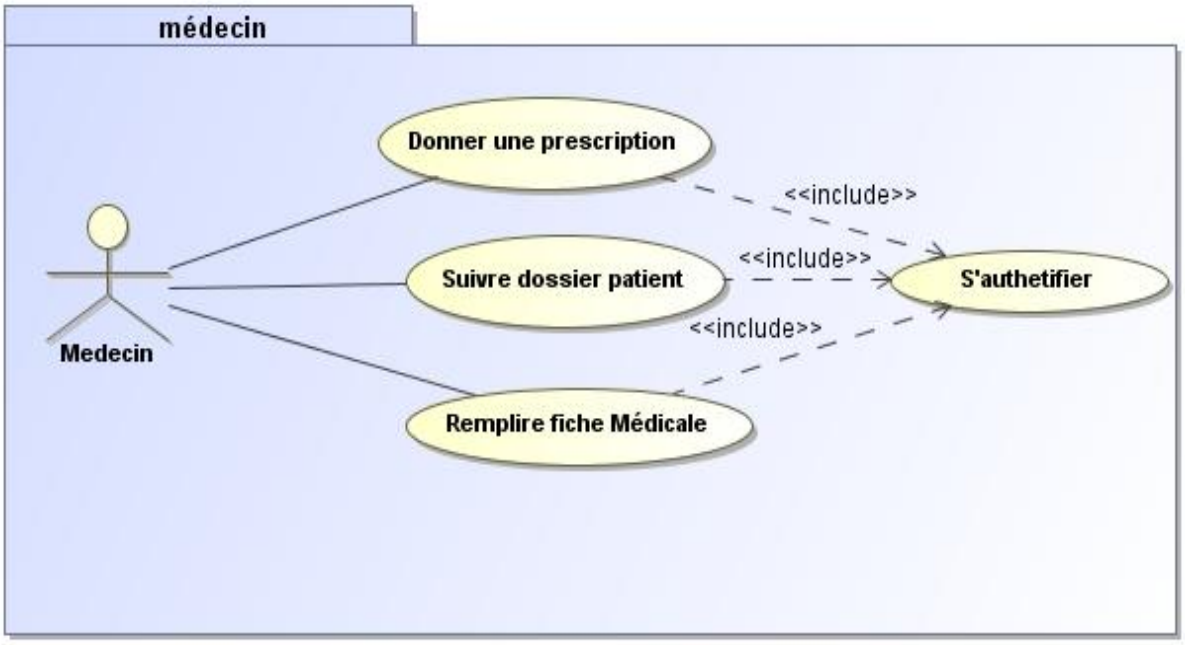


Figure 2 : Diagramme use case médecin

description de la figure : la **Figure 2** ci dessus montre ce qu'un médecin réalise comme tâches dans le système

Titre : Gestion des médecins
Objectifs :
Résumé : Cette fonctionnalité permet :
- le médecin donner une prescription, suivre le dossier du patient, remplis fiche médicale.

Acteurs : médecin

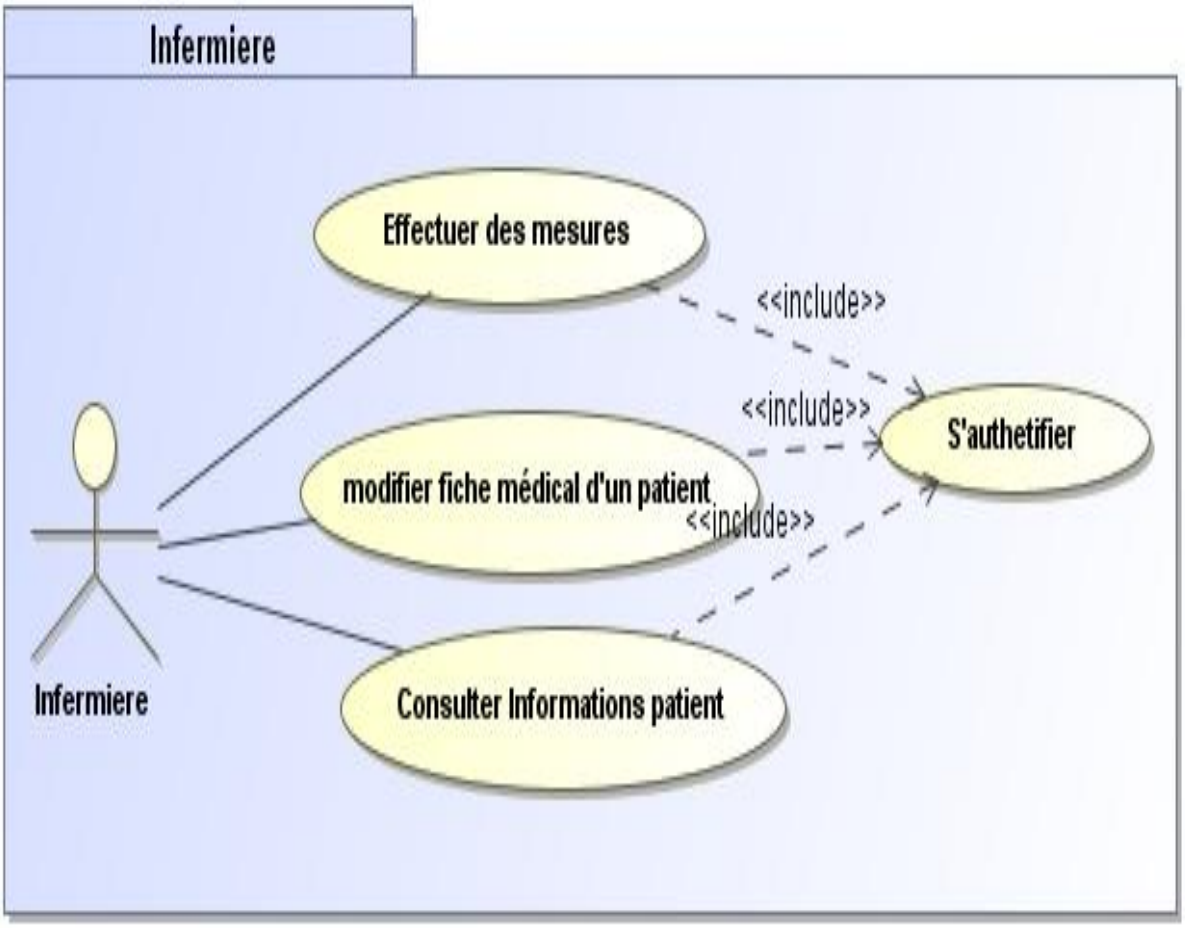


Figure 3 : Diagramme use case infermière

description de la figure : la **Figure 3** ci dessus montre ce que l'infermière réalise comme tâches dans le système

Titre : Gestion des infermières
Objectifs :
Résumé : Cette fonctionnalité permet :
- l'infermière de modifier une fiche médicale, effectuer des mesures, consulter informations patient
Acteurs : infermière

Description détaillée :

- **Pré conditions :** Les différents acteurs doivent s'authentifier pour avoir accès aux fonctionnalités du système.
- **Description du traitement nominal :**

La Secrétaire peut :

1. Ajouter un dossier ;
2. Modifier un dossier;
3. Archiver un dossier des médecins;
4. Planifier des chambres patientes;
5. Planifier l'horaire des médecins.
6. Affecter un médecin a un patient.
7. Affecter un patient a un médecin.
8. ajouter un rendez vous
9. modifier un rendez vous
10. supprimer un rendez vous
11. ajouter une visite
12. modifier une visite
13. annuler une visite

Le infermière peut :

1. modifier fiche médicale patient
2. affecter mesures
3. Consulter dossier patient.

Le médecin peut :

1. donner une prescription;
2. suivre le dossier ;
3. remplis fiche médicale;

- **Exceptions :**

[Exception 1 : ChampsObligatoires] : Message d'erreur si l'un des champs obligatoires n'est pas rempli.

[Exception 2 : ErreurDate] : Message d'erreur si la date de début est postérieure à la date de fin d'un projet ou d'une tâche.

UC2 : Administration

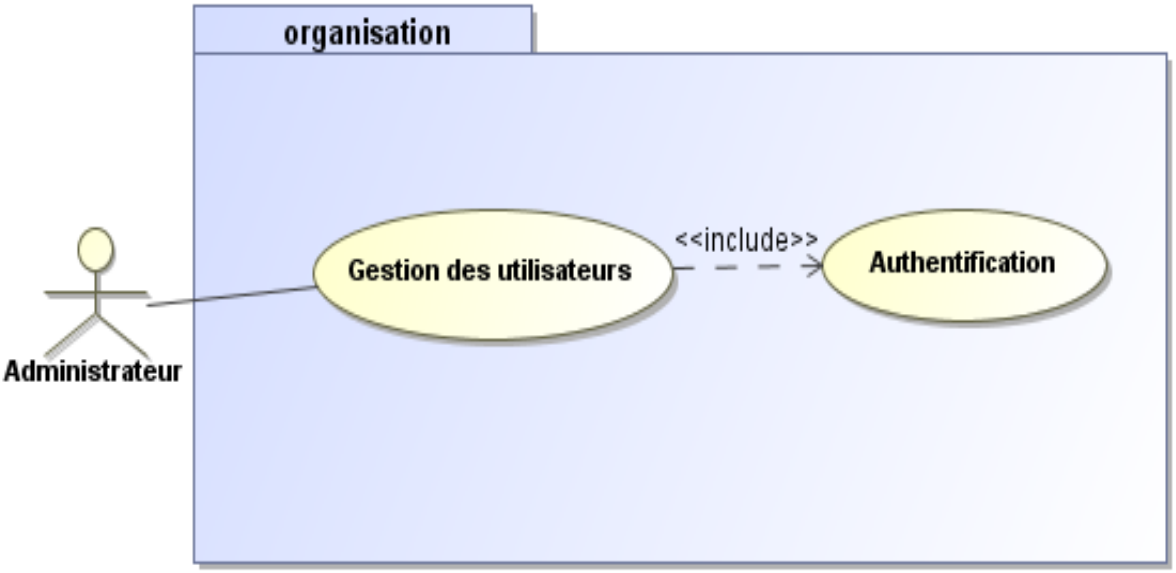


Figure 4 : Diagramme use case Administrateur

description de la figure : la **Figure 4** ci dessus montre ce que l'administrateur réalise comme tâches dans le système

Sommaire D'identification :

Titre : Gestion des utilisateurs
But : Ajouter, modifier et supprimer des utilisateurs (secrétaires, infirmière, médecin).
Résumé : Cette fonctionnalité permet à l'administrateur d'ajouter, modifier et supprimer des utilisateurs
Acteur : Administrateur.

Description Détaillée :

- **Pré conditions :** L'administrateur s'est authentifié sur le système
- **Description du traitement nominal :** l'acteur peut :
  1. Ajouter un utilisateur ;
  2. Modifier un utilisateur ;
  3. Supprimer un utilisateur ;
  4. Chercher un utilisateur.
- **Exceptions :**
  [Exception 1 : Champs Obligatoires] : Message d'erreur si l'un des champs obligatoires n'est pas rempli.

c. Diagramme de classes

I. Définition

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il est le seul obligatoire lors d'une telle modélisation.

Alors que le diagramme de cas d'utilisation montre un système du point de vue des acteurs, le diagramme de classes en montre la structure interne. Il permet de fournir une représentation abstraite des objets du système qui vont interagir ensemble pour réaliser les cas d'utilisation. Il est important de noter qu'un même objet peut très bien intervenir dans la réalisation de plusieurs cas d'utilisation. Les cas d'utilisation ne réalisent donc pas une partition1 des classes du diagramme de classes. Un diagramme de classes n'est donc pas adapté (sauf cas particulier) pour détailler, décomposer, ou illustrer la réalisation d'un cas d'utilisation particulier.

Il s'agit d'une vue statique car on ne tient pas compte du facteur temporel dans le comportement du système. Le diagramme de classes modélise les concepts du domaine d'application ainsi que les concepts internes créés de toutes pièces dans le cadre de l'implémentation d'une application. Chaque langage de Programmation Orienté Objets donne un moyen spécifique d'implémenter le paradigme objet (pointeurs ou pas, héritage multiple ou pas, etc.), mais le diagramme de classes permet de modéliser les classes du système et leurs relations indépendamment d'un langage de programmation particulier.

Les principaux éléments de cette vue statique sont les classes et leurs relations : association, généralisation et plusieurs types de dépendances, telles que la réalisation et l'utilisation.

II. Diagrammes de classes

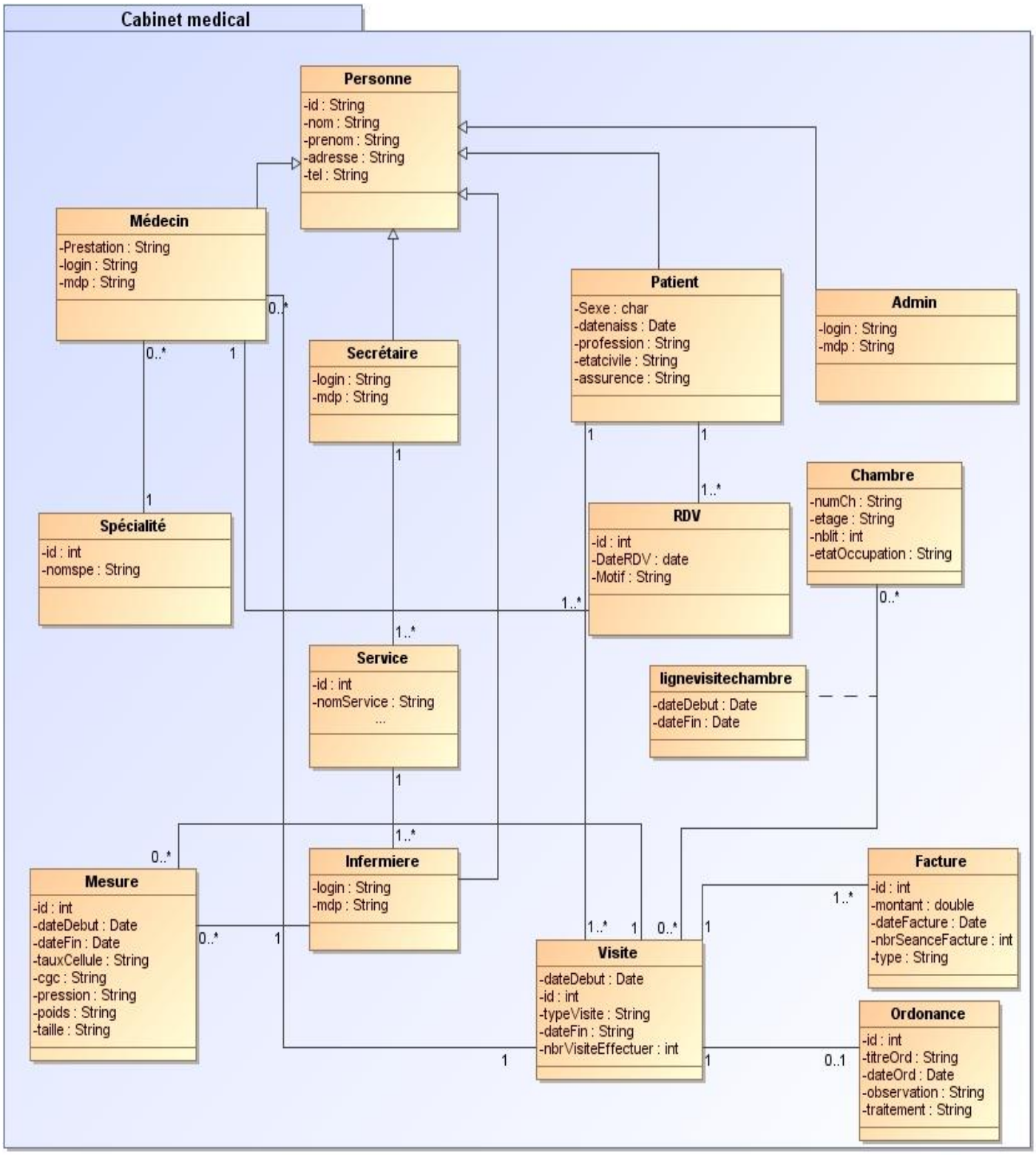


Figure 5 : Diagramme de classe

description de la figure : la **Figure 5** ci dessus montre les relations présente entre les objets du système

a. Diagrammes de séquences du système « AjoutRDV » , « AjoutVisite »:

I. Définition

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique dans la formulation UML.

II. Diagramme de séquence du système : ajout d'un rendez-vous

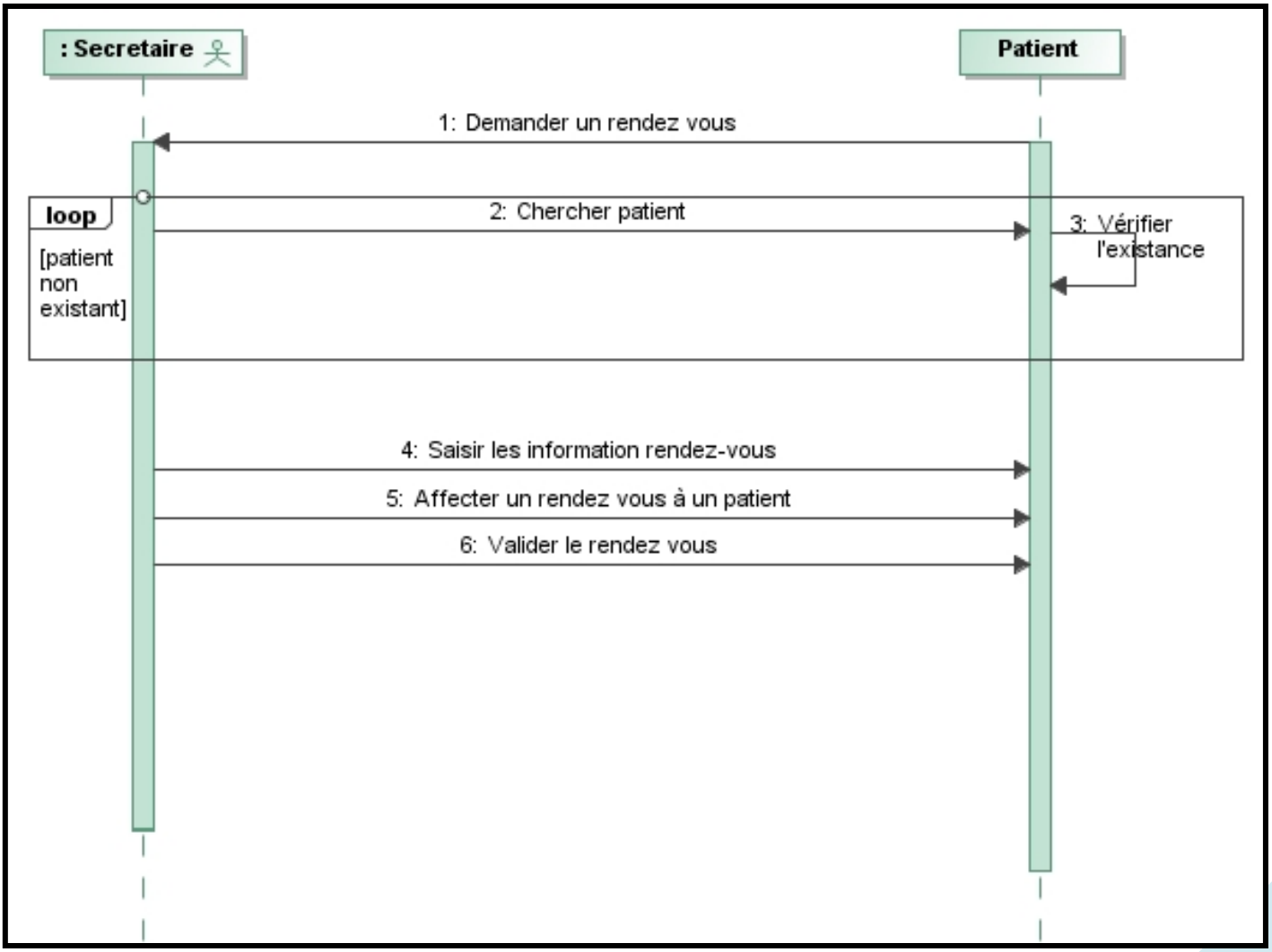


Figure 6 : Diagramme de séquence boîte blanche du cas d'utilisation "Ajout d'un rendez vous"

description de la figure : la **Figure 6** ci dessus décrit l'interaction de la secrétaire avec l'objet patient pour réaliser le cas "ajout d'un rendez vous"

I. Diagramme de séquence système : ajout d'une visite

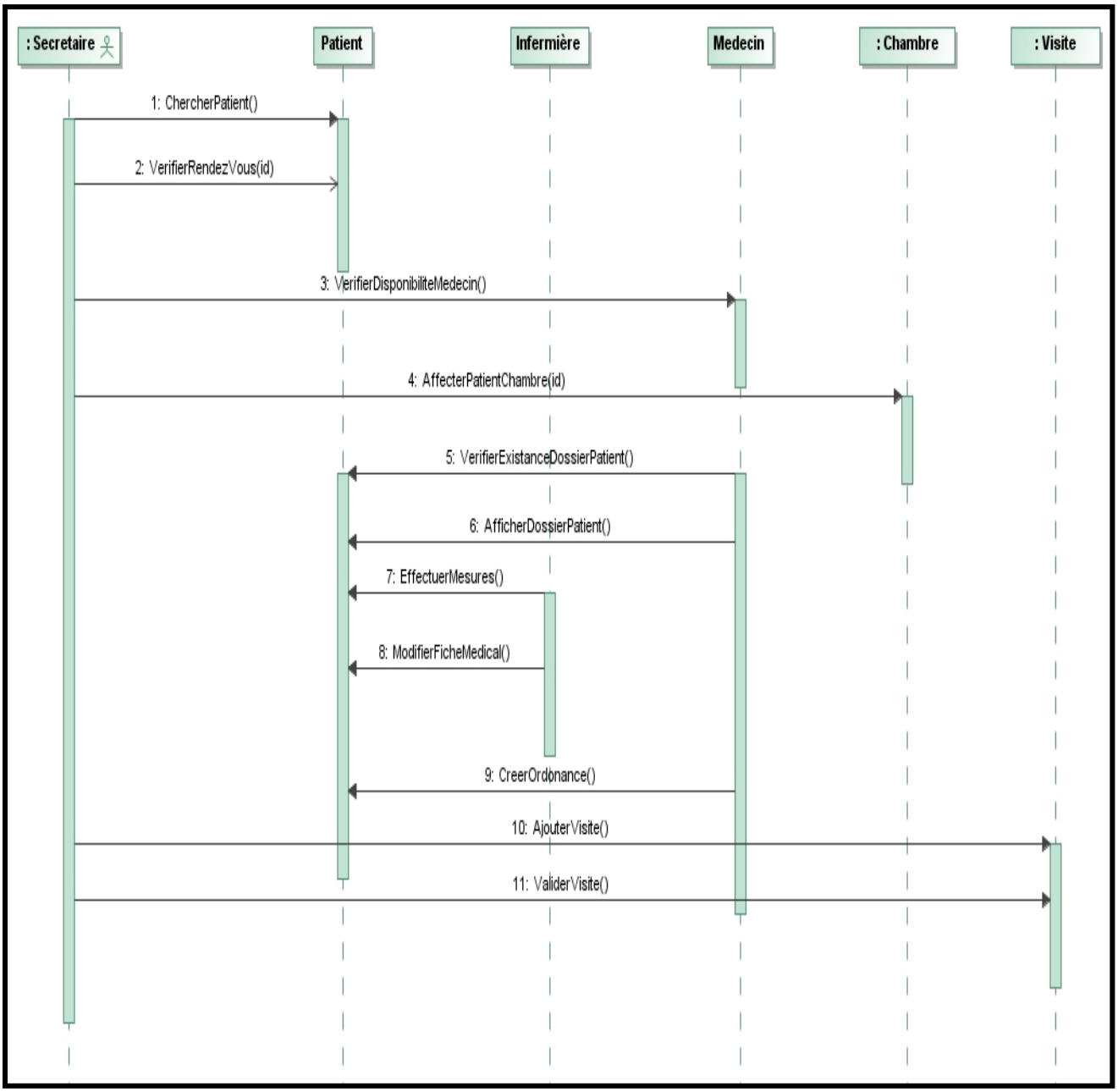


Figure 7 : Diagramme de séquence boîte blanche du cas d'utilisation "d'ajout d'une visite"

description de la figure : la **Figure 7** ci dessus décrit l'interaction de la secrétaire avec les différents objet (patient, médecin, visite, chambre) du système afin de réaliser le cas d'utilisation "d'ajout d'une visite"

a. Diagramme d'état de transition "Patient" et "Facture"

I. Définition

Les diagrammes d'états-transitions permettent de décrire les changements d'états d'un objet ou d'un composant, en

réponse aux interactions avec d'autres objets/composants ou avec des acteurs.

Un état se caractérise par sa durée et sa stabilité, il représente une conjonction instantanée des valeurs des attributs d'un objet. Une transition représente le passage instantané d'un

état vers un autre. Une transition est déclenchée par un événement. En d'autres termes : c'est

l'arrivée d'un événement qui conditionne la transition. Les transitions peuvent aussi être automatiques, lorsqu'on ne spécifie pas l'événement qui la déclenche.

En plus de spécifier un événement précis, il est aussi possible de conditionner une transition, à l'aide de "gardes" : il s'agit d'expressions booléennes, exprimées en langage naturel

(et encadrées de crochets).

II. Diagramme d'état de transition de patient

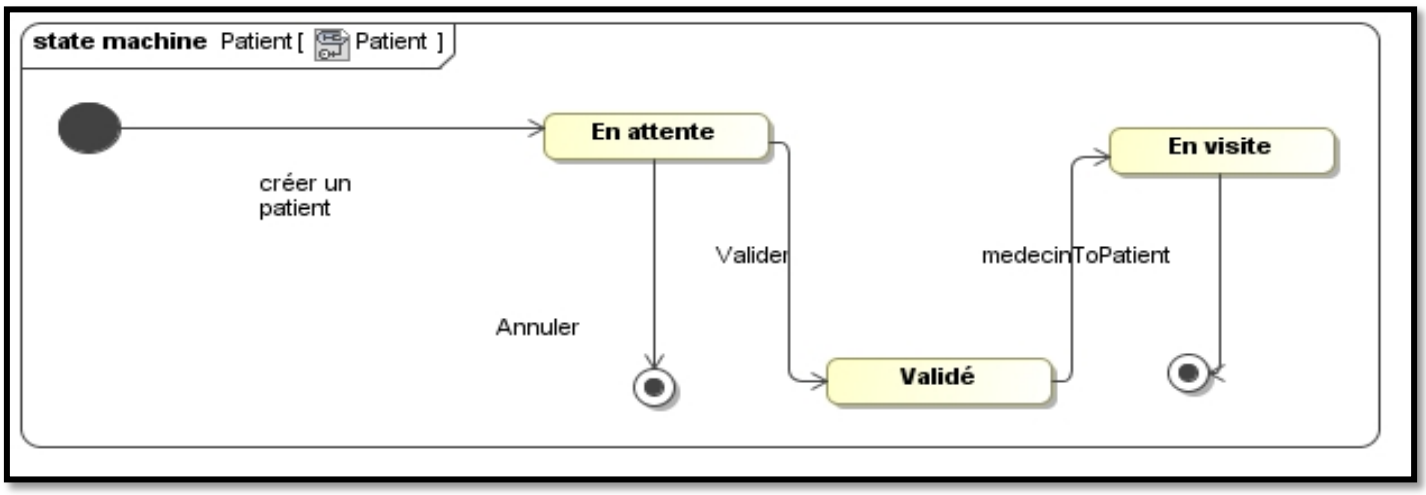


Figure 8 : Diagramme d'état de transition de l'objet patient

description de la figure : la **Figure 8** ci dessus montre les états par lesquelles passe l'objet patient

Description détaillée :	
Etat	**Action**
En attente	Ajouter un patient .
Validée	Valider une facture .
En Visite	Affecter un patient a un médecin .

III. Diagramme d'état de transition d'une facture

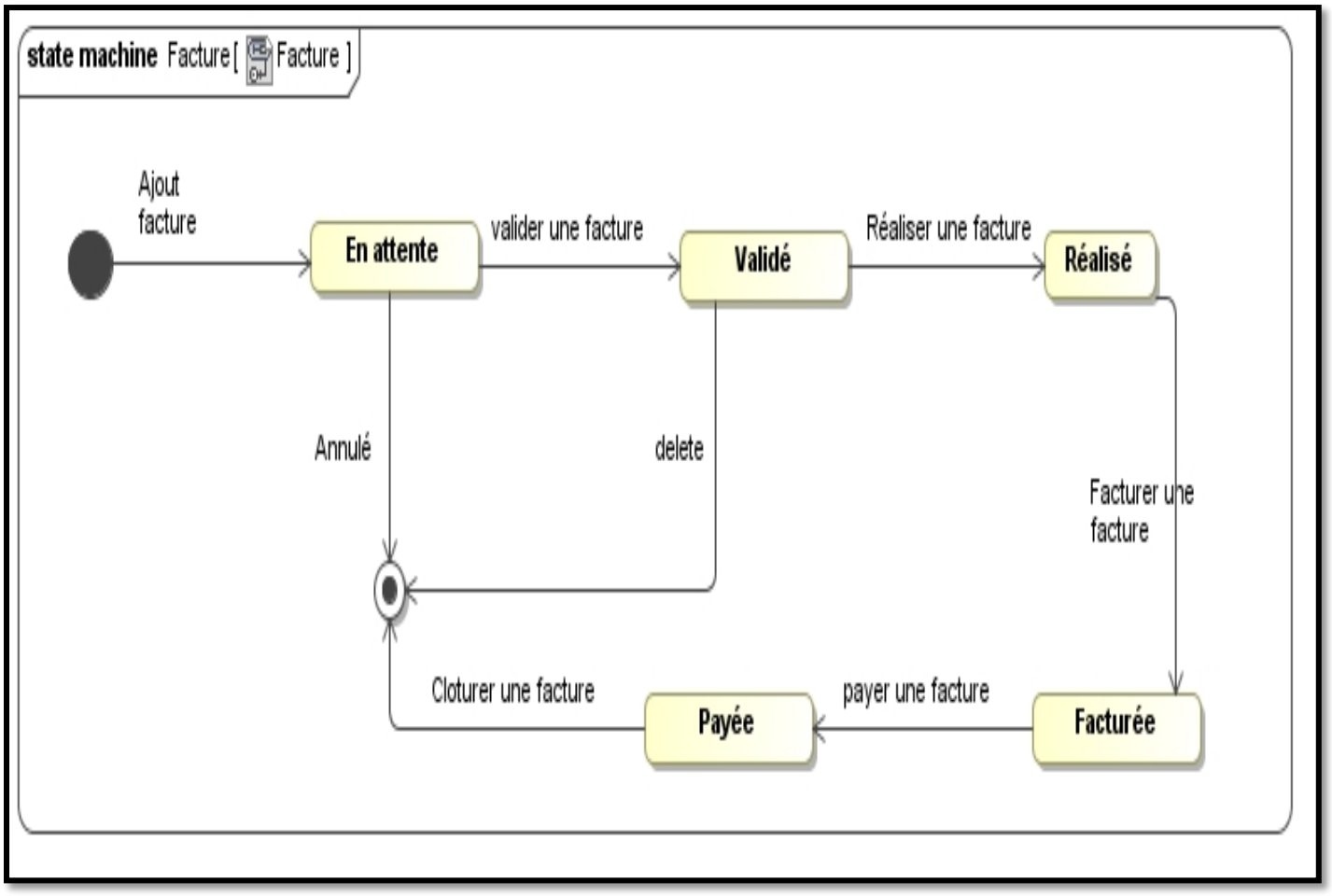


Figure 9 : Diagramme d'état de transition de l'objet facture

description de la figure : la **Figure 9** ci dessus montre les états par lesquelles passe l'objet facture

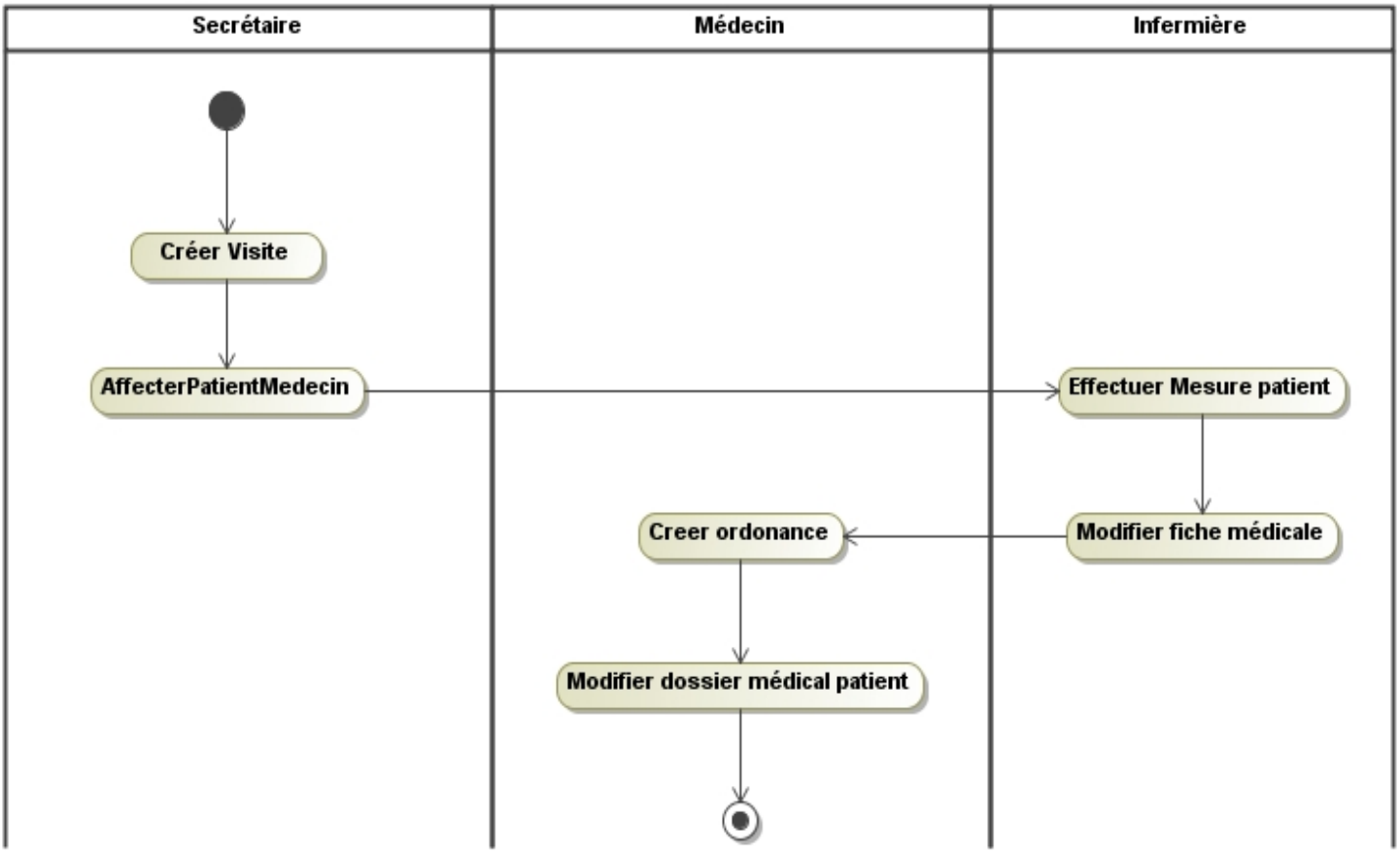
Description détaillée :	
Etat	**Action**
En attente	Ajouter une facture .
Validée	Valider une facture .
Réalisée	Réaliser une facture .
Facturée	Facturer une facture .
Payée	Payer une facture .

b. Diagramme d'Activité d'une Visite et d'une Facture:

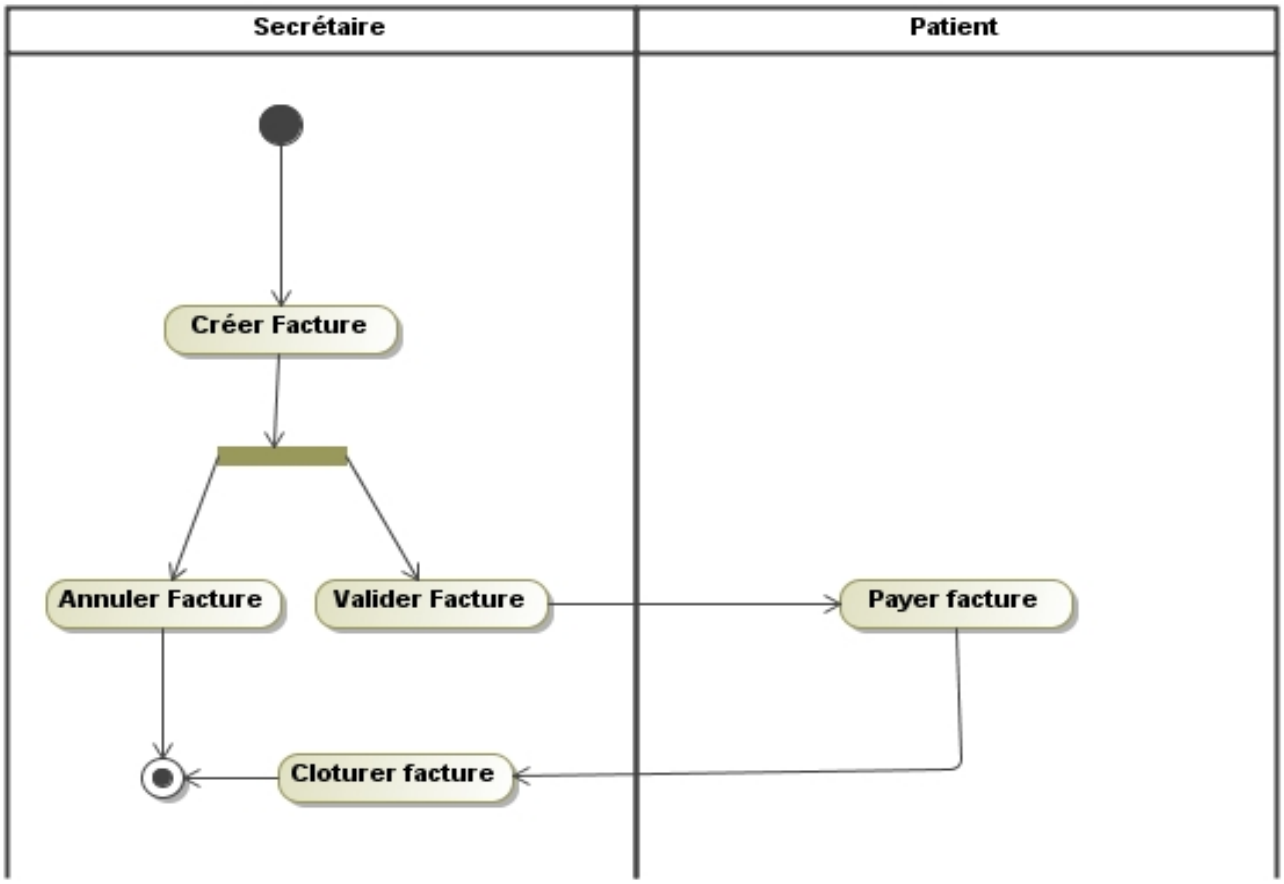
I. Définition

permet de décrire sous forme de flux ou d'enchaînement d'activités le comportement du système ou de ses composants.

II. Diagramme d'activité Visite



III. Diagramme d'activité Facture



Conclusion

Dans ce chapitre, nous avons présenté les différents diagrammes élaborés qui nous ont aidés à cerner les différentes fonctionnalités du futur système.
Dans le chapitre suivant, nous aborderons l'architecture du système et présenterons les différents outils utilisés.