REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud MAMMERI de Tizi-Ouzou

Faculté de Génie Electrique et d'informatique

Département d'informatique

Mémoire de Fin d'étude

En vue de l'obtention du diplôme de Master en informatique spécialité ingénierie des systèmes d'information

Conception et réalisation d'une application REST pour la gestion de rendez-vous d'un cabinet médical sous Android

Proposé et dirigé par :

M[r] : S. SADOU

Réalisé par :

M[r] : AMIAR ABDERAZAK

Promotion 2015/2016

Remerciements

Louange à Allah le tout puissant de m'avoir donné patience, courage et volonté pour réussir mon mémoire de fin d'étude.

Je souhaite exprimer ma profonde gratitude à tous ceux qui de près ou de loin ont participé à la réalisation du présent travail.

J'adresse à cet effet mes remerciements à :
Mr : S. SADOU, mon promoteur, pour m'avoir proposé ce sujet et de m'avoir dirigé tout au long de sa réalisation.

A tous les enseignants qui ont assuré ma formation durant mon parcours universitaire, pour l'ensemble des connaissances que j'ai consenti à leur égard.

Ces quelques mots ne traduisent guère tous ce que j'ai pu recevoir d'eux, mais je souhaite qu'ils y trouvent l'expression de mon infinie reconnaissance.

Dédicaces

Je dédie ce travail :

A ma chère grand-mère Fatma qu'Allah l'accueil dans son vaste paradis.
A mes chers parents, à mon cher oncle Belkacem et à ma chère sœur Tassadit pour toutes les valeurs qu'ils m'ont inculquées mais aussi pour l'affection et l'attention qu'ils m'ont prêtées
durant toute ma vie. Ainsi que leurs sacrifices et soutiens tout au long de
mes études.
A ma chère femme Tassadit.
A ma chère grand-mère Kaltouma qu'Allah la protège.
A mes chers frères Abdelmadjid, Slimane et Allaoua.
A ma chère sœur Hayet.
A mes chers petits neveux Abdelhakim et Abdelouahed
A ma chère cousine Rachida
A mes chers amis Samir, Abdelkarim, Nabil, Amine, Ahmed.
A mes camarades de la section informatique(ISI).
A toute la famille Amiar.
A toute la famille Chabi.
A toute la famille Brahami.
A toutes mes copains.
A tous mes proches et mes amis.

Sommaire

Premier chapitre : Généralités

Introduction

Aujourd'hui en Algérie, la prise de rendez-vous médical est souvent plus pénible que la maladie du patient elle-même. Un patient doit se lever avant l'aube, afin de pouvoir s'inscrire sur une liste d'attente collée généralement sur les portes des cabinets médicaux. Se lever tôt ne suffit pas dans certain cas pour consulter un médecin, car la liste d'attente est généralement modifiée ou déchirée et voir même encombrée avec d'autre rendez-vous différés. L'ennui de la file d'attente est répandu dans la société ces dernières années, dû à la croissance de la population et au manque des services et infrastructures sanitaires.
A ce jour, le patient est cerné entre les gears de la mauvaise gestion des rendez-vous et l'incapacité des cabinets médicaux à la prise en charge des besoins des patients. Résoudre un tel problème requiert de poser une panoplie de questions qui doivent traiter les mesures que doit-on prendre, les outils et les moyens nécessaires, le rôle que jouent les nouvelles technologies dans la résolution de ce problème.

Parmi les mesures incontournables à prendre en charge afin de répondre à la bonne gestion de rendez-vous médical, on citera l'inclusion des Technologies de l'Informations et de la Communications.

L'intégration des TIC dans la gestion des cabinets médicaux réduira sans doute les ennuis d'attente des patients et ceux des médecins à se préoccuper de la gestion des listes des inscrits, la gestion de la file d'attente, dans certain cas, assurée par un agent qui sera une charge financière de plus pour le médecin.

Ce travail qu'on est en train de réaliser, est justement l'une des solutions qui inclut des réponses aux questions posées précédemment en optant à l'usage et l'intégration des TIC dans la gestion des cabinets médicaux.

En effet, le domaine des TIC est si vaste qu'on devra préciser dans ce qui suit les outils et les moyens nécessaire pour la réalisation de ce type de projet.

Choisir une approche, un outil, un moyen et une technologie adéquate à la résolution d'un problème doit être précédé d'une analyse et une étude profonde, ce qui permettra d'aboutir à une solution fiable et optimale.

Avant c'était la révolution de l'ordinateur bureautique qui a changé le monde en quelques décennies, ensuite l'invention des ordinateur personnel portable, qui sont devenus le modèle idéal pour les professionnels en tout ce qui offre d'avantage par rapport à l'ordinateur bureautique (portabilité, légèreté, haute performance…).

De nos jours, le Smartphone à fait encore une révolution plus large que les inventions antérieures.
Autrefois, le portable nous était utile pour passer un coup de fil, envoyer des sms et se distraire avec des simples jeux installés sur l'appareil.

Maintenant, téléphoner ou écrire un sms est une fonctionnalité basic sur les Smartphones… La concurrence entre les différentes compagnies téléphoniques a fait que l'évolution du téléphone portable se fasse très rapidement. Le divertissement a pris plus d'importance que les fonctionnalités de base.

En un espace d'une vingtaine d'année, le téléphone portable nous a fait changer notre comportement. Dans une réunion, pendant le sommeil, à l'école ou, encore, dans un enterrement nos mobiles restent allumés ou sous vibreur, pour ne pas manquer un appel, important ou pas, nous devons être joignables.

Le fait d'être joignable favorise la dépendance. De plus, certaines entreprises obligent ses employés à posséder un téléphone portable. Exemple, l'Hôpital Universitaire de Genève fourni à ses employés un téléphone pour pouvoir les contacter.

Nous favorisons de plus en plus le contact téléphonique que le contact physique. Nous préférons utiliser le moyen de communication le plus répandu dans le monde, le téléphone ou Smartphone. Nous sommes presque tous dépendants du portable.

Tous ces arguments nous conduisent du premier coup d'œil à choisir cet appareil(Smartphone) comme un moyen nécessaire pour manipuler et transmettre l'information, en effet c'est l'outil de communication le plus proche à l'être humain.

A présent, l'offre technologique nous permet non seulement de choisir le type d'appareil de communication mais aussi le choix du system d'exploitation.
Actuellement, au niveau mondial, en 2016, le marché des Smartphones ne devrait croître que de 3,2% à 1,48 milliard de terminaux livrés. Les terminaux Android devraient tirer leur épingle du jeu avec 6% de croissance et 84% de part de marché **[1]**.

Ces statistiques nous montrent, sans doute, la dominance d'Android sur le marché, évidement c'est un argument suffisant pour choisir cette plate-forme afin de solutionner un problème social tel que la **Gestion des rendez-vous dans les cabinets médicaux**.

Comme nous l'avons mentionné précédemment, le choix d'outils, d'approches et des moyens pour la résolution d'un problème quelconque nécessite une analyse et une étude profonde pour pouvoir cerné ce problème dans un contexte bien défini. Ce raisonnement que nous avons suivi, nous a conduit à choisir la plate-forme d'Android pour réaliser notre projet qui s'intitule : **Conception et réalisation d'une application REST pour la gestion de rendez-vous d'un cabinet médical.**

Le titre de ce projet nous reflète deux sous-titres : **Conception** et **Réalisation**, qui seront deux grands chapitres a détaillé.
Le chapitre conception va nous éclaircir le plan architectural de l'application ainsi que la méthodologie et l'approche suivi pour concevoir cette architecture.
Le chapitre réalisation va nous montrer comment et avec quels moyens et outils avons-nous développé cette application.
Mais avant de commencer a détaillé ces deux grands chapitres, nous devons débuter ce projet par quelques généralités sur la plate-forme Android et les applications mobiles, cela sera utile pour pouvoir se situer dans les chapitres qui vont suivre.

1. Généralités sur Android et applications mobiles

1.1. Introduction

Afin de pouvoir se situer dans les prochains chapitres
nous allons voir dans cette partie quelques généralités à propos d'Android et les applications mobiles, cela nous permet d'avoir un aperçu sur : l'historique, la chronologie des versions et l'architecture d'Android; l'environnement et les prérequis de développement pour Android; l'historique, le développement, l'objectif, les avantages et les inconvénients des applications mobiles et quelques statistiques concernant le nombre de téléchargement d'applications mobiles; les web service et l'architecture REST .

1.2. Présentation d'Android

1.2.1. Open Handset Alliance [2]

Rachetée par Google en 2005, Android était initialement une startup qui développa un système d'exploitation pour appareil mobile.

Le 5 novembre 2007 fut annoncée la création de l'OHA (Open Handset Alliance) : un consortium crée à l'initiative de Google réunissant une trentaine d'entreprises à ses débuts.

La plus grande partie de ces entreprises étaient des opérateurs mobiles, des constructeurs, des industriels et des éditeurs logiciels.

Le rôle de l'OHA est de favoriser l'innovation sur les appareils mobiles en fournissant une plate-forme véritablement ouverte et complète incluant un système d'exploitation, le middleware (les logiciels intermédiaires), une interface utilisateur et des applications phares.

Le 12 novembre 2007 l'OHA annonça la sortie du premier SDK (Software Development Kit) ou kit de développement logiciel Android permettant aux développeurs de pouvoir créer des applications pour la plate-forme Android.

En septembre 2008 sortie la première version de la plate-forme Android notée 1.0.

1.2.2. Chronologie des versions d'Android [3]

Développer avec Android nécessite de savoir la chronologie des versions de cette plate-forme afin de pouvoir choisir la version adéquate et les outils nécessaires.
Android **1.0** est la première version commerciale du système parue en septembre 2008 sur le Smartphone HTC-Dream avec les caractéristiques suivantes :

- ✓ Téléchargement et mises à jour des applications via Android Market
- ✓ Navigateur supportant les sites web en HTML et XHTML
- ✓ Support de l'appareil photo
- ✓ Support des dossiers d'applications
- ✓ Accès aux serveurs e-mail POP3, IMAP4 et SMTP
- ✓ Synchronisation de Gmail, Contacts et Google Agenda avec leurs applications dédiées
- ✓ Google Maps avec Latitude et Street View pour utiliser le service de cartographie de Google, de recherche d'adresses et pour utiliser son téléphone comme un GPS

- ✓ Synchronisation des contacts, mails et agenda
- ✓ Recherche sur internet avec le moteur de recherche Google
- ✓ Service de messagerie instantanée avec Google Talk
- ✓ Envoi de SMS et de MMS
- ✓ Lecteur multimédia pour lire ses fichiers audio et vidéo
- ✓ Notifications dans la barre de statut, possibilité de personnaliser les sonneries, le vibreur et la LED
- ✓ Synthèse vocale pour chercher un numéro
- ✓ Possibilité de personnaliser le fond d'écran
- ✓ Application YouTube
- ✓ Autres applications incluses: alarme, calculatrice, menu d'appel, écran d'accueil, galerie photo et menu paramètres
- ✓ Support du Wi-Fi et du Bluetooth

La version **1.1** mise en ligne en février 2009 pour le HTC Dream, corrige quelques bugs, et apporte de légers changements sur la version 1.0 pour passer à la version 1.1 qui se caractérise par :

- ✓ Résolution de plusieurs problèmes
- ✓ Changements des API
- ✓ Avis et détails ajoutés à Maps
- ✓ Délai d'arrêt de l'écran plus long lors de l'utilisation du haut-parleur
- ✓ "Afficher" & "Cacher" le pavé numérique, inclus dans le menu d'appel
- ✓ Support pour sauvegarder les fichiers attachés aux MMS
- ✓ Support des marquées dans les layouts

Le 30 avril 2009, la mise à jour **Android 1.5 (Cupcake)**, basée sur le noyau Linux 2.6.27, est publiée. Il y a eu plusieurs nouvelles fonctionnalités et mises à jour de l'interface graphique, la version 1.5 se caractérise :

- ✓ Support pour les claviers virtuels en trois parties avec prédiction des mots et dictionnaire personnalisé.
- ✓ Support pour les Widgets, qui permettent d'accéder rapidement à certaines informations de l'application à laquelle ils sont rattachés.
- ✓ Enregistrement vidéo dans les formats MPEG-4 et 3GP.
- ✓ Support du Bluetooth A2DP et AVRCP.
- ✓ Ajout de la fonction copier/coller dans le navigateur.
- ✓ Ajout de photos pour les contacts enregistrés comme favoris.
- ✓ Ajout de la date et de l'heure dans le menu d'appel, et ajout d'un accès rapide aux contacts depuis ce même menu.
- ✓ Animations lors d'un changement d'écran.
- ✓ Ajout d'une option de rotation automatique.
- ✓ Ajout de l'actuelle animation de démarrage.
- ✓ Envoi de vidéos vers YouTube et Picasa.

Le 15 septembre 2009, c'est la sortie du SDK pour **Android 1.6 (Donut)**, basé sur le noyau Linux 2.6.29 est publié. La mise à jour comprend:

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.
- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.
- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA
- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit)

Le 26 octobre 2009, le SDK pour **Android 2.0 (Eclair)**, basé sur même noyau que Donut, est publié. La mise à jour comprend:

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.
- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.
- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA.

- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit).

Le 20 mai 2010, le SDK pour **Android 2.2 (Froyo)**, basé sur le noyau 2.6.32, est publié. La mise à jour comprend:

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.
- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.
- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA.
- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit).

Le 6 décembre 2010, le SDK pour **Android 2.3 (Gingerbread)**, basé sur le noyau 2.6.35, est publié. La mise à jour comprend:

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.
- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.

- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA.
- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit).

Le 22 février 2011, le SDK pour **Android 3.0 (Honeycomb)**, basé sur le noyau 2.6.36, est publié. Réservé aux tablettes tactiles, cette mise à jour comprend de nombreux changements dans l'interface :

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.
- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.
- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA.
- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit).

Le 19 octobre 2011, le SDK pour **Android 4.0 (Ice Cream Sandwich)**, basé sur le noyau 3.0.1, est publié. Cette nouvelle version unifiée pour Smartphones et tablettes tactiles apporte de nombreux changements :

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.

- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.
- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA.
- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit).

Le 16 décembre 2011 c'est la sortie de la **version 4.0.3(Ice Cream Sandwich)**qui se caractérise de :

- ✓ Nombreuses corrections de bugs et d'optimisations.
- ✓ Améliorations de certaines fonctionnalités dont le Bluetooth.
- ✓ Nouvelles APIs pour les développeurs.
- ✓ Amélioration du calendrier.
- ✓ Amélioration de l'accessibilité pour les lecteurs d'écrans.
- ✓ Diverses modifications de l'interface de l'appareil photo.

Le 27 juin 2012 lors du Google I/O 2012, Google annonce la **version 4.1 dénommée Jelly Bean** basée sur le noyau Linux 3.0.31 et dont la principale nouveauté est une amélioration des fonctionnalités et des performances de l'interface utilisateur. Les améliorations des performances font partie du Project Butter (Projet Beurre en français) qui utilise le triple buffering, une amélioration de Vsync (en) et une fréquence de rafraîchissement de l'écran porté à 60 FPS afin de créer l'impression que l'interface est plus fluide que sous Ice Cream Sandwich. Jelly Bean a été publié à l'Open Handset Alliance le 9 juillet 2012 et le premier appareil à être équipé de Jelly Bean, la tablette Nexus 7 a été commercialisé à partir du 13 juillet 2012 aux États-Unis, cette version se caractérise :

- ✓ Mise à jour de la recherche, autorisant la recherche dans les favoris, l'historique, les contacts, et Internet depuis l'écran d'accueil.

- ✓ Mise à jour de la recherche vocale, plus rapide et une meilleure intégration avec les applications natives, incluant la possibilité d'appeler ses contacts et support de plusieurs langues supplémentaires.
- ✓ Possibilité pour les développeurs d'intégrer leurs contenus dans les résultats de recherche.
- ✓ Interface de l'Android Market améliorée.
- ✓ Interface native pour l'appareil photo, la caméra et la galerie.
- ✓ Galerie : autorise les utilisateurs à sélectionner plusieurs photos pour suppression.
- ✓ Mise à jour du support pour CDMA/EVDO, 802.1x, VPNs, et une synthèse vocale.
- ✓ Support des écrans avec une résolution WVGA.
- ✓ Amélioration de la rapidité dans la recherche et les applications utilisant la caméra.
- ✓ Framework de reconnaissance de Gestes et outil de développement GestureBuilder.
- ✓ Google Navigation (GPS Gratuit).

Le 3 septembre 2013, Google annonce que la **version 4.4**, portera le nom de **KitKat**, alors que plusieurs rumeurs annonçaient le nom de Key Lime Pie (tarte au citron vert). Cette version est sortie le 4 novembre 2013, en même temps que le Nexus 5. La version 4.4.2 est sortie le 9 décembre 2013, la version 4.4.3 le 2 juin 2014, et la version 4.4.4 le 20 juin 2014.

Voici la liste des nouveautés :

- ✓ apparition d'un mode immersif, qui cache la barre de notifications pendant les jeux ou le visionnage d'un film, et qui permet, à la différence des autres versions, de la faire réapparaître en glissant le doigt à partir du bord de l'écran.
- ✓ mise à jour de l'application Téléphone, qui met en place un nouveau design, et qui cherche la personne qui appelle sur Google+ si elle n'est pas dans les contacts.
- ✓ Hangouts remplace l'application Messages et centralise les SMS, MMS et appels vidéo.
- ✓ mise à jour du clavier Google, qui ajoute des emojis, en plus d'un changement de design.
- ✓ apparition de Google Cloud Print, qui permet d'imprimer des photos, des documents ou des pages web à distance à partir d'un smartphone ou d'une tablette via une imprimante connectée.
- ✓ compatibilité avec le Bluetooth MAP et avec Chromecast.

- ✓ mise à jour de l'application Téléchargements permettant une meilleure gestion avec de nouvelles options de triage et une nouvelle interface.
- ✓ mise à jour de l'application E-mail avec des dossiers imbriqués, des photos des contacts et une navigation améliorée.
- ✓ Nouvelle politique de gestion des cartes mémoires plus restrictive.

La version 5.0 Annoncée le 15 octobre 2014 et sortie le 3 novembre 2014, **Android 5.0 Lollipop** est une évolution majeure d'Android qui propose de nombreuses modifications et nouveautés, et qui étend sa disponibilité sur de nouveaux supports tels que la télévision, la voiture ou les montres connectées.

Android Lollipop est proposée dès sa sortie sur les appareils Nexus, notamment les Nexus 6, 9 et Player qui sont disponibles dès le 3 novembre 2014, et ceux qui sont certifiés Google Experience Edition.

Voici la liste des principales nouveautés :

- ✓ Refonte totale de l'interface graphique avec un nouveau design nommé Material Design.
- ✓ Nouveau moteur d'exécution ART (en) qui compile les applications dès leur installation plutôt que la compilation JIT de Dalvik.
- ✓ Projet Volta qui permet d'optimiser la consommation d'énergie et de gagner en autonomie.
- ✓ Amélioration du système de notifications.
- ✓ Activation par défaut du chiffrement des données utilisateur.
- ✓ Disponibilité d'Android TV et Android Auto.

C'est fin mai 2015, lors de la Google I/O, que la firme de Mountain View a dévoilé les détails sur **Android 6.0 (Marshmallow**). La version grand public a été quant à elle déployée au courant du troisième trimestre 2015, en même temps que la sortie du nexus 5x et 6p, cette version se caractérise :

- ✓ Intégration finale du mode Multi-fenêtre.
- ✓ Réorganisation du gestionnaire des permissions.
- ✓ Barre d'action rapide pour tablette.
- ✓ Intégration Option informations.
- ✓ Lancement Google Now en débloquant l'écran d'accueil Option dédié Google Now on Tap.
- ✓ Possibilité désinstallation application Native.
- ✓ Information application barre d'état.
- ✓ Intégration de "Adoptable Storage" qui permet d'utiliser une carte microSD comme stockage interne.

1.2.3. Architecture d'Android [2]

Le système Android est basé sur le noyau Linux 2.6. Ce noyau prend en charge la gestion des couches basses, telles que les processus, la gestion de la mémoire, les droits utilisateurs et la couche matérielle à l'aide des drivers.
Au-dessus de ce noyau, figure la couche des bibliothèques principales du système fournies par des tiers. Celles-ci, de bas niveaux, sont écrites en C et/ou C++. Elles fournissent des services essentiels tels que la gestion de l'affichage 2D et 3D, un moteur de base de donnée SQLite, la lecture et l'enregistrement audio et vidéo, un moteur de navigateur web…
Les fonctionnalités offertes par ces bibliothèques sont ensuite reprises et utilisées par la couche supérieure sous forme de bibliothèque Java. Celles-ci fournissent des bibliothèque et composants réutilisables spécifiques à des domaines particuliers. On y retrouve par exemple les bibliothèques de base de données, de téléphonie, de localisation géographique, de communication en champ proche…
Enfin, la couche de plus haut niveau est celle des applications. Ces applications sont celles fournies par défaut, comme l'application d'accueil ou bureau, l'application de lancement d'applications, le navigateur web, l'application de téléphone.
Bien qu'il soit possible de développer des applications comportant du code C et/ou C++ via le NDK(Native Development Kit) afin notamment d'améliorer les performances.
Par défaut, chaque application s'exécute dans une machine virtuelle Java elle-même confinée dans un processus Linux dédié. Cette machine virtuelle est spécifique à la plate-forme Android et spécialisée pour les environnements embarqués. Elle est nommée machine virtuelle Dalvik.

1.2.4. Environnement de développement [2]

L'environnement de développement peut se restreindre à la simple ligne de commande et de l'éditeur de texte léger, mais puissant, l'accompagnant jusqu'à aller à l'environnement de développement intégré (Eclipse, NetBeans, AndroidStudio… etc) , gourmand en ressources mais fonctionnel.

1.2.5. Prérequis [2]

Pour pouvoir développer des applications sous Android, il faut tout d'abord s'assurer que le poste de développement est compatible avec les critères requis.
Systèmes d'exploitation supportés :

- ✓ Windows XP (32 bits) et les versions de Windows supérieure, Vista, Windows 7, Windows 8 (32 ou 64 bits)
- ✓ Mac OS X 10.5.8 ou supérieur (x86 seulement)
- ✓ Linux (Bibliothèque GNU C (glib) 2.11 ou supérieure ; les distributions 64 bits doivent être capable d'exécuter des application 32 bits).
- ✓ JDK (Java Development Kit) de Java 5 ou supérieur est requis.
- ✓ SDK(Software Development Kit) : Le kit de développement d'Android est un ensemble complet d'outils de développement. Il inclut un débogueur, des bibliothèques logicielles, un émulateur basé sur QEMU (un logiciel libre de machine virtuelle, pouvant émuler un processeur et, plus généralement, une architecture différente si besoin.), de la documentation, des exemples de code et des tutoriaux. Les plateformes de développement prises en charge par ce kit sont les distributions sous Noyau Linux, Mac OS X 10.5.8 ou plus, Windows XP ou version ultérieure. L'IDE officiellement supporté était Eclipse combiné au plugin d'outils de développement d'Android (ADT), mais depuis 2015, Google officialise Android Studio qui devient alors l'IDE officiel pour le SDK Android.

1.3. Applications mobiles

1.3.1. Présentation[9]

Une application mobile est un logiciel applicatif développé pour un appareil électronique mobile, tel qu'un assistant personnel, un téléphone portable, un « Smartphone », un baladeur numérique, une tablette tactile, ou encore certains ordinateurs fonctionnant avec le système d'exploitation Windows Phone.

Elles sont pour la plupart distribuées depuis des plateformes de téléchargement (parfois elles-mêmes contrôlées par les fabricants de Smartphones) telles que l'App Store (plateforme d'Apple), le Google Play (plateforme de Google / Android), ou encore le Windows Phone Store (plateforme de Microsoft). Mais des applications peuvent aussi être installées sur un ordinateur, grâce par exemple au logiciel ITunes distribué par Apple pour ses appareils. Les applications distribuées à partir des magasins d'applications sont soit payantes, soit gratuites, mais généralement avec des publicités.

Sur certaines plateformes, les applications peuvent aussi être installées à partir de sources tierces, via un site non affilié au distributeur d'origine. Sur Android, cela est possible en activant le mode développeur. Sur iOS, cette manipulation est possible soit en étant développeur Apple, soit en possédant un appareil Jailbreaké.

1.3.2. Historique sur les applications mobiles [9]

Les applications mobiles sont apparues dans les années 1990, elles sont liées aux développements d'Internet et des télécommunications, des réseaux sans fils et des technologies agents, et à l'apparition et la démocratisation des terminaux mobiles : Smartphones, tablettes tactiles...

1.3.3. Développement des applications mobiles [9]

Les applications mobiles sont développées sur des ordinateurs ; le langage utilisé dépend du système sous lequel l'application sera exécutée. Les applications pour les terminaux Apple sont développées dans un langage principalement dédié à ces applications mobiles, l'Objective C. Celles pour Windows Mobile, sont développées en C#, langage aussi utilisé pour les programmes exécutables .exe. Le système Android utilise, quant à lui, un langage universel, le Java, langage pouvant être utilisé pour les ordinateurs, le développement Web (JEE).

Pour publier votre application mobile sur les principaux App Store, le logiciel doit remplir plusieurs conditions. La combinaison de techniques qui offrent une visibilité aux applications dans les magasins est appelé App Store Optimization (ASO).

1.3.4. Objectif des applications mobiles [9]

Elles visaient d'abord à améliorer la productivité et à faciliter la récupération d'informations telles que courrier électronique, calendrier électronique, contacts, marché boursier et informations météorologiques.

Vers 2005, elles gagnent les entreprises puis, la demande du public et la disponibilité d'outils de développement ont conduit à une expansion rapide dans d'autres domaines, comme :

- ✓ les jeux mobiles ;
- ✓ les automatismes industriels ;
- ✓ le GPS et les services permettant la localisation ;
- ✓ les opérations bancaires ;
- ✓ les suivis des commandes, l'achat de billets ;
- ✓ des applications médicales mobiles ;
- ✓ la réalité virtuelle ;
- ✓ l'écoute de musiques ou de radios ;
- ✓ la visualisation de vidéos ou de chaines de télévision ;
- ✓ la consultation d'Internet ;
- ✓ les réseaux sociaux généraux (type Facebook) ;
- ✓ les réseaux sociaux spécialisés.

1.3.5. Statistique sur les Apps Mobiles [9]

Statistiquement, environ 200 milliards d'applications mobiles ont été téléchargés jusqu'en 2015, alors qu'en 2009, seul 2 milliards l'ont été. Selon les prévisions, le compteur devrait atteindre 200 milliards de téléchargement en 2017.

De 2009 à 2015, le nombre de téléchargement d'applications mobiles gratuites pourrait atteindre 167 milliards. En 2017, ce chiffre pourra atteindre 253 milliards.

De 2011 à 2015, les applications mobiles ont générés un revenu de 45.37 milliards de dollars. En 2017, les revenus devraient atteindre les 76.5 milliards de dollars.

1.3.6. Avantages des applications mobiles [10]

- ✓ Une application pour Smartphone est bien souvent plus ergonomique qu'un site internet mobile. Elle rend l'expérience utilisateur bien meilleure grâce à une navigation et des contenus spécialement adaptés aux usages mobiles.

- ✓ Une application mobile peut aussi tirer parti des fonctions spécifiques aux Smartphones telles que le GPS ou l'appareil photo. Grâce à ces applications, il est même possible de recevoir des notifications en push, chose impossible avec un site mobile.

- ✓ Autre atout non négligeable : une application ne nécessite pas forcément de connexion à internet. Ainsi, plus besoin d'attendre l'ouverture de la page, plus de déconnexion intempestive et plus de raisons de s'énerver…

- ✓ Très faciles à trouver, les applications sont directement disponibles sur l'Apple store avec l'iPhone ou sur le Play store avec un Smartphone Android.

1.3.7. Inconvénients des applications mobiles [10]

- ✓ Le respect des règles des plateformes mobiles
  L'Apple store, le Play store et le Windows store imposent un certain nombre de règles pour les développeurs. Parfois contraignantes, elles sont un passage obligé si l'on souhaite pouvoir distribuer une application Smartphone de manière optimale.
- ✓ Un cout de développement élevé
  Le coût de création d'une application mobile est relativement élevé et peut décupler vos dépenses e-marketing, surtout si vous souhaitez que votre application soit disponible sur tous les systèmes d'exploitation pour mobiles…
- ✓ La mise à jour est contraignante pour l'utilisateur
  En cas de mise à jour de l'application, le mobinaute doit se rendre une nouvelle fois sur les plateformes de téléchargement. Le site mobile peut quant à lui évoluer sans contraintes pour le visiteur.
- ✓ Controverse entre applications web et applications mobiles

En 2012, Tim Berners-Lee (créateur du web) critique les applications mobiles fermées faisant usage du Web. Le Web symbolise un monde ouvert, tandis que le monde des applis mobiles est cloisonné. Il propose aux développeurs de logiciels une solution (open web Apps) utilisant le standard HTML5**[9]**

- ✓ Craintes en termes de non-protection des données personnelles, Les obligations légales (information de l'utilisateur via les mentions légales, et respect de la loi pour la confiance dans l'économie numérique notamment) des applications sont les mêmes que celles des sites web. En 2014, selon le parlement et la commission européenne « Neuf Européens sur dix (92 %) disent qu'ils sont préoccupés par les applications mobiles (App) susceptibles de collecter leurs données personnelles sans leur consentement » et « Sept Européens sur dix sont préoccupés par l'utilisation potentielle que les entreprises peuvent faire de l'information divulguée ». Le parlement européen prépare une nouvelle stratégie et une réglementation sur la protection des données **[9]**.

1.3.8. Les services web

1.3.8.1. Introduction [6]

Dans cette partie nous allons présenter les services web qui sont une notion très importante dans la réalisation de ce type projet. Cependant, comprendre le fonctionnement des services web nécessite de donner un exemple de projet reconnu utilisant cette notion. Le World Wide Web est le grand projet mondial utilisant la technologie des services web, 3W est une application conçue selon l'architecture REST (representational state transfer). L'architecture du Web remplace donc les concepts applicatifs clients et serveurs par les concepts agents et ressources. Des agents interagissent avec des ressources pour créer, accéder, modifier ou supprimer une ressource. Jusqu'à présent, on parlait surtout de l'interaction entre agents utilisateurs, principalement les navigateurs avec les ressources.

Aujourd'hui, on parle de plus en plus de l'interaction entre agents ressources ; c'est-à-dire la relation entre les ressources : une ressource devient l'agent d'une autre ressource, mais reste elle-même une ressource accessible par d'autres agents. C'est exactement l'architecture décrite par l'exemple d'implémentation applicative des Mashups (application composite : une application qui combine du contenu ou du service provenant de plusieurs applications plus ou moins hétérogènes).

Les services web traitent donc d'agents ressources là où le mode opératoire classique du Web parle d'agents utilisateurs. Mais les deux concepts reposent sur la même architecture : REST.

Il n'y a donc pas de différence fondamentale entre l'interaction d'un navigateur avec une ressource et celle d'un Service Web avec une ressource. La principale différence se situe au niveau du format de la représentation des données : HTML pour les navigateurs ou agents utilisateurs, **XML** ou **JSON** pour les Services Web ou agents ressources...

XML [13] : Extensible Markup Language (XML, « langage de balisage extensible » en français) est un métalangage informatique de balisage générique qui dérive du SGML. Cette syntaxe est dite « extensible » car elle permet de définir différents espaces de noms, c'est-à-dire des langages avec chacun leur vocabulaire et leur grammaire, comme XHTML, XSLT, RSS, SVG… Elle est reconnaissable par son usage des chevrons (<>) encadrant les balises. L'objectif initial est de faciliter l'échange automatisé de contenus complexes (arbres, texte riche…) entre systèmes d'informations hétérogènes (interopérabilité). Avec ses outils et langages associés, une application XML respecte généralement certains principes :

- ✓ la structure d'un document XML est définie et validable par un schéma ;
- ✓ un document XML est entièrement transformable dans un autre document XML.

JSON [14]: JavaScript Object Notation, est un format de données textuelles dérivé de la notation des objets du langage JavaScript. Il permet de représenter de l'information structurée comme le permet XML par exemple. Créé par Douglas Crockford entre 2002 et 2005, il est décrit par la RFC 7159 de l'IETF.

Un document JSON a pour fonction de représenter de l'information accompagnée d'étiquettes permettant d'en interpréter les divers éléments, sans aucune restriction sur le nombre de celles-ci.

Un document JSON ne comprend que deux types d'éléments structurels :

- ✓ des ensembles de paires nom / valeur ;
- ✓ des listes ordonnées de valeurs.

Ces mêmes éléments représentent trois types de données :

- ✓ des objets ;
- ✓ des tableaux ;
- ✓ des valeurs génériques de type tableau, objet, booléen, nombre, chaîne ou null.

On peut donc définir un Service Web comme l'implémentation logicielle d'une ressource, identifiée par une URL, et accessible en utilisant les protocoles internet. Les agents s'occupent du contenu, de la représentation de leur état, pas du type de contenu. Il faut donc voir les Services Web comme le moyen de manipuler l'information, et non comme un simple fournisseur de services.

1.3.8.2. Définition [4]

Les Web Services sont des applications qui relient des programmes, des objets, des bases de données ou des processus d'affaires à l'aide de XML ou JSON et de protocoles Internet standards. Les Web Services sont des compléments aux programmes et applications existantes, développées à l'aide de langages tel que Visual Basic, C, C++, C#, Java ou autre, et servent de pont pour que ces programmes communiquent entre eux. Les Web Services définissent non seulement les données transmises entre deux applications, mais aussi comment traiter ces données et les relier à l'interne et à l'externe d'une application logicielle sous-jacente. De ce fait, les Web Services permettent aux entreprises et individus de publier des liens vers leurs applications de la même manière qu'ils publient des liens vers leurs pages Web. Conséquemment, les Web Services peuvent faire en sorte que toutes les ressources informatiques dont une entreprise a besoin soient des ressources distribuées à la grandeur de l'Internet. Contrairement à la plupart des applications de type client/serveur, les Web Services ne fournissent pas d'interface usager.

Ils sont utilisés afin d'envoyer des données destinées à être lues par des machines. Cependant, les programmeurs peuvent tout de même développer une interface graphique pour l'utilisateur, auxquels ils pourront ajouter une panoplie de Web Services afin de personnaliser une page Web ou pour offrir une fonctionnalité spécifique à des utilisateurs.

Dans sa présentation la plus générale, un service web se concrétise par un agent, réalisé selon une technologie informatique précise par un fournisseur du service. Un demandeur, à l'aide d'un agent de requête, utilise ce service. Fournisseur et demandeur partagent une même sémantique du service web, tandis qu'agent et agent de requête partagent une même description du service pour coordonner les messages qu'ils échangent **[5]**

1.3.8.3. Type de service web

- ✓ **REST (Representational state transfer) [7]**
  Les services web de type Representational state transfer (REST| Services Web RESTful| Agents Ressources | Robots Logiciels) exposent entièrement ces fonctionnalités comme un ensemble de ressources (URI) identifiables et accessibles par la syntaxe et la sémantique du protocole HTTP. Les Services Web de type REST sont donc basés sur l'architecture du web et ses standards de base : HTTP et URI.

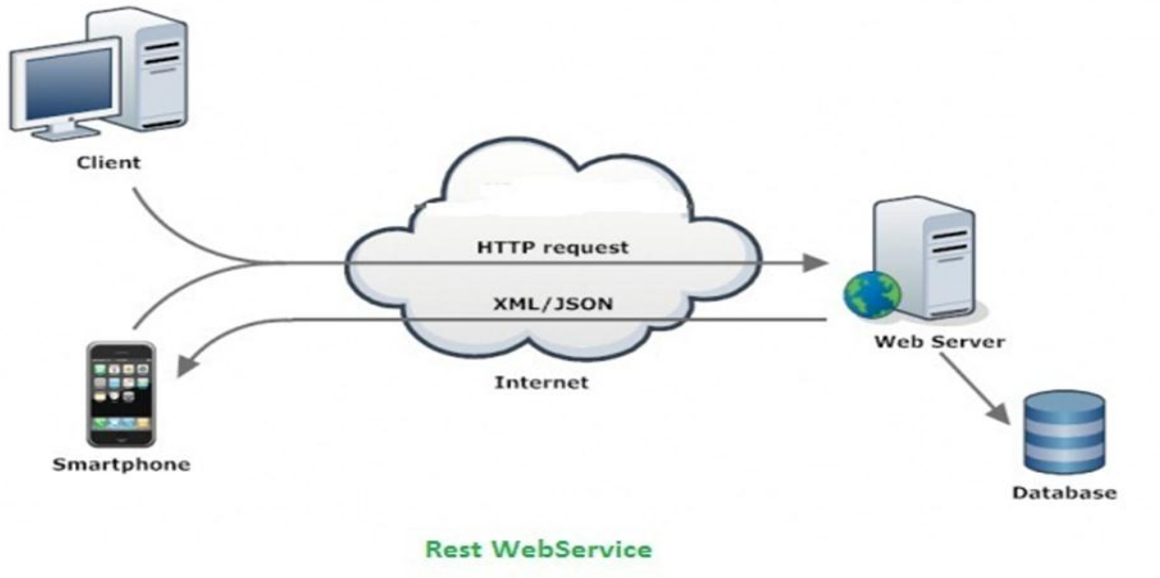


Image-Source : http://phpflow.com/php/web-service-types-soapxml-rpcrestful/

✓ **Les Services Web WS-* [8]**

Les Services Web WS-* désignent l'implémentation logicielle des spécifications WS-* et reposent tous sur un ensemble de protocoles et de standards de base utilisés pour l'échange de données entre applications dans des environnements hétérogènes :

le SOAP (Simple Object Access Protocol) pour l'échange de messages ; le WSDL (Web Service Description Language) pour la description : des services web, de leurs opérations, des messages utilisés, des types de données utilisées, des protocoles utilisés et de leur localisation au sens internet (URI / URL) ; les annuaires UDDI qui peuvent référencer des services web.
Ces Services Web WS-* sont par ailleurs définis selon le type d'architecture SOA(service oriented architecture).

Les logiciels écrits dans divers langages de programmation et sur diverses plates-formes peuvent employer des Services Web WS-* pour échanger des données à travers des réseaux informatiques comme Internet. L'OASIS et le World Wide Web Consortium (W3C) sont les comités de coordination responsables de l'architecture et de la standardisation des services Web. Pour améliorer l'interopérabilité entre les réalisations de service Web, l'organisation WS-I a développé une série de profils pour faire évoluer les futures normes impliquées.

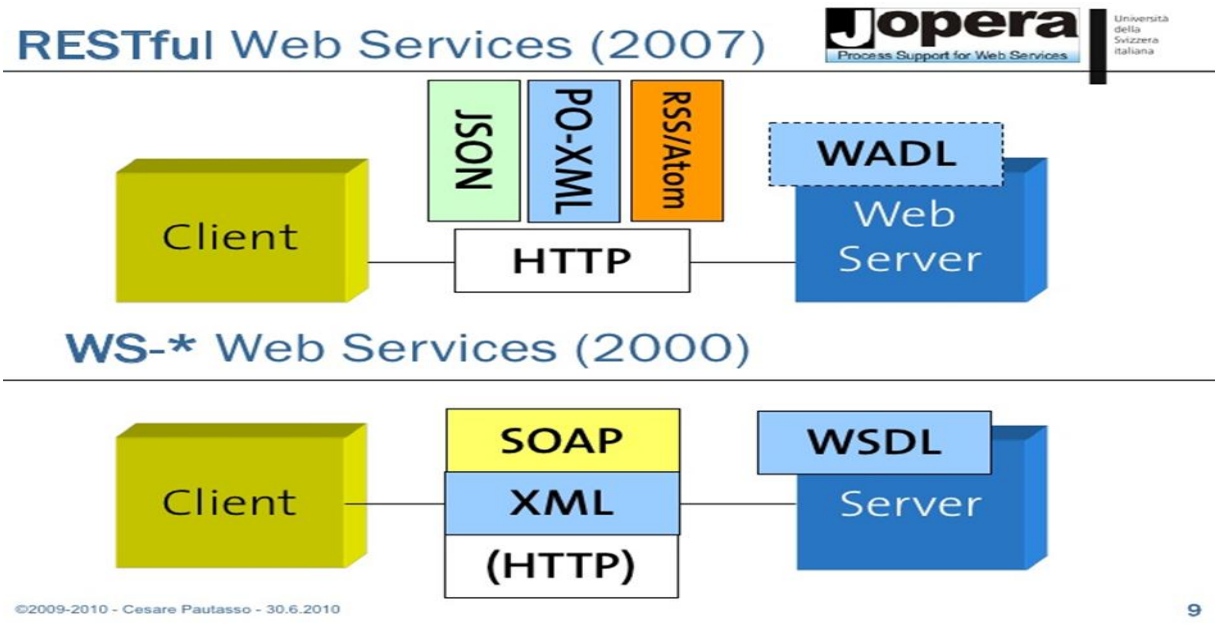


Image-Source : http://www.slideshare.net/cesare.pautasso/ws-vs-restful-services.

1.3.8.4. Avantage des services web [8]

Les services Web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
Les services Web utilisent des standards et protocoles ouverts.
Les protocoles et les formats de données sont au format texte dans la mesure du possible, facilitant ainsi la compréhension du fonctionnement global des échanges.
Les outils de développement, s'appuyant sur ces standards, permettent la création automatique de programmes utilisant les services Web existants.

1.3.8.5. Inconvénient des services web [8]

Les normes de services Web dans certains domaines sont actuellement récentes.
Les services Web souffrent de performances faibles comparées à d'autres approches de l'informatique répartie telles que le RMI, CORBA, ou DCOM.

1.3.8.6. L'Architecture REST [11]

Notre projet va se baser sur l'architecture REST (representational state transfer), qui est un style d'architecture pour les systèmes hypermédia distribués, créé par Roy Fielding en 2000 dans le chapitre 5 de sa thèse de doctorat. Il trouve notamment des applications dans le World Wide Web.

Une architecture REST doit respecter les contraintes suivantes :

- ✓ **Client-serveur** : les responsabilités sont séparées entre le client et le serveur. L'interface utilisateur est séparée de celle du stockage des données. Cela permet aux deux d'évoluer indépendamment.
- ✓ **Sans état** : chaque requête d'un client vers un serveur doit contenir toute l'information nécessaire pour permettre au serveur de comprendre la requête, sans avoir à dépendre d'un contexte conservé sur le serveur. Cela libère de nombreuses interactions entre le client et le serveur.

- ✓ **Mise en cache** : le serveur envoie une réponse qui donne l'information sur la propension de cette réponse à être mise en cache, comme la fraîcheur, sa date de création, si elle doit être conservée dans le futur. Cela permet à des serveurs mandataires de décharger les contraintes sur le serveur et aux clients de ne pas faire de requêtes inutiles. Cela permet également d'améliorer l'extensibilité des serveurs.
- ✓ **Une interface uniforme** ; cette contrainte agit selon quatre règles essentielles :
  - l'identification des ressources : chaque ressource est identifiée unitairement ;
  - la manipulation des ressources à travers des représentations : les ressources ont des représentations définies ;
  - un message auto-descriptif : les messages expliquent leur nature. Par exemple, si une représentation en HTML est codée en UTF-8, le message contient l'information nécessaire pour dire que c'est le cas ;
  - hypermédia comme moteur d'état de l'application : chaque accès aux états suivants de l'application est décrit dans le message courant.
- ✓ **Un système hiérarchisé par couche** : les états de l'application sont identifiés par des ressources individuelles. Toute l'information n'est pas envoyée dans une seule ressource unique. Les requêtes/réponses entre le client et le serveur augmentent et donc peuvent faire baisser la performance d'où l'importance de la mise en cache, etc. Le bénéfice est que cela rend beaucoup plus flexible l'évolution du système.
- ✓ **Code-on-demand (facultatif)** : la possibilité pour les clients d'exécuter des scripts obtenus depuis le serveur. Cela permet d'éviter que le traitement ne se fasse que du côté serveur et permet donc de faire évoluer les fonctionnalités du client au cours du temps. En revanche cela réduit la visibilité de l'organisation des ressources. Un état devient dépendant du client et non plus du serveur ce qui contredit la règle. Il faut donc être prudent en utilisant cette contrainte.

1.3.8.6.1. Avantage de l'architecture REST [11]

La thèse de Roy Fielding précise les avantages de ce style architectural par rapport à d'autres styles d'architectures d'applications Web. Entre autres :

- ✓ L'application est plus simple à entretenir, car elle désolidarise la partie client de la partie serveur. La nature hypermédia de l'application permet d'accéder aux états suivants de l'application par inspection de la ressource courante.
- ✓ L'absence de gestion d'état du client sur le serveur conduit à une plus grande indépendance entre le client et le serveur. Elle permet également de ne pas avoir à maintenir une connexion permanente entre le client et le serveur. Le serveur peut ainsi répondre à d'autres requêtes venant d'autres clients sans saturer l'ensemble de ses ports de communication. Cela devient essentiel dans un système distribué.
- ✓ L'absence de gestion d'état du client sur le serveur permet également une répartition des requêtes sur plusieurs serveurs : une session client n'est pas à maintenir sur un serveur en particulier (via une sticky session d'un loadbalancer), ou à propager sur tous les serveurs (avec des problématiques de fraîcheur de session). Cela permet aussi une meilleure évolutivité et tolérance aux pannes (un serveur peut être ajouté facilement pour augmenter la capacité de traitement, ou pour en remplacer un autre).
- ✓ Dans un contexte Web :
  - l'utilisation du protocole HTTP permet de tirer parti de son enveloppe et ses en-têtes ;
  - l'utilisation d'URI comme représentant d'une ressource permet d'avoir un système universel d'identification des éléments de l'application ;
  - la nature auto-descriptive du message permet la mise en place de serveurs cache : les informations nécessaires sur la fraîcheur, la péremption de la ressource sont contenues dans le message lui-même.

1.3.8.6.2. Inconvénients de l'architecture REST [11]

Le principal inconvénient de REST est la nécessité pour le client de conserver localement toutes les données nécessaires au bon déroulement d'une requête, ce qui induit une consommation en bande passante réseau plus grande [évasif]. Notamment dans l'univers des appareils mobiles, chaque aller-retour de connexion est consommateur de bande passante. La latence de la connexion rend également l'interaction moins fluide.

1.3.8.6.3. Présentation de l'architecture client- serveur [12]

1.3.8.6.3.1. Définition

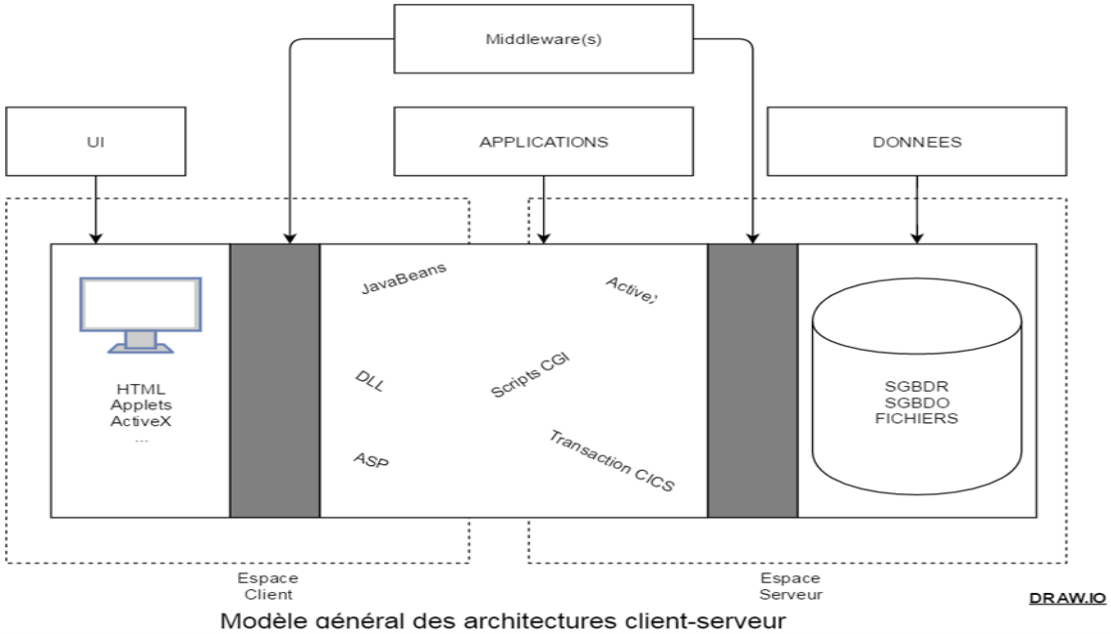
L'architecture client-serveur désigne un mode de communication à travers un réseau entre plusieurs programmes ou logiciels : l'un, qualifié de client, envoie des requêtes ; l'autre ou les autres, qualifiés de serveurs, attendent les requêtes des clients et y répondent. Par extension, le client désigne également l'ordinateur sur lequel est exécuté le logiciel client, et le serveur, l'ordinateur sur lequel est exécuté le logiciel serveur.

1.3.8.6.3.2. Modèle général d'architecture

Toutes les architectures informatiques client-serveur présentent des caractéristiques communes :

- ✓ Elle intègre une interface utilisateur (UI), souvent graphique (GUI)
- ✓ Elles fonctionnent, bien sûr, grâce à des applications ;
- ✓ Les applications qui les animent manipulent des données.

La figure ci-dessous présente le modèle général des architectures client-serveur



Modèle général des architectures client-serveur

Source-image: Livre les Architectures client-serveur et internet et intranet des CGI aux EJB [Pierre-Yves Cloux, David Doussot, Aurélien Géron]

- **L'espace serveur** n'est pas aussi monolithique que la figure ci-dessus le présente : il n'y a rarement qu'un seul serveur (que l'on parle du serveur physique ou des serveurs applicatifs qu'il héberge).

On peut citer quelques exemples de serveur :

- ✓ **Le serveur d'objet** qui exécute la logique applicative, auquel le client se connecte par l'intermédiaire d'un protocole d'objets distribués. Les objets qu'héberge ce serveur peuvent éventuellement être conditionnés sous la forme de composants.
- ✓ **Le moniteur transactionnel** dont le but est de coordonner l'exécution de transactions distribuées sur plusieurs machines.
  Le moniteur transactionnel est très utile pour coordonner entre plusieurs transactions simultanées et s'assurer de leur atomicité.
- ✓ **Le serveur Web** qui, selon les cas, a rôle de serveur de fichiers ou celui de middleware.
- ✓ **Le serveur de sécurité** qui gère l'authentification des utilisateurs et le chiffrement des flux de communication.

- **L'espace client** est souvent unique pour un utilisateur donnée, et sa localisation physique est en principe indéterminée. En effet, le logiciel client n'est pas lié à la machine qui le fait tourner, il doit être portable et mobile.
- **L'interface utilisateur(UI)**est un type d'interface informatique qui permet à un usager de manipuler la machine. Elle coordonne les interactions homme-machine **[3]**.

Des machines comme le Macintosh d'Apple et le système d'exploitation Windows de Microsoft ont popularisé les interfaces graphiques dans les années80.
Aujourd'hui les interfaces utilisateur sont interactives, multimédia, intelligentes et, de préférence, portables (l'interface graphique peut alors fonctionner sous tout système d'exploitation et conserver toujours le même aspect).

- **Les applications**

Le choix en matière de langage de développement est considérable. Il faut savoir qu'ils sont le plus souvent de haut niveau, c'est-à-dire qu'ils restent éloignés des préoccupations « bassement matérielles » comme la gestion de la mémoire ou la programmation réseau : ces détails sont réglés automatiquement. Le développeur peut donc se concentrer sur la logique de l'application.

- **Les données**

La très grande majorité des données de gestion sont encore à l'heure actuelle stockées dans des bases de données non relationnelles dans des mainframes. Jusqu'à une époque très récente, la plupart des éditeurs de solutions intranet avaient purement et simplement oublié ce fait.
Les accès se limitaient aux données relationnelles (Sybase, Oracle, etc., à travers SQL), souvent exclusivement au travers de l'interface ODBC (Open Data base Connectivity) de Microsoft.

- **Middlewares**

Pour lier efficacement les différentes composantes d'une application client-serveur (UI, application et donnée) il est parfois nécessaire d'avoir recours à des logiciels spécifiques : les middlewares. Comme leur nom l'indique, ce sont des intermédiaires. Ils servent à simplifier le travail du développeur en se chargeant d'un point technique particulier. Ils sont de plusieurs types, les principaux étant :

  - **Les middlewares d'accès aux données** (drivers de base de données relationnelles par exemple). Ils mettent en communication les applications avec les différentes sources de données de l'entreprise ;

  - **Les Object Request Broker** (Corba, DCOM, ou RMI). Leur but premier est de rendre plus transparente la communication avec des applications situées sur des machines distantes, grâce à ce qu'on appelle des « objets distribués » ;

  - **Les moniteurs transactionnels**. Leurs fonction est de coordonner des transactions distribuées sur plusieurs machines ;

- **Les Messages Oriented Middleware** (MOM) ils s'agissent de serveurs qui servent d'intermédiaires dans la communication entre applications. Ils peuvent stocker des messages dans une file d'attente et attendre que l'application destinataire soit prête à recevoir le message qui lui est adressé. Ainsi, on découple les applications sur le réseau, ce qui permet d'avoir un système plus performant et plus stable. Les MOM offrent aussi une bonne résistance aux interruptions de services.

1.4. Conclusion du chapitre

Dans ce premier chapitre nous avons eu un aperçu sur Android, les applications mobiles et les services web, cela va nous faciliter d'entamer les prochains chapitres qui sont la conception et la réalisation de l'application mobile sous Android.

Deuxième chapitre : Conception

Introduction

Dans le premier chapitre nous avons clarifié les notions nécessaires pour pouvoir comprendre la suite de ce travail et entamer la partie conception et réalisation de ce projet.
Nous avons cerné le problème posé dans un contexte bien défini, nous avons fait une analyse et nous avons proposé des solutions adéquates à ce type de problème.
Comme nous l'avons précisé dans le chapitre précédent, aboutir à une solution fiable et optimale pour un problème quelconque, nécessite une analyse et une étude bien déterminées, cependant il existe une panoplie d'approches, de méthodes et de langages standards et professionnelles permettant de bien concevoir et mieux structurer son projet.
Parmi ces langages on citera UML, un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en développement logiciel et en conception orientée objet.
UML se définit comme un langage de modélisation graphique et textuel
destiné à comprendre et décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue. **[15]**
UML est utilisé pour spécifier, visualiser, modifier et construire les documents nécessaires au bon développement d'un logiciel orienté objet. UML offre un standard de modélisation, pour représenter l'architecture logicielle.

Les différents éléments représentables sont :

- ✓ Activité d'un objet/logiciel
- ✓ Acteurs
- ✓ Processus
- ✓ Schéma de base de données
- ✓ Composants logiciels
- ✓ Réutilisation de composants

UML 2 s'articule autour de treize types de diagrammes, chacun d'eux étant dédié à la représentation des concepts particuliers d'un système logiciel.
Ces types de diagrammes sont répartis en deux grands groupes :

- **Six diagrammes structurels:**

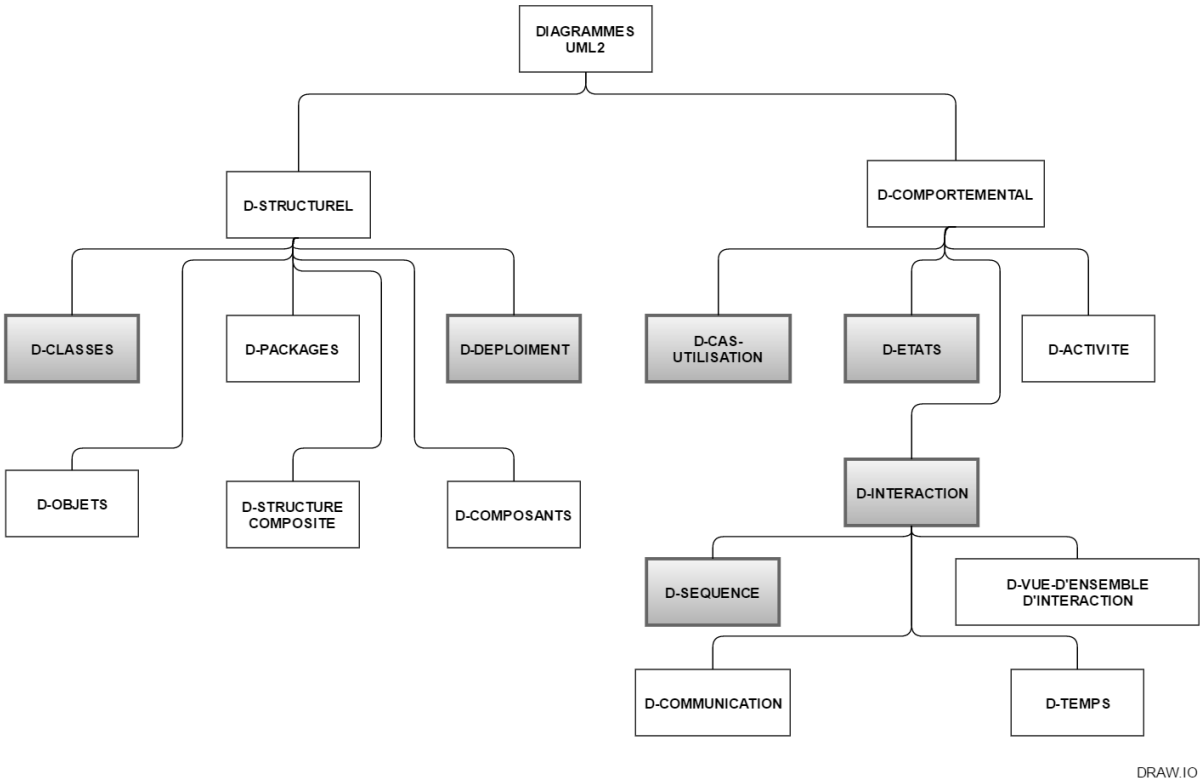
  1. **Diagramme de classes:** Il montre les briques de base statiques : classes, associations, interfaces, attributs, opérations, généralisations, etc.
  2. **Diagramme d'objets:** Il montre les instances des éléments structurels et leurs liens à l'exécution.
  3. **Diagramme de packages:** Il montre l'organisation logique du modèle et les relations entre packages.
  4. **Diagramme de structure composite:** Il montre l'organisation interne d'un élément statique complexe.
  5. **Diagramme de composants:** Il montre des structures complexes, avec leurs interfaces fournies et requises.
  6. **Diagramme de déploiement:** Il montre le déploiement physique des « artefacts » sur les ressources matérielles.

- **Sept diagrammes comportementaux:**

  1. **Diagramme de cas d'utilisation:** Il montre les interactions fonctionnelles entre les acteurs et le système à l'étude.
  2. **Diagramme de vue d'ensemble des interactions:** Il fusionne les diagrammes d'activité et de séquence pour combiner des fragments d'interaction avec des décisions et des flots.
  3. **Diagramme de séquence**: Il montre la séquence verticale des messages passés entre objets au sein d'une interaction.

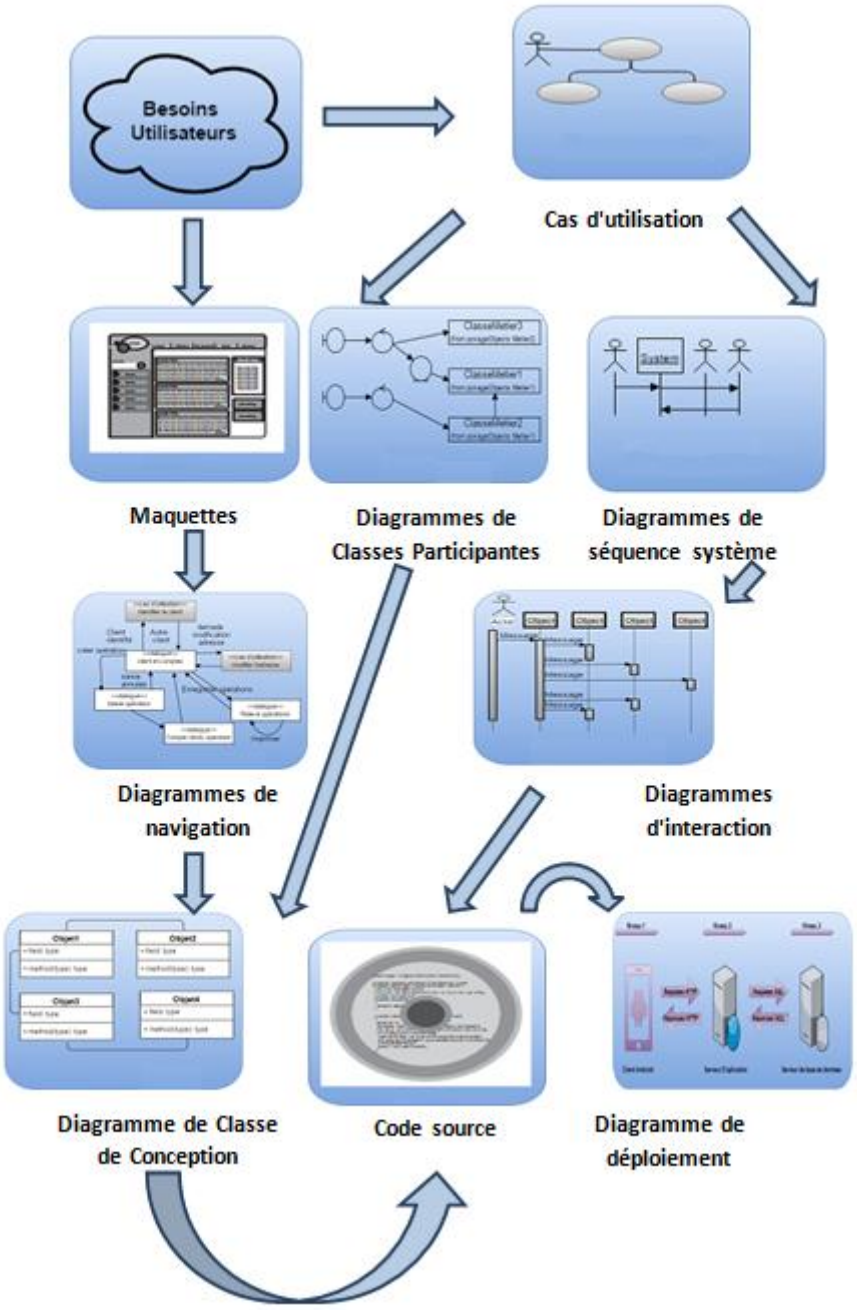
4. **Diagramme de communication:** Il montre la communication entre objets dans le plan au sein d'une interaction.
5. **Diagramme de temps:** Il fusionne les diagrammes d'états et de séquence pour montrer l'évolution de l'état d'un objet au cours du temps.
6. **Diagramme d'activité:** Il montre l'enchaînement des actions et décisions au sein d'une activité.
7. **Diagramme d'états:** Il montre les différents états et transitions possibles des objets d'une classe.

La figure ci-dessous montre les six diagrammes utilisés dans ce projet



DIAGRAMMES UML UTILISES DANS NOTRE PROJET

Le processus de développement utilisé dans notre projet se résume dans la figure si dessous :



Nous allons commencer par la définition des besoins des utilisateurs, c'est l'étape préliminaire du processus de développement de ce projet.

1. Besoins des utilisateurs

1.1. Expression initiale des besoins utilisateurs

Un praticien a besoin de bien gérer son cabinet médical en planifiant d'une manière régulière son calendrier des rendez-vous et en répondant aux déférents besoins (médical ou informationnel) des patients.

Un patient a besoin de prendre rendez-vous chez le praticien le plus proche selon la spécialité et le lieu voulu, choisir le jour et l'heure du rendez-vous selon les disponibilités et le contacter en temps voulu.

1.2. Objectifs

- Faciliter la gestion des rendez-vous aux praticiens.
- Faciliter la prise de rendez-vous aux patients.

1.3. Exigences fonctionnelles

❖ **Praticiens :**

1. Inscription:
   Un praticien doit s'inscrire en introduisant ses coordonnées pour avoir une session personnelle lui permettant de gérer son compte.
2. Authentification:
   Un praticien déjà inscrit doit s'authentifier en introduisant son adresse email et son mot de passe afin de pouvoir gérer son calendrier des rendez-vous, gérer sa messagerie…etc.
3. Planification du calendrier des rendez-vous:
   Un praticien a la possibilité de planifier son calendrier des rendez-vous selon le jour et l'heur de sa disponibilité.
4. Consultation du catalogue des rendez-vous:
   Un praticien a la possibilité de consulter son catalogue des rendez-vous.
5. Gestion de la messagerie:
   Un praticien a la possibilité de contacter ses patients et recevoir des messages

.

❖ **Patients :**

1. Inscription:
   Un patient doit s'inscrire en introduisant ses coordonnées pour avoir une session personnelle lui permettant de gérer son compte.
2. Authentification:
   Un patient déjà inscrit doit s'authentifier en introduisant son adresse email et son mot de passe afin de pouvoir prendre un rendez-vous, gérer sa messagerie…etc.
3. Recherche d'un praticien:
   Un patient a la possibilité de rechercher un praticien selon ses coordonnées, sa localité et sa spécialité.
4. Consultation du calendrier des rendez-vous du praticien:
   Un patient a la possibilité de consulter le calendrier des rendez-vous du praticien.
5. Prise de rendez-vous:
   Un patient a la possibilité de prendre un rendez-vous en choisissant le jour et l'heure voulu.
6. Gestion de la messagerie:
   Un patient a la possibilité de contacter ses praticiens et recevoir des messages.

❖ **Administrateur:**

1. Authentification:
   L'administrateur doit s'authentifier afin de pouvoir gérer les utilisateurs
2. Gestion des utilisateurs:
   L'administrateur a la possibilité de bloquer ou supprimer définitivement un compte utilisateur, il a la possibilité de contacter un utilisateur ou de répondre à ses messages.
3. Gestion de la messagerie
   L'administrateur a la possibilité de contacter ou de répondre aux messages des utilisateurs.

1.4. Exigences non fonctionnelles

- **Exigences de qualité:**
  - Ergonomie sobre;
  - Clarté et lisibilité de l'application;
  - Facilité de navigation au sein de l'application;
- **Exigences de performance**
  - L'application fournit un service de géolocalisation Google Map;
  - L'application dispose d'un calendrier des rendez-vous dynamique et ajustable;

1.5. Contraintes de conception

- Un utilisateur doit d'abord s'inscrire pour avoir un compte utilisateur (praticien ou patient) personnalisé.
- Un praticien doit s'authentifier pour pouvoir planifier son calendrier des rendez-vous, consulter son catalogue des rendez-vous, gérer sa messagerie…etc.
- Un patient doit s'authentifier pour pouvoir consulter le calendrier des rendez-vous du praticien, prendre un rendez-vous, gérer sa messagerie…etc.
- Un patient doit introduire le nom et le prénom du praticien, sa ville et sa spécialité afin de pouvoir faire la recherche.
- Langages de développement : Java, PHP, MySQL.
- Système d'exploitation : Android.

2. Cas d'utilisation

2.1. Diagramme des cas d'utilisation

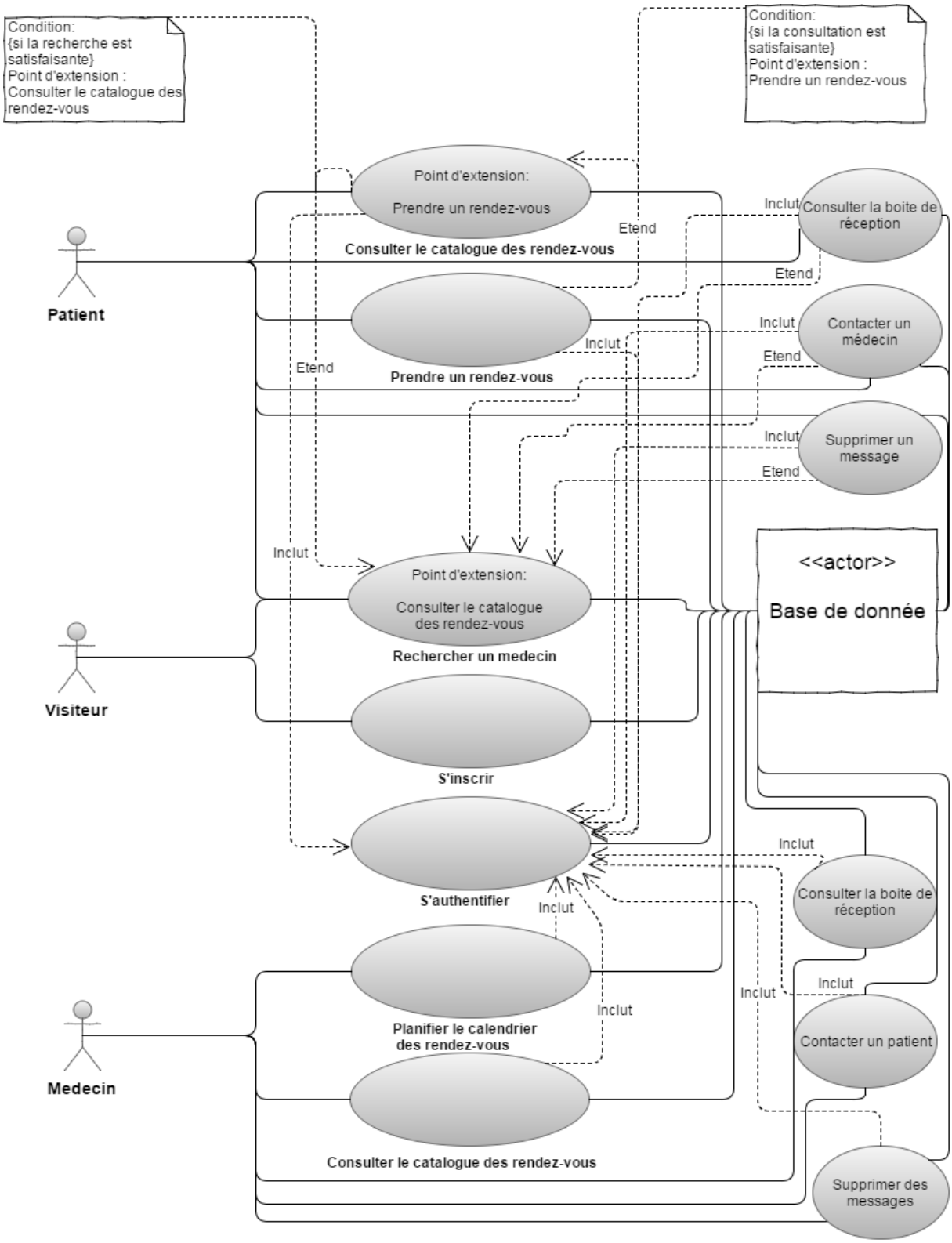


Diagramme des cas d'utilisation pour la gestion des rendez-vous d'un cabinet midical

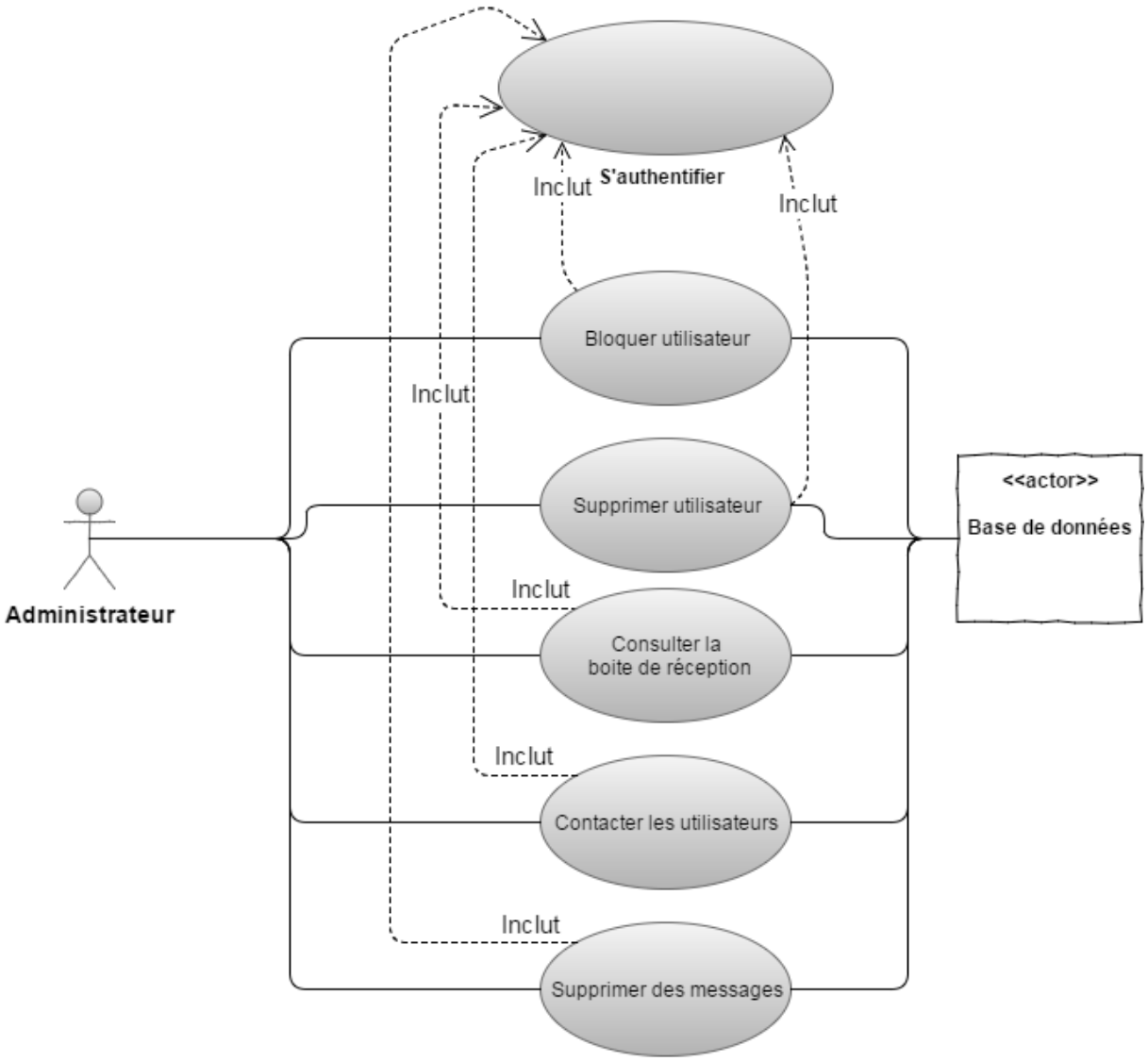


Diagramme des cas d'utilisation du SGRCM Administrateur

2.2. Description textuelle des cas d'utilisation

Description du casd'utilisation :S'inscrire

Titre : s'inscrire
Auteur : visiteur
Objectif : ce cas d'utilisation permet à un visiteur de s'inscrire autant qu'un patient ou médecin
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le visiteur doit choisir le statut de son compte (patient ou médecin). • Le visiteur doit renseigner le formulaire d'inscription.
Scénario nominal : 1. Le visiteur remplit le formulaire d'inscription ; 2. Le système client vérifie la validité des champs saisie par le visiteur ; 3. Le système client envoie les données saisie par le visiteur vers le système serveur ; 4. Le système serveur traite les données envoyées par le client ; 5. Le système serveur enregistre les données envoyées par le client dans la base de donnée ; 6. Le système serveur renvoie une notification pour confirmer au visiteur que l'inscription est réussie.
Enchaînement alternatif : A1: Les informations saisies par le visiteur ne sont pas valide. L'enchaînement alternatif démarre au point 3 du scénario nominal : 3. Le système client indique au visiteur que les informations saisies sont incorrecte ; 4. Le système client demande au visiteur de vérifier les informations saisies. Le scénario nominal reprend au point 1.
Post-condition : • Le visiteur est inscrit dans la base de données • Un message de confirmation de l'inscription est envoyé au visiteur.

Description du cas d'utilisation : S'authentifier

Titre : s'authentifier
Auteur : patient et médecin
Objectif : ce cas d'utilisation permet à un patient ou médecin de s'authentifier afin d'accéder à son compte personnel.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le visiteur doit renseigner le formulaire d'authentification
Scénario nominal : 1. Le patient/médecin remplit le formulaire d'authentification ; 2. Le système client vérifie la validité des champs saisie par le patient/médecin ; 3. Le système client envoie les données saisie par le patient/médecin vers le serveur ; 4. Le système serveur traite les données envoyées par le client ; 5. Le système serveur enregistre les données envoyées par le client dans la base de donnée ;
Enchaînement alternatif : A1: Les informations saisies par le patient/médecin ne sont pas valide. L'enchaînement alternatif démarre au point 3 du scénario nominal : 3. Le système client indique au patient/médecin que les informations saisies sont incorrecte ; 4. Le système client demande au patient/médecin de vérifier les informations saisies. Le scénario nominal reprend au point 1.
Post-condition : • L'authentification du patient/médecin est réussie • Le patient/médecin pourra se déconnecter de son compte

Description du cas d'utilisation : Planifier le calendrier des rendez-vous

Titre : Planifier le calendrier des rendez-vous
Auteur : médecin
Objectif : ce cas d'utilisation permet à un médecin de planifier son calendrier des rendez-vous afin de permettre aux patients de prendre des rendez-vous.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le médecin doit s'authentifier;
Scénario nominal : 1. Le médecin renseigne ses journées et ses horaires de disponibilité ; 2. Le système client envoie les données renseignées par le médecin vers le serveur ; 3. Le système serveur traite les données envoyées par le médecin ; 4. Le système serveur enregistre les données envoyées par le médecin dans la base de données ; 5. Le système serveur renvoie une notification pour confirmer au médecin que la planification est réussie.
Enchaînement alternatif : A1: Les informations saisies par le médecin ne sont pas valides. L'enchaînement alternatif démarre au point 2 du scénario nominal : 2. Le système client indique au médecin que les informations saisies sont incorrecte ; 3. Le système client demande au médecin de vérifier les informations saisies. Le scénario nominal reprend au point 1.
Post-condition : • Planification des rendez-vous réussie • Le médecin pourra modifier son calendrier

Description du cas d'utilisation : Consulter le catalogue des rendez-vous

Titre : Consulter le catalogue des rendez-vous
Auteur : médecin
Objectif : ce cas d'utilisation permet à un médecin de consulter son catalogue des rendez-vous.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Médecin doit s'authentifier.
Scénario nominal : 1. Le médecin consulte le catalogue des rendez-vous. 2. Le système affiche tous les rendez-vous pris par les patients.
Post-condition : • Consultation réussie

Description du cas d'utilisation :rechercher un médecin

Titre : Rechercher un médecin
Auteur : patient/visiteur
Objectif : ce cas d'utilisation permet à un patient/visiteur de faire la recherche des médecins.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le patient/visiteur doit renseigner le formulaire de recherche.
Scénario nominal : 1. Le patient/visiteur remplit le formulaire de recherche ; 2. Le système client vérifie la validité des champs saisie par le patient ; 3. Le système client envoie les données saisie par le patient vers le serveur ; 4. Le système serveur traite les données envoyées par le client ; 5. Le système serveur répond à la demande du client.
Enchaînement alternatif : A1: Les informations saisies par le patient/visiteur ne sont pas valide. L'enchaînement alternatif démarre au point 3 du scénario nominal : 3. Le système client indique au patient/visiteur que les informations saisies sont incorrecte ; 4. Le système client demande au patient/visiteur de vérifier les informations saisies. Le scénario nominal reprend au point 1. A2: Les informations recherchées par le patient/visiteur ne sont pas disponible L'enchaînement alternatif démarre au point 5 du scénario nominal : 5. Le système serveur indique au patient que les informations recherchées ne sont pas disponible. Le scénario nominal reprend au point 1.
Post-condition : • Trouver le médecin ou la listes des médecins recherchés ;

Description du cas d'utilisation : Consulter le catalogue des rendez-vous

Titre : consulter le catalogue des rendez-vous
Auteur : patient
Objectif : ce cas d'utilisation permet à un patient de consulter le catalogue des rendez-vous d'un médecin afin de pouvoir prendre un rendez-vous.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le patient doit s'authentifier. • Le patient doit faire la recherche d'un médecin.
Scénario nominal : 1. Le patient accède à l'espace de consultation du catalogue des rendez-vous ; 2. Le système serveur envoie le calendrier des rendez-vous planifié par le médecin ; 3. Le patient fait la consultation.
Post-condition : • Consultation réussie.

Description du cas d'utilisation : Consulter la boite de réception

Titre : Consulter la boite de réception
Auteur : médecin
Objectif :ce cas d'utilisation permet au médecin de consulter sa boite de réception.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le médecin doit s'authentifier. • Le médecin doit accéder à sa messagerie.
Scénario nominal : 1. Le médecin consulte sa boite de réception; 2. Le système serveur renvoie les messages envoyés par les patients ; 3. Le médecin fait la consultation.
Post-condition : • Consultation réussie.

Description du cas d'utilisation : Consulter la boite de réception

Titre : Consulter la boite de réception
Auteur : patient
Objectif :ce cas d'utilisation permet au patient de consulter sa boite de réception.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le patient doit s'authentifier. • Le patient doit accéder à sa messagerie.
Scénario nominal : 1. Le patient consulte sa boite de réception; 2. Le système serveur renvoie les messages envoyés par les patients ; 3. Le patient fait la consultation.
Post-condition : • Consultation réussie.

Description du cas d'utilisation : prendre un rendez-vous

Titre : prendre un rendez-vous
Auteur : patient
Objectif :ce cas d'utilisation permet au patient de prendre un rendez-vous.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le patient doit s'authentifier ; • Le patient doit faire la rechercher d'un médecin et consulter le calendrier des rendez-vous du médecin.

Scénario nominal : 1. Le patient prend un rendez-vous ; 2. Le système client envoie les informations nécessaires au système serveur ; 3. Le système serveur enregistre la demande du patient ; 4. Le système serveur renvoie une notification pour confirmer au patient que la prise de rendez-vous est réussie.
Post-condition : • Prise de rendez-vous réussie.

Description du cas d'utilisation : contacter un médecin

Titre :contacter un médecin
Auteur : patient
Objectif : ce cas d'utilisation permet au patient de contacter un médecin par la messagerie.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le patient doit s'authentifier. • Le patient doit faire une rechercher un médecin
Scénario nominal : 1. Le patient saisie un message pour le médecin; 2. Le système client vérifie la validité des coordonnées du médecin; 3. Le système client envoie le message au système serveur ; 4. Le système serveur fait des traitements ; 5. Le système serveur enregistre le message dans la base de données.
Enchaînement alternatif : A2: Les coordonnées saisies par le patient ne sont pas valides. L'enchaînement alternatif démarre au point 3 du scénario nominal : 3. Le système indique au patient que les coordonnées saisies ne sont pas valides. 4. Le système demande au patient de corriger les coordonnéesdu médecin. Le scénario nominal reprend au point 1.
Post-condition : • Message envoyé.

Description du cas d'utilisation : contacter un patient

Titre :contacter un patient
Auteur : patient
Objectif : ce cas d'utilisation permet au médecin de contacter un patient par la messagerie.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le médecin doit s'authentifier. • Le médecin doit consulter la messagerie.
Scénario nominal : 1. Le médecin saisie un message; 2. Le système client vérifie la validité des coordonnées du patient ; 3. Le système client envoie le message au système serveur ; 4. Le système serveur fait des traitements ; 5. Le système serveur enregistre le message dans la base de données.
Enchaînement alternatif : A3: Les coordonnées saisies par le médecin ne sont pas valides. L'enchaînement alternatif démarre au point 3 du scénario nominal : 3. Le système indique au médecin que les coordonnées saisies ne sont pas valides. 4. Le système demande au médecin de corriger les coordonnées du patient. Le scénario nominal reprend au point 1.
Post-condition : • Message envoyé.

Description du cas d'utilisation : Supprimer un message

Titre : Supprimer un message
Auteur : Médecin
Objectif : ce cas d'utilisation permet au médecin de supprimer un ou plusieurs messages de sa boite de réception.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le médecin doit s'authentifier. • Le médecin doit accéder à sa messagerie.
Scénario nominal : 1. Le médecin consulte sa boite de réception; 2. Le médecin choisit les messages à supprimer; 3. Le système client envoie la requête au système serveur; 4. Le système serveur supprime les messages; 5. Le système serveur renvoie une notification pour confirmer au médecin que la suppression est réussie.
Post-condition : • Suppression réussie.

Description du cas d'utilisation : Supprimer un message

Titre : Supprimer un message
Auteur : Patient
Objectif : ce cas d'utilisation permet au patient de supprimer un ou plusieurs messages de sa boite de réception.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • Le patient doit s'authentifier. • Le patient doit accéder à sa messagerie.
Scénario nominal : 1. Le patient consulte sa boite de réception; 2. Le patient choisit les messages à supprimer; 3. Le système client envoie la requête au système serveur; 4. Le système serveur supprime les messages; 5. Le système serveur renvoie une notification pour confirmer au patient que la suppression est réussie.
Post-condition : • Suppression réussie.

Description du cas d'utilisation : bloquer un compte utilisateur

Titre : bloquer utilisateur
Auteur : administrateur
Objectif : ce cas d'utilisation permet à l'administrateur de gérer les médecins/patients en bloquant un compte médecin/patients soit à cause d'une mauvaise réputation ou par demande d'un utilisateur.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • L'administrateur doit s'authentifier.
Scénario nominal : 1. L'administrateur remplit le formulaire d'authentification ; 2. Le système vérifier l'authenticité de l'administrateur ; 3. L'administrateur consulte la liste des médecins/patients inscrit ;
Post-condition : • blocage d'un compte réussi.

Description du cas d'utilisation : S'authentifier

Titre : S'authentifier
Auteur : administrateur
Objectif : ce cas d'utilisation permet à l'administrateur de s'authentifier afin d'accéder à son compte personnel.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition :

• L'administrateur s'inscrire.
Scénario nominal : 1. L'administrateur remplit le formulaire d'authentification ; 2. Le système vérifier l'authenticité de l'administrateur ; 3. L'administrateur accède à son espace personnel;
Post-condition : • Authentification réussie.

Description du cas d'utilisation : Consulter la boite de réception

Titre : Consulter la boite de réception
Auteur : Administrateur
Objectif : ce cas d'utilisation permet à l'administrateur de consulter sa boite de réception.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • L'administrateur doit s'authentifier. • L'administrateur doit accéder à sa messagerie.
Scénario nominal : 1. Le administrateur consulte sa boite de réception; 2. Le système serveur renvoie les messages envoyés par les utilisateurs ; 3. L'administrateur fait la consultation.
Post-condition : • Consultation réussie.

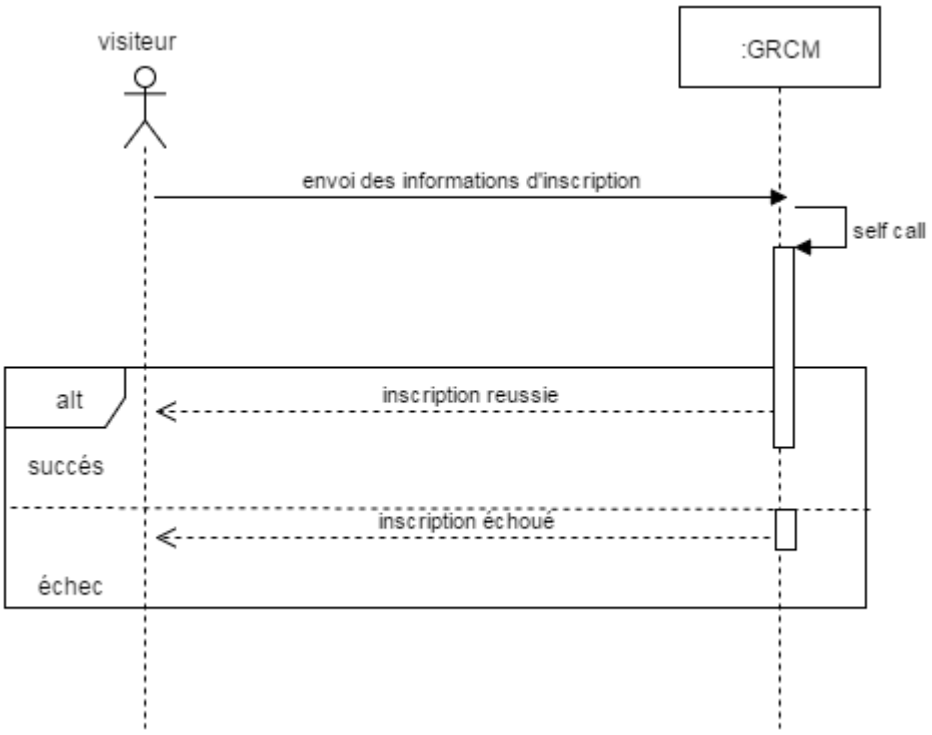
Description du cas d'utilisation : contacter les utilisateurs

Titre : contacter les utilisateurs
Auteur : administrateur
Objectif : ce cas d'utilisation permet à l'administrateur de contacter les utilisateurs.
Date de création : 26/05/2016
Date de mise à jour : 26/05/2016
Version : 1.0
Responsable : Amiar Abderazak
Pré-condition : • L'administrateur doit s'authentifier. • L'administrateur doit consulter la messagerie.
Scénario nominal : 1. L'administrateur saisie un message; 2. Le système client vérifie la validité des coordonnées de l'utilisateur; 3. Le système client envoie le message au système serveur; 4. Le système serveur fait des traitements; 5. Le système serveur enregistre le message dans la base de données;
Enchaînement alternatif : A1: Les coordonnées saisies par l'administrateur ne sont pas valides. L'enchaînement alternatif démarre au point 3 du scénario nominal : 1. Le système indique à l'administrateur que les coordonnées saisies ne sont pas valides. 2. Le système demande à l'administrateur de corriger les coordonnées de l'utilisateur. Le scénario nominal reprend au point 1.
Post-condition : • Message envoyé.

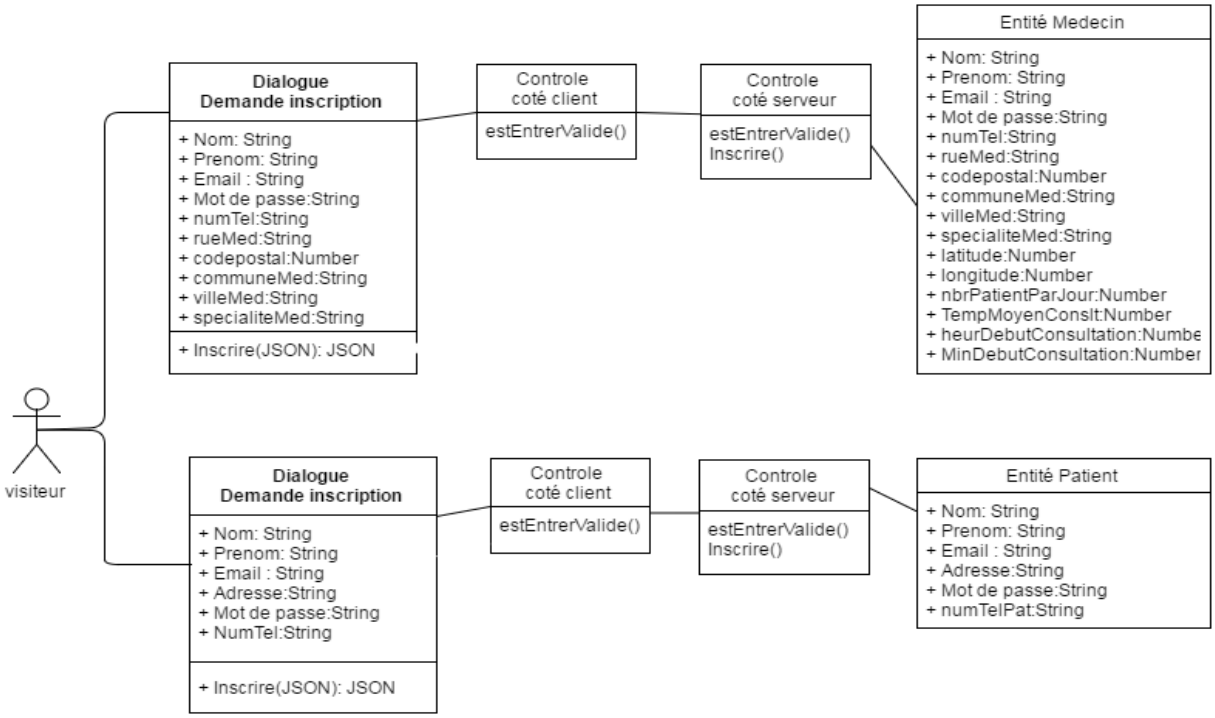
3. Processus du développement des cas d'utilisation

3.1. Inscription visiteur:

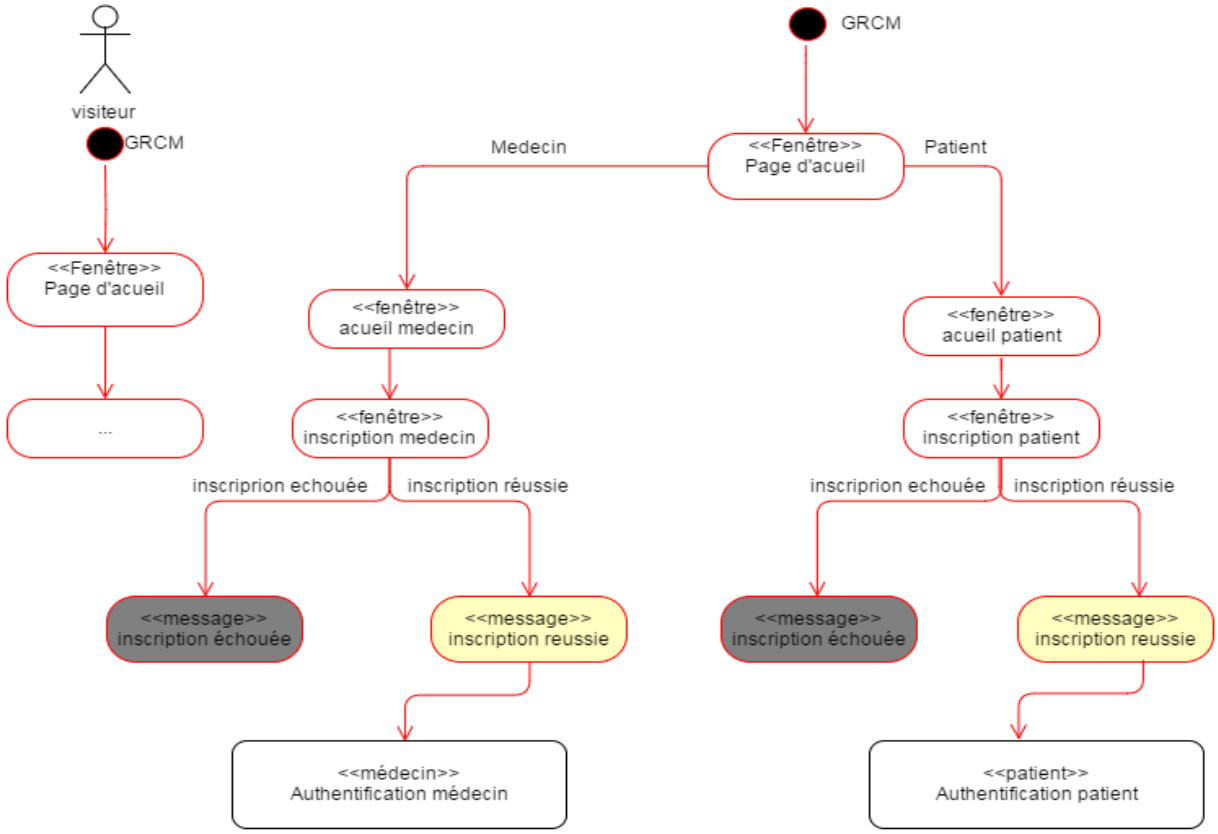
3.1.1. Diagramme de séquence système



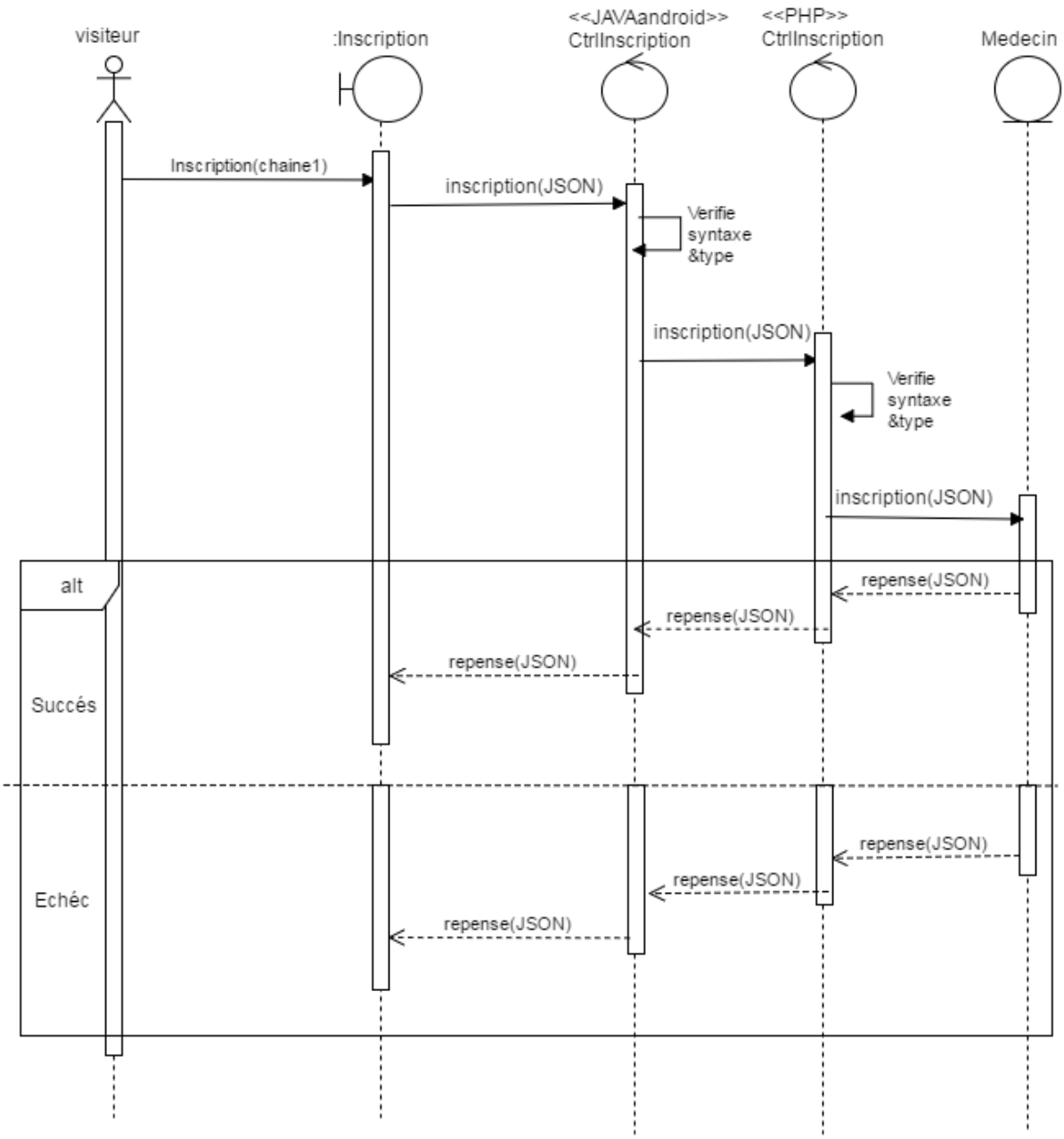
3.1.2. Diagramme de classes participantes



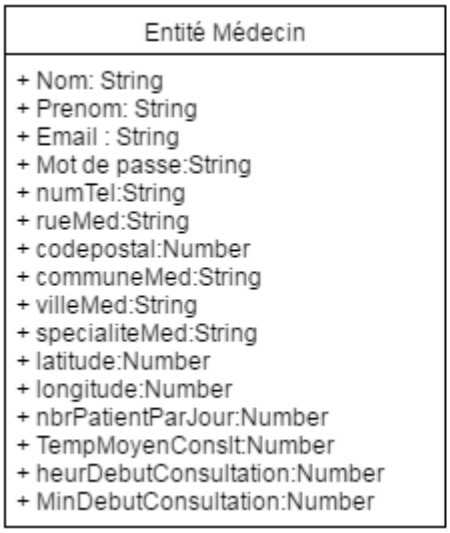
3.1.3. Diagramme de navigation

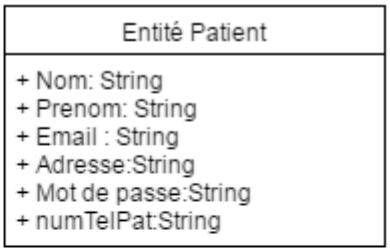


3.1.4. Diagramme d'interaction



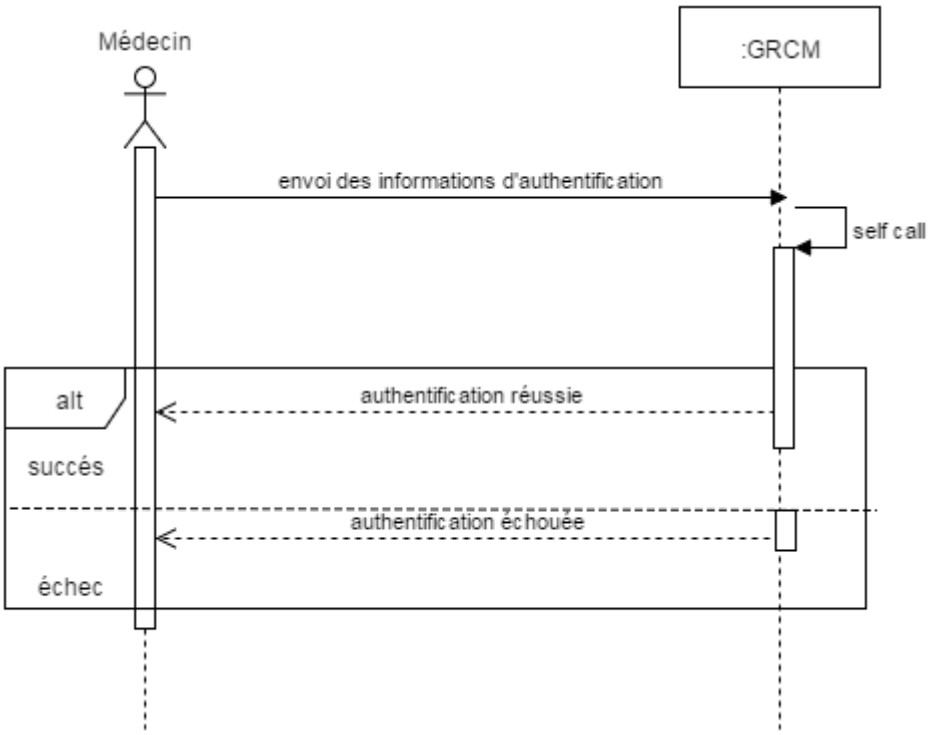
3.1.5. Diagramme de classe de conception préliminaire



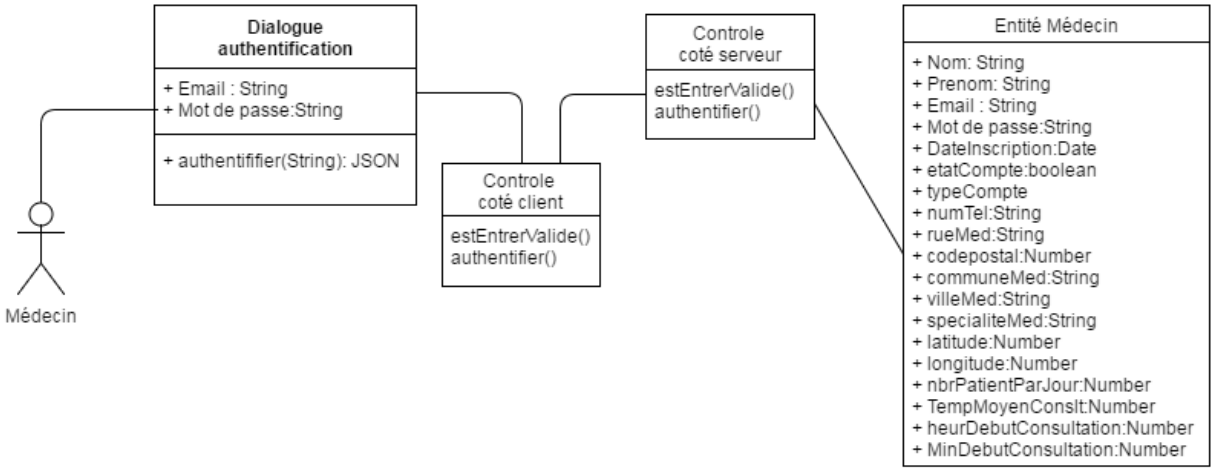


3.2. Authentification médecin:

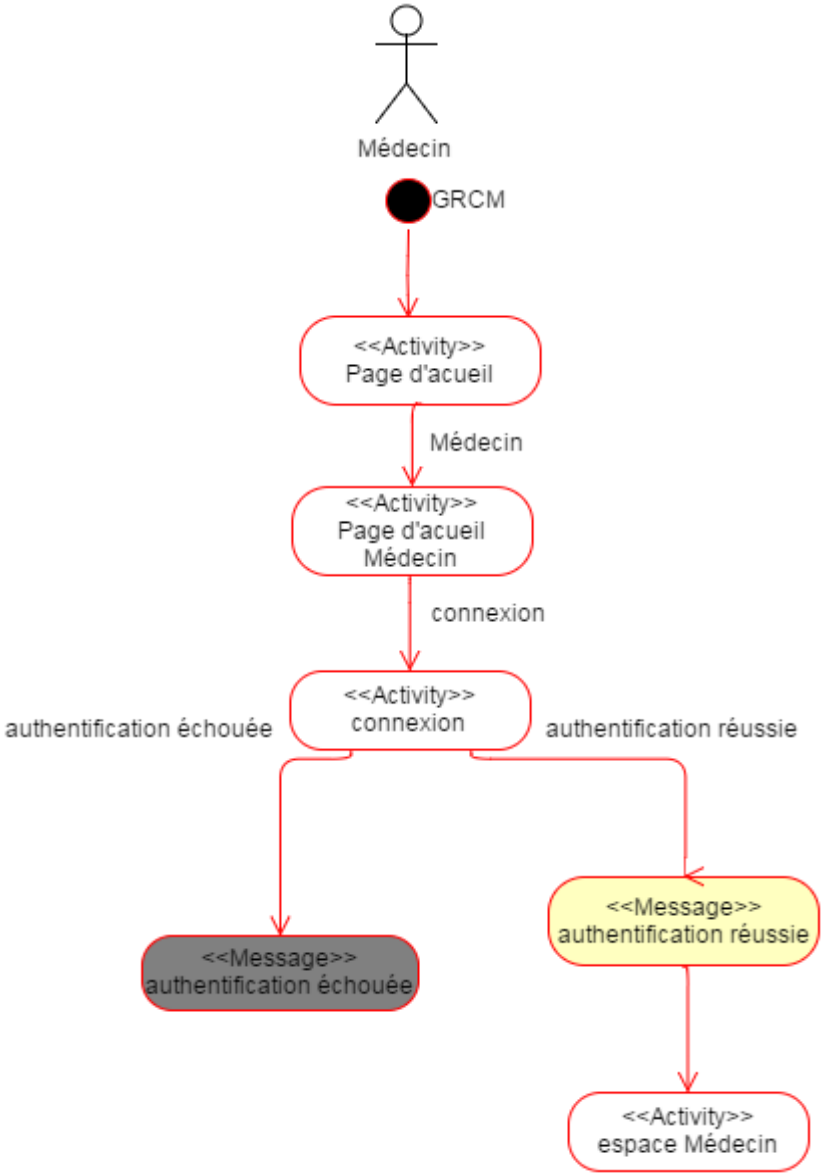
3.2.1. Diagramme de séquence système



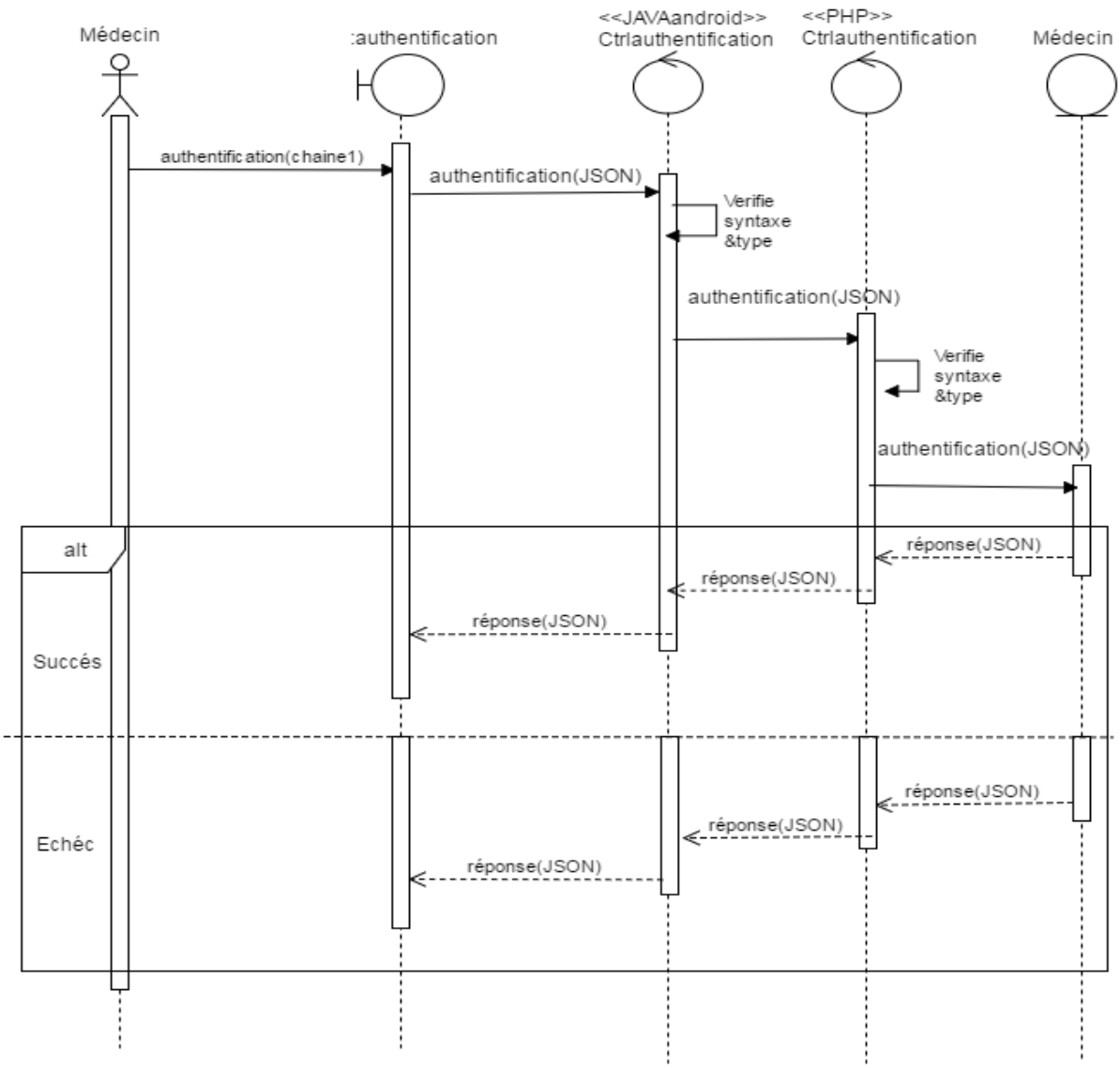
3.2.2. Diagramme de classe participantes



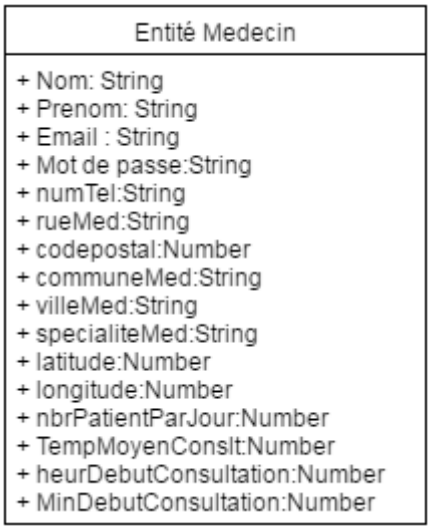
3.2.3. Diagramme de navigation



3.2.4. Diagramme d'interaction

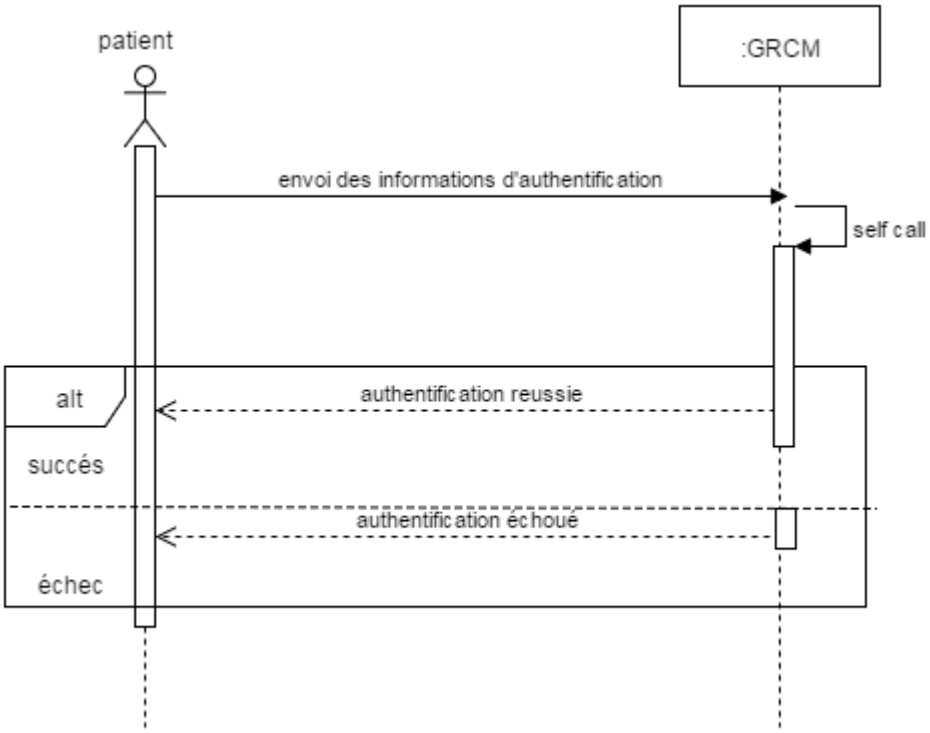


3.2.5. Diagramme de classes de conception préliminaire

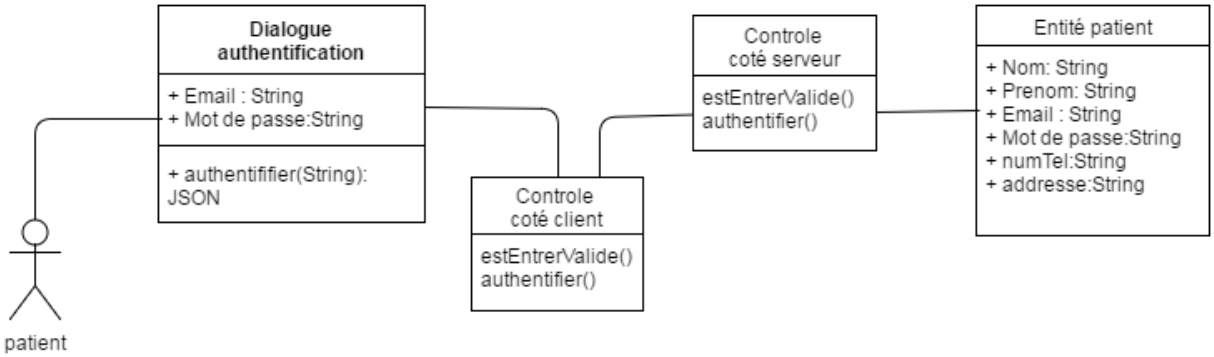


3.3. Authentification patient

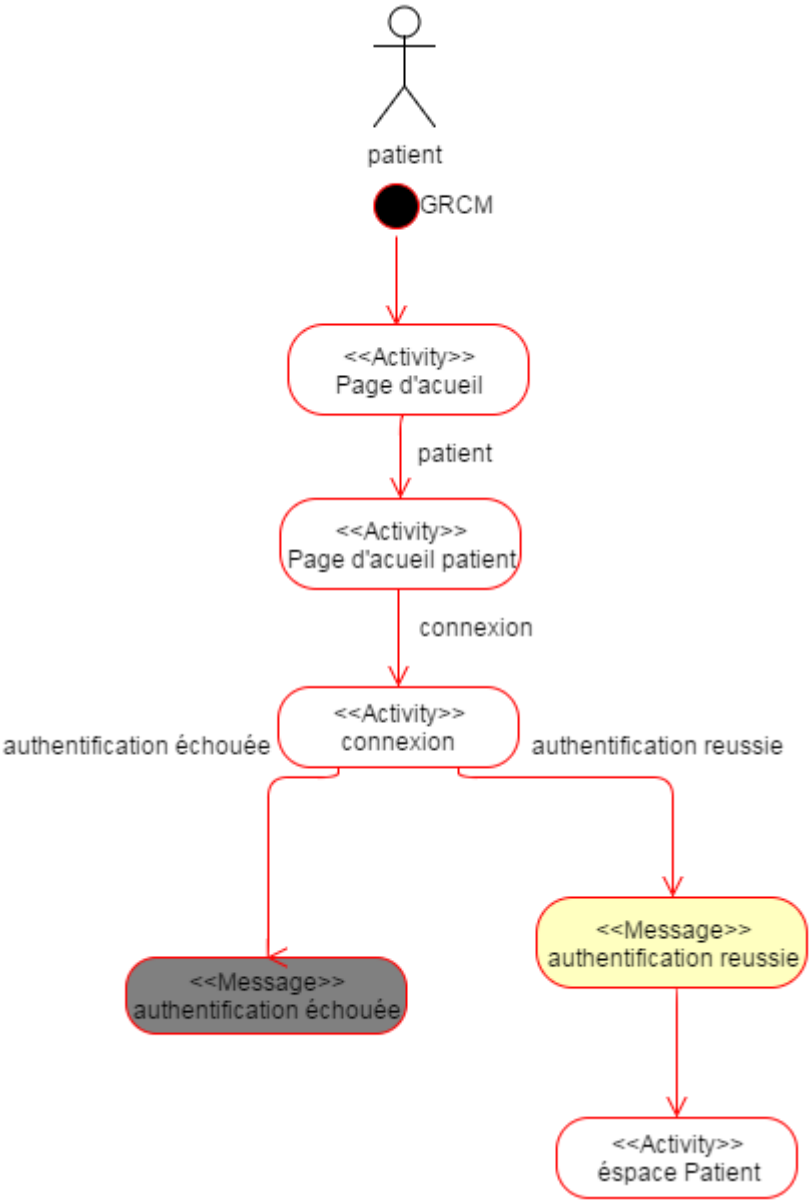
3.3.1. Diagramme de séquence système



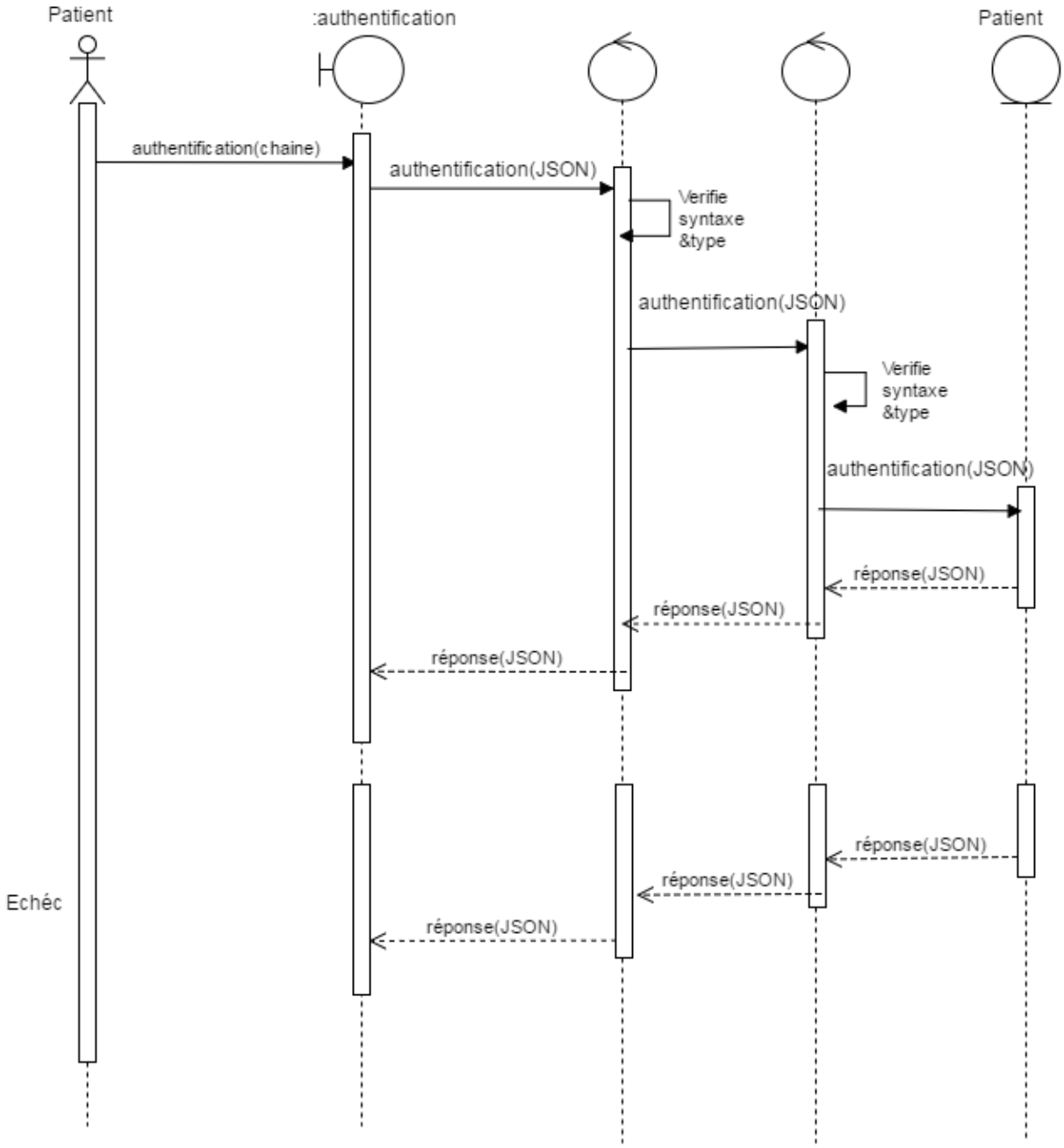
3.3.2. Diagramme de classe participantes



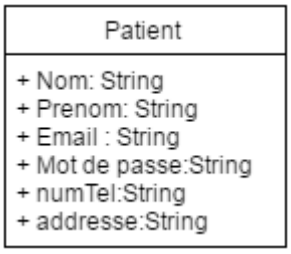
3.3.3. Diagramme de navigation



3.3.4. Diagramme d'interaction

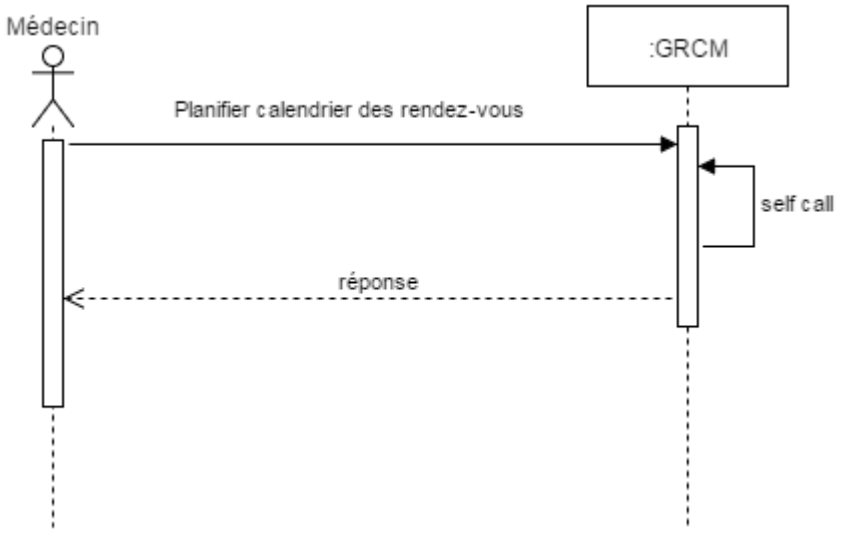


3.3.5. **Diagramme de classe de conception préliminaire**

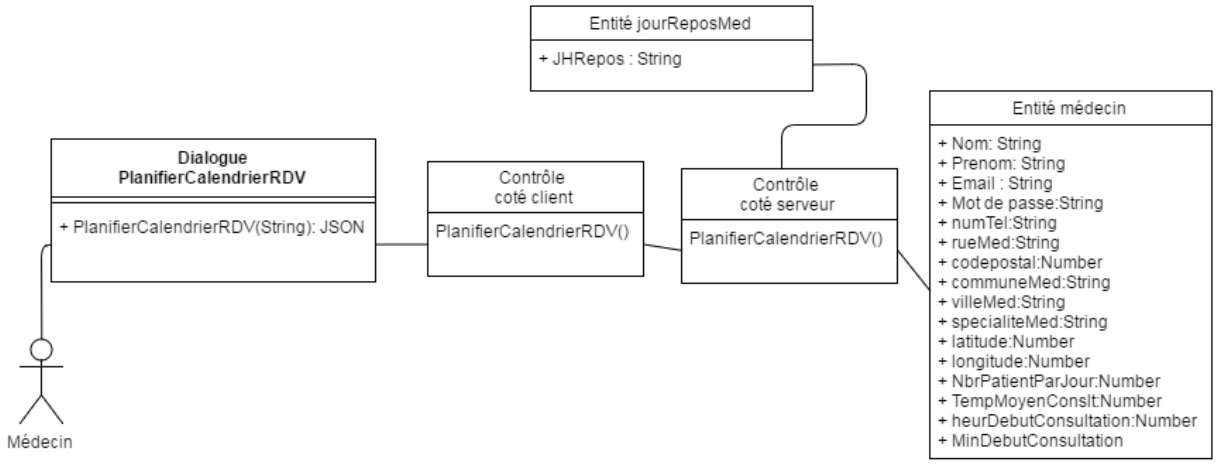


3.4. Planifier calendrier des rendez-vous

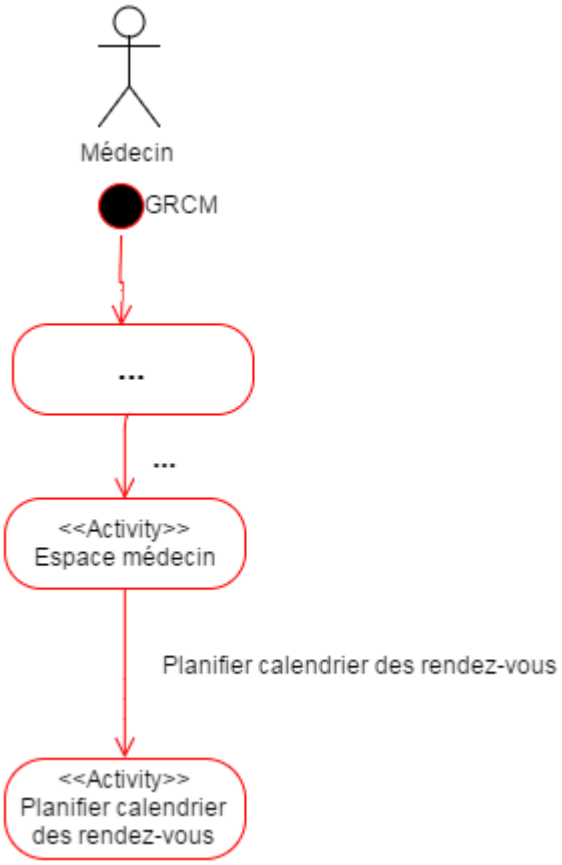
3.4.1. Diagramme de séquence système



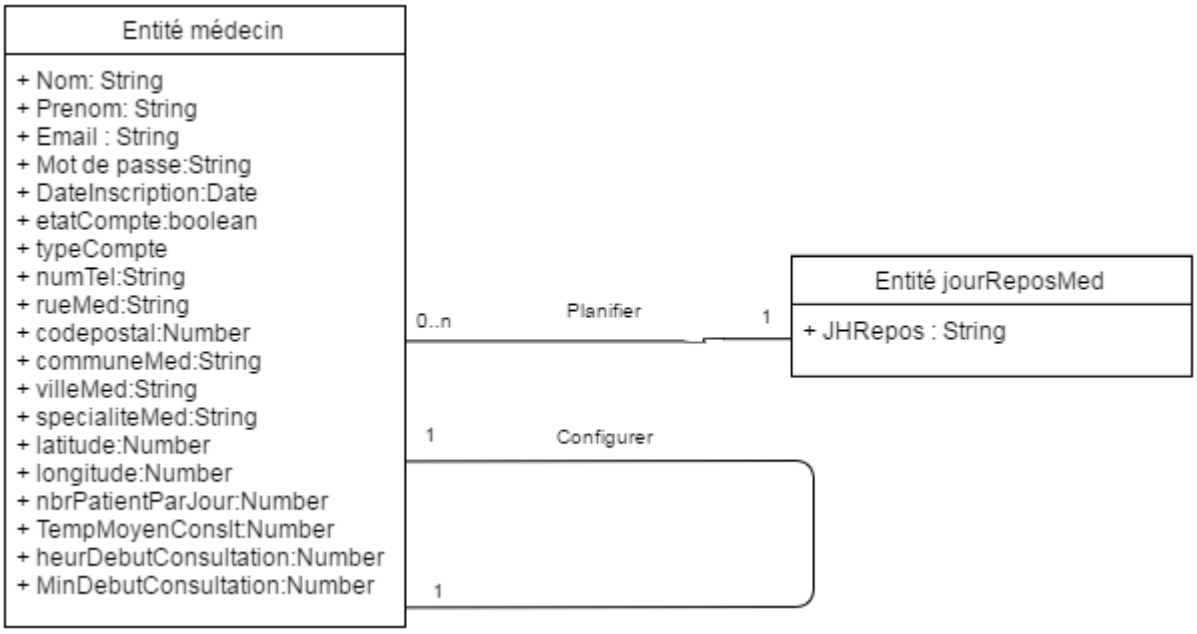
3.4.2. Diagramme de classe participantes



3.4.3. Diagramme de navigation

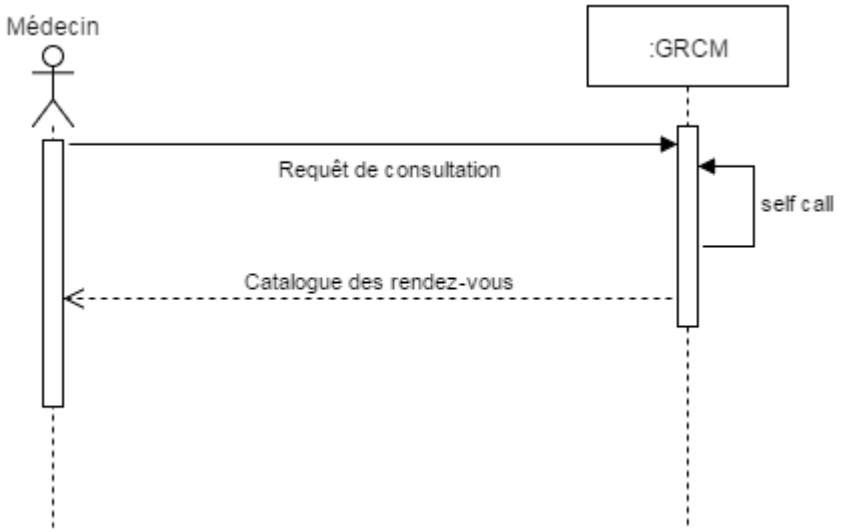


3.4.4. **Diagramme de classe de conception préliminaire**

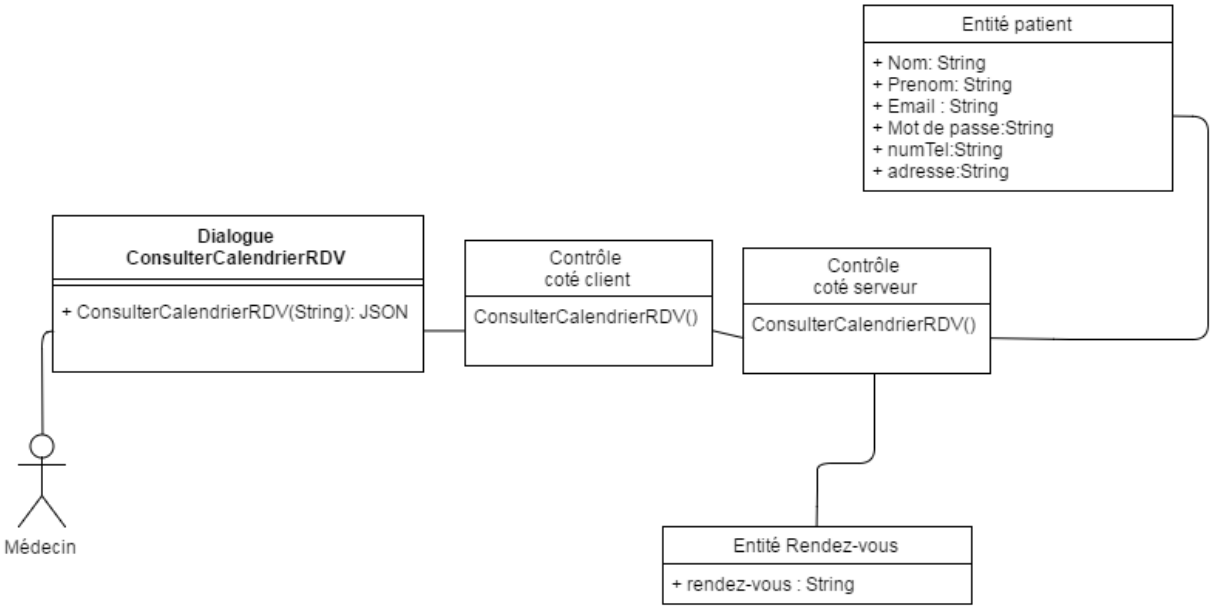


3.5. Consulter catalogue des rendez-vous

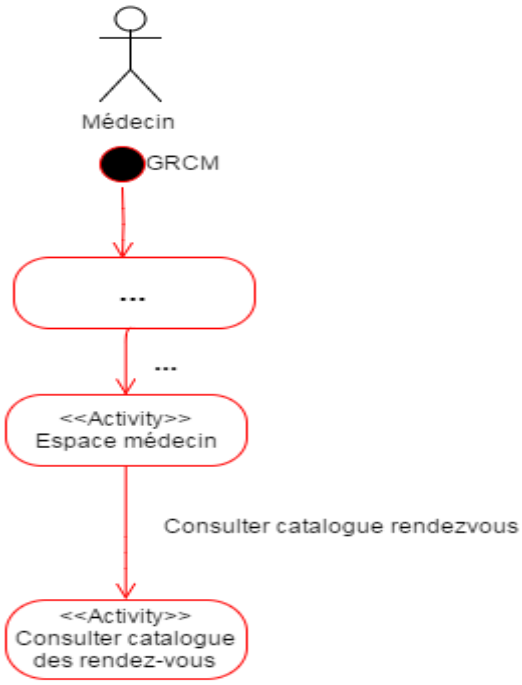
3.5.1. Diagramme de séquence système



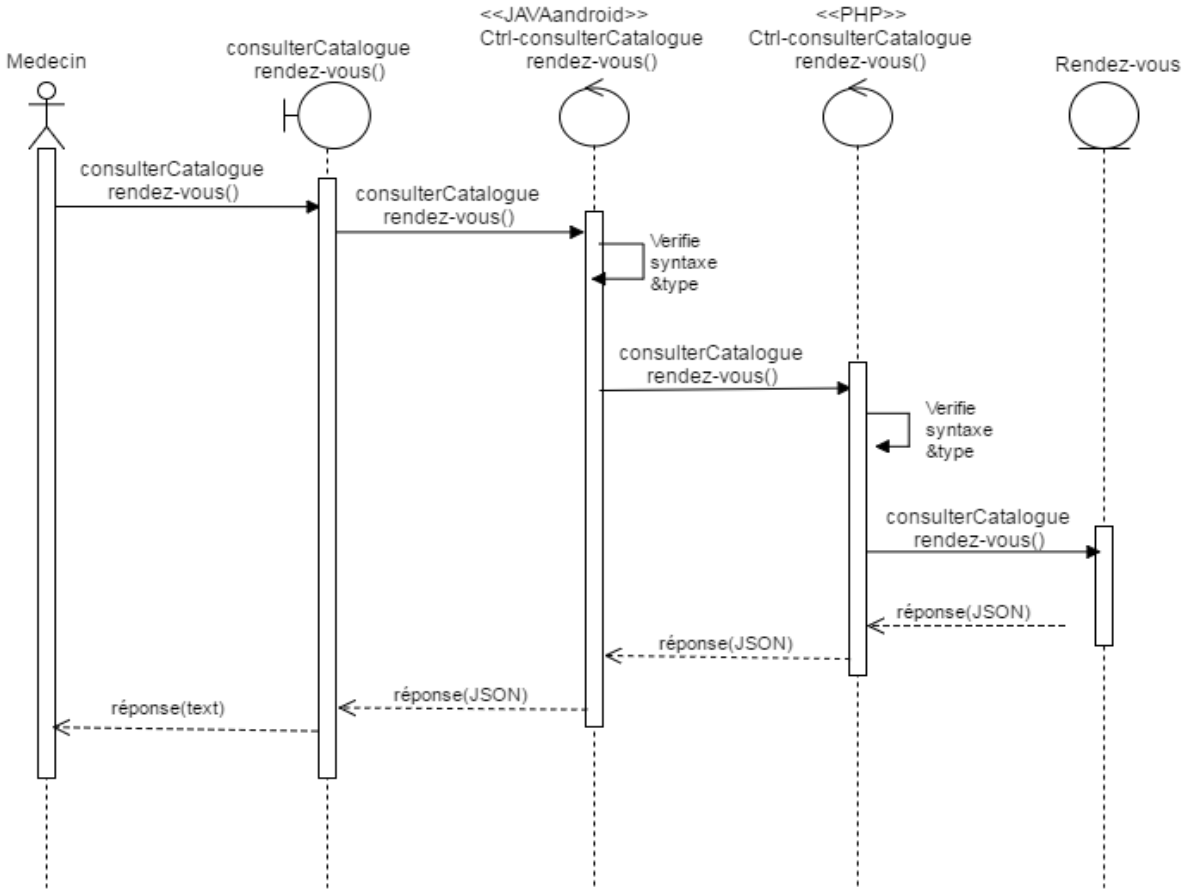
3.5.2. Diagramme de classes participantes



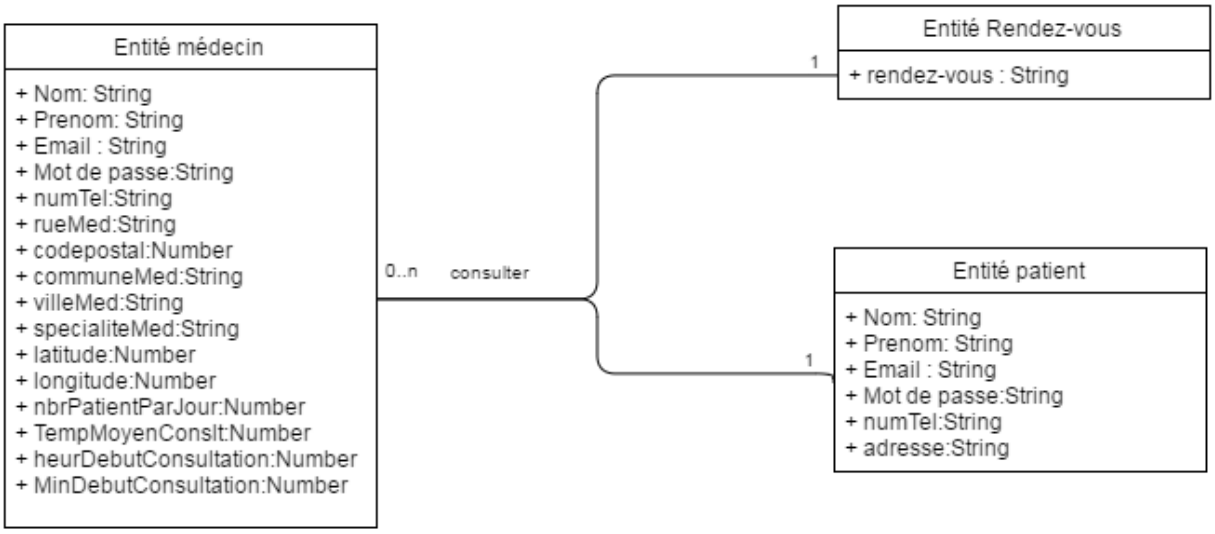
3.5.3. Diagramme de navigation



3.5.4. Diagramme d'interaction

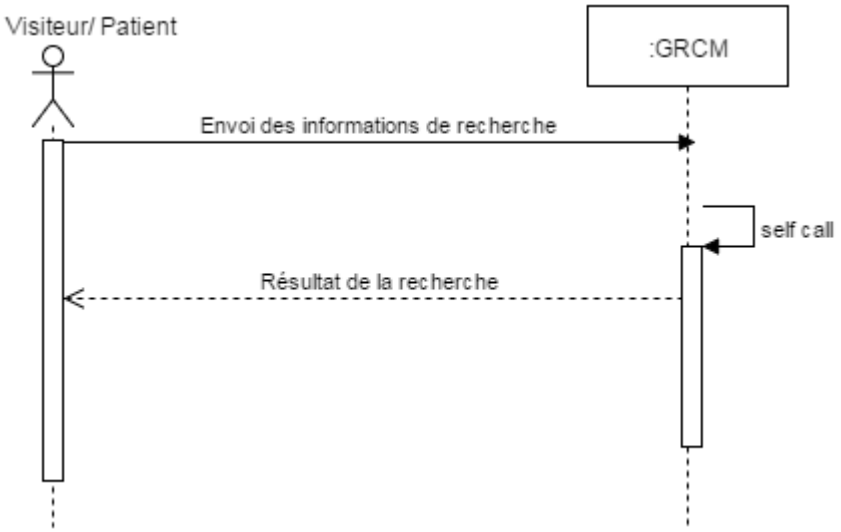


3.5.5. Diagramme de classes de conception préliminaires

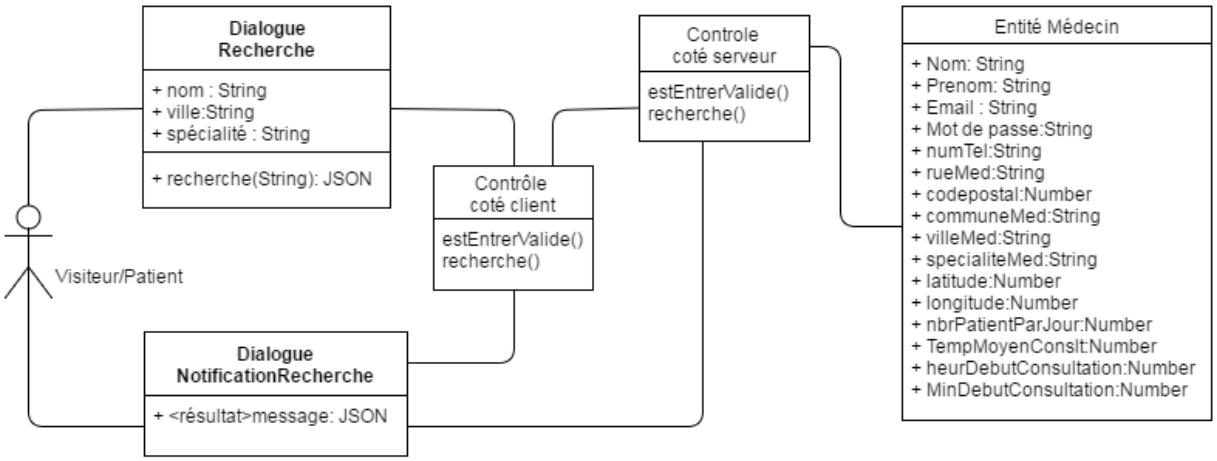


3.6. rechercher un médecin

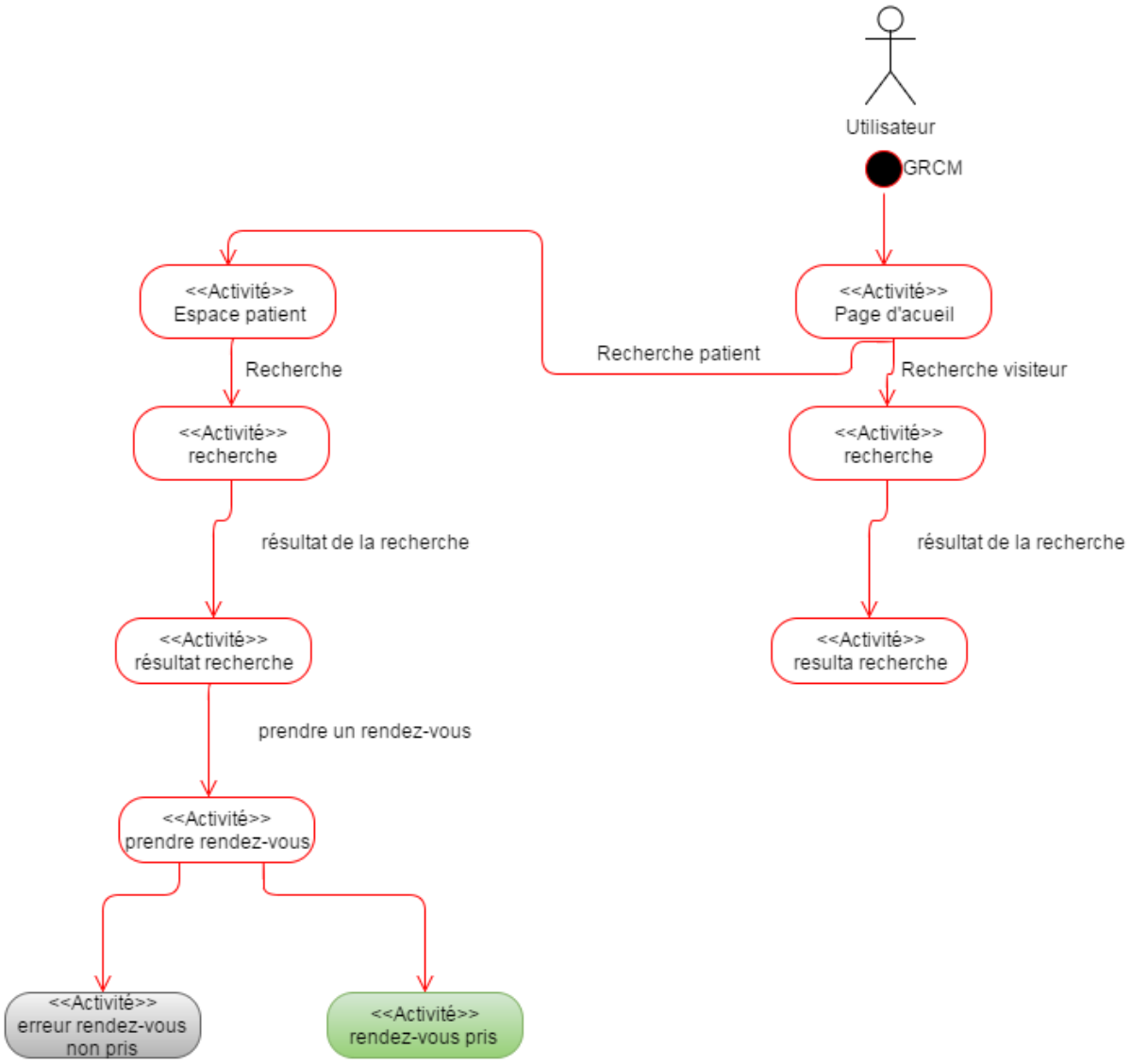
3.6.1. Diagramme de séquence système



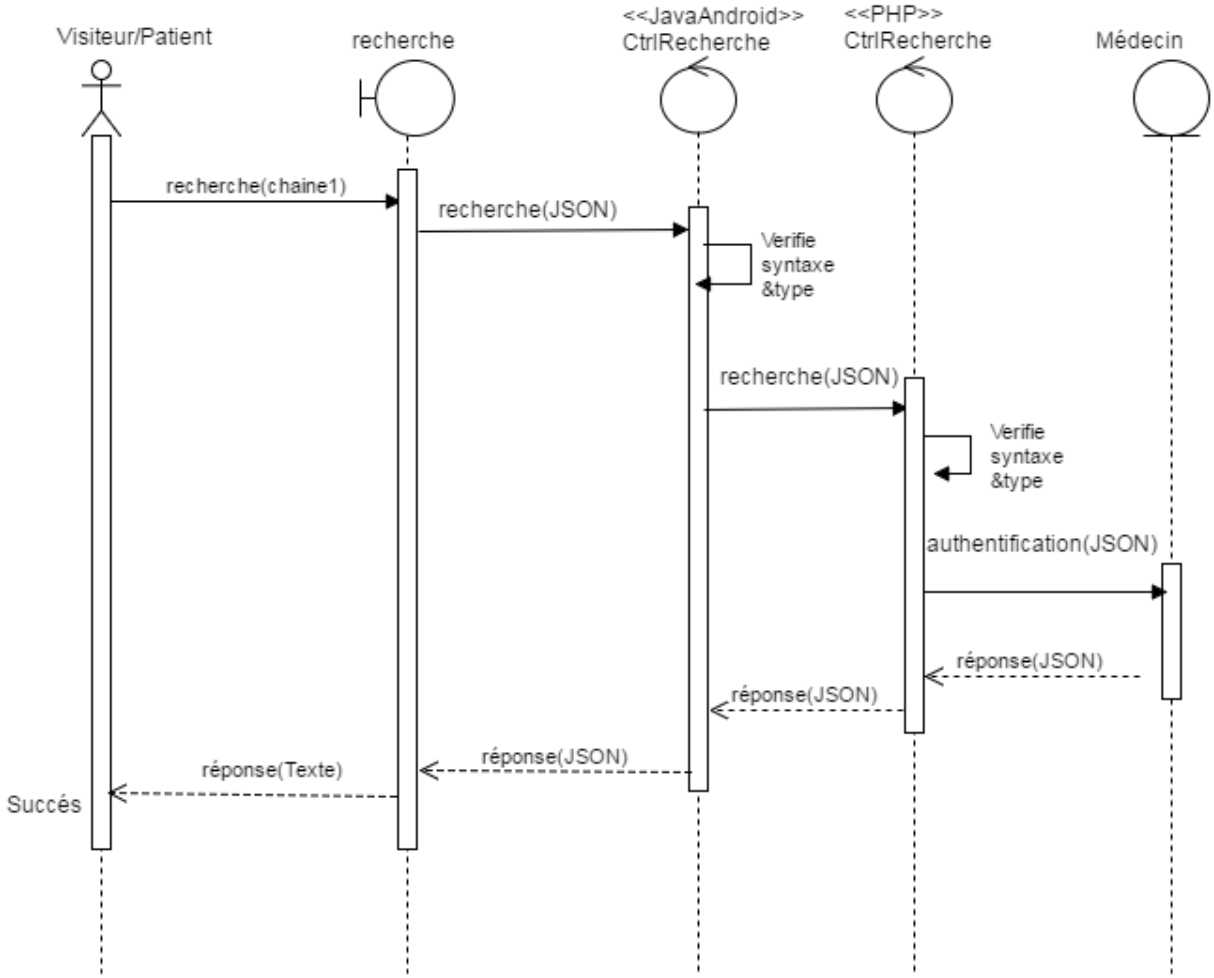
3.6.2. Diagramme de classes participantes



3.6.3. Diagramme de navigation



3.6.4. Diagramme d'interaction

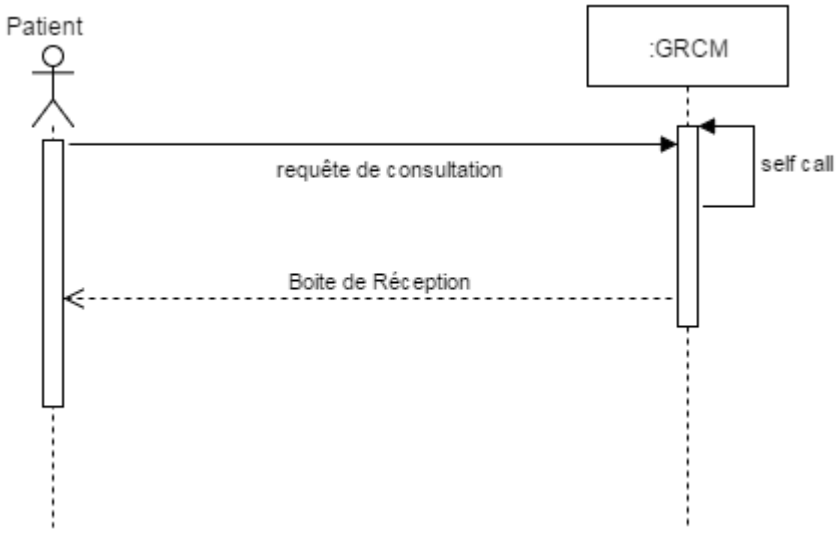


3.6.5. Diagramme de classes de conception préliminaires

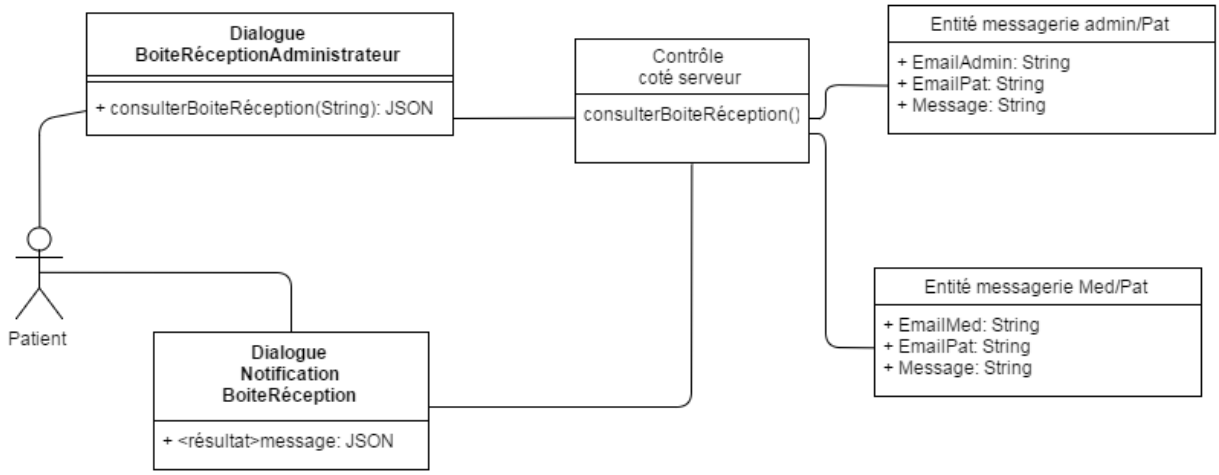
Entité medecin
+ Nom: String + Prenom: String + Email : String + Mot de passe:String + numTel:String + rueMed:String + codepostal:Number + communeMed:String + villeMed:String + specialiteMed:String + latitude:Number + longitude:Number + nbrPatientParJour:Number + TempMoyenConslt:Number + heurDebutConsultation:Number + MinDebutConsultation:Number

3.7. Consulter la boite de réception patient

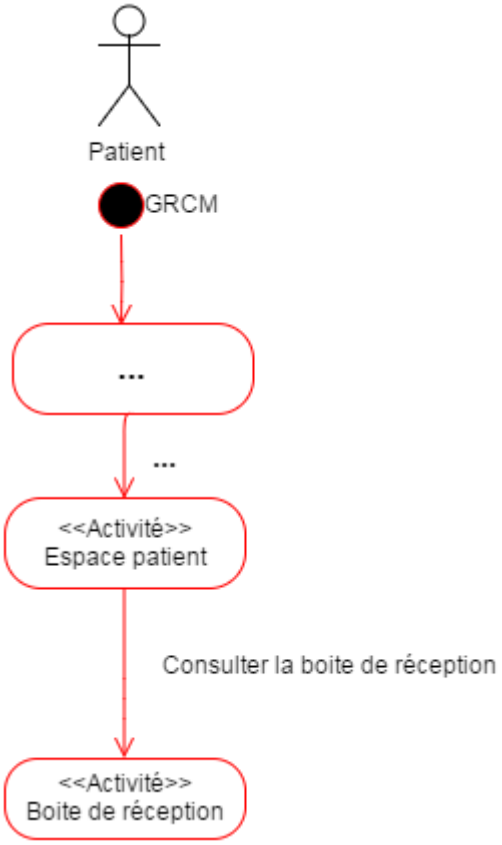
3.7.1. Diagramme de séquence système



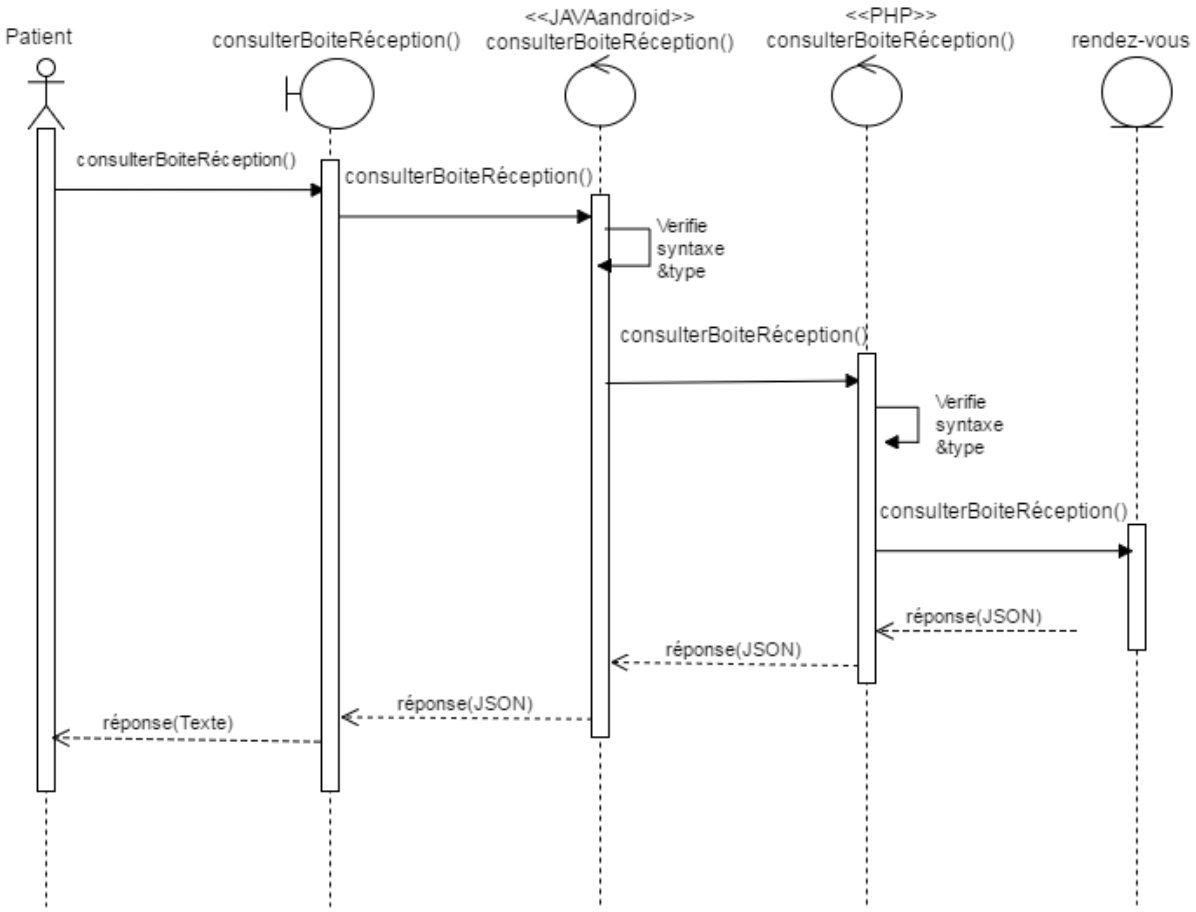
3.7.2. Diagramme de classes participantes



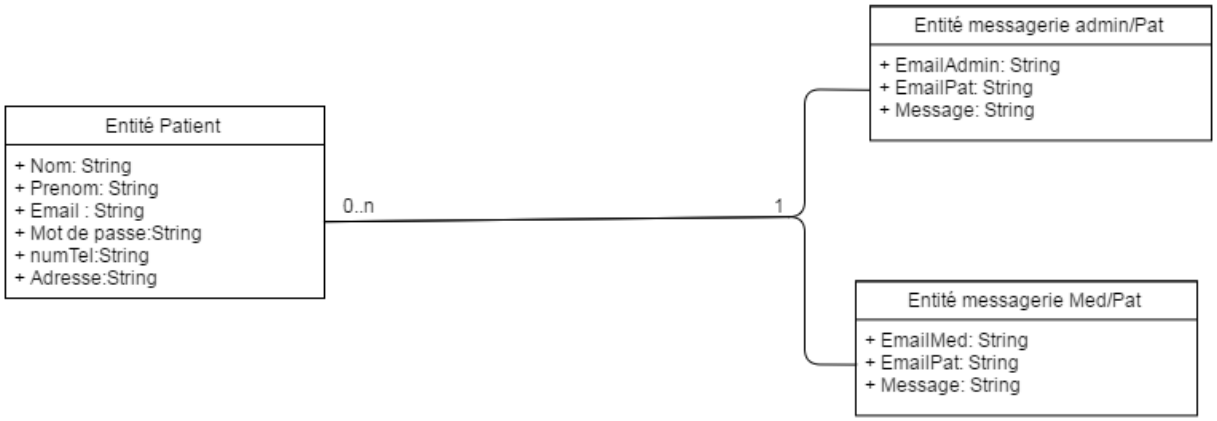
3.7.3. Diagramme de navigation



3.7.4. Diagramme d'interaction

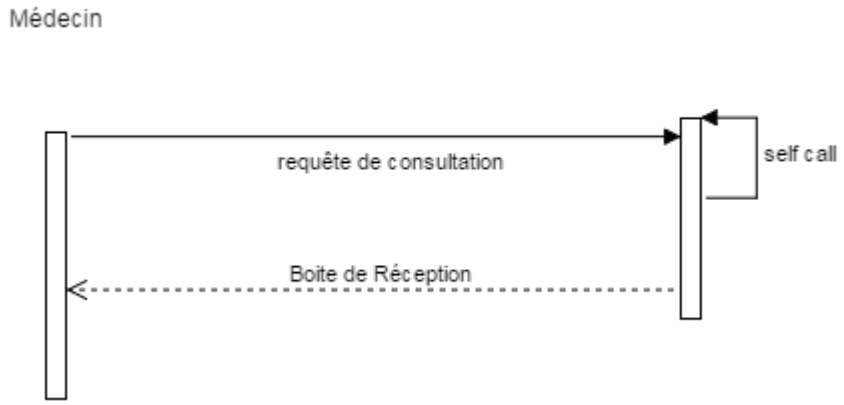


3.7.5. Diagramme de classes de conception préliminaires

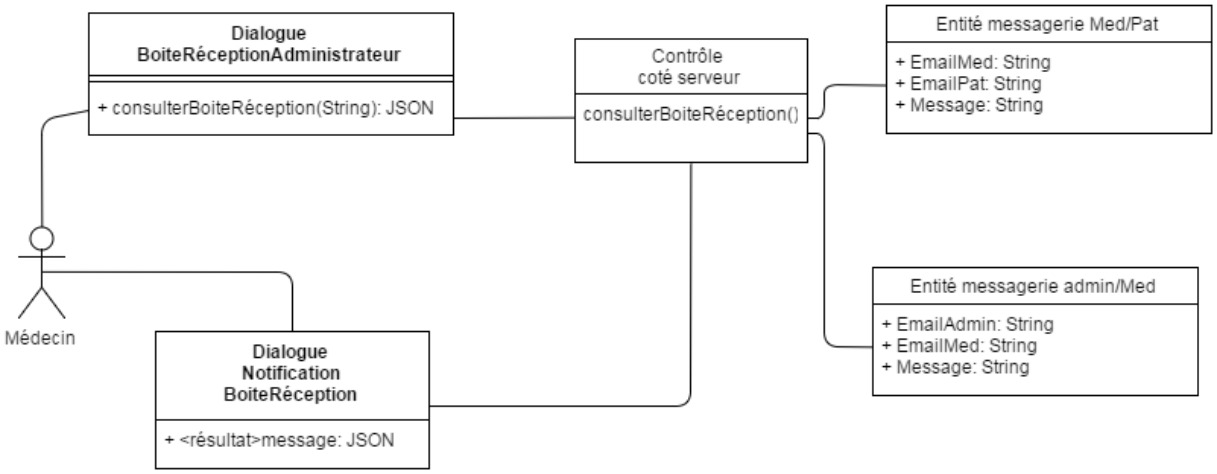


3.8. Consulter la boite de réception médecin

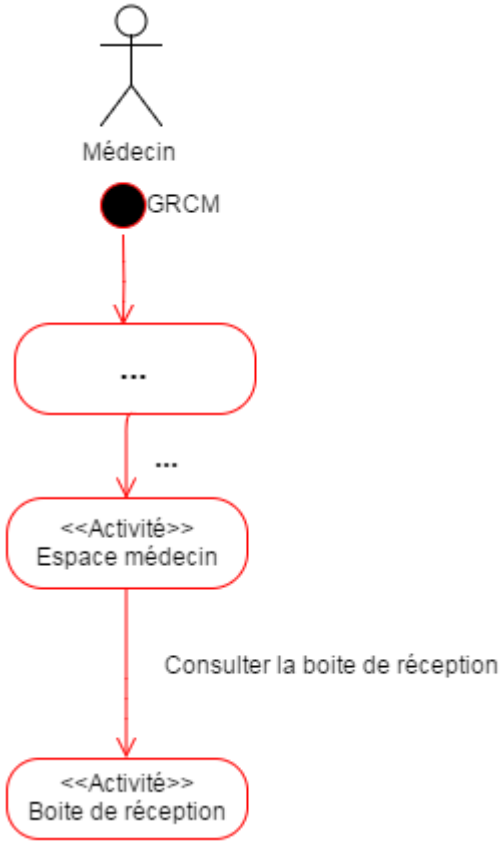
3.8.1. Diagramme de séquence système



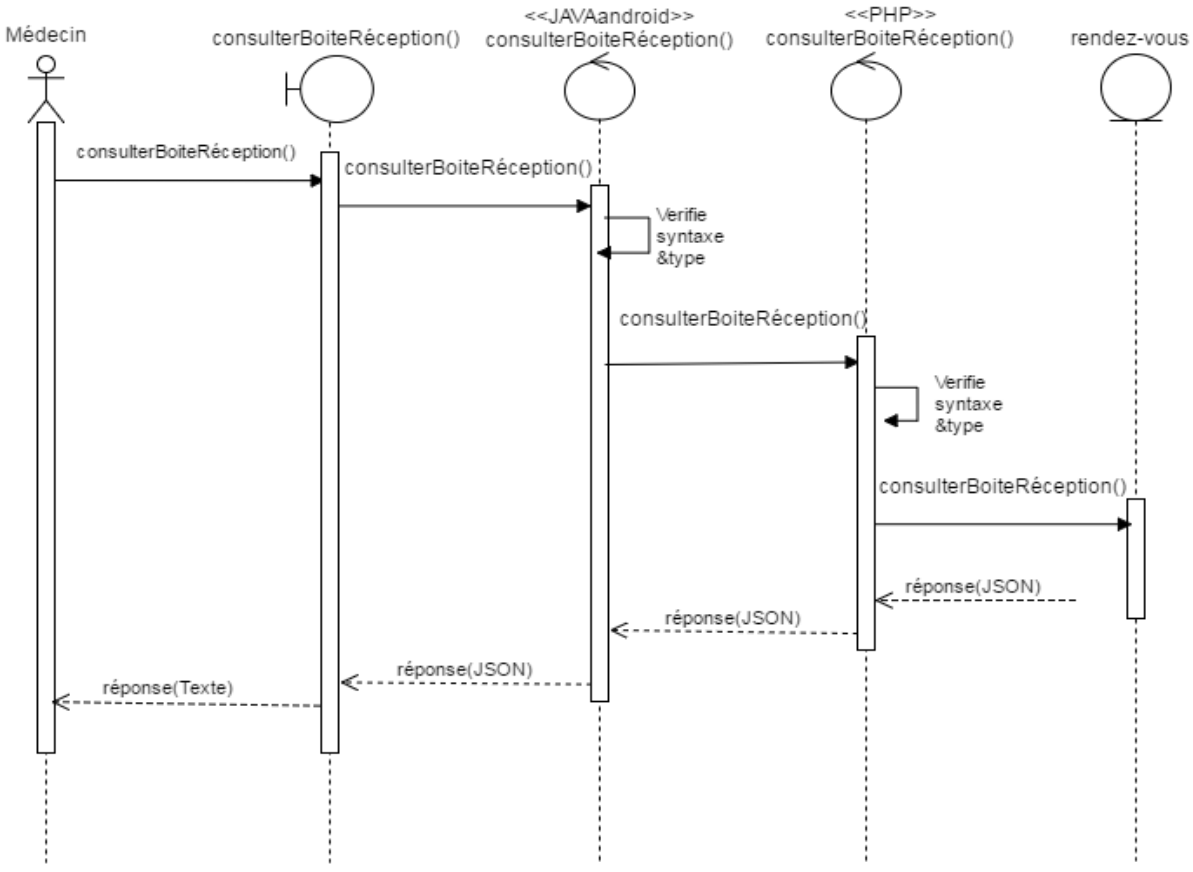
3.8.2. Diagramme de classes participantes



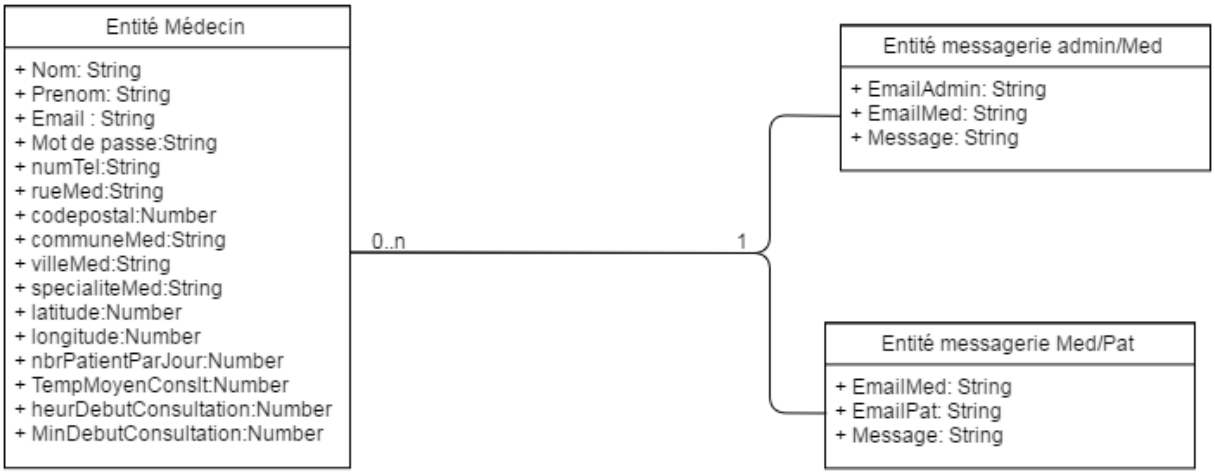
3.8.3. Diagramme de navigation



3.8.4. Diagramme d'interaction

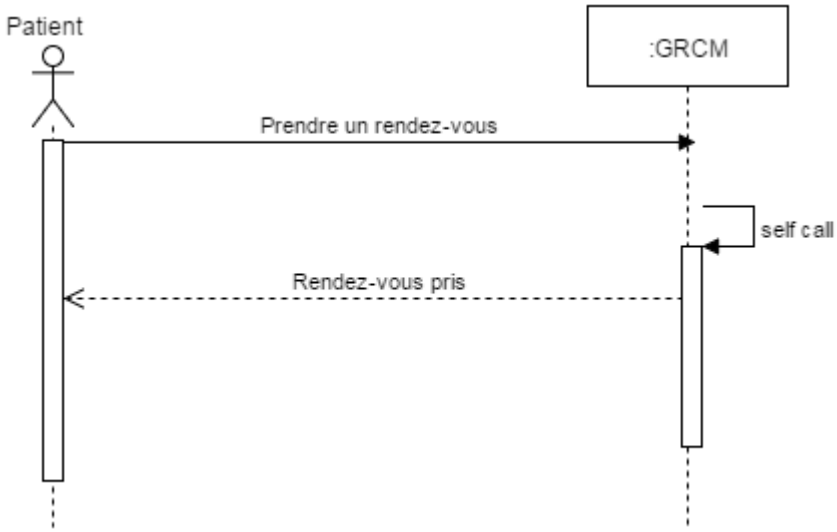


3.8.5. Diagramme de classes de conception préliminaires

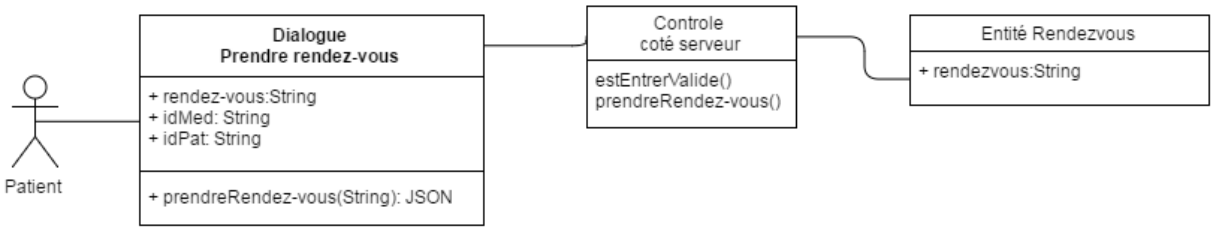


3.9. prendre un rendez-vous

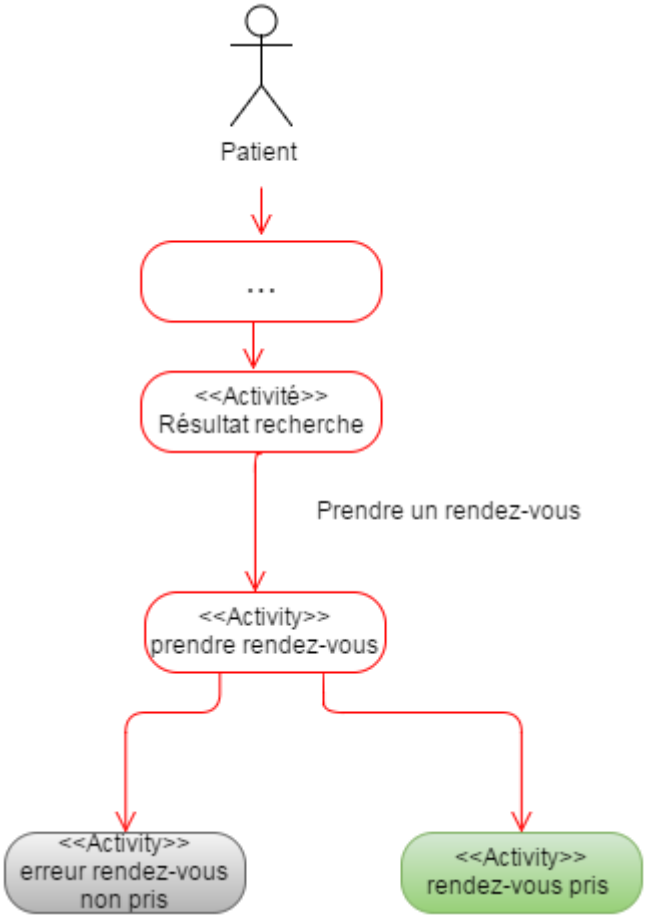
3.9.1. Diagramme de séquence système



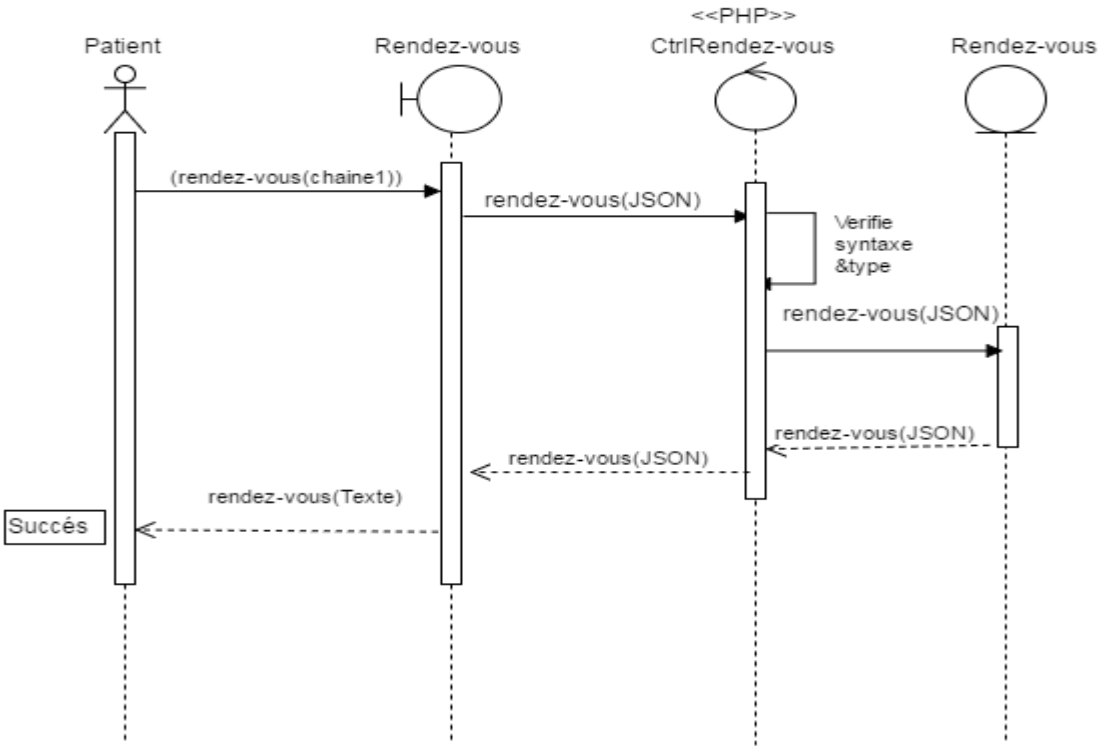
3.9.2. Diagramme de classes participantes



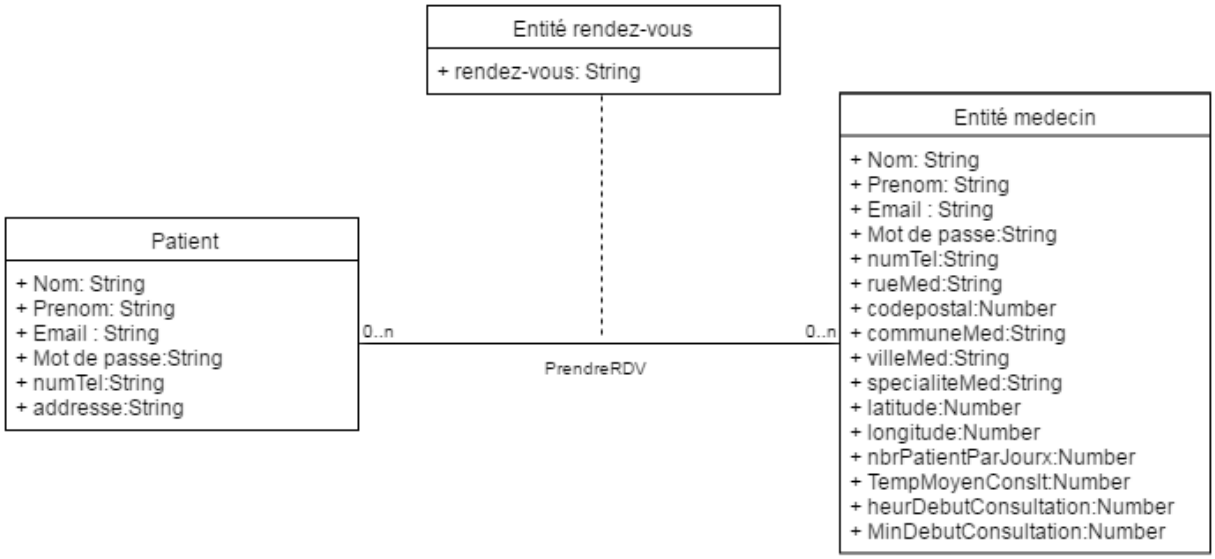
3.9.3. Diagramme de navigation



3.9.4. Diagramme d'interaction

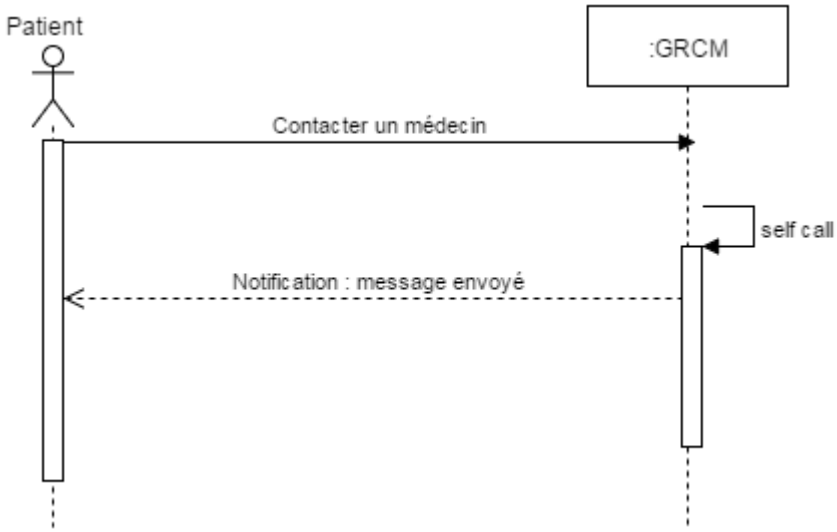


3.9.5. Diagramme de classes de conception préliminaires

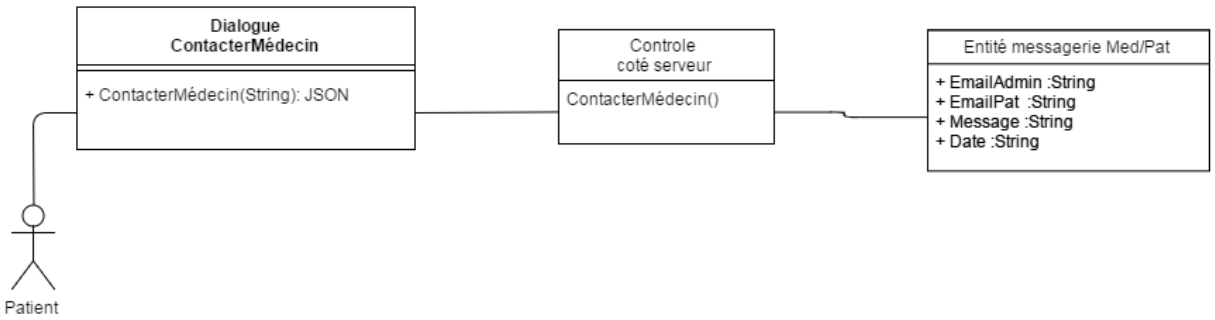


3.10. contacter un médecin

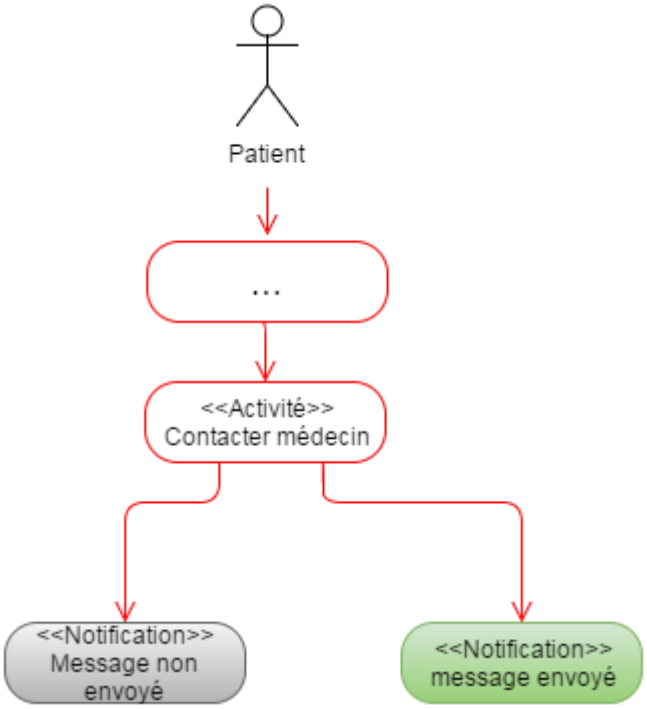
3.10.1. Diagramme de séquence système



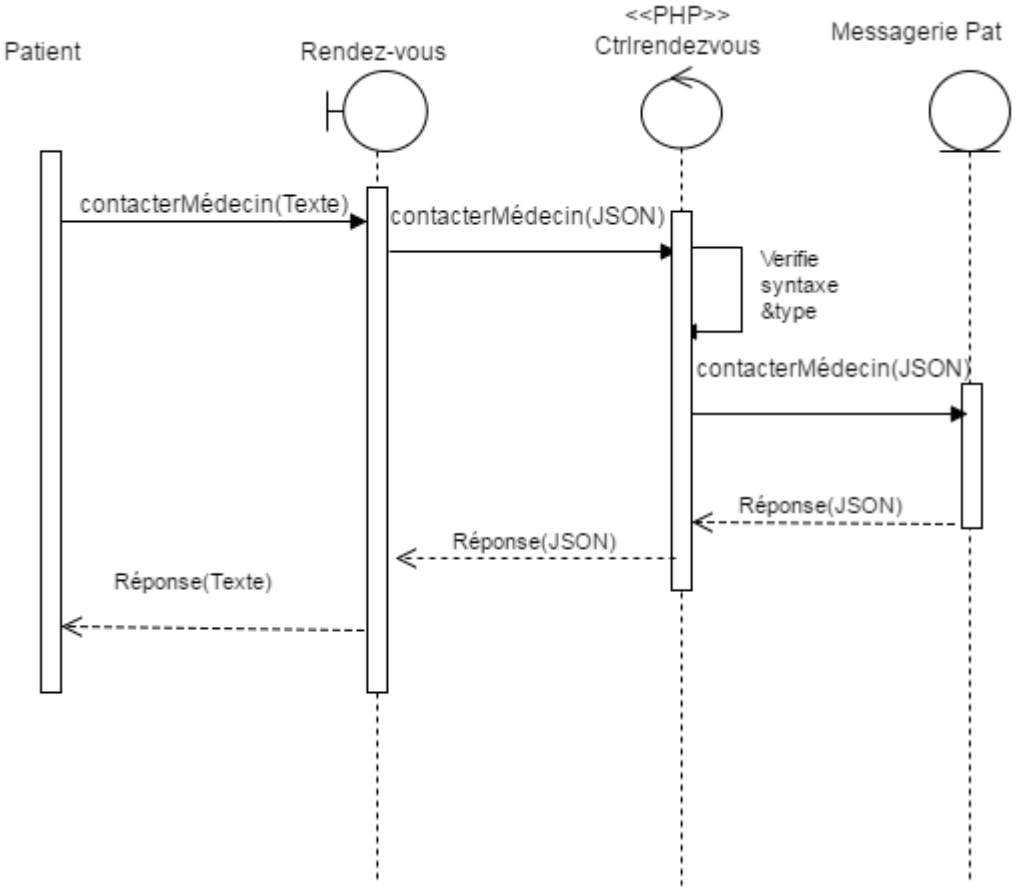
3.10.2. Diagramme de classes participantes



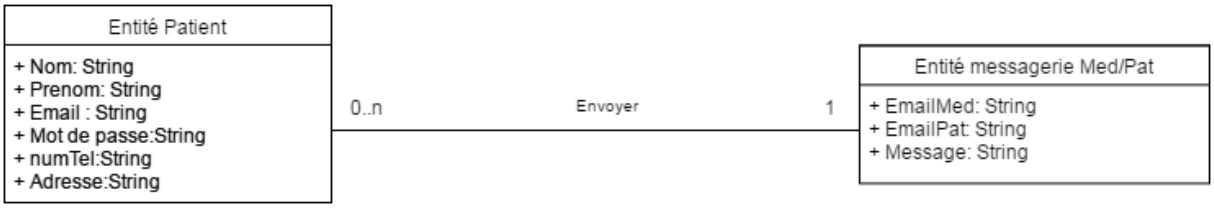
3.10.3. Diagramme de navigation



3.10.4. Diagramme d'interaction

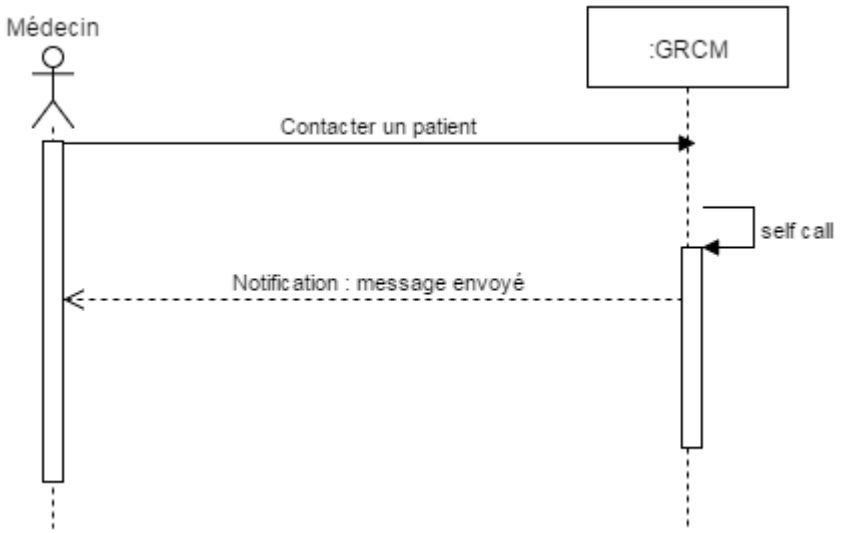


3.10.5. Diagramme de classes de conception préliminaires

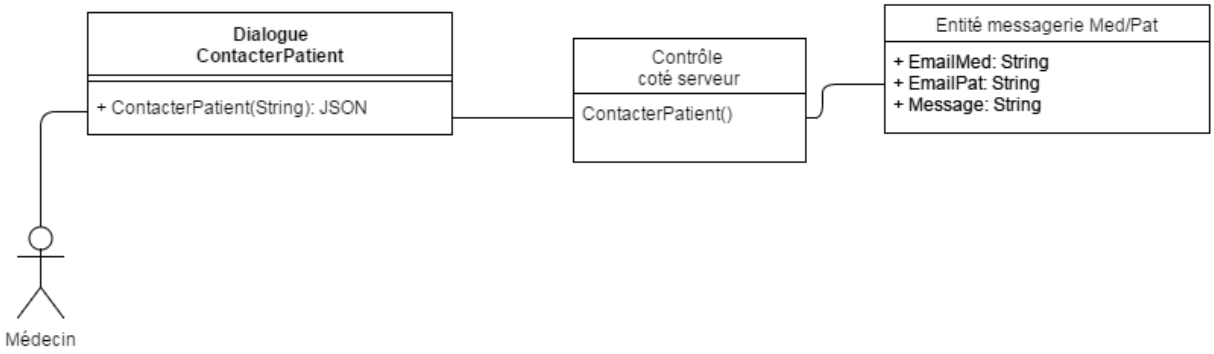


3.11. contacter un patient

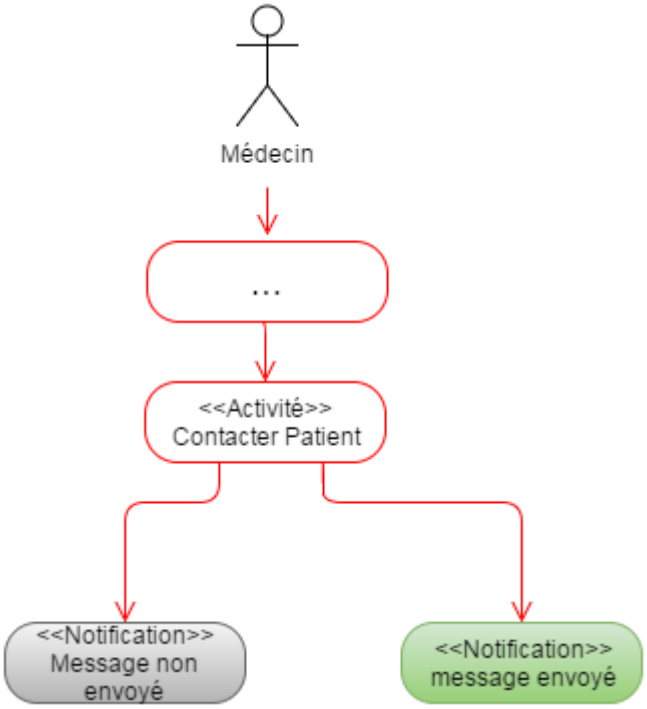
3.11.1. Diagramme de séquence système



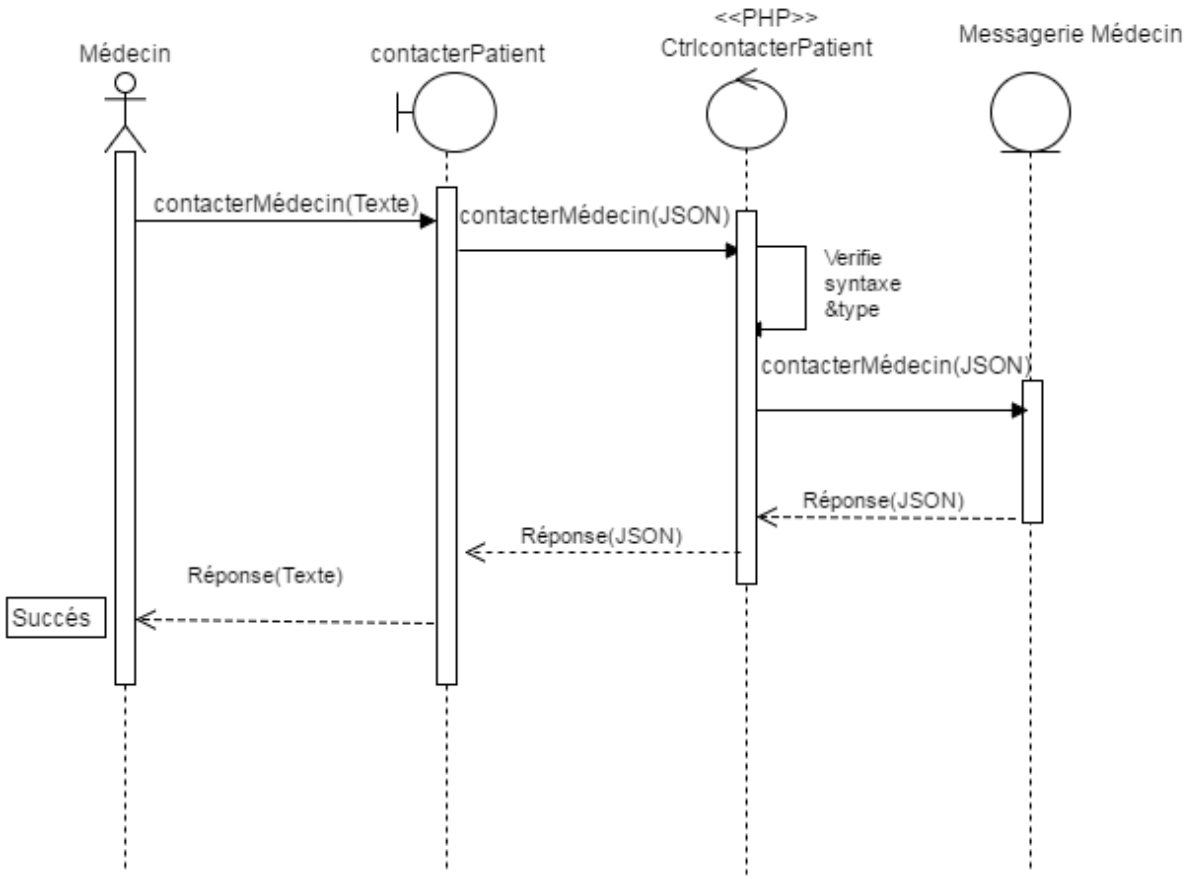
3.11.2. Diagramme de classes participantes



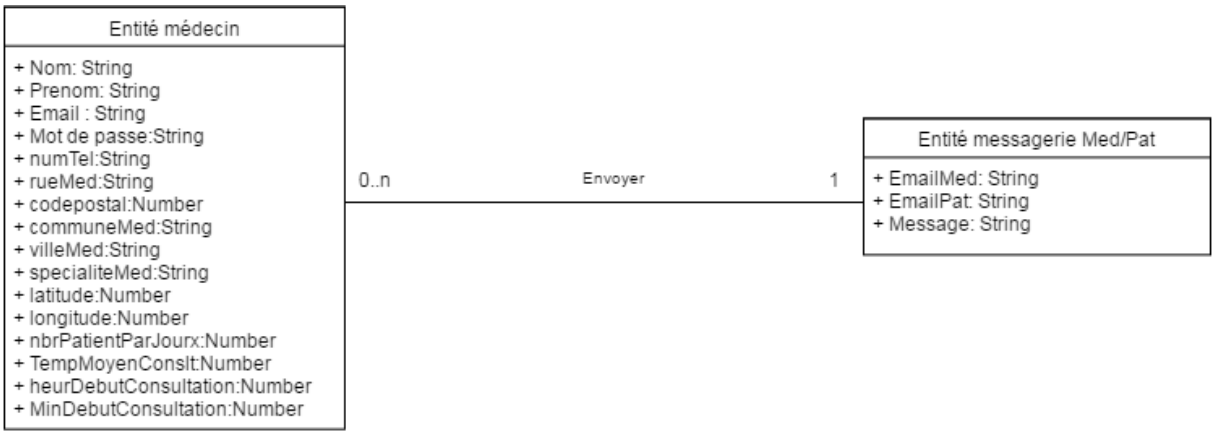
3.11.3. Diagramme de navigation



3.11.4. Diagramme d'interaction

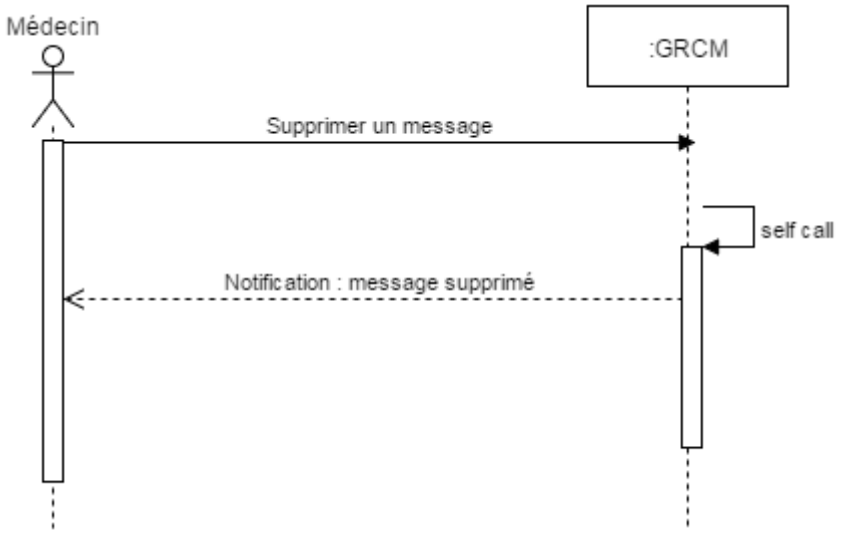


3.11.5. Diagramme de classes de conception préliminaires

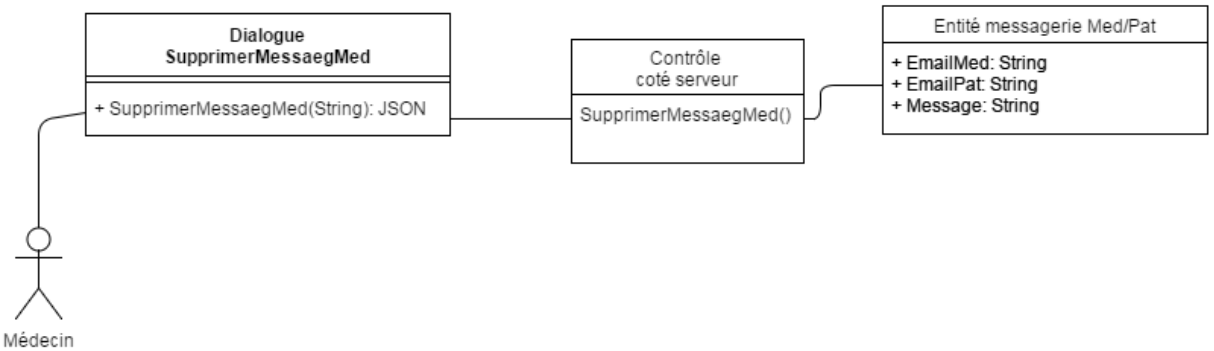


3.12. Supprimer un message médecin

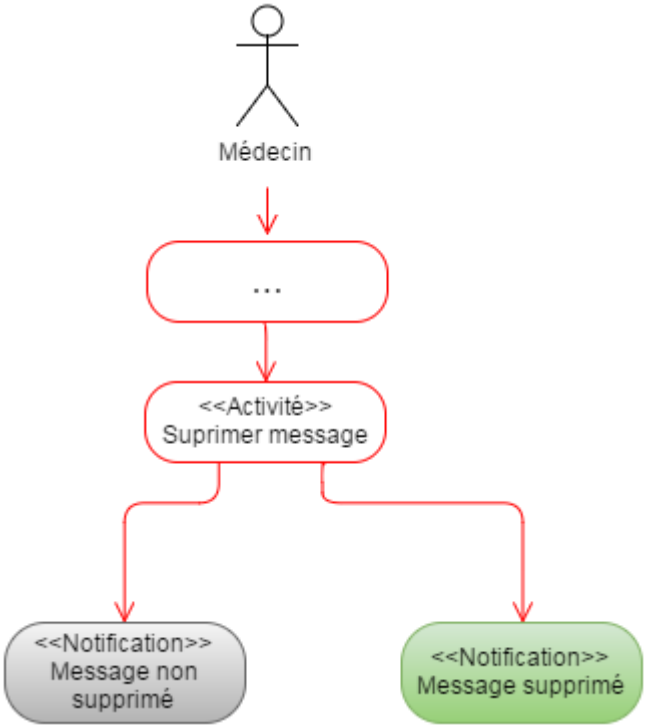
3.12.1. Diagramme de séquence système



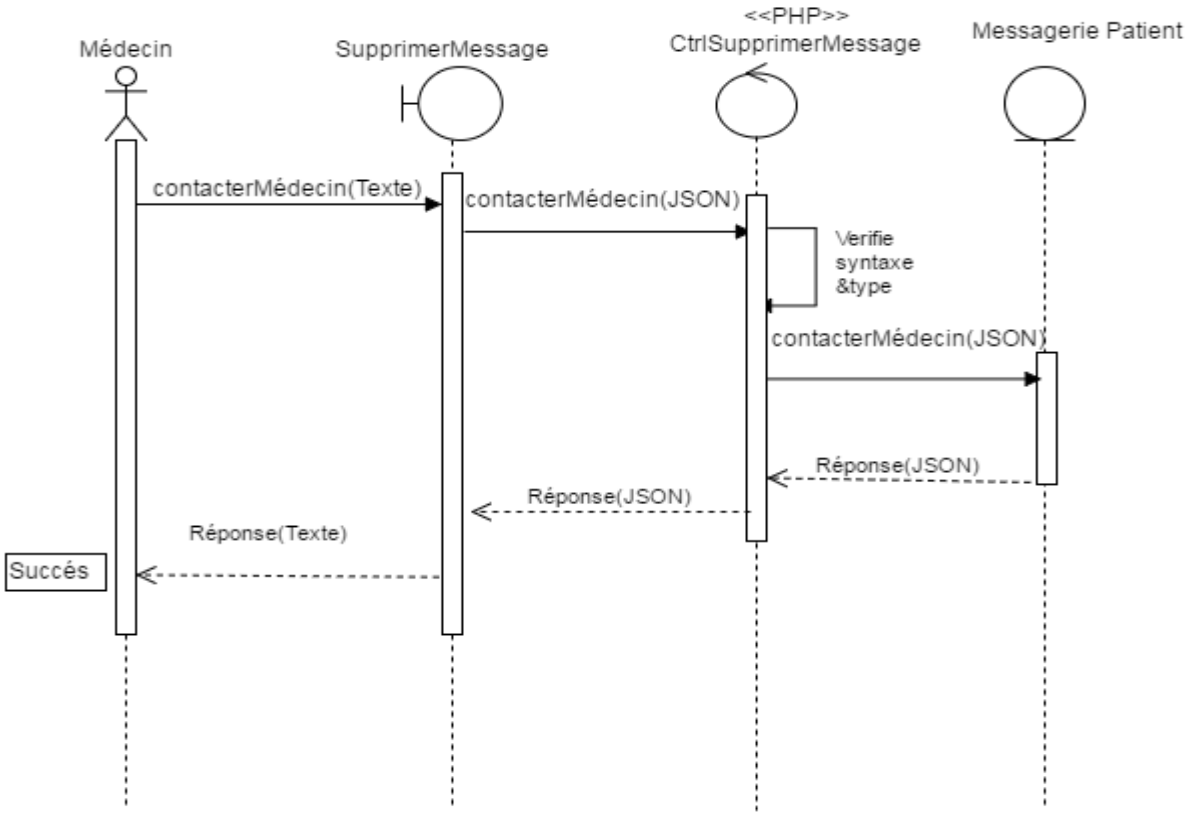
3.12.2. Diagramme de classes participantes



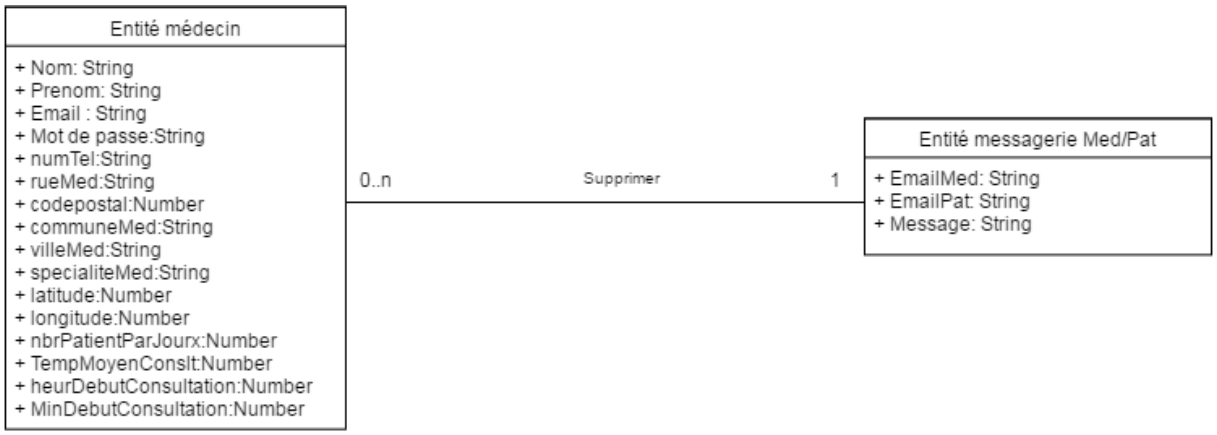
3.12.3. Diagramme de navigation



3.12.4. Diagramme d'interaction

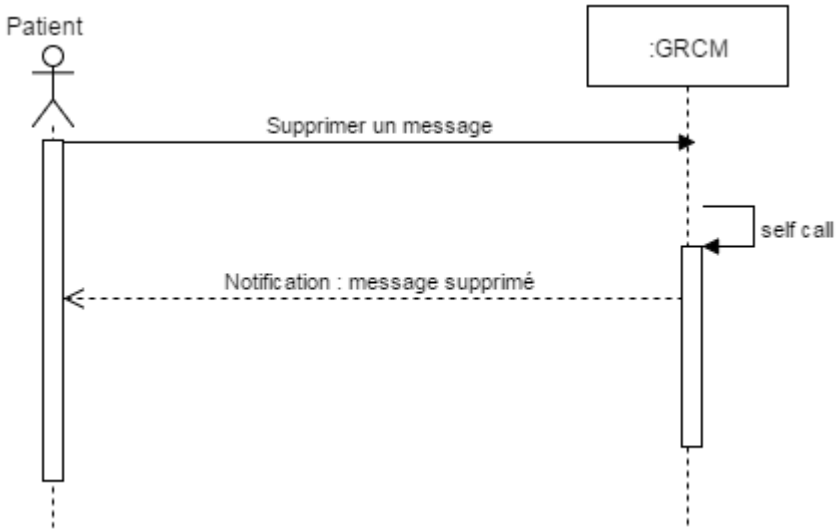


3.12.5. Diagramme de classes de conception préliminaires

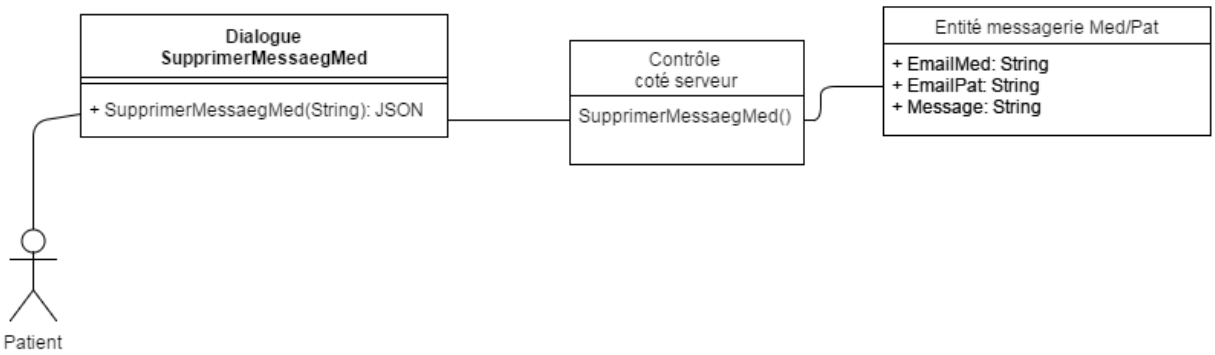


3.13. Supprimer un message patient

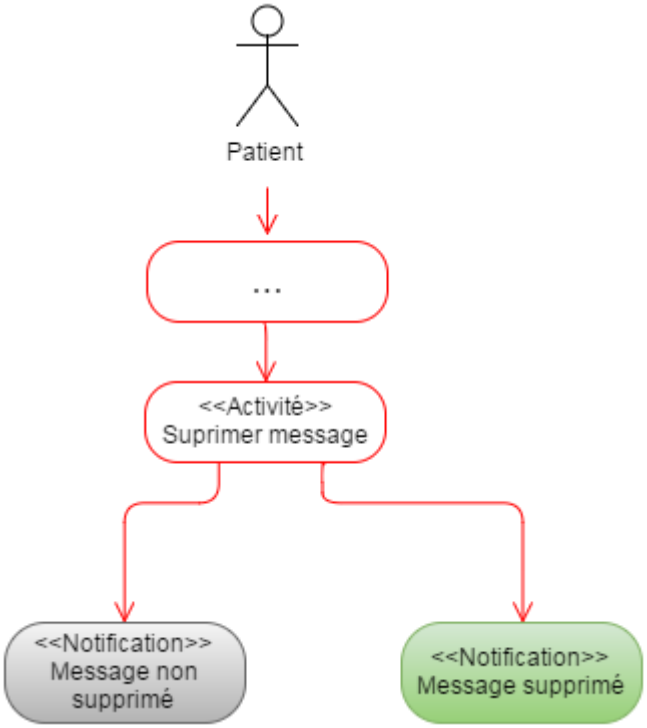
3.13.1. Diagramme de séquence système



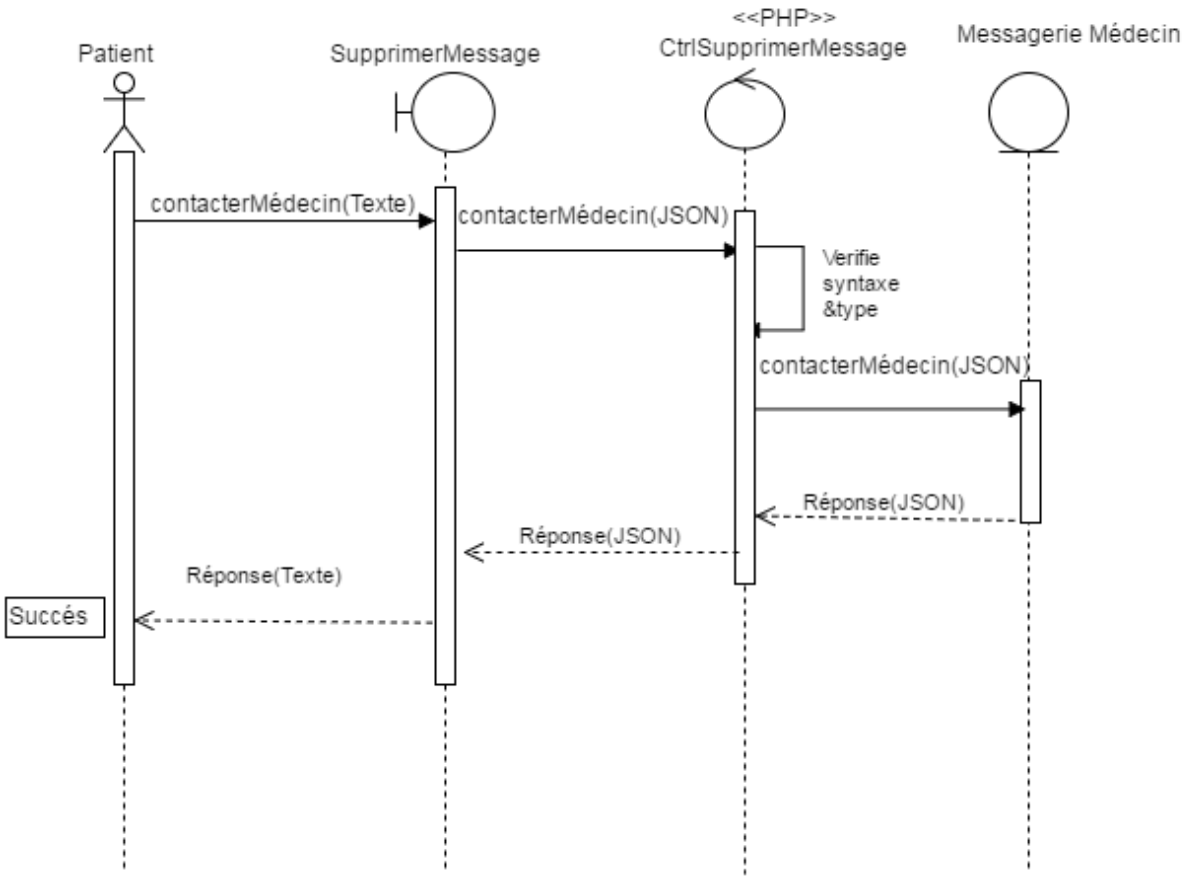
3.13.2. Diagramme de classes participantes



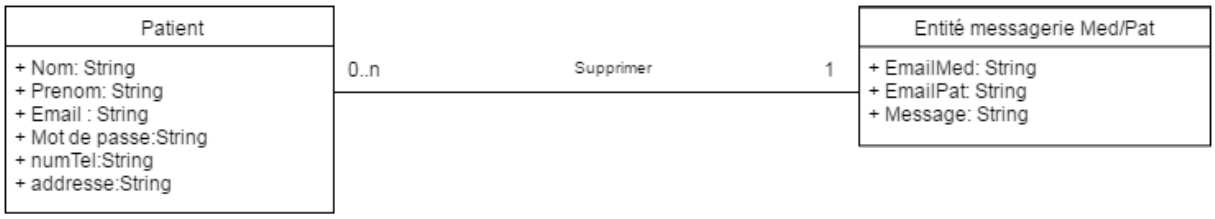
3.13.3. Diagramme de navigation



3.13.4. Diagramme d'interaction

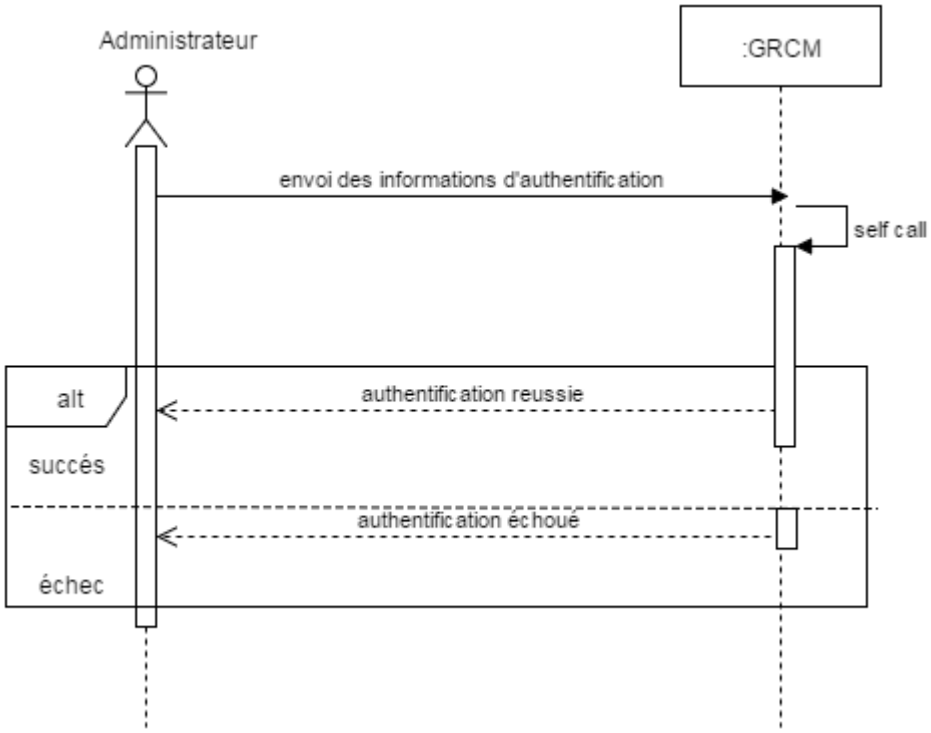


3.13.5. Diagramme de classes de conception préliminaires

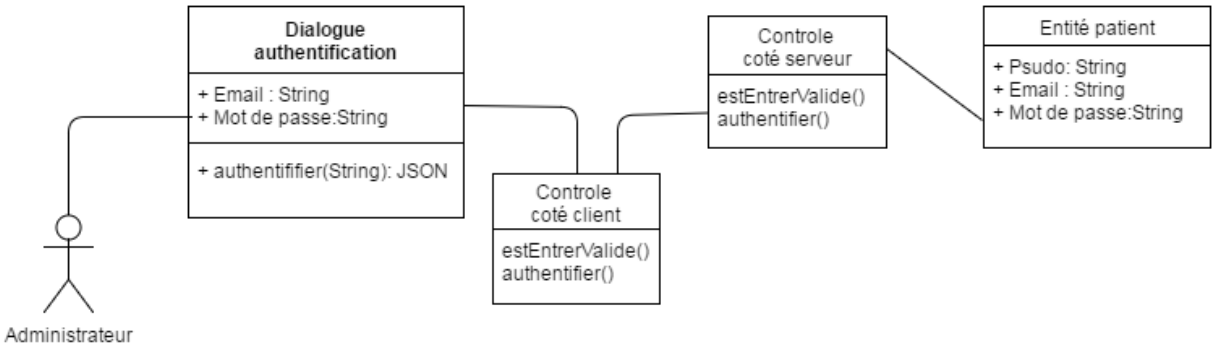


3.14. S'authentifier administrateur

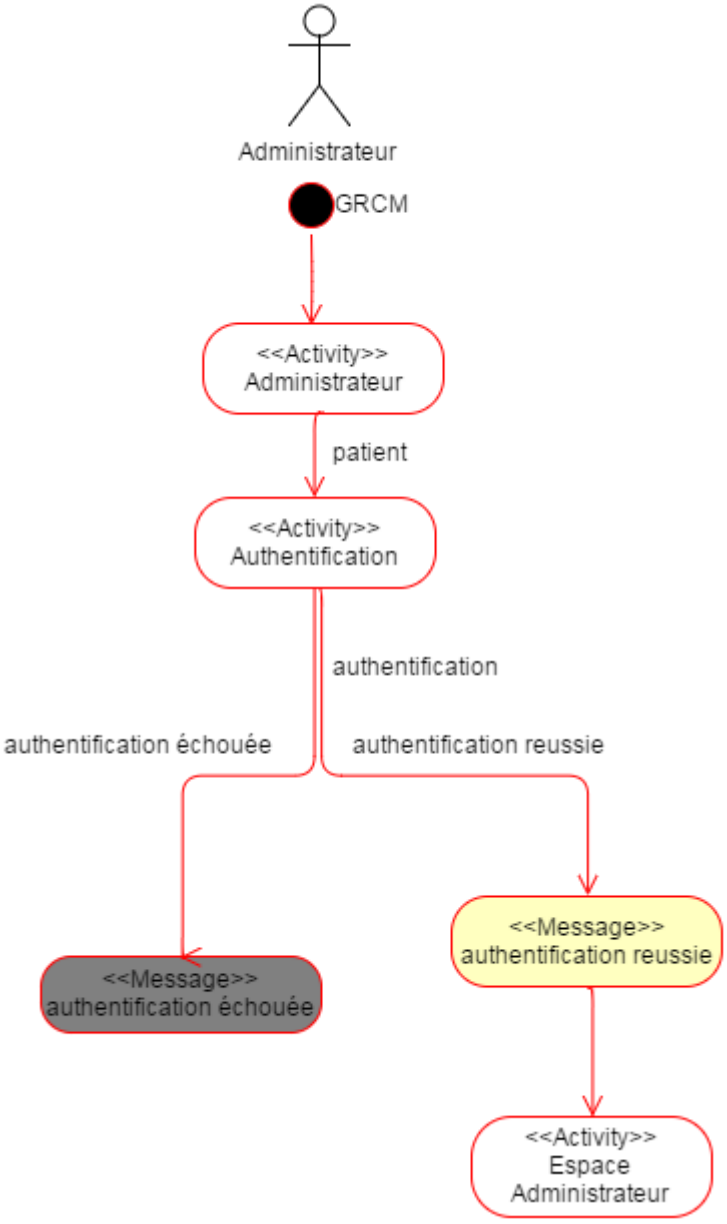
3.14.1. Diagramme de séquence système



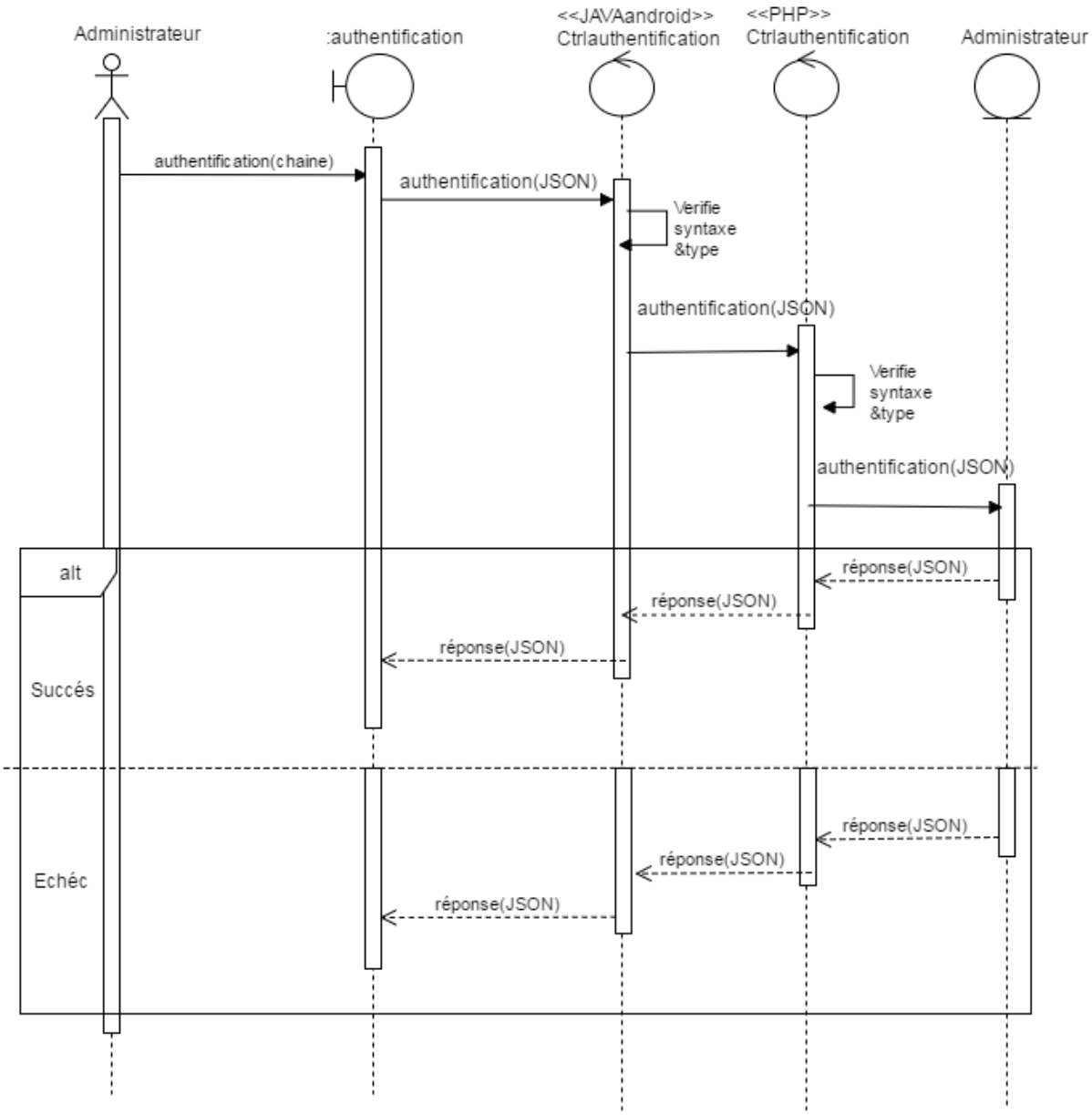
3.14.2. Diagramme de classes participantes



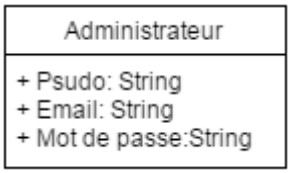
3.14.3. Diagramme de navigation



3.14.4. Diagramme d'interaction

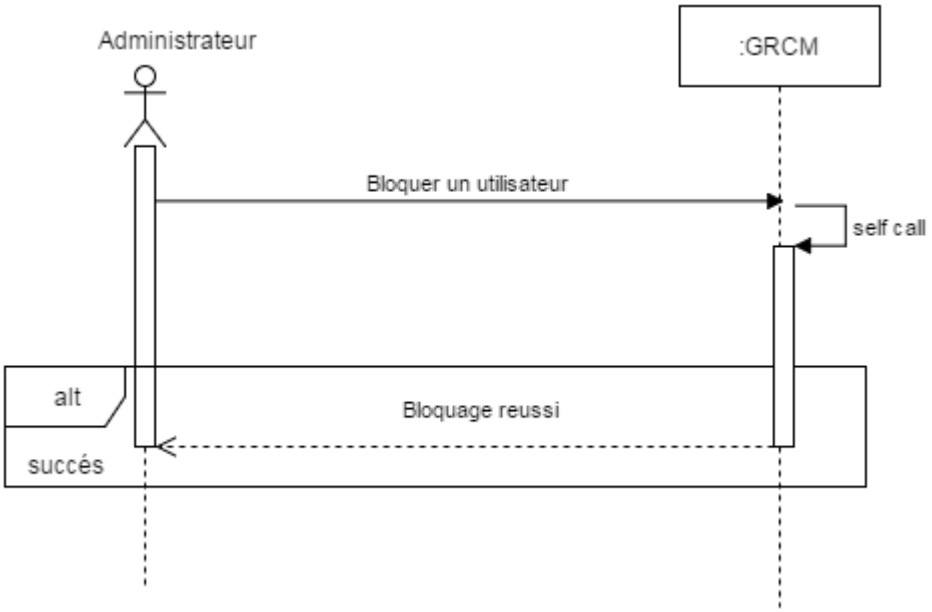


3.14.5. Diagramme de classes de conception préliminaires

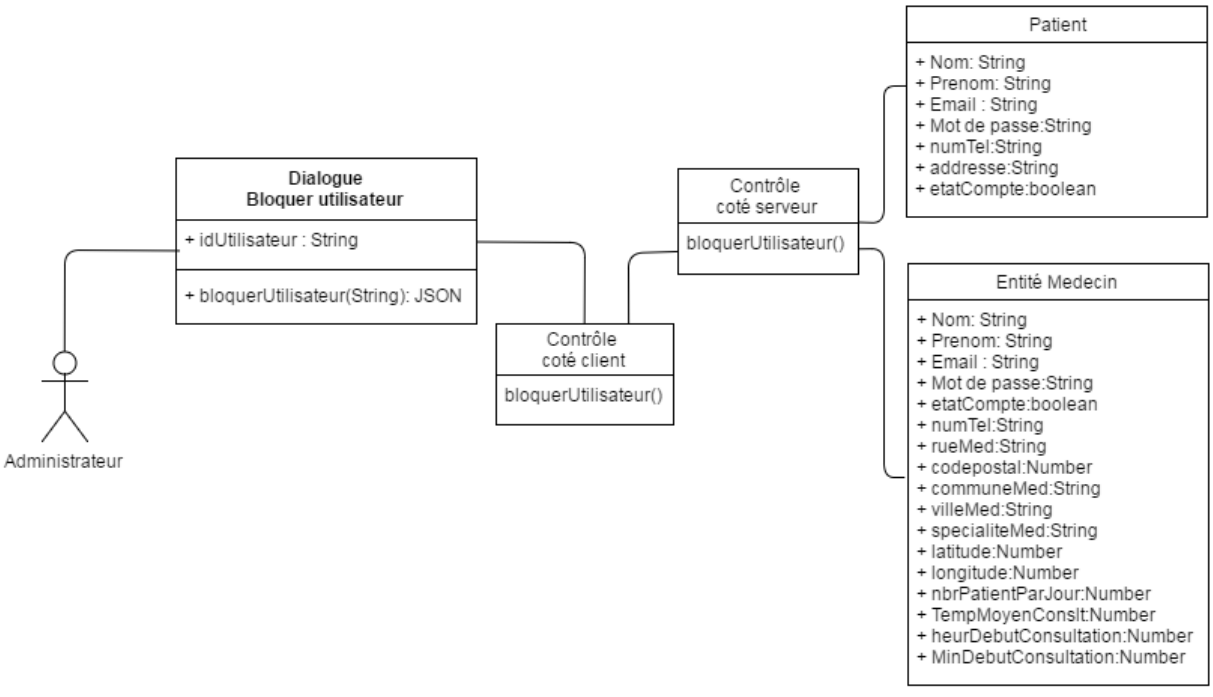


3.15. Bloquer un utilisateur

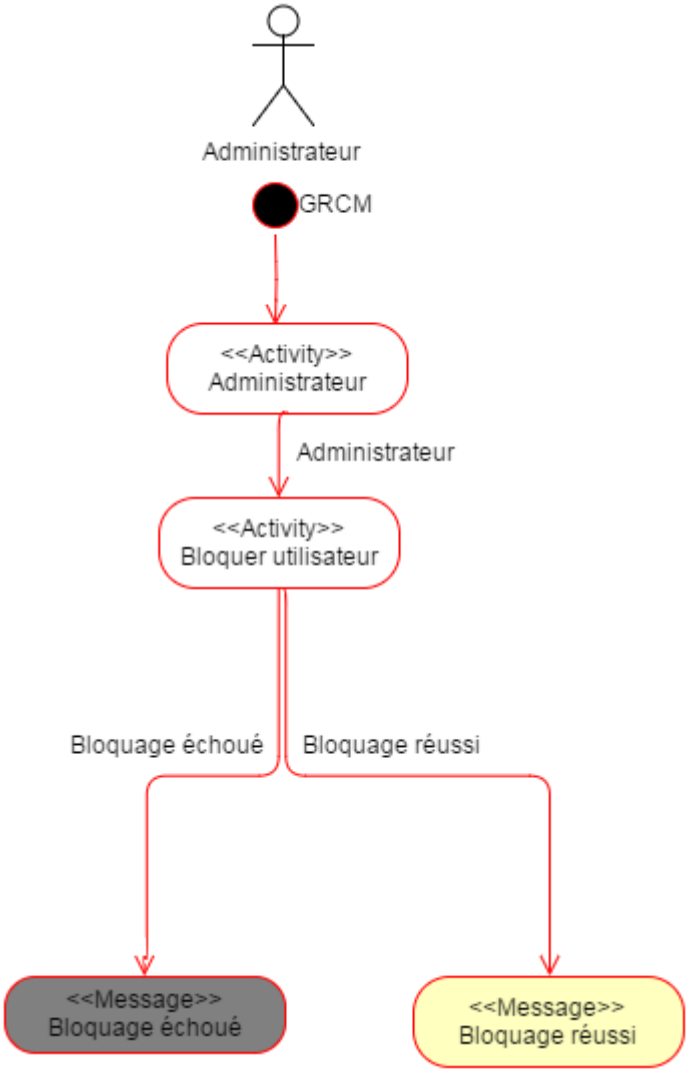
3.15.1. Diagramme de séquence système



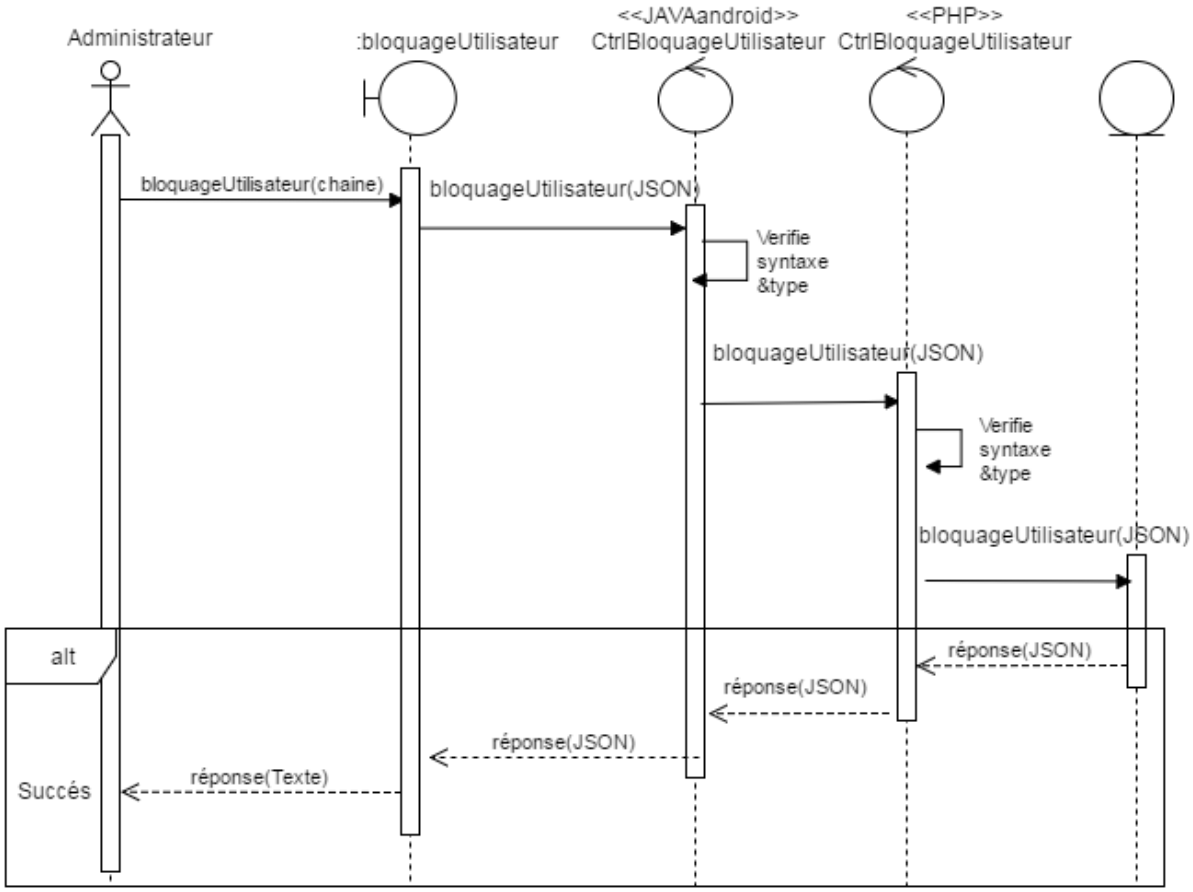
3.15.2. Diagramme de classes participantes



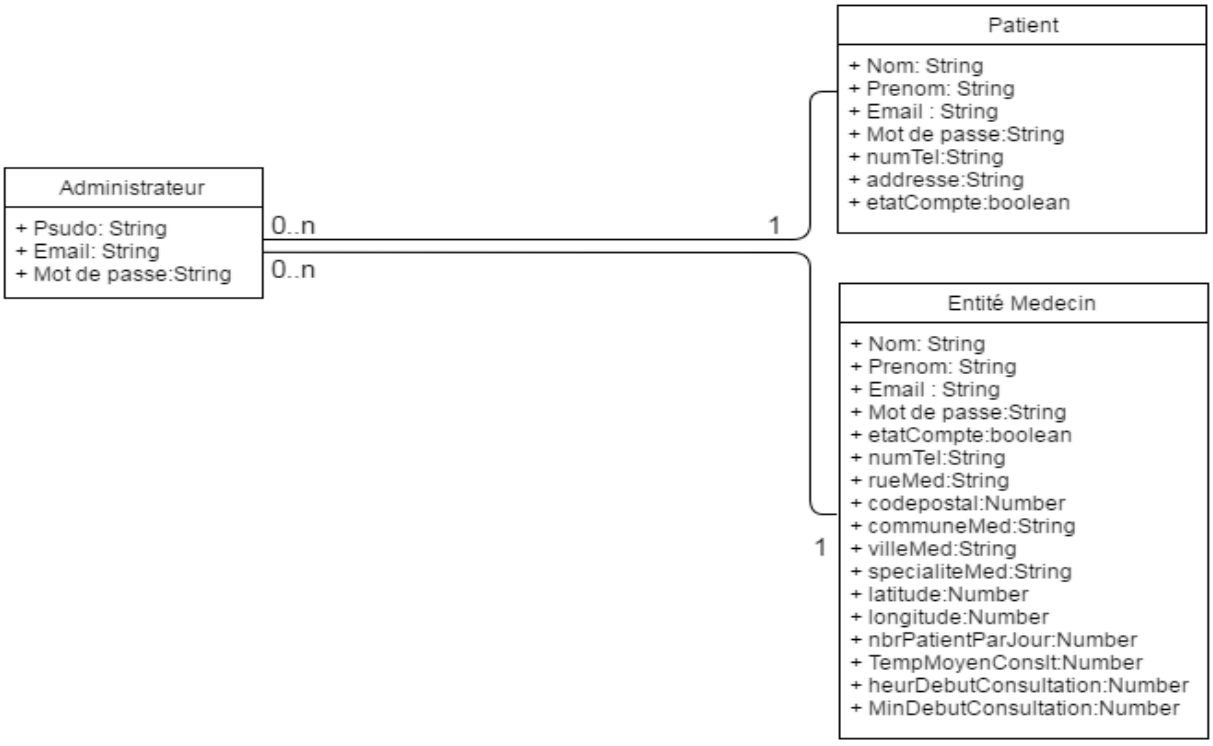
3.15.3. Diagramme de navigation



3.15.4. Diagramme d'interaction

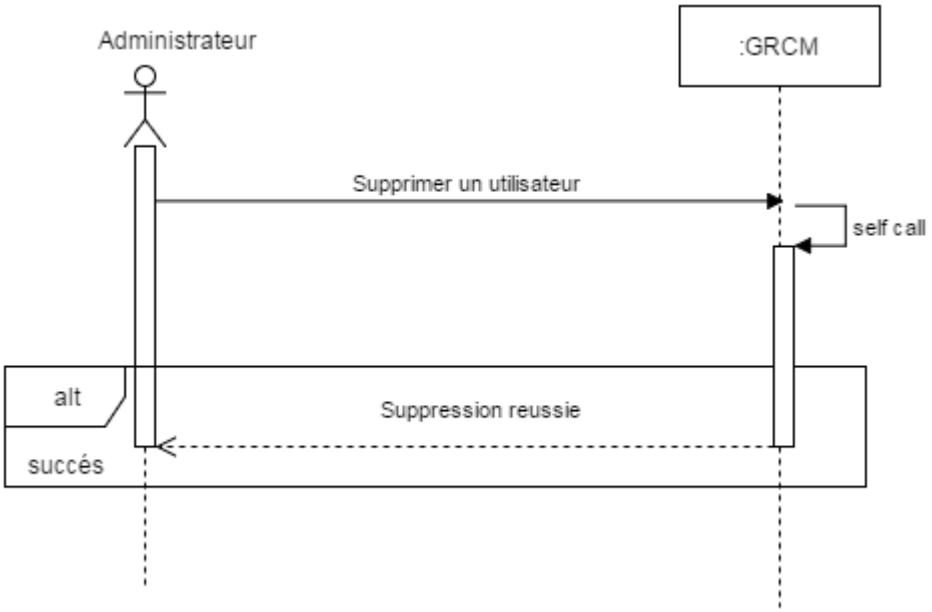


3.15.5. Diagramme de classes de conception préliminaires

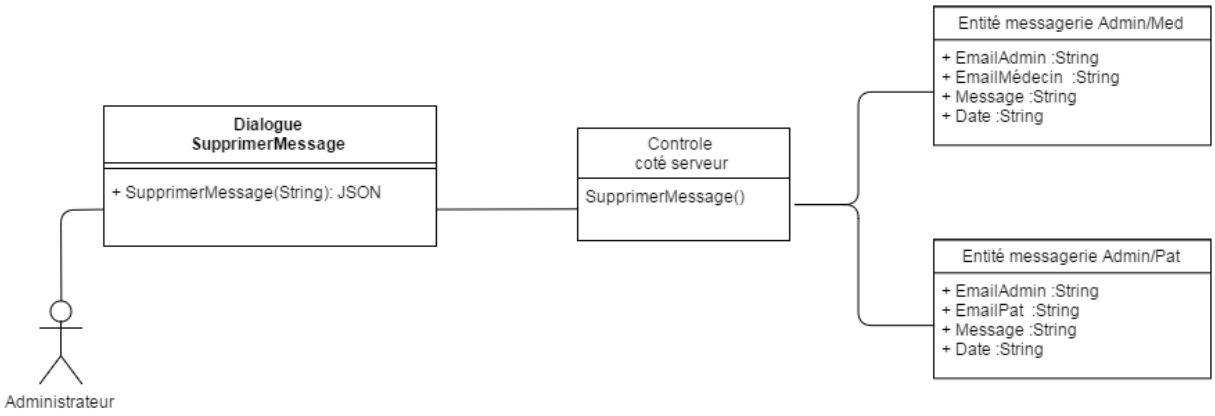


3.16. Supprimer utilisateur administrateur

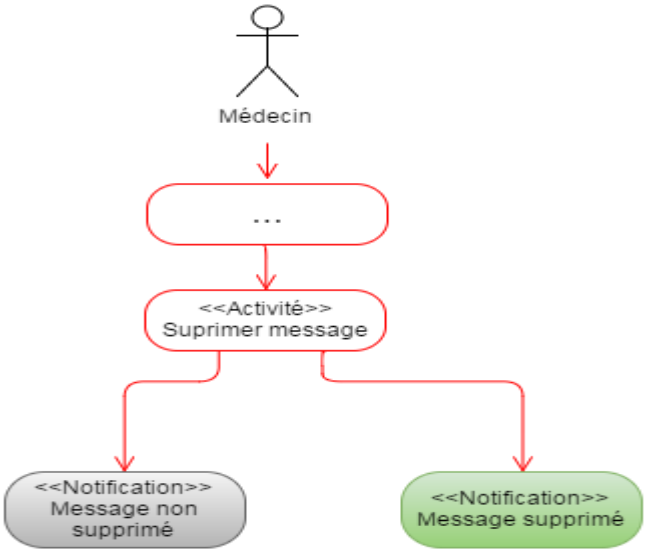
3.16.1. Diagramme de séquence système



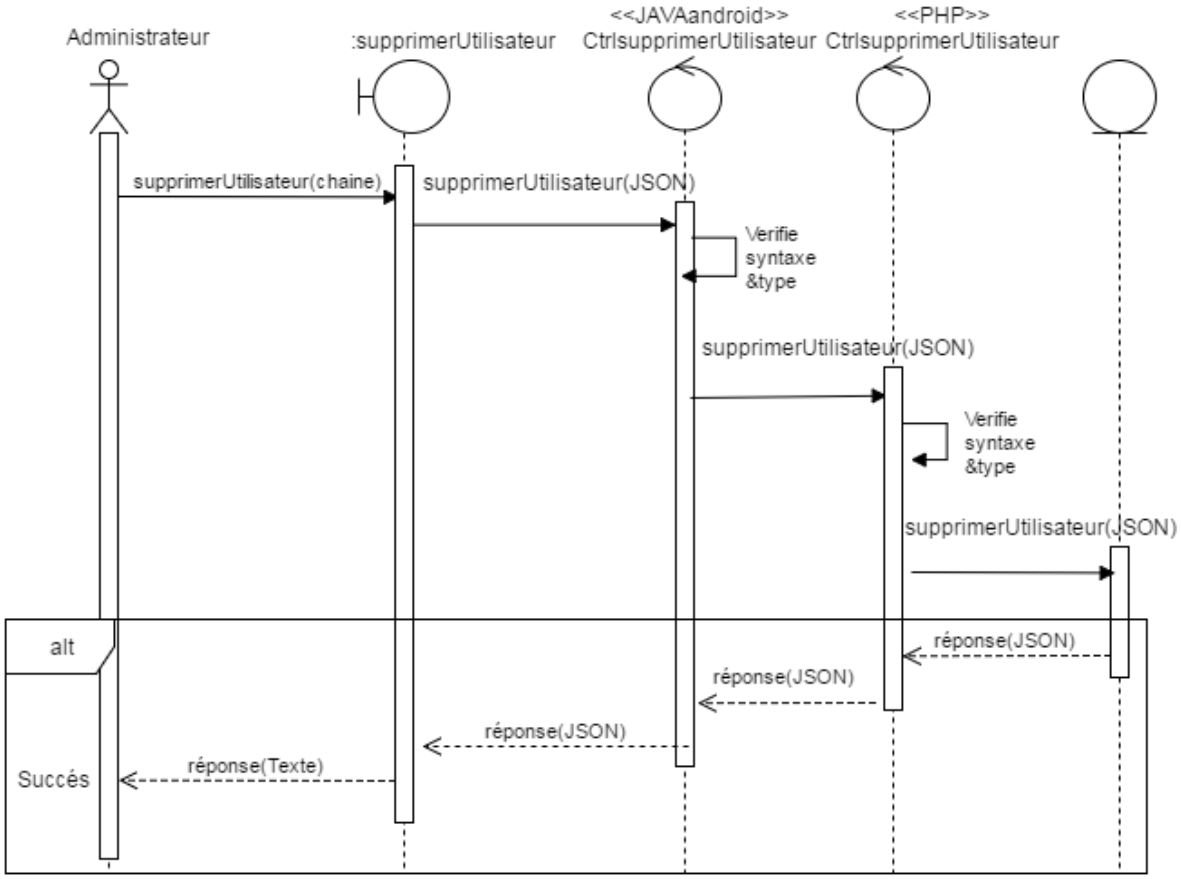
3.16.2. Diagramme de classes participantes



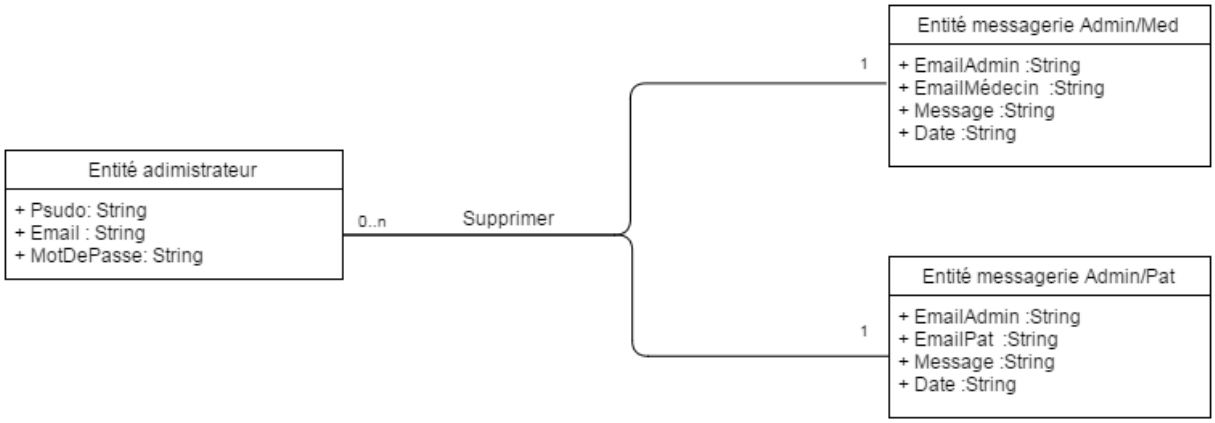
3.16.3. Diagramme de navigation



3.16.4. Diagramme d'interaction

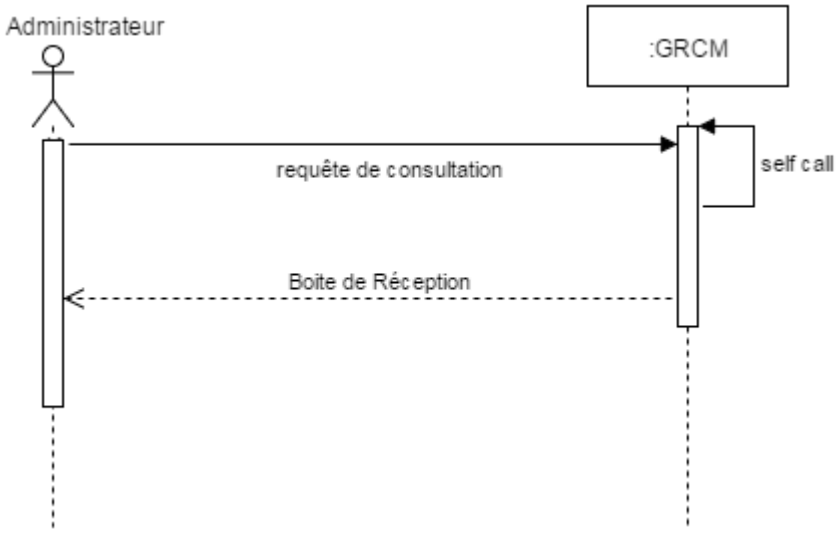


3.16.5. Diagramme de classes de conception préliminaires

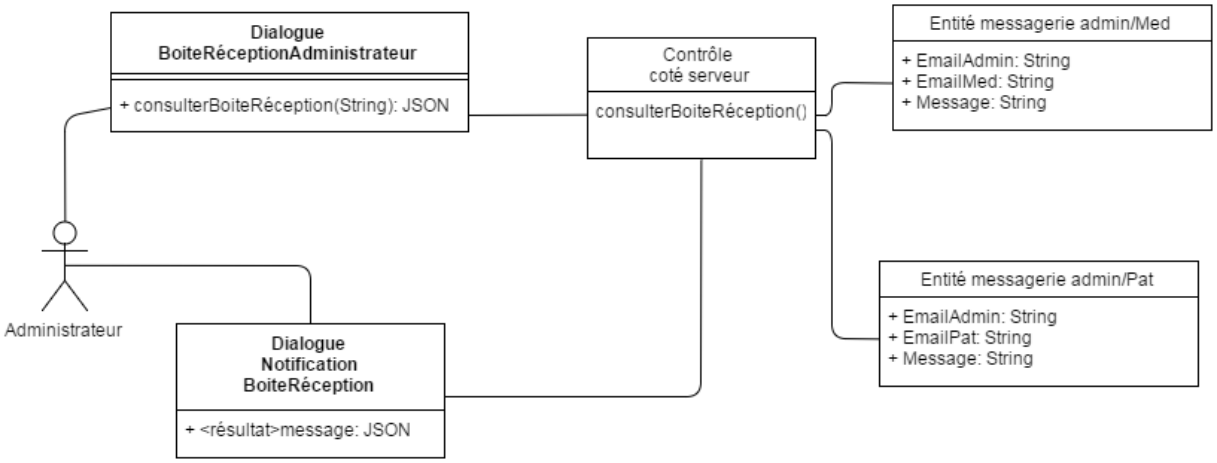


3.17. Consulter la boite de réception administrateur

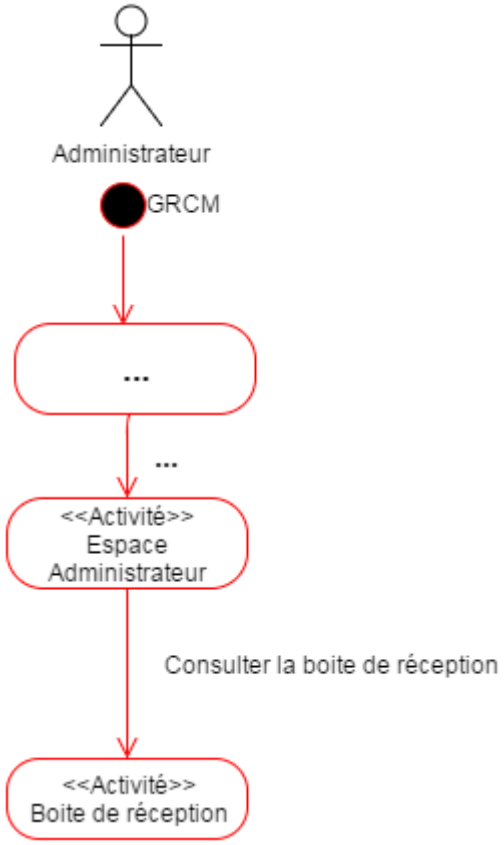
3.17.1. Diagramme de séquence système



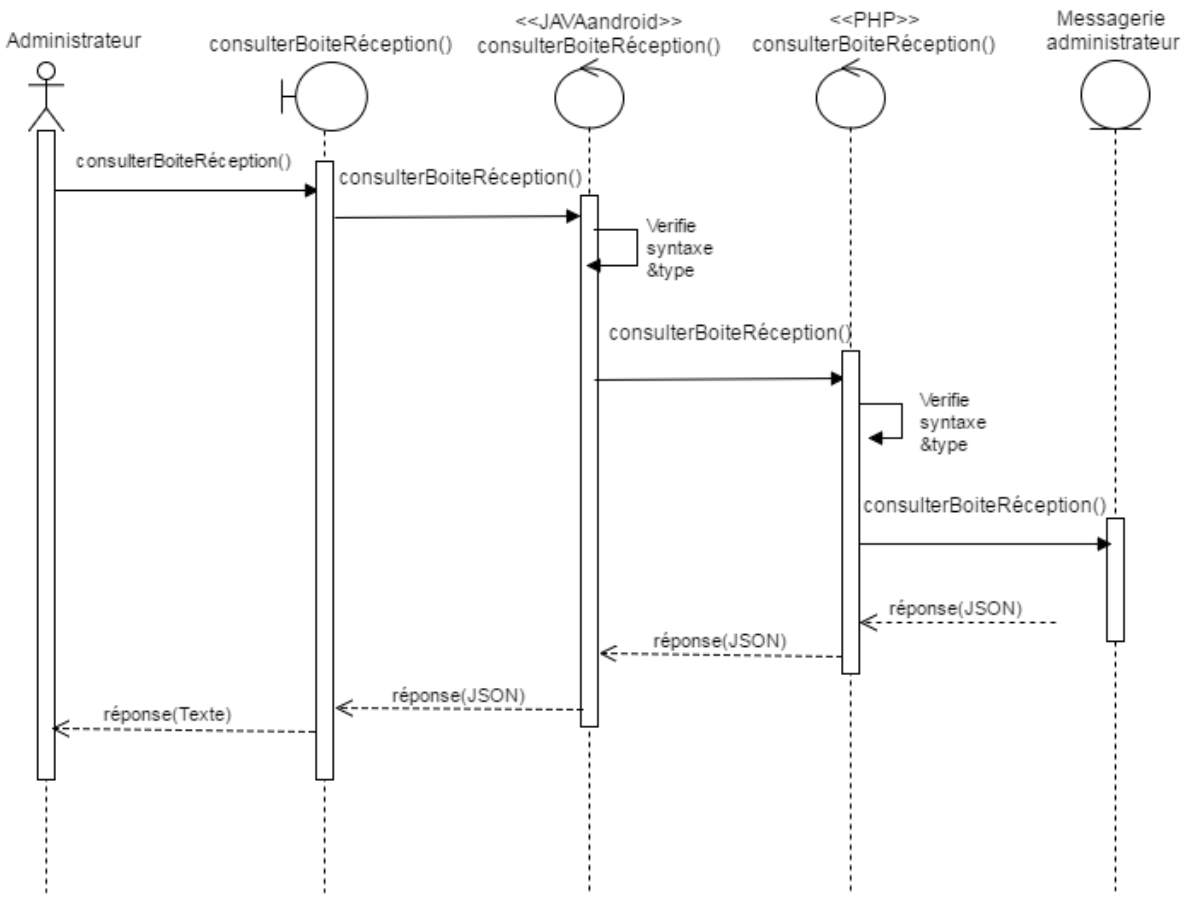
3.17.2. Diagramme de classes participantes



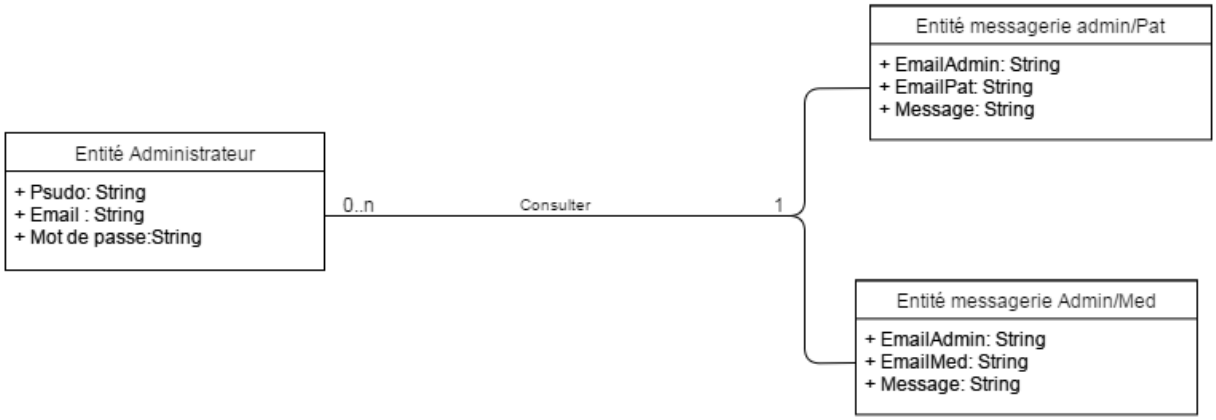
3.17.3. Diagramme de navigation



3.17.4. Diagramme d'interaction

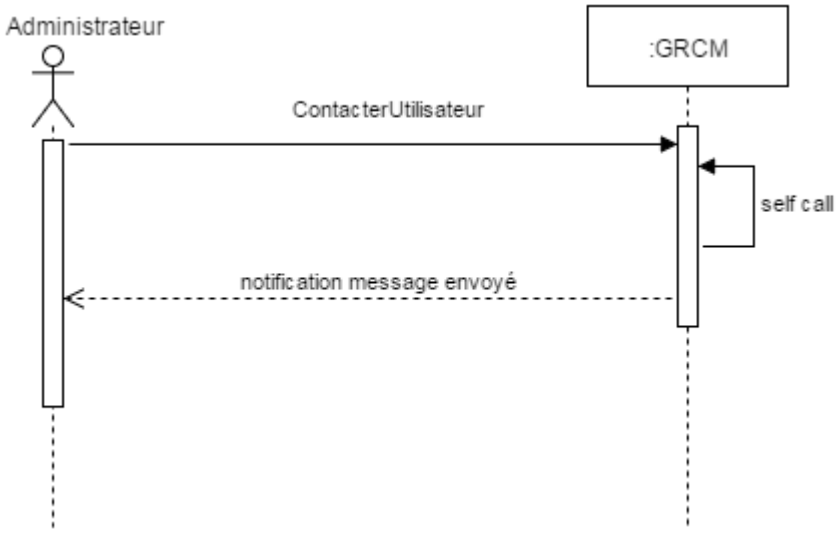


3.17.5. Diagramme de classes de conception préliminaires

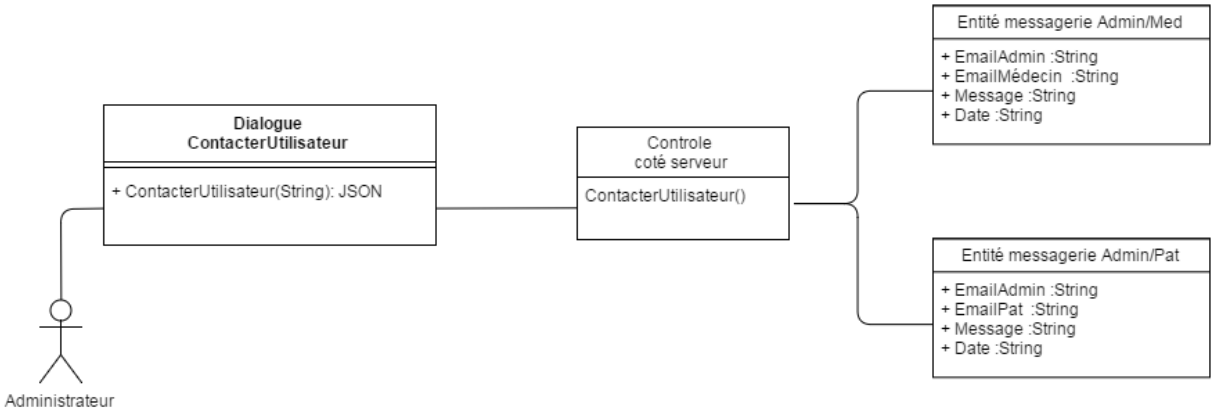


3.18. contacter les utilisateurs

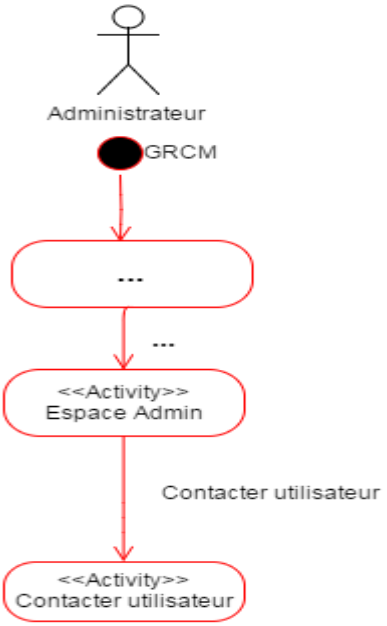
3.18.1. Diagramme de séquence système



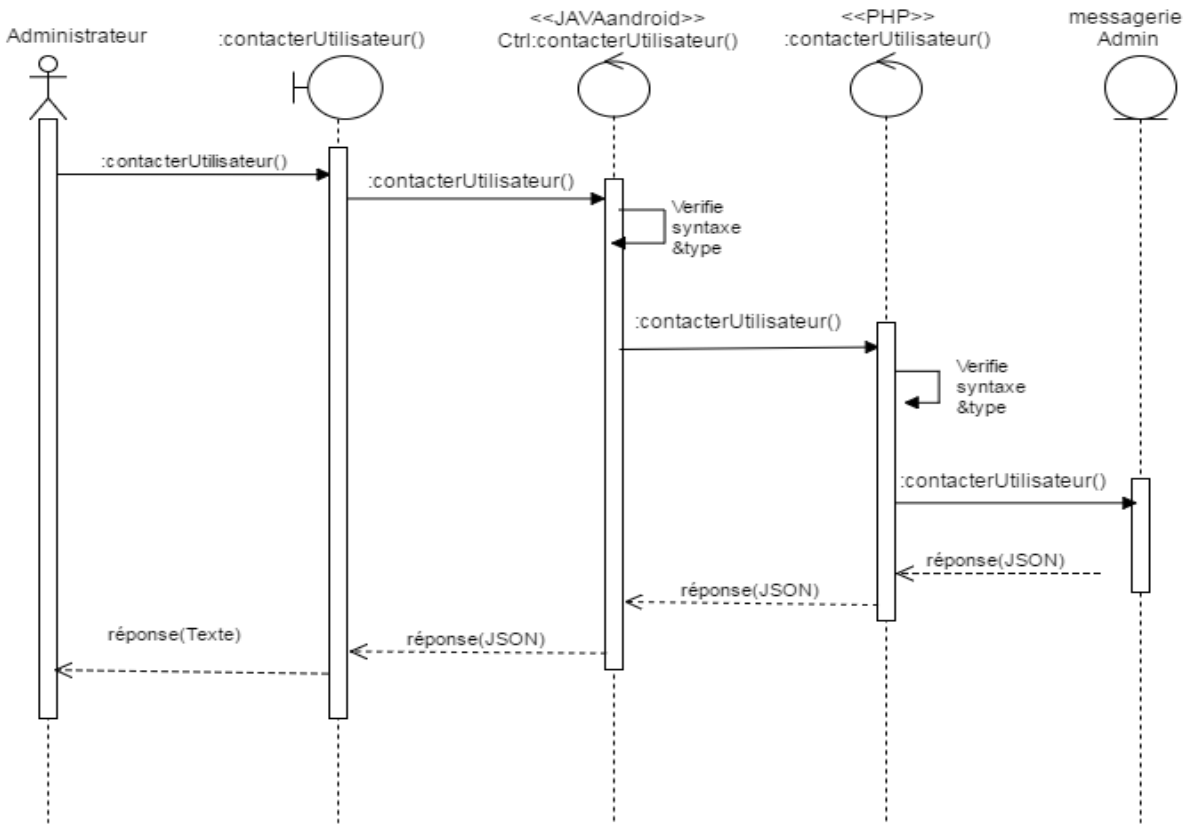
3.18.2. Diagramme de classes participantes



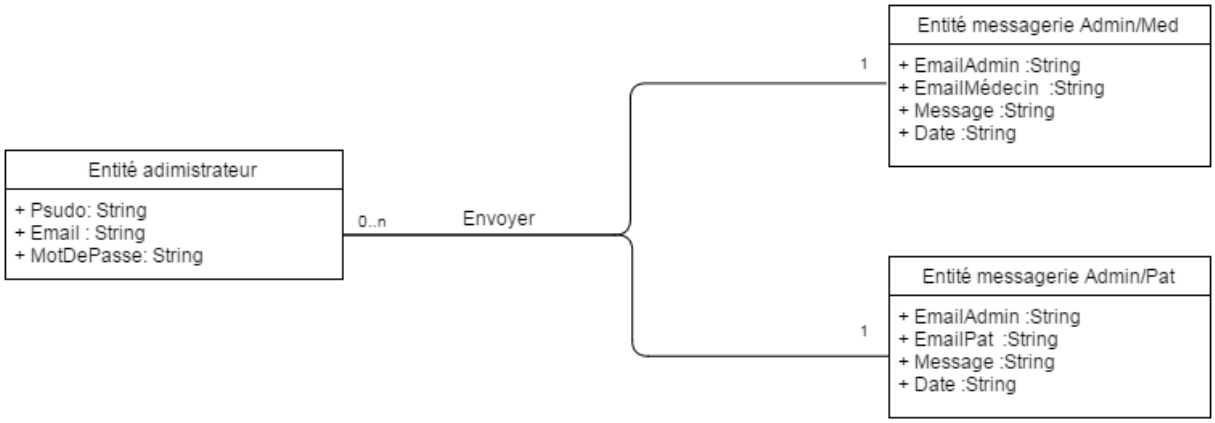
3.18.3. Diagramme de navigation



3.18.4. Diagramme d'interaction

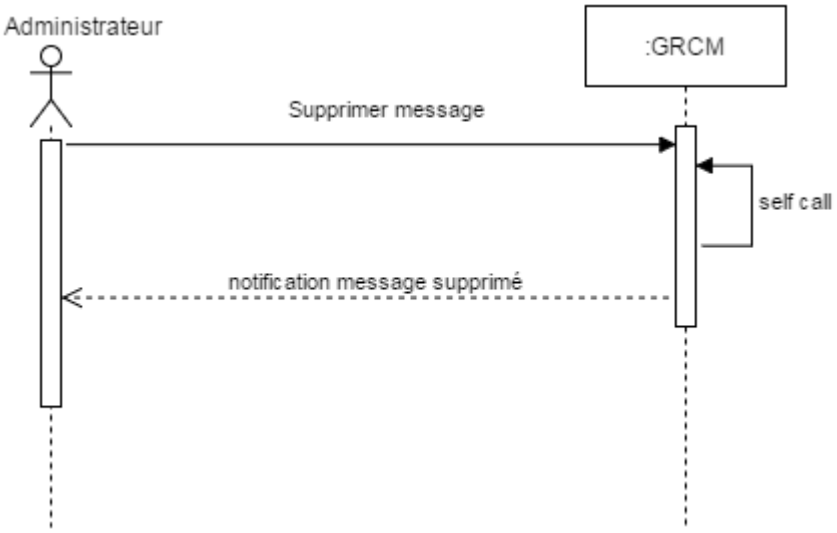


3.18.5. Diagramme de classes de conception préliminaires

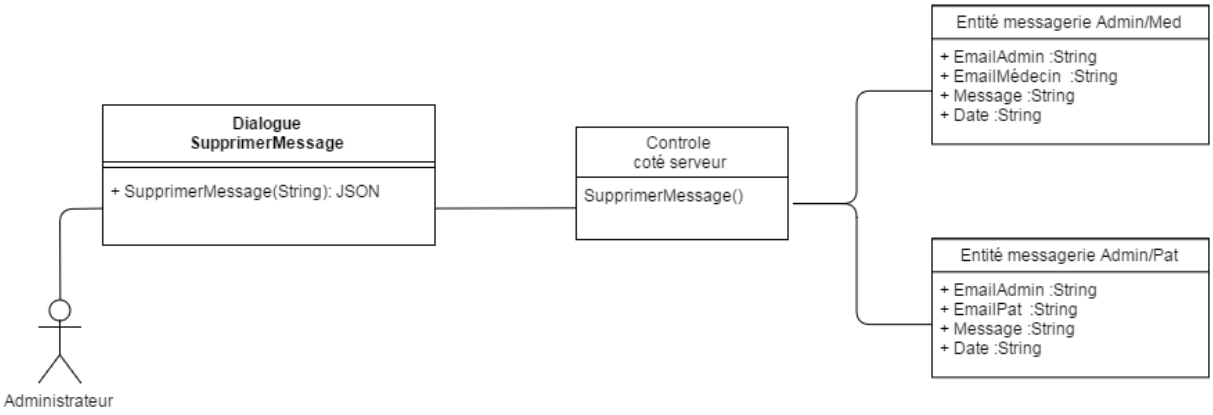


3.19. Supprimer des messages

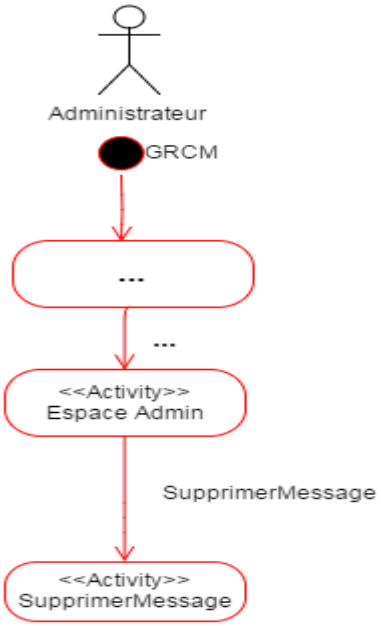
3.19.1. Diagramme de séquence système



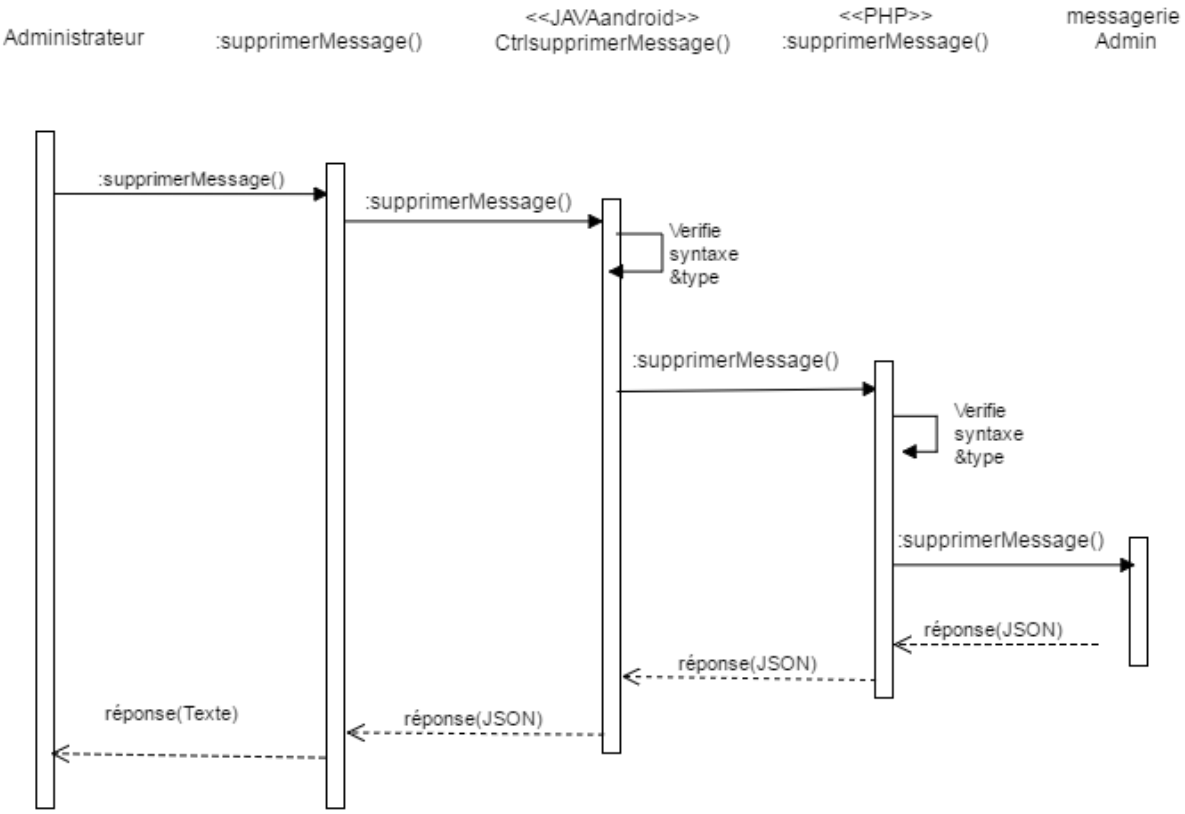
3.19.2. Diagramme de classes participantes



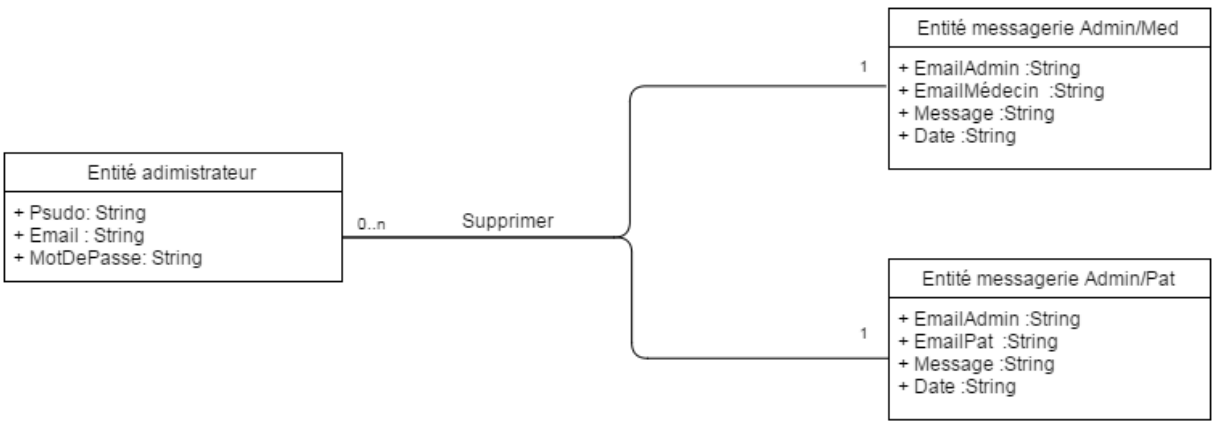
3.19.3. Diagramme de navigation



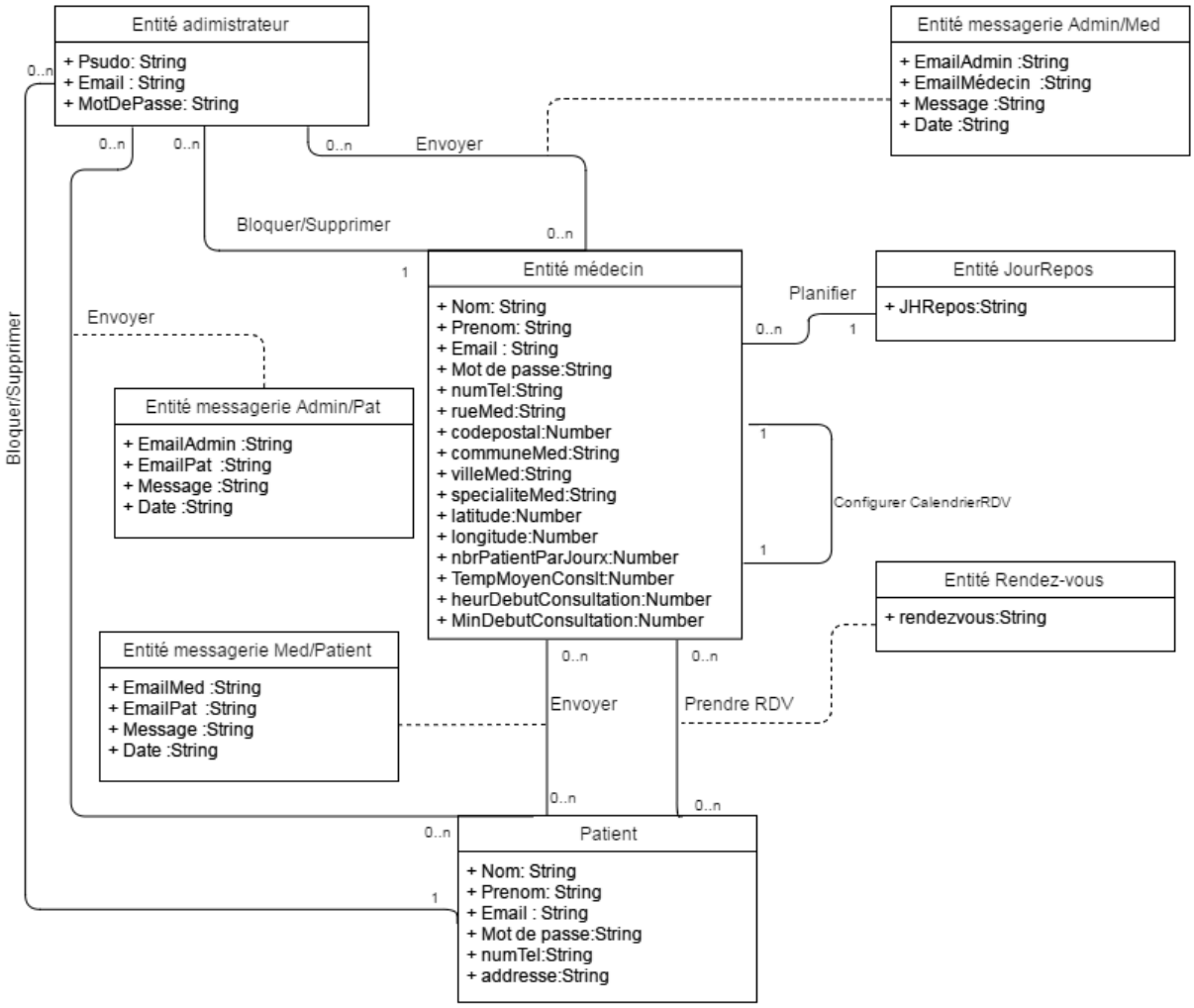
3.19.4. Diagramme d'interaction



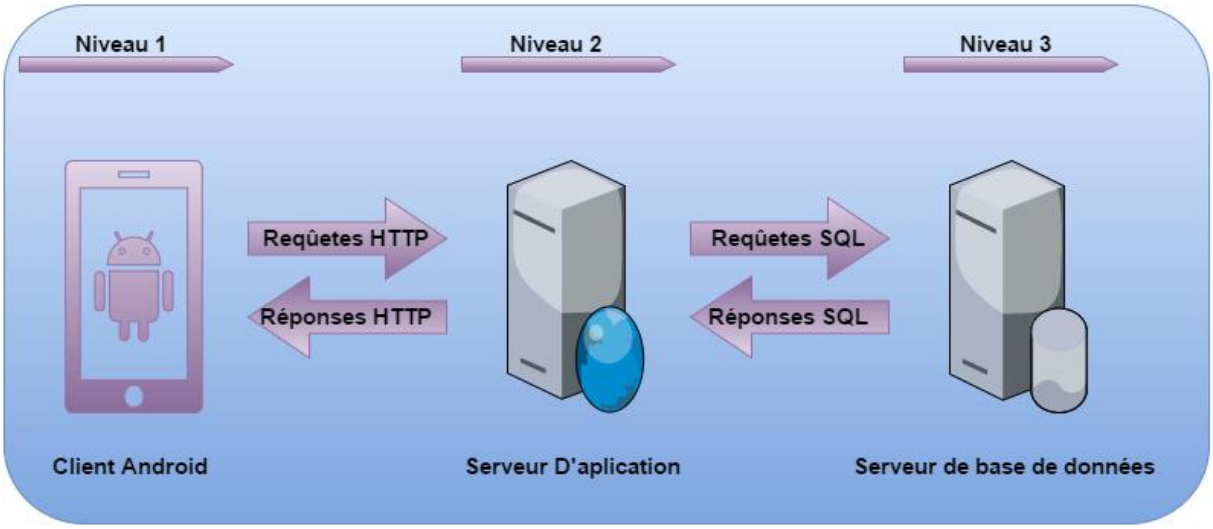
3.19.5. Diagramme de classes de conception préliminaires



4. Diagramme de classes de conception préliminaire général



5. Diagramme de déploiement



6. Conclusion du chapitre

Dans ce chapitre nous avons conçu le système de gestion des rendez-vous de cabinet médical à l'aide du langage de modélisation graphique UML qui fournir une méthode normalisée pour visualiser la conception d'un système.
Dans le chapitre suivant nous allons présenter les outils et les langages de développement de ce projet.

Troisième chapitre : Réalisation

1. Introduction

Ce chapitre couvre la création et la mise en œuvre des différents programmes, interfaces et bases de données, qui servent à la constitution de notre application et de ses fonctionnalités.
Nous décrivons l'environnement de création du système et de la base de données, ensuite nous présenterons quelques interfaces résultantes.

2. Les outils et langages de développement

2.1. IDE éclipse [16]

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Son objectif est de produire et fournir des outils pour la réalisation de logiciels, englobant les activités de programmation (notamment environnement de développement intégré et frameworks) mais aussi d'AGL recouvrant modélisation, conception, test, gestion de configuration, reporting… Son EDI, partie intégrante du projet, vise notamment à supporter tout langage de programmation à l'instar de Microsoft Visual Studio.

Bien qu'Eclipse ait d'abord été conçu uniquement pour produire des environnements de développement, les utilisateurs et contributeurs se sont rapidement mis à réutiliser ses briques logicielles pour des applications clientes classiques. Cela a conduit à une extension du périmètre initial d'Eclipse à toute production de logiciel : c'est l'apparition du framework Eclipse RCP en 2004.

Figurant parmi les grandes réussites de l'Open source, Eclipse est devenu un standard du marché des logiciels de développement, intégré par de grands éditeurs logiciels et sociétés de services. Les logiciels commerciaux *Lotus Notes 8*, *IBM Lotus Symphony* ou *WebSphere Studio Application Developer* sont notamment basés sur Eclipse.

2.2. Android [17]

Android est un système d'exploitation mobile basé sur le noyau Linux et développé actuellement par Google.

Lancé en juin 2007 à la suite du rachat par Google en 2005 de la startup du même nom, le système avait d'abord été conçu pour les Smartphones et tablettes tactiles, puis s'est diversifié dans les objets connectés et ordinateurs comme les télévisions (Android TV), les voitures (Android Auto), les ordinateurs (Android-x86) et les smart Watch (Android Wear).

En 2015, Android est, face à Ios d'Apple, le système d'exploitation le plus utilisé dans le monde avec plus de 80 % de parts de marché dans les Smartphones.

2.3. WAMP serveur [18]

WAMP est un acronyme informatique signifiant :

- « **W**indows »
- « **A**pache »
- « **M**ySQL »
- « **P**HP » dans la majorité des cas mais aussi parfois, « **P**erl », ou « **P**ython ».

Il s'agit d'un néologisme basé sur LAMP.

2.3.1. L'interface PhpMyAdmin [19]

phpMyAdmin (PMA) est une application Web de gestion pour les systèmes de gestion de base de données MySQL réalisée en PHP et distribuée sous licence GNU GPL.

2.3.2. ServeurApach [20]

Le logiciel libre **Apache HTTP Server** (**Apache**) est un serveur HTTP créé et maintenu au sein de la fondation Apache. C'est le serveur HTTP le plus populaire du World Wide Web. Il est distribué selon les termes de la licence Apache.

2.3.3. Serveur Mysql [21]

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, Informix et Microsoft SQL Server.

Son nom vient du prénom de la fille du cocréateur Michael Widenius, My. SQL fait référence au *Structured Query Language*, le langage de requête utilisé.

MySQL AB a été acheté le 16 janvier 2008 par Sun Microsystems pour un milliard de dollars américains. En 2009, Sun Microsystems a été acquis par Oracle Corporation, mettant entre les mains d'une même société les deux produits concurrents que sont Oracle Database et MySQL. Ce rachat a été autorisé par la Commission européenne le 21 janvier 2010.

Depuis mai 2009, son créateur Michael Widenius a créé MariaDB pour continuer son développement en tant que projet Open Source.

2.4. Java [22]

Le langage **Java** est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au *SunWorld*.

La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java.

La particularité et l'objectif central de Java est que les logiciels écrits dans ce langage doivent être très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou GNU/Linux, avec peu ou pas de modifications. Pour cela, divers plateformes et frameworks associés visent à guider, sinon garantir, cette portabilité des applications développées en Java.

2.5. PHP [23]

PHP: Hypertext Preprocessor, plus connu sous son sigle *PHP* (acronyme récursif), est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.

PHP a permis de créer un grand nombre de sites web célèbres, comme Facebook, Wikipédia, etc. Il est considéré comme la base de la création des sites Internet dits dynamiques.

2.6. JSON [24]

JSON (***JavaScript Object Notation)*** est un format de données textuelles dérivé de la notation des objets du langage JavaScript. Il permet de représenter de l'information structurée comme le permet XML par exemple. Créé par Douglas Crockford entre 2002 et 2005, il est décrit par la RFC 7159 de l'IETF.

Un document JSON a pour fonction de représenter de l'information accompagnée d'étiquettes permettant d'en interpréter les divers éléments, sans aucune restriction sur le nombre de celles-ci.

Un document JSON ne comprend que deux types d'éléments structurels :

- des ensembles de paires nom / valeur ;
- des listes ordonnées de valeurs.

Ces mêmes éléments représentent trois types de données :

- des objets ;
- des tableaux ;
- des valeurs génériques de type tableau, objet, booléen, nombre, chaîne ou null.

3. Diagramme de classe de conception préliminaire au modèle relationnel

3.1. Introduction

A partir de la description conceptuelle que nous avons effectuée, on peut réaliser le modèle relationnel; vu que le système d'information ne peut pas le manipulé directement; et ça en utilisons des règles de passages de l'UML vers le relationnel.

3.2. Quelques notions essentielles

Domaine : c'est l'ensemble des valeurs d'un attribut.
Relation : c'est un sous ensemble du produit cartésien d'une liste de domaines. C'est en fait un tableau à deux dimensions dont les colonnes correspondent aux Domaines et dont les lignes contiennent des tuples. On associe un nom à Chaque colonne.
Attribut : c'est une colonne d'une relation, caractérisé par un nom.
Tuple : c'est la liste des valeurs d'une ligne d'une relation.
Cardinalité : elle permet de définir les conditions de participation d'une entité à une relation. Toutefois, une entité peut participer à plusieurs relations.
L'arité : est le nombre d'attributs d'une relation.
Clé : On distingue deux types de clés:

Clé primaire : ensemble d'attributs dont les valeurs permettent de distinguer les n-uplets les uns des autres (notion d'identifiant).

Clé étrangère : Attribut qui est clé primaire d'une autre entité.

NB : pour la notation, nous avons choisi de souligner les clés primaires et de mettre * à la fin de chaque clé étrangère.

3.3. règles de passages

1- **Transformation des classes:** chaque classe du diagramme UML devient une relation, il faut choisir un attribut de la classe pouvant jouer le rôle de clé.
2- **Transformation des associations :** Nous distinguons trois familles d'associations**.**
   a. **Association 1.. :** 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.
   b. **Association *..* et n-aire et classes-association** : la classe-association devient une relation. La clé primaire de cette relation est la concaténation des identifiants des classes connectées à l'association.

c. **Association 1 .. 1 :** 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 de la classe connectée à l'association. Si les deux multiplicités minimales sont à un, il est préférable de fusionner les deux classes en une seule.

En appliquant ces règles de transformation d'un diagramme de classe vers un modèle relationnel, nous avons abouti au schéma relationnel suivant :

Médecins (idMed, nomMed, prenomMed, numTelMed, emailMed, passwordMed, rueMed, communeMed, codePostalMed, villeMed, SpecialiteListe, latitude, longitude, nbrPatientParJour, TempMoyenConsIt, heurDebutConsultation, MinDebutConsultation)

Patients (idPat, nomPat, prenomPat, adressePat, numTelPat, emailPat, passwordPat)

Administrateur (Email, Psudo, password)

JourReposMed (idJR, idMed*, JHRepos)

Messagerie_Admin/Med (idMsg, EmailAdmin*, EmailMed*, DateMsg)

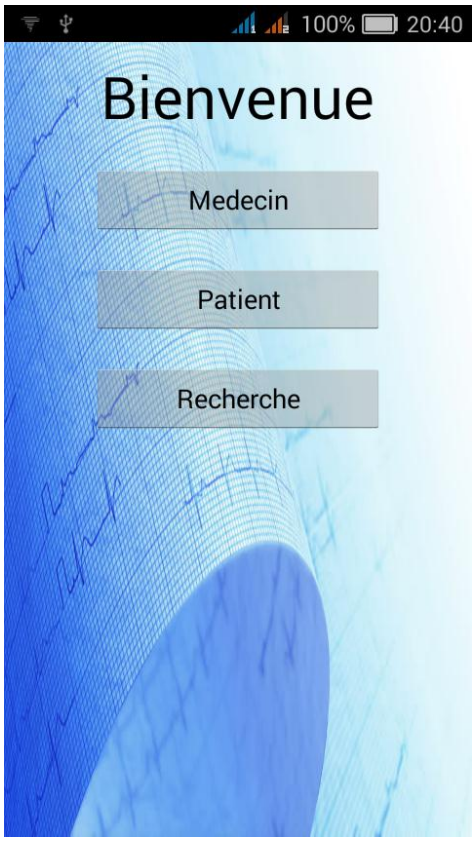
Messagerie_Admin/Pat (idMsg, EmailAdmin*, EmailPat*, DateMsg)

Messagerie_Med/Pat (idMsg, EmailMed*, EmailPat, DateMsg)

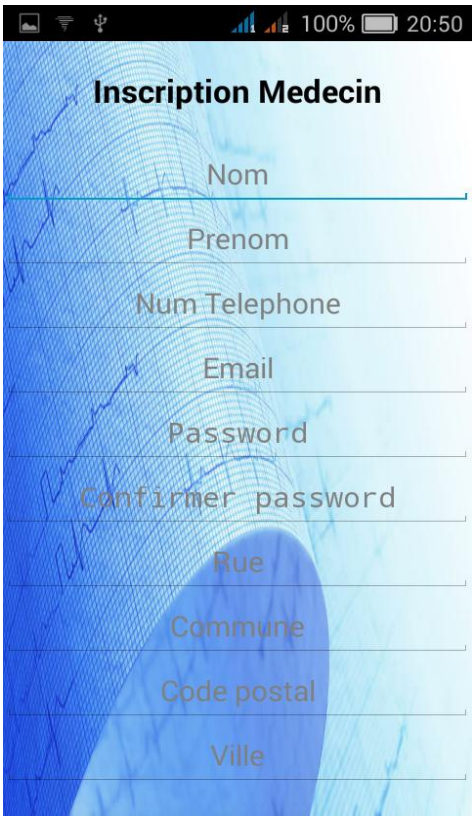
Rendez-vous (idrdv, idMed*, idPat*, rdv)

4. Interfaces de l'application

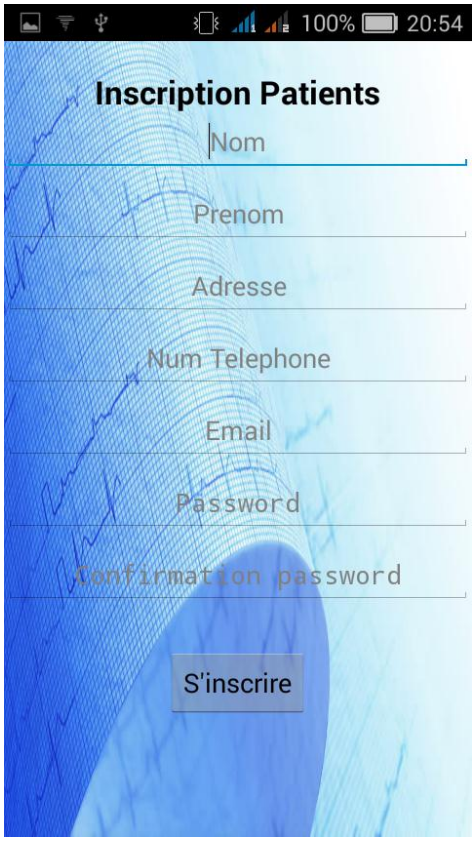
4.1. Home



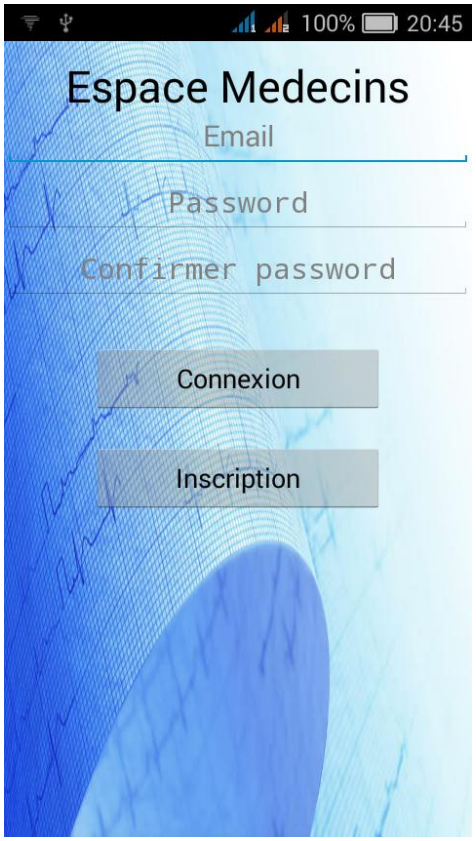
4.2. Inscription du médecin



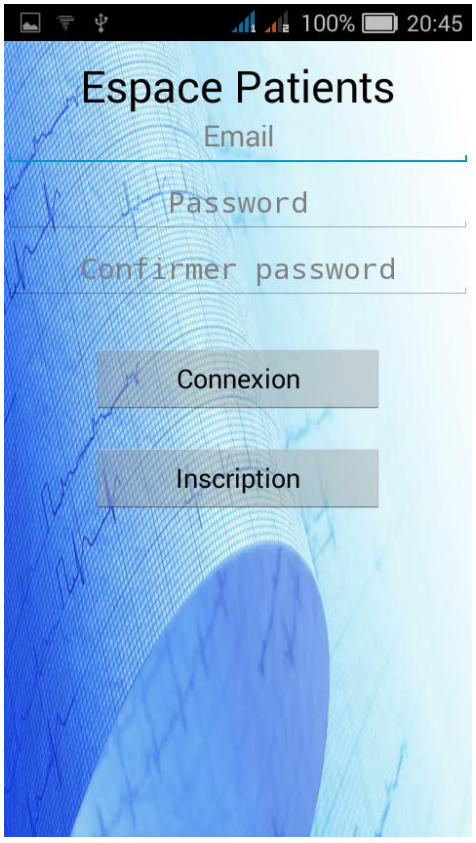
4.3. Inscription du patient



4.4. Connexion médecin



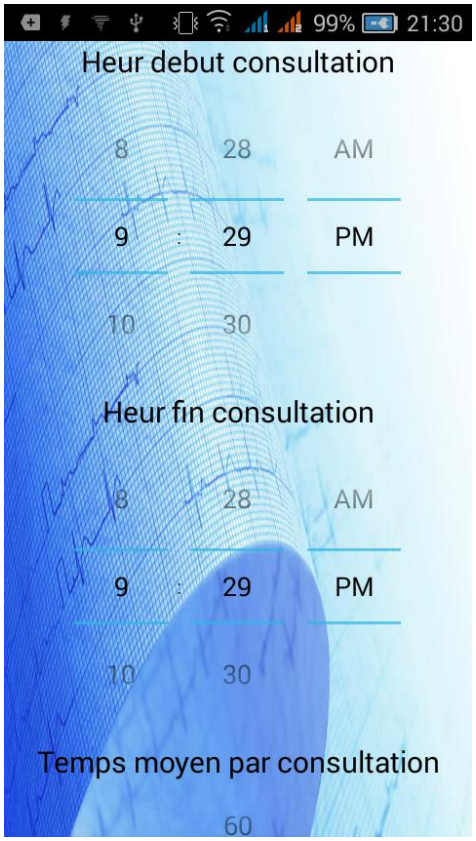
4.5. Connexion patient



4.6. Espace médecin

4.7. Paramétrer le calendrier des rendez-vous

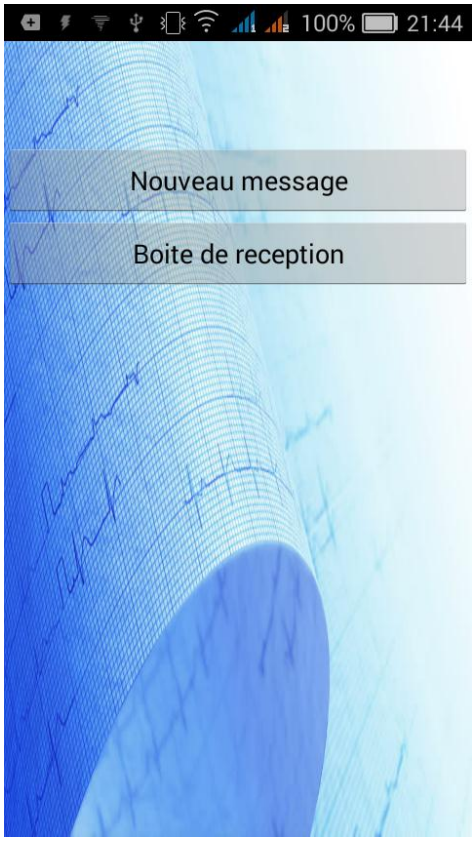
4.8. Planifier les horaires de travail



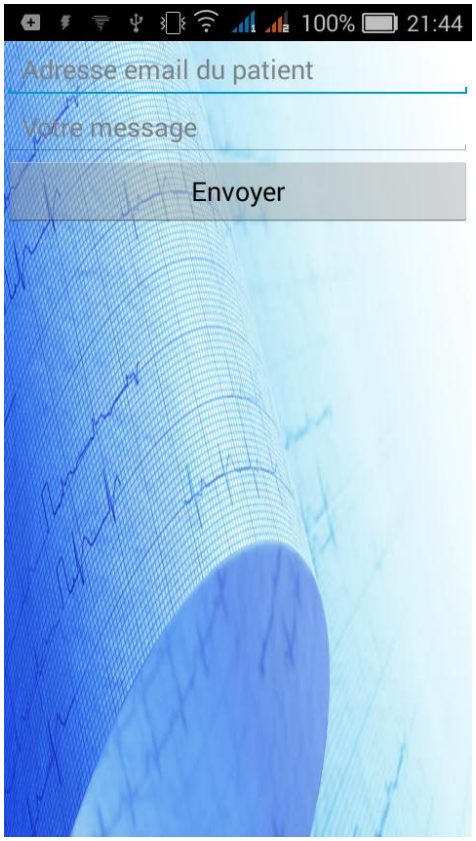
4.9. Planifier les horaires du repos

4.10. Consulter le catalogue des rendez-vous

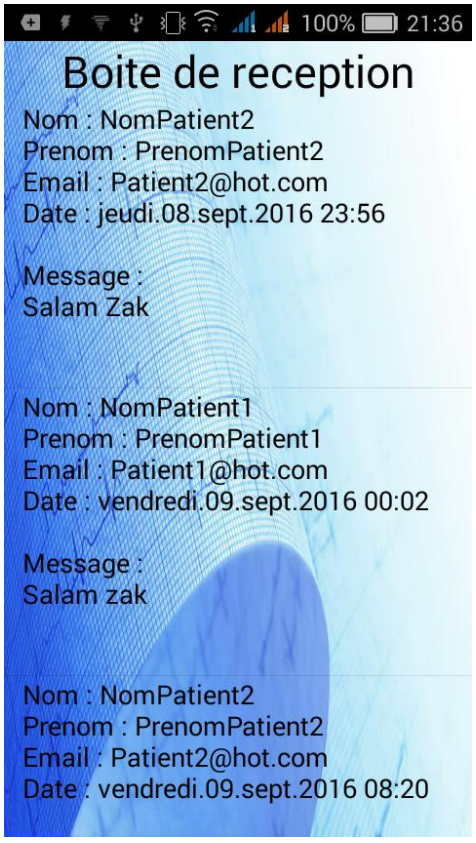
4.11. Messagerie du médecin



4.12. Envoyer un message au patient



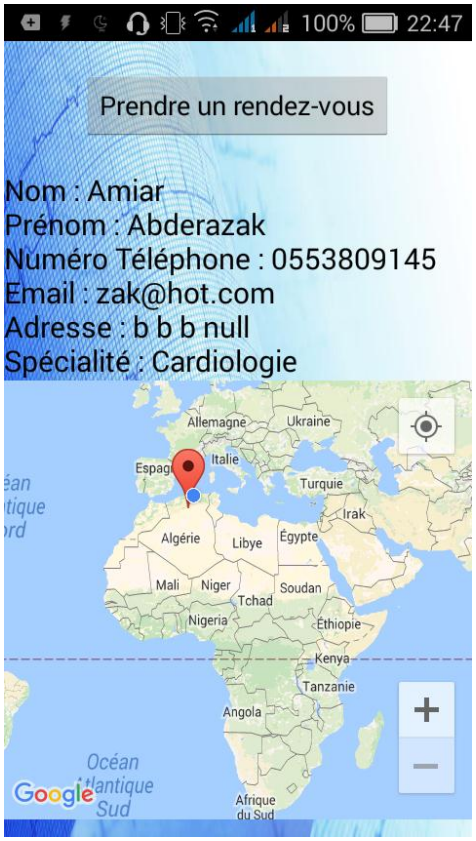
4.13. Boite de réception du médecin



4.14. Espace patient

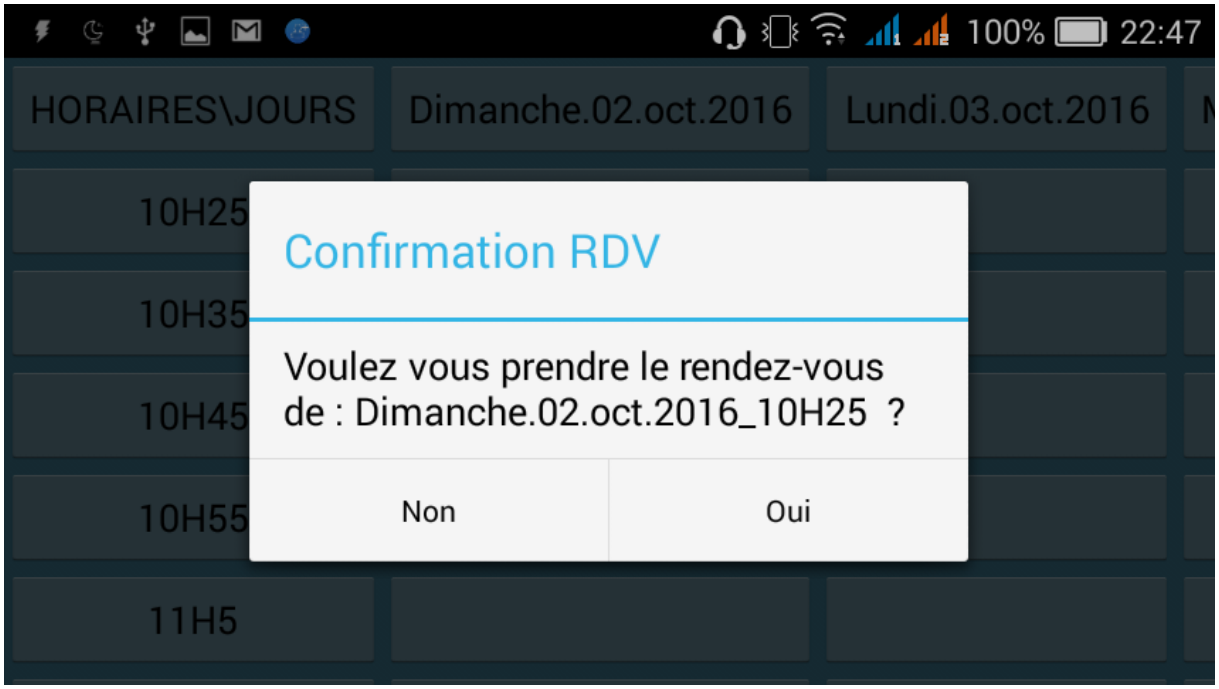
4.15. Recherche médecin

4.16. Résultat de la recherche

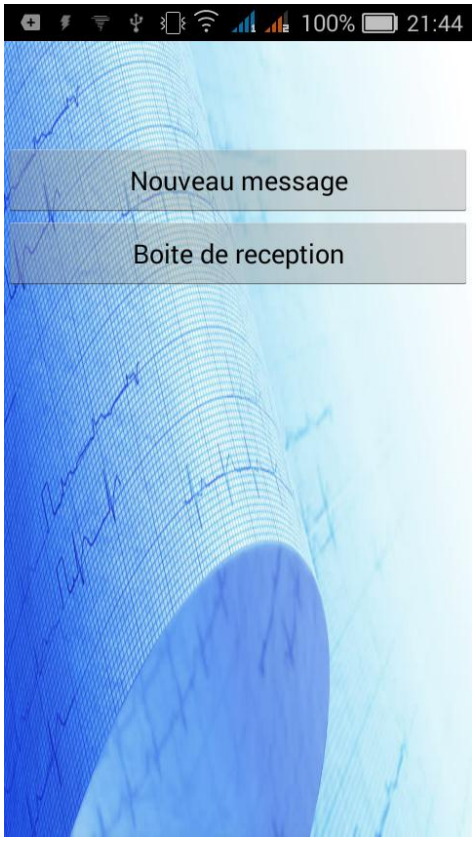


4.17. Consulter le calendrier des rendez-vous

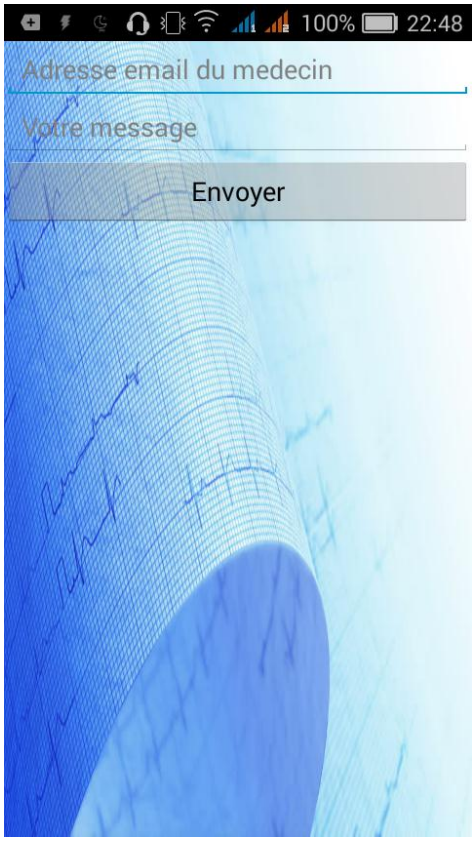
4.18. Prendre un rendez-vous



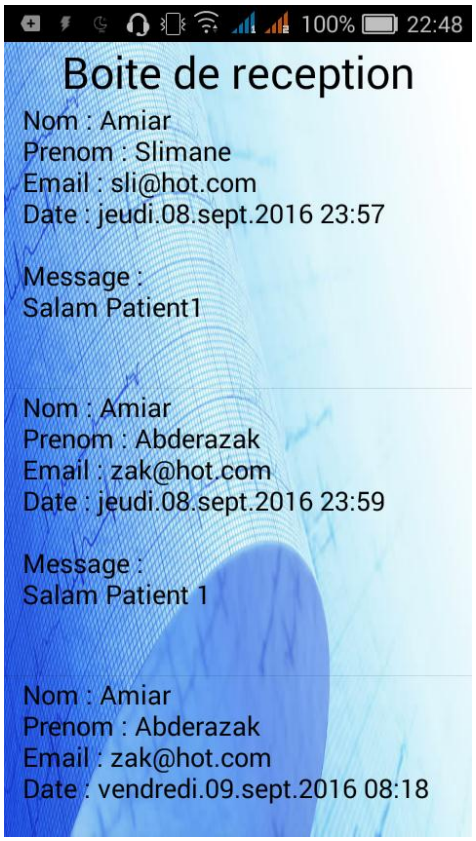
4.19. Messagerie du patient



4.20. Envoyer un message au médecin



4.21. Boite de réception du patient



5. Conclusion du chapitre

Dans ce chapitre nous avons présenté les outils et les langages utilisés dans ce projet afin de pouvoir aboutir à une solution logiciel finale productive.

6. Conclusion général

L'objectif de ce travail était de développer une application de gestion de rendez-vous médical mobile sous le système d'exploitation Android au profit des praticiens et des patients.

La réalisation de ce travail m'a permis :

- D'acquérir de nouvelles connaissances sur les langages UML, JAVA, PHP, Système d'exploitation Android, et aussi de se familiariser avec l'IDE Eclipse.
- D'approfondir mes connaissances théoriques et pratiques en rapport avec la conception des systèmes d'information, les services web de type REST et les bases de données, les techniques de programmation d'application mobile...
- De me familiariser avec un certain nombre d'outils informatiques de développement tel que l'environnement WAMP serveur, le système de gestion de base de données PhpMyadmin.

Je pense avoir atteint les objectifs fixés au départ, j'espère que ce travail sera à l'origine d'une mise en place réelle de mon application au sein des cabinets médicaux.

Bibliographie et webographie

[1] http://www.zdnet.fr/actualites/chiffres-cles-les-os-pour-smartphones-39790245.htm

[2] Livre : Android Guide de développement d'application pour Smartphones et Tablettes, Sébastien Perochon.

[3] Web: https://fr.wikipedia.org/wiki/Historique_des_versions_d'Android

[4] Livre: LES WEB SERVICES ET LEUR IMPACT SUR LE COMMERCE B2B, Gilbert Babin& Michel Leblanc.

[5] Web: https://www.w3.org/TR/ws-arch/#whatis

[6] Web: https://fr.wikipedia.org/wiki/Service_web#Introduction

[7] Web:https://fr.wikipedia.org/wiki/Service_web#Les_services_web_de_type_Representational_state_transfer_.28REST.29

[8] Web: https://fr.wikipedia.org/wiki/Service_web#Les_Services_Web_WS

[9] Web: https://fr.wikipedia.org/wiki/Application_mobile

[10] Web:https://www.redacteur.com/blog/avantages-inconvenients-dune-application-mobile/

[11] Web: https://fr.wikipedia.org/wiki/Representational_state_transfer

[12] Livre: Les Architectures client-serveur et internet et intranet des CGI aux EJB [Pierre-Yves Cloux, David Doussot, Aurélien Géron]

[13] Web: https://fr.wikipedia.org/wiki/Extensible_Markup_Language

[14] Web : https://fr.wikipedia.org/wiki/JavaScript_Object_Notation

[16] Web: https://fr.wikipedia.org/wiki/Eclipse_%28projet%29

[17]: Web: https://fr.wikipedia.org/wiki/Android

[18]: Web: https://fr.wikipedia.org/wiki/WAMP

[19]: Web: https://fr.wikipedia.org/wiki/PhpMyAdmin

[20]: Web: https://fr.wikipedia.org/wiki/Apache_HTTP_Server

[21]: Web: https://fr.wikipedia.org/wiki/MySQL

[22]: Web: https://fr.wikipedia.org/wiki/Java_%28langage%29

[23]: Web: https://fr.wikipedia.org/wiki/PHP

[24]: Web: https://fr.wikipedia.org/wiki/JavaScript_Object_Notation

Autres ouvrages de référence

- UML 2 modéliser une application web, Pascal Roques.
- En action de l'analyse des besoins à la conception, Pascal Roques et Franck Vallée.
- Conception de bases de données avec UML, Gilles Roy.
- Les web services et leur impact sur le commerce B2B, Gilbert Babin et Michel Leblanc.
- Mémoire de fin d'étude, gestion de cabinet médical, Djellil Djamel-Edine, université Abou Bakr Belkaid- Tlemcen.
- Créez des applications pour Android, Frédéric Espiau.
- Android Guide de développement d'application pour Smartphones et Tablettes, Sébastien Perochon.
- Concevez votre site web avec php et mysql, Mathieu Nebra.
- Sérialisez vos objets au format JSON, http://user.oc-static.com/pdf/573354-serialisez-vos-objets-au-format-json.pdf.
- Apprenez à programmer en java, http://user.oc-static.com/pdf/10601-apprenez-a-programmer-en-java.pdf

Autres Site web de référence

- http://www.uml-diagrams.org/
- http://stackoverflow.com
- https://developer.android.com
- www.developpez.com
- http://docs.oracle.com/javase/8/docs/api/
- https://developers.google.com/maps/documentation/android-api/
- http://php.net/docs.php