République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieure et de la recherche scientifique

N Attribué par la bibliothèque

Année Univ : 2017/2018

Développement d'une application échange messages sécurisé

Mémoire présenté en vue de l'obtention du diplôme de

Master Académique

Université Dr Tahar Moulay - Saïda

Discipline : INFORMATIQUE

Spécialité : Sécurité

par

Abdelhadi Mohamed Reda
Yahia Cherif Fethi

Soutenue le – Mai 2018 devant le jury composé de :

Dr :	Université Dr Tahar Moulay - Saïda	Président
Dr :	Université Dr Tahar Moulay - Saïda	Rapporteur
Mme :	Université Dr Tahar Moulay - Saïda	Examinatrice
Dr : Khobzaoui	Université Dr Tahar Moulay - Saïda	Encadreur

REMERCIEMENTS

Je tiens tout d'abord à remercier Mr Khobzaoui, notre encadreur, pour tout ses précieux conseils et ses encouragements durant la réalisation de ce mémoire.

Je tiens ensuite à remercier mes parents pour le soutien, moral et financier, inconditionnel dont ils ont fait preuve tout au long de mes études. Si je suis ici aujourd'hui, c'est grâce à eux.

Je souhaite aussi remercier, mes frères et mes sœurs pour leur accompagnement durant mes cinq années d'études.

Je remercie également toutes personnes qui, de près ou de loin, a participé à l'élaboration de ce manuscrit.

Mes remerciements vont également aux membres du jury qui nous ont honoré par leur participation au jury et pour l'intérêt qu'ils ont bien voulu porter à ce travail.

Enfin, je remercie mes amis et camarades de promotion avec qui j'ai vécue des meilleurs moments.

Dédicaces

Que ce travail témoigne de mes respects :

A mes parents : Qui avec à leurs tendres encouragements et leurs grands sacrifices, ont pu créer un climat favorable à la poursuite de mes études. Enfaite aucune dédicace ne pourrait exprimer mon respect, ma considération et mes profonds sentiments envers eux. Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

A mes sœurs et mes frères.

A la famille Abdelhadi et Yahia Cherif .

à qui j'exprime tous mes sentiments de respect et de reconnaissance pour le soutien qu'ils n'ont cessé de me porter.

A tous mes enseignants dont leur générosité et leur soutien m'oblige de leur témoigner mon profond respect et ma loyale considération.

A tous mes amis et collègues :

qu'ils trouvent ici le témoignage d'une fidélité et d'une amitié infinie.

Résumé

Notre travail consiste à concevoir et réaliser une application d'échange de messagerie sécurisée. Cette application que nous l'avons baptisé **SIC-CHAT** prend en charge les conversations avec une, ou plusieurs, personne(groupe), l'envoi de textes, de photos, de stickers et d'emojis. via un réseau local ou via internet. Les conversations sont protégées, au chois, avec deux systèmes de chiffrement(Cryptage Symétrique DES ou un cryptage Asymétrique RSA).

Comme outil de modélisation et de conception nous avons choisie l'UML qui se présente comme un langage universel de conception et la modélisation. La réalisation quant à elle nécessite des outils de développements bien adaptés au contexte de l'application. Dans le cadre de ce projet nous avons fait recours aux outils suivants : MySql, apache Derby pour la gestion de notre base de données et Java sous NetBeans comme langage de programmation et plus d'autre outils tel que....

Mots clés : Messagerie Sécurisée, Chiffrement, Déchiffrement, Cryptage Symétrique, cryptage Asymétrique

Abstract

Our job is to design an application handles sending and receiving conversations,The application supports conversations with one or more people (group), the application SIC-CHAT manages some functions in addition to sending texts : sending photos, stickers and emojis, messages of groups of course. Finally, SIC-CHAT makes it possible to carry out conversations and go through a Local network or via a Wi-Fi access point.

The circulation of our messages are protected with two Encryption System, We can choose between them (Symmetric Encryption DES or RSA Asymmetric Encryption), our willingness to let people communicate without barriers.

Each creation requires a modeling with a well-specified universal language such as UML, the realization as for it requires development tools well adapted to the context of the application. For databases, use a DBMS such as MySQL, Apache Derby, and so on. is essential.

The SIC-CHAT message exchange application (Secure Chat) for the encryption of data circulating between users. , the design and implementation of a system to optimize the management of message exchanges.

The application has been developed using different computer programs such as Apache derby, JavaFX, Adobe Photoshop, ... etc. The programming language used is Java.

Keywords : SIC-CAT, DES, RSA, MySQL, Wifi, Apache Derby, JavaFX, Adobe Photoshop.

Table des matières

Table des figures

Liste des tableaux

Glossaire

SIGLES ET ABREVIATIONS

SIC-CHAT : Sécurité Chat

MI : messagerie instantanée

RC : Relay Chat

MS : messagerie sécurisé

INTERNET : Interconnected Networks

IP : Internet Protocol

TLS : Transport Layer Security

PKI : Public Key Infrastructure

RSA : Ron Rivest, Adi Shamir et Leonard Adleman ; Algorithme

DES : Data Encryption Standard

AES : Advanced Encryption Standard

TCP : Transport Control Protocol

TCP/IP : Transmission Control Protocol/Internet Protocol

UP : Processus Unifié

UML : Unified Modeling Language

E-Mail : Electronic Mail

LAN : Local Area Network

MDA : Model Driven Architecture

UML : Unified Modeling Language

SQL : Structured Query Language

FTP : File Transfer Protocol

XML : Extensible Markup Language

Introduction générale

Le besoin de dissimuler les informations a préoccupé l'homme depuis le début de la civilisation, La confidentialité apparait comme un facteur primordial dans toute lutte pour l'accès au pouvoir que ce soit dans un cadre militaire ou diplomatique. Aujourd'hui, de plus en plus d'applications dites civiles sont utilisées pour véhiculer ou échanger des données, entre deux ou plusieurs interlocuteurs, via les réseaux de télécommunications actuels.
Ainsi, les banques utilisent ces applications pour communiquer avec leurs clients leurs permettant ainsi de suivre leurs comptes et d'effectuer des transaction à distance. Les laboratoires de recherche s'en servent pour échanger informations et données classifiées secrétés entre ces membres. Les chefs militaires et politiques pour transmettre et/ou échanger des informations utra-secrétés comme des ordres et des rapports qui nécessite d'âtre confidentiels. Toutes ces informations et données doivent être cachées ou dissimulée a fin qu'elle ne soient pas compréhensibles par des tierces personnes non autorisées. La survie de ces institutions et organisation en dépend. Cette caractéristique de non "compréhensibilité" est assurée par un vaste champ de recherche en pleine extension qui est la cryptographie. Enfaite, le mot cryptographie est un terme assez générique regroupant un ensemble de techniques de chiffrement et de déchiffrement permettant d'assurer les caractères de confidentialité et d'intégrité des données échangées via les réseaux de télécommunication actuelles. Ces derniers sont considérés comme le moyen le plus sûr pour protéger ces données. Toute fois susceptibles à des attaques pouvant les casser.

L'objectif principal de notre projet de fin d'étude consiste à concevoir et développer une application d'échange ou de conversation sécurisée. Cette sécurisation est assurée par un chiffrement des données échangées à la source et de les déchiffrée à la destination. Le présent mémoire est structuré en quatre chapitres comme suite :

- Le chapitre 1 : Introduit les concepts de base relatifs à la messagerie électronique. On y présentera les composantes logicielles d'une application de messagerie électroniques, son fonctionnement et la façon de sa sécurisation.
- Le chapitre 2 : Introduit les concepts de base de la cryptographie. On y présentera, en particulier, les principaux algorithmes cryptographiques.
- Le chapitre 3 : Sera consacré à l'étude conceptuelle de notre projet. On y présentera les outils de conception utilisées et les différents diagrammes UML relatifs à notre application.
- Le chapitre 4 : quant à lui sera consacré à la phase d'implémentation de notre application.
- une conclusion générale clôtura ce manuscrit.

1 Système de messagerie électronique

1.1 Définition

Un système de messagerie électronique est l'ensemble des éléments contribuant à transmettre un message via un réseau informatique. Globalement un système messagerie électronique est constitué des composantes suivantes : Un serveur de messagerie, un client de messagerie un ensemble de trois agents de messagerie et un ensemble de protocoles de messagerie.

1.1.1 Serveur de messagerie

Un serveur de messagerie électronique est un logiciel serveur de courrier électronique. Il a pour vocation de transférer les messages électroniques d'un serveur à un autre. Un utilisateur n'est jamais en contact direct avec ce serveur mais utilise soit un client de messagerie, soit un courrielleur web, qui se charge de contacter le serveur pour envoyer ou recevoir les messages. Parmi les serveurs de messagerie les plus connus on peut citer :

- Sendmail qui est un serveur de messagerie qui s'occupe de la livraison des messages électroniques et en plus qu'il est "Open" source, il est caractérisé par sa résistance à une grande charge, sa haute sécurité comme il est Multi plate-forme(MAC OS, GNU/LINUX). Mais il est aussi connue pour sa lenteur, sa difficile configurer et maintenance car son architecture est aussi vieille que complexe.
- Postfix qui est un serveur de messagerie gratuit, adapté aux gros besoins, facile à configurer et à maintenir, Multi plate-forme (MAC OS, GNU/LINUX). et accessible en mobilité. Ce serveur ne compte pas d'inconvénients majeurs.

1.1.2 Client de messagerie

Un client de messagerie, également appelé courrielleur est un logiciel qui permettant la lecture, l'écriture, l'expédition et le stockage des courriers. Il est installé sur les postes des utilisateurs qui se connectent au serveur de messagerie. Ce logiciel utilise des protocoles de communication (POP, IMAP, SMTP) pour recevoir et envoyer des emails et des serveurs de messagerie pour communiquer entre eux. il existe deux types clients à savoir lourds et léger.

Les clients de messagerie de type léger sont des logiciels installés sur des postes clients, permettent de se connecter au serveur de messagerie via un navigateur web. Ils fonctionnent uniquement en mode connecté et ne copie pas en local les messages stockés sur le serveur. Ainsi, en mode hors connexion nous n'avons plus accès à nos courriers. Comme exemples de clients de messagerie léger on cite : RoundCube, Zimbra, Squirrelmail,

Un client de messagerie lourd englobe toutes les applications que l'on peut installer sur chaque poste de travail des différents collaborateurs. Ces applications peuvent être liées à un serveur qui sauvegardera les données. Avec un client de messagerie "lourd" nous avons l'avantage de ne pas laisser nos données personnelles sur des serveurs tiers, de pouvoir y accéder en n'étant pas connecté au réseau, nous disposons également des fonctions de recherches bien plus abouties que sur la majorité des webmails. Parmi les client de messagerie lourd les plus connus on peut citer : Microsoft Outlook, Mozilla Thunderbird.

1.1.3 Les agents de la messagerie

Comme déja mentionné ci-haut, on compte trois agents de messagerie :

- Le Mail Transfert Agent (MTA) : Qui est un programme permettant d'envoyer le message d'un serveur à un autre. Ce logiciel est situé sur chaque serveur de messagerie. Il est composé d'un agent de routage et d'un agent de transmission. Il envoie le message via un protocole sortant. Les protocoles sortants permettent de gérer la transmission du courrier entre les systèmes de messagerie.
- Le Mail User Agent (MUA) : Cet agent se pressente, d'une part, comme un un logiciel client pour le MTA dont le rôle consiste à formater les messages en partance afin de les donner au MTA, et d'autre part comme un un client pour le MDA qui formate les messages de la boîte aux lettres afin de les afficher à l'écran.
- Le Mail Delivery Agent (MDA) : Cet agent prend en charge charge la gestion des boites aux lettres. Il prélève le courrier dans les files d'attentes du MTA et le dépose dans le répertoire de boites aux lettres de l'utilisateur. Pour cela il est souvent considéré comme le point final d'un système de messagerie. Il est possible de placer des fonctions de sécurité à ce niveau comme des antivirus ou des anti-spam.

1.1.4 Les protocoles de la messagerie

Un protocole est définie comme étant un ensemble de règles et de procédures à respecter pour émettre et recevoir des données sur un réseau. Ils sont généralement classés en deux catégories :

1. Les protocoles orientés connexion : Il s'agit des protocoles opérant un contrôle de transmission des données pendant une communication établie entre deux machines.
2. Les protocoles non orientés connexion : lorsqu'aucune connexion n'est établie entre la machine émettrice et celle réceptrice.

Les trois principaux protocoles utilisés par un serveur de messagerie sont le SMTP (Simple Mail Transfer Protocol), le POP (Post Office Protocol) et l'IMAP (Internet Message Access Protocol).

- SMTP (Simple Mail Transfert Protocol) : Est le protocole standard permettant de transférer le courrier d'un serveur à un autre en connexion point à point. Il s'agit d'un protocole fonctionnant en mode connecté, encapsulé dans une trame TCP/IP. Le courrier est remis directement au serveur de courrier du destinataire. Le protocole

SMTP fonctionne grâce à des commandes textuelles envoyées au serveur SMTP (par défaut sur le port 25). Chacune des commandes envoyées par le client est suivi d'une réponse du serveur SMTP composée d'un numéro et d'un message descriptif.

- POP (Poste Office Protocol) : Permet comme son nom l'indique d'aller récupérer son courrier sur un serveur distant (le serveur POP). Il est nécessaire pour les personnes n'étant pas connectées en permanence à Internet afin de pouvoir consulter les mails reçus hors connexion. Il existe deux principales versions de ce protocole, POP2 et POP3, auxquels sont affectés respectivement les ports 109 et 110 et fonctionnant à l'aide de commandes textuelles radicalement différentes tout comme dans le cas du protocole SMTP
- Le protocole IMAP (Internet Message Access Protocol) est un protocole alternatif au protocole POP3 mais beaucoup plus complet et offrant beaucoup plus de possibilités. Ainsi il permet de gérer plusieurs accès simultanés, de gérer plusieurs boîtes aux lettres et enfin de trier le courrier selon plus qu'un critères. Le protocole IMAP permet de répondre beaucoup mieux à des besoins de déplacement. Il minimise également les échanges de données sur le réseau. La plupart des clients de messagerie implémentent le protocole IMAP puisque celui-ci est largement utilisé par les différents fournisseurs d'accès à Internet.

1.2 Fonctionnement d'une messagerie électronique

La figure 1.1 présente le transfert d'un courriel d'un expéditeur à un destinataire.

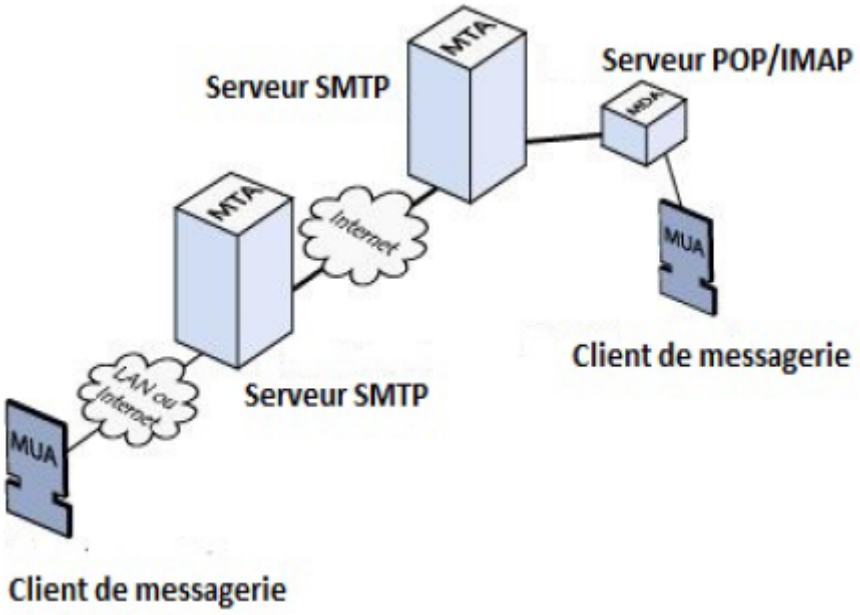


FIGURE 1.1 – fonctionnement de la messagerie électronique.

Le fonctionnement d'un système messagerie est décomposé en au moins deux étapes significatives et indépendantes : l'envoi et la réception et peut suit le scénario suivant :

1. L'expéditeur communique son courriel via le MUA.

2. Le MUA transmet ce courriel au MTA (la plupart des MUA intègre des clients SMTP).
3. Le MTA du système de l'émetteur établit un canal de transmission avec le MTA du système du destinataire, par émissions successives de requêtes bidirectionnelles.
4. Une fois le canal établit, le courriel est transmis d'un système à un autre par les MTA.
5. Dans le système du destinataire, Le MTA transmet le courrier reçu au serveur IMAP ou POP3.
6. Le MDA récupère le courriel du serveur IMAP / POP 3, et le met à disposition du MUA.
7. Le MUA dépose le courriel dans les boîtes aux lettres du destinataire qui pourra le consulter à tout moment, sur authentification.

1.3 La sécurité de la messagerie électronique

Comme tout système informatique, la messagerie se trouve face à des risques et menaces qui touchent à l'intégrité et la confidentialité des données. On peut identifier trois catégories de risque :

1. **Perte de confidentialité :** les messages et les pièces jointes associées, quand ils sont envoyés en clair sur un réseau, peuvent être facilement lus par des tierces parties(voir figure 1.2). Cela est particulièrement vrai lors de l'envoi d'informations sensibles, documents techniques ou commerciaux[9].
   La figure 1.2 présente le risque de perte de confidentialité du message sur un réseau clair .

FIGURE 1.2 – Risque : perte de confidentialité du message .

2. **Usurpation d'identité :** Un message peut être " forgé ", c'est à dire fabriqué de toutes pièces, en indiquant une fausse identité pour l'expéditeur. Un attaquant peut alors prendre l'identité d'une personne connue ou inconnue pour déstabiliser l'entreprise[9]. Voir figure 1.3.
   La figure 1.3 présente le risque Usurpation d'identité a partir d'un message forgé .

FIGURE 1.3 – Risque : Usurpation d'identité .

3. **Modification du contenu d'un message :** Un Message peut être intercepté, modifié, puis relayé à son destinataire. Par exemple, il est techniquement possible d'intercepter un courriel, de modifier le corps du texte ou les pièces jointes (documents Word. . .) puis de le relayer normalement. Rien ne garantit que le message que l'on reçoit correspond bien à celui qui a été envoyé par son expéditeur[9]. Voir figure 1.4.
   La figure 1.4 présente le risque Modification du contenu d'un message .

FIGURE 1.4 – Risque : Modification du contenu d'un message.

Ces risques sont réels, et la sensibilisation de tous les acteurs est essentielle. Ainsi les questions pertinentes à se poser sont les suivantes :
► Quel serait l'impact si une information commerciale confidentielle venait à être connue par un de mes concurrents ? (risque de confidentialité)
► Quel serait l'impact si un message envoyé à un de mes partenaires était modifié à mon insu ? (risque d'intégrité)
► Comment établir une relation de confiance avec mes partenaires commerciaux et les rassurer sur l'authenticité de mes messages ? (risque d'authentification)
► Quel est le coût ou le nombre de jours / hommes nécessaires pour mettre en place une solution simple pour garantir la confidentialité de mes échanges ?[9].

La cryptographie permet de répondre de façon sûre et efficace à ces 3 types de risques. Le prochain chapitre sera, entièrement, consacré à l'introduction des concepts de bases de la cryptographie

Un bonne solution de sécurité consiste, globalement, à :

▶ Utiliser des techniques de chiffrement et de signatures électroniques des messages.

▶ Utiliser des protocoles sécurisés pour sécuriser les communications entre les MTA et entre le serveur et les clients de messagerie. Parmi ces protocoles on cite[26].

- SSL (Secure Socket Layer) qui est développé pour permettre de la communication sécurisée en mode client/serveur pour des applications réseaux utilisant TCP/IP.
- Le protocole TLS (Transport Layer Security) est une évolution de SSL réalisé par l'IETF et qui sert de base à HTTPS par exemple.

▶ Faire face virus en analysant et en filtrant les messages dés leurs entrée sur le réseau.

▶ Faire face au spams par l'analyse lexicale et la mise en place d'une listes noire et blanche, d'utiliser des Realtime Blackhole List pour les adresses ou de relais SMTP et en utiliser des signatures.

1.4 Conclusion

Dans ce chapitre, nous avons tenter de présenter les principaux concepts reliés à la messagerie électronique. Nous avons parles des différents composants d'un système de messagerie électroniques, son fonctionnement et comment il peut être sécurisé. Comme dans le cadre de ce mémoire, on est appelé à concevoir et à réaliser une application de messagerie sécurisée à base de techniques de cryptographie, nous avons juger nécessaire de consacrer un chapitre à l'introduction ces techniques. Ainsi, le prochain chapitre sera dédié à la présentation des principaux concepts de la cryptographie.

2 Généralités sur la cryptographie

2.1 Introduction

La cryptographie est une science très ancienne qui date de 1900 ans avant J.-C. Des recherches indiquent qu'un scribe égyptien a employé des hiéroglyphes non conformes à la langue pour écrire un message. De ce temps-là et au long de l'histoire, la cryptographie a été utilisée exclusivement à des fins militaires. Aujourd'hui, les réseaux informatiques exigent une phase de cryptographie comme mécanisme fondamental afin d'assurer la confidentialité et l'intégrité des échanges de messages en présence d'un ou plusieurs attaquants, ou intrus, potentiels.

En se rapportant à des événements historiques de notre ère informatisée, nous pouvons dir que l'histoire de développement de la cryptographie a été marqué par les faits suivants :

- La machine Enigma créée par Dr Arthur Scherbius en 1923 et qui a été utilisée largement dans la seconde guerre mondiale ;
- La théorie de l'information de Claude Shannon en 1949 ;
- L'algorithme lucifer développé par IBM en 1970 ;
- la théorie du système à clef publique par Whitfield Diffie et Martin Hellman en 1976 qui a marqué le début de la cryptographie moderne ;
- Le premier standard pour le cryptage, l'algorithme DES en 1976 ;
- Les résultats de la cryptographie quantique par Charles H. Bennett et Gilles Brassard en 1990,
- Le standard AES en 2000, qui a remplacé le DES[6].

2.2 Lexique de base :

Nous commençons ce chapitre par l'introduction de certains concepts qui faciliteront la compréhension les objectifs de notre application :

- **Notion de sécurité**
  Il est possible de classer les cryptosystèmes en trois catégories par rapport à la sécurité : parfaite, calculatoire et sémantique[6].
  Sécurité parfaite. La notion de sécurité parfaite ou inconditionnelle a été créée par Shannon. Un cryptosystème est à sécurité parfaite si aucune information ne peut être obtenue sur le texte clair, à partir du texte crypté correspondant. En réalité, cette approche est impraticable dans la plupart des scénarios. Diffie et Hellman ont

proposé le remplacement de la sécurité parfaite par le concept de sécurité calculatoire pour une meilleure évaluation des cryptosystèmes.

Sécurité calculatoire
La sécurité calculatoire est liée à la quantité de ressources informatiques. Elle suppose que si on ne dispose que d'une puissance de calcul limitée, alors il est impossible de déduire le texte clair. C'est-à-dire, qu'il est impossible de résoudre certains problèmes de calculs très complexes (la factorisation de grands nombres ou l'extraction de logarithme discret par exemple) dans un temps raisonnable. La sécurité des algorithmes est démontrée en montrant que la capacité d'un adversaire à casser le procédé avec les ressources existantes et avec une probabilité significative est impossible. Shordans, a proposé une méthode basée sur un ordinateur quantique (théorique) pour résoudre les problèmes difficiles. Cette méthode suscite que la sécurité calculatoire est un fait directement lié aux avancements technologiques.

Sécurité sémantique
La sécurité sémantique, introduite par Goldwasser et Micali dans, s'adresse aux cryptosystèmes asymétriques, et coïncide avec la notion de la sécurité parfaite limitée aux chiffrements probabilistes. L'exigence de la sécurité sémantique est très forte et dit qu'on ne peut pas extraire une information, même partielle, sur le texte clair à partir du texte chiffré et de la clef publique. La sécurité sémantique est considérée comme insuffisante parce qu'elle prévoit que l'attaquant a connaissance seulement du texte chiffré et de la clef publique. Elle ne prend pas en compte les autres types d'attaques comme l'attaque à texte chiffré choisi.

- **texte chiffré ou Cryptogramme :** Message chiffré ou codé.
- **Cryptographie** : Discipline incluant les principes, les moyens et les méthodes de transformation des données, dans le but de masquer leur contenu, d'empêcher leur modification ou leur utilisation illégale.
- **Cryptologie :** Science des messages secrets. Se décompose en cryptographie et cryptanalyse.
- **Cryptanalyse** : analyse des textes chiffrés pour retrouver les informations dissimulées
- **Chiffrer** : Ensemble de procédés et ensemble de symboles (lettres, nombres, signes, etc.) employés pour remplacer les lettres du message à chiffrer. On distingue généralement les chiffres à transposition et ceux à substitution. Chiffrer=Crypter : Transformer un message afin qu'il ne soit lisible qu'à l'aide d'une clef.
- **Decrypter** : Parvenir à restaurer des données qui avaient été chiffrées, donc à leur faire retrouver leur état premier ("en clair"), sans disposer des clefs théoriquement nécessaires.
- **Clef** : Dans un système de chiffrement, elle correspond à un nombre, un mot, une phrase, etc. qui permet, grâce à l'algorithme de chiffrement, de chiffrer ou de déchiffrer un message.

Il est à noter que les termes « **cryptage** » et « **crypter** » sont des anglicismes, dérivés de l'anglais **to encrypt**, souvent employés incorrectement à la place de chiffrement et chiffrer.

2.3 L'usage de la cryptographie

La cryptographie est traditionnellement utilisée pour dissimuler des messages aux yeux de certains utilisateurs. Cette utilisation a aujourd'hui un intérêt d'autant plus grand que les communications via internet circulent dans des infrastructures dont on ne peut garantir la fiabilité et la confidentialité. Désormais, la cryptographie sert non seulement à préserver la confidentialité des données mais aussi à garantir leur intégrité et leur authenticité.

- La confidentialité : consiste à rendre l'information intelligible à d'autres personnes que les acteurs de la transaction.
- L'intégrité consiste à déterminer si les données n'ont pas été altérées durant la communication.
- L'authentification : consiste à assurer l'identité d'un utilisateur, c.-à-d de garantir à chacun des correspondants que son partenaire est bien celui qu'il croit être. Un contrôle d'accès peut permettre (par exemple par le moyen d'un mot de passe qui devra être crypté) l'accès à des ressources uniquement aux personnes autorisées.
- La non répudiation : de l'information est la garantie qu'aucun des correspondants ne pourra nier la transaction.

2.4 Mécanisme de la cryptographie

Un algorithme de cryptographie ou un chiffrement est une fonction mathématique utilisée lors du processus de cryptage et de décryptage. Cet algorithme est associé à une clé(un mot, un nombre ou une phrase), afin de crypter une donnée. Avec des clés différentes, le résultat du cryptage variera également. La sécurité des données cryptées repose entièrement sur deux éléments : l'invulnérabilité de l'algorithme de cryptographie et la confidentialité de la clé. Les clés doivent être stockées de manière sécurisée et de manière à ce que seul leur propriétaire soit en mesure de les atteindre et de les utiliser.

2.5 Algorithmes de chiffrement à clef :

La confidentialité est le premier problème posé à la cryptographie. Il se résout par la notion de chiffrement. Il existe deux grandes familles d'algorithmes cryptographiques à base de clef : Les algorithmes à clef secrète ou algorithmes symétriques, et les algorithmes à clef publique ou algorithmes asymétriques.

2.5.1 Chiffrement symétrique :

Autrement appelée cryptage à clé privée, ce type se base sur l'utilisation d'une clé pour crypter et décrypter les messages(figure 2.1). La sécurité de cette solution repose sur le fait que la clé est connue uniquement par l'émetteur et le récepteur du message.
Pour une meilleure sécurité, elle doit être détruite et réinitialisée après chaque conversation[7].
La figure 2.1 présente le fonctionnement de Chiffrement symétrique .

Le cryptage symétrique fonctionne selon deux procédés différents :

• le cryptage par flot : le cryptage s'effectue en continu, bit par bit
• le cryptage par bloc : le cryptage s'effectue sur les blocs de bits

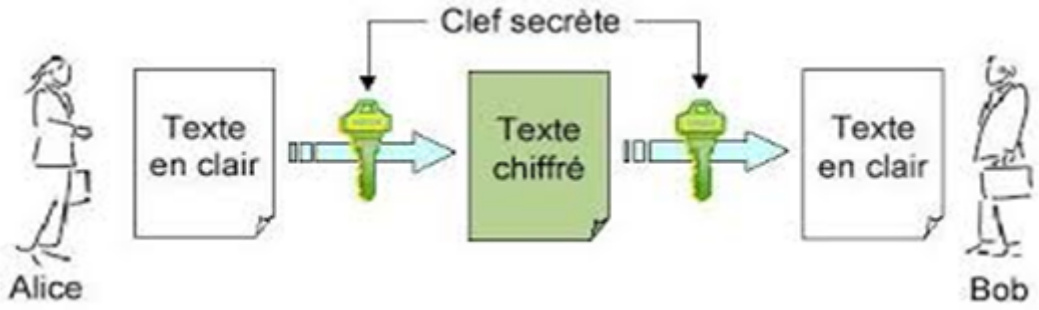


FIGURE 2.1 – Schéma de cryptage symétrique .

Les avantages du cryptage symétrique :
• la rapidité d'exécution (une seule clé utilisée).
• la simplicité d'implémentation (gestion d'une seule clé).
• Permet de concevoir différents mécanismes cryptographiques (fonctions de hachage, etc...)
• Clés relativement courtes.

Les inconvénients du cryptage symétrique :
• la complexité de fonctionnement : une obligation d'avoir le nombre de clés privées égal au nombre de destinataires.
• la sécurisation de la chaîne de transmission de la clé.
• Impossibilité de garantir la propriété de non-répudiation dans les schémas de signature électronique[7].

2.5.2 Chiffrement asymétrique

La cryptographie asymétrique(figure 2.2), ou cryptographie à clé publique, est une méthode de chiffrement qui s'oppose à la cryptographie symétrique. Elle repose sur l'utilisation d'une clé publique (qui est diffusée) et d'une clé privée (gardée secrète), l'une permettant de coder le message et l'autre de le décoder. Ainsi, l'expéditeur peut utiliser la clé publique du destinataire pour coder un message que seul le destinataire (en possession de la clé privée) peut décoder, garantissant la confidentialité du contenu. Inversement, l'expéditeur peut utiliser sa propre clé privée pour coder un message que le destinataire peut décoder avec la clé publique ; c'est le mécanisme utilisé par la signature numérique pour authentifier l'auteur d'un message[7].
La figure 2.2 présente le fonctionnement de Chiffrement asymétrique.

Principe :

La cryptographie asymétrique, ou cryptographie à clé publique est fondée sur l'existence de fonctions à sens unique - une fois la fonction appliquée à un message, il est extrêmement difficile de retrouver le message original.

En réalité, on utilise en cryptographie asymétrique des fonctions à sens unique et à brèche secrète. Une telle fonction est difficile à inverser, à moins de posséder une information particulière, tenue secrète, nommée clé privée.

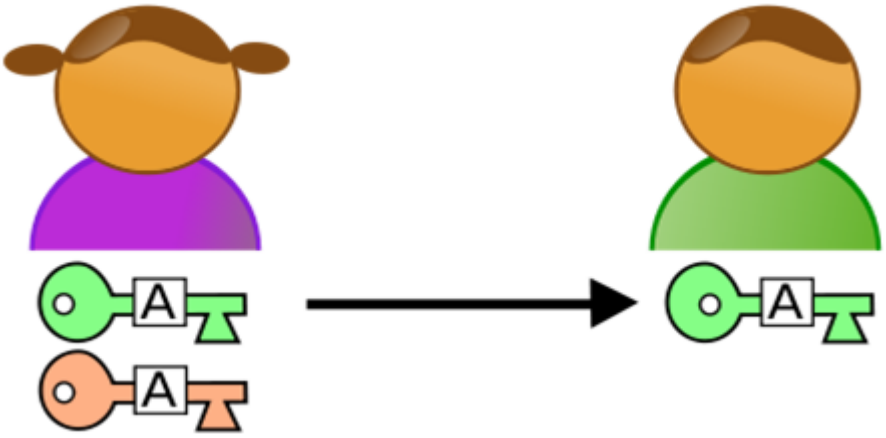


FIGURE 2.2 – cryptage asymétrique .

À partir d'une telle fonction, voici comment se déroulent les choses(figure 2.3 : Alice souhaite pouvoir recevoir des messages chiffrés de n'importe qui. Elle génère alors une valeur à partir d'une fonction à sens unique et à brèche secrète à l'aide d'un algorithme de chiffrement asymétrique ,par exemple RSA.
La figure 2.3 présente exemple de fonctionnement cryptage asymétrique.

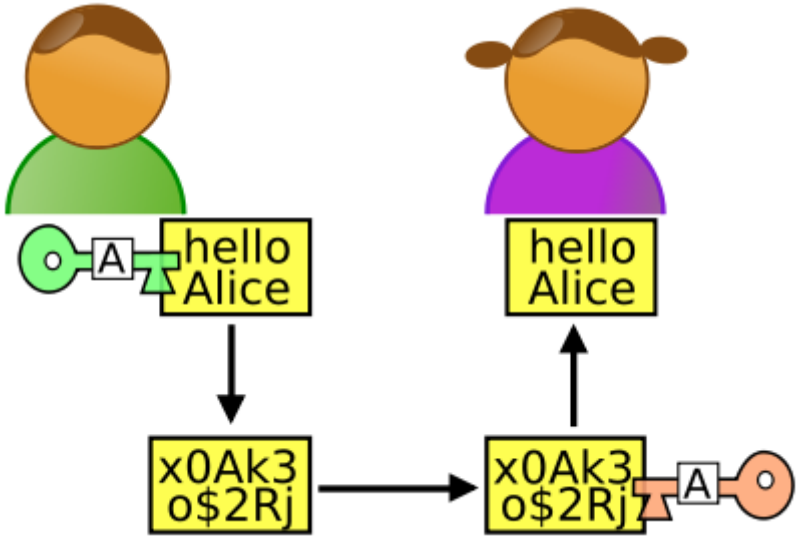


FIGURE 2.3 – cryptage symétrique .

Alice diffuse à tout le monde la fonction pour coder les messages (notée clé publique) mais garde secrète la fonction de décodage (notée clé privée).

1ér étape : Alice génère deux clés. La clé publique (verte) qu'elle envoie à Bob et la clé privée (rouge) qu'elle conserve précieusement sans la divulguer à quiconque.

2^{éme} et 3^{éme} étapes : Bob chiffre le message avec la clé publique d'Alice et envoie le

texte chiffré. Alice déchiffre le message grâce à sa clé privée.

2.5.3 Chiffrement :

Un des rôles de la clé publique est de permettre le chiffrement ; c'est donc cette clé qu'utilisera Bob pour envoyer des messages chiffrés à Alice. L'autre clé - l'information secrète - sert à déchiffrer. Ainsi, Alice, et elle seule, peut prendre connaissance des messages de Bob. La connaissance d'une clé ne permet pas de déduire l'autre[7].

Authentification de l'origine :
D'autre part, l'utilisation par Alice de sa clé privée sur le condensat d'un message, permettra à Bob de vérifier que le message provient bien d'Alice : il appliquera la clé publique d'Alice au condensat fourni (condensat chiffré avec la clé privée d'Alice) et retrouve donc le condensat original du message. Il lui suffira de comparer le condensat ainsi obtenu et le condensat réel du message pour savoir si Alice est bien l'expéditeur. C'est donc ainsi que Bob sera rassuré sur l'origine du message reçu : il appartient bien à Alice. C'est sur ce mécanisme notamment que fonctionne la signature numérique[7].

Faiblesses :
Comme dans le cadre de la cryptographie symétrique, les faiblesses peuvent tout d'abord venir des faiblesses de l'algorithme de chiffrement (voir Cryptanalyse).
Le risque principal dans l'utilisation des clés asymétriques est celui de l'attaque de l'homme du milieu, c'est-à-dire la possibilité qu'une partie adverse intercepte les clés publiques échangées pour les remplacer par les siennes. Il pourrait alors déchiffrer et signer tous les messages échangés. La résolution de cette faiblesse nécessite une infrastructure à clés publiques afin de certifier que les clé publiques appartiennent bien aux parties.[7].

2.5.4 Algorithme RSA

Le premier système à clé publique solide à avoir été inventé, et le plus utilisé actuellement, est le système RSA. Publié en 1977 par Ron Rivest, Adi Shamir et Leonard Adleman de l'Institut de technologie du Massachusetts (MIT), le RSA est fondé sur la difficulté de factoriser des grands nombres, et la fonction à sens unique utilisée est une fonction "puissance"[14].

L'algorithme de chiffrement

Départ :
Il est facile de fabriquer de grands nombres premiers p et q (+- 100 chiffres) Etant donné un nombre entier n = pq, il est très difficile de retrouver les facteurs p et q **(1) Création des clés :**
La clé secrète : 2 grands nombres premiers p et q
La clé publique : n = pq ; un entier e premier avec (p-1)(q-1)
(2) Chiffrement :
le chiffrement d'un message M en un message codé C se fait suivant la transformation suivante : $C = M^e \bmod n$

(3) Déchiffrement :
il s'agit de calculer la fonction réciproque
$M = C^d \bmod n$
tel que $e.d = 1 \bmod [(p-1)(q-1)]$

La signature électronique :

Après la confidentialité de la transmission d'un message subsiste un problème : son authenticité. Alice voudrait bien envoyer un message M à Bob de telle façon que celui-ci soit sûr qu'elle est réellement l'émettrice du message, et qu'un intrus ne tente pas de venir semer la confusion.
Le système RSA fournit une solution à ce problème :
Rappelons les données :
Alice seule détient la clé secrète d et diffuse la clé publique (n,e)
Alice va se servir de la clé publique pour chiffrer le message M
(1) Alice accompagne son message chiffré de sa signature, qui correspond à : M^d
(2) Bob va donc voir si l'égalité $(M^d)^e \bmod n = M$ est vérifiée. Si c'est le cas, Alice est bien l'émettrice du message.

Sécurité du système : primalité, factorisation

Comme pour les protocoles fondés sur le logarithme discret, la sécurité du système RSA est calculatoire : elle dépend essentiellement de la difficulté de factoriser un entier qui est le produit de deux grands nombres premiers. Si on sait factoriser n, il est facile de trouver d. Il est à noter qu'il est conseiller d'utiliser des clés de 1024 bits (+- 309 chiffres décimaux).
Le principal objet des attaques est l'implémentation, la mise en oeuvre du RSA. C'est la manière de se servir du système qui peut poser problème, pas le système lui-même (par exemple, prendre une clé trop petite...).
Signalons enfin que le réel problème du RSA (et des autres systèmes à clé publique) n'est pas la sécurité, mais la lenteur. Tous les algorithmes à clé publique sont 100 à 1000 fois plus lents que les algorithmes à clé secrète, quelle que soit leur implémentation (logicielle ou matérielle) ![14].

Exemple : chiffrer BONJOUR

1) Alice crée ses clés :
La clé secrète : p = 53 , q = 97 (Note : en réalité, p et q devraient comporter plus de 100 chiffres !)
La clé publique : e = 7 (premier avec 52*96), n = 53*97 = 5141
2) Alice diffuse sa clé publique (par exemple, dans un annuaire).
3) Bob ayant trouvé le couple (n,e), il sait qu'il doit l'utiliser pour chiffrer son message. Il va tout d'abord remplacer chaque lettre du mot BONJOUR par le nombre correspondant à sa position dans l'alphabet :
B = 2, O = 15, N = 14, J = 10, U = 21, R = 18
BONJOUR = 2 15 14 10 15 21 18

4) Ensuite, Bob découpe son message chiffré en blocs de même longueur représentant chacun un nombre plus petit que n. Cette opération est essentielle, car si on ne faisait pas des blocs assez longs (par exemple, si on laissait des blocs de 2 chiffres), on retomberait sur un simple chiffre de substitution que l'on pourrait attaquer par **l'analyse des fréquences**
BONJOUR = 002 151 410 152 118

5) Bob chiffre chacun des blocs que l'on note B par la transformation C = $B^e \bmod n$ (où C est le bloc chiffré) :
$C_1 = 2^7 \bmod 5141 = 128$
$C_2 = 151^7 \bmod 5141 = 800$
$C_3 = 410^7 \bmod 5141 = 3761$
$C_4 = 152^7 \bmod 5141 = 660$
$C_5 = 118^7 \bmod 5141 = 204$

On obtient donc le message chiffré C : 128 800 3761 660 204.

2.6 Le chiffrement à clé secrète :

Par opposition au chiffrement à clé publique, le chiffrement à clef secrète est aussi appelé chiffrement symétrique. La clé de chiffrement peut être calculée à partir de la clé de déchiffrement et vice versa. En général, les clés de chiffrement et de déchiffrement sont identiques. L'émetteur et le destinataire doivent se mettre d'accord préalablement sur une clé qui doit être gardée secrète, car la sécurité d'un tel algorithme repose sur cette clé.

Les schémas de chiffrement symétrique peuvent être classés en deux catégories, le chiffrement par blocs et le chiffrement par flot[27].

Les schémas de chiffrement par flot et appelé aussi chiffrement en continu, traitent l'information bit à bit, et sont très rapides. Ils sont parfaitement adaptés à des moyens de calcul et de mémoire (cryptographie en temps réel) comme la cryptographie militaire, ou la cryptographie entre le téléphone portable GSM et son réseau. Leur principe est d'effectuer un chiffrement de Vernam en utilisant une clé pseudo-aléatoire, c'est à dire une clé qui ne soit pas choisie aléatoirement parmi tous les mots binaires de longueur n. Cette clé (qu'on appellera suite pseudo-aléatoire) est générée par différents procédés à partir d'une clé secrète d'une longueur juste suffisante pour résister aux attaques exhaustives.

Dans un schéma de chiffrement par blocs, le message est divisé en blocs de bits, de longueur fixe. Chaque bloc est chiffré l'un après l'autre. Le chiffrement peut être effectué par substitutions (les bits d'un bloc sont substitués par d'autres bits) et par transpositions (les bits d'un bloc sont permutés entre eux). La substitution permet d'ajouter de la confusion, c'est-à-dire de rendre la relation entre le message et le texte chiffré aussi complexe que possible. La transposition permet d'ajouter de la diffusion, c'est-à-dire de réarranger les bits du message afin d'éviter que toute redondance dans le message ne se retrouve dans le texte chiffré

On distingue le chiffrement par blocs itératifs. Une fonction constituée de combinaisons complexes de substitutions et/ou de transpositions, appelée fonction de tour ou fonction de ronde, est appliquée itérativement. Une itération est appelée un tour ou une ronde. Chaque ronde prend en entrée la sortie de la ronde précédente et chiffre cette entrée à l'aide de la fonction de ronde et d'une sous-clé de ronde générée à partir de la clé secrète K. La fonction de chiffrement n'est pas la fonction de ronde, mais elle est constituée par

l'ensemble de toutes les rondes.

Le 15 mai 1973 le NBS (National Bureau of Standards, aujourd'hui appelé NIST - National Institute of Standards and Technology) a lancé un appel dans le Federal Register (l'équivalent aux Etats-Unix du Journal Officiel en France) pour la création d'un algorithme de chiffrement répondant aux critères suivants :

• posséder un haut niveau de sécurité lié à une clé de petite taille servant au chiffrement et au déchiffrement
• être compréhensible
• ne pas dépendre de la confidentialité de l'algorithme
• être adaptable et économique
• être efficace et exportable

Fin 1974, IBM propose " Lucifer ", qui, grâce à la NSA (National Security Agency), est modifié le 23 novembre 1976 pour donner le DES (Data Encryption Standard). Le DES a finalement été approuvé en 1978 par le NBS. Le DES fut normalisé par l'ANSI (American National Standard Institute) sous le nom de ANSI X3.92, plus connu sous la dénomination DEA (Data Encryption Algorithm)[16].

2.6.1 Principe du DES

Il s'agit d'un système de chiffrement symétrique par blocs de 64 bits, dont 8 bits (un octet) servent de test de parité (pour vérifier l'intégrité de la clé). Chaque bit de parité de la clé (1 tous les 8 bits) sert à tester un des octets de la clé par parité impaire, c'est-à-dire que chacun des bits de parité est ajusté de façon à avoir un nombre impair de '1' dans l'octet à qui il appartient. La clé possède donc une longueur " utile " de 56 bits, ce qui signifie que seuls 56 bits servent réellement dans l'algorithme.

L'algorithme consiste à effectuer des combinaisons, des substitutions et des permutations entre le texte à chiffrer et la clé, en faisant en sorte que les opérations puissent se faire dans les deux sens (pour le déchiffrement). La combinaison entre substitutions et permutations est appelée code produit .

La clé est codée sur 64 bits et formée de 16 blocs de 4 bits, généralement notés k_1 à k_16. Etant donné que " seuls " 56 bits servent effectivement à chiffrer, il peut exister 2^56 (soit $7.2*10^16$) clés différentes ![16].

2.6.2 L'algorithme du DES :

Les grandes lignes de l'algorithme sont les suivantes(figure 2.4) :

- Fractionnement du texte en blocs de 64 bits (8 octets) ;
- Permutation initiale des blocs ;
- Découpage des blocs en deux parties : gauche et droite, nommées G et D ;
- Etapes de permutation et de substitution répétées 16 fois (appelées rondes) ;
- Recollement des parties gauche et droite puis permutation initiale inverse.

La figure 2.4 présente Schéma de cryptage de l'algorithime DES .

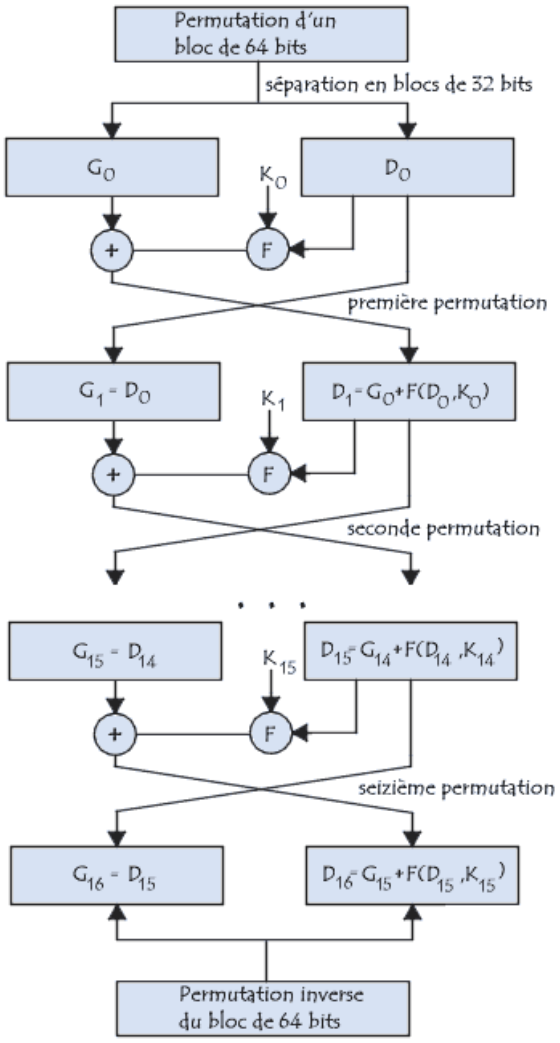


FIGURE 2.4 – Schéma de cryptage DES.

Fractionnement du texte :

Permutation initiale :

Dans un premier temps, chaque bit d'un bloc est soumis à la permutation initiale, pouvant être représentée par la matrice de permutation initiale (notée PI) suivante :

PI								
	58	50	42	34	26	18	10	2
	60	52	44	36	28	20	12	4
	62	54	46	38	30	22	14	6
	64	56	48	40	32	24	16	8
	57	49	41	33	25	17	9	1
	59	51	43	35	27	19	11	3
	61	53	45	37	29	21	13	5
	63	55	47	39	31	23	15	7

Cette matrice de permutation indique, en parcourant la matrice de gauche à droite puis de haut en bas, que le 58ème bit du bloc de texte de 64 bits se retrouve en première position, le 50ème en seconde position et ainsi de suite.

Scindement en blocs de 32 bits :

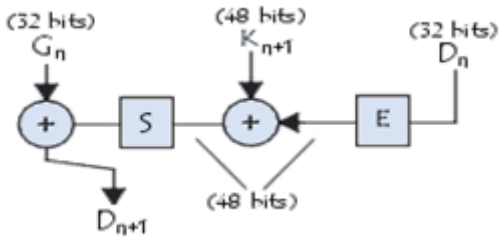
Une fois la permutation initiale réalisée, le bloc de 64 bits est scindé en deux blocs de 32 bits, notés respectivement G et D (pour gauche et droite, la notation anglo-saxonne étant L et R pour Left and Right). On note G_0 et D_0 l'état initial de ces deux blocs :

G_0								
	58	50	42	34	26	18	10	2
	60	52	44	36	28	20	12	4
	62	54	46	38	30	22	14	6
	64	56	48	40	32	24	16	8

D_0								
	57	49	41	33	25	17	9	1
	59	51	43	35	27	19	11	3
	61	53	45	37	29	21	13	5
	63	55	47	39	31	23	15	7

Il est intéressant de remarquer que G_0 contient tous les bits possédant une position paire dans le message initial, tandis que D_0 contient les bits de position impaire.

Rondes Les blocs Gn et Dn sont soumis à un ensemble de transformation itératives appelées rondes, explicitées dans ce schéma, et dont les détails sont donnés plus bas :



Fonction d'expansion :

Les 32 bits du bloc D_0 sont étendus à 48 bits grâce à une table (matrice) appelé table d'expansion(notée E), dans laquelle les 48 bits sont mélangés et 16 d'entre eux sont dupliqués :

E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Ainsi, le dernier bit de D_0 (c'est-à-dire le 7ème bit du bloc d'origine) devient le premier, le premier devient le second, De plus, les bits 1,4,5,8,9,12,13,16,17,20,21,24,25,28 et 29 de D_0 (respectivement 57, 33, 25, l, 59, 35, 27, 3, 6l, 37, 29, 5, 63, 39, 31 et 7 du bloc d'origine) sont dupliqués et disséminés dans la matrice.

OU exclusif avec la clé :

La matrice résultante de 48 bits est appelée D'_0 ou bien E[D_0]. L'algorithme DES procède ensuite à un OU exclusif entre la première clé K1 et E[D_0]. Le résultat de ce OU exclusif est une matrice de 48 bits que nous appelerons D_0 par commodité (il ne s'agit pas du D_0 de départ !).

Fonction de substitution :

D_0 est ensuite scindé en 8 blocs de 6 bits, noté D_0i. Chacun de ces blocs passe par des fonctions de sélection (appelées parfois boîtes de substitution ou fonctions de compression), notées généralement Si. Les premiers et derniers bits de chaque D_0i détermine (en binaire) la ligne de la fonction de sélection, les autres bits (respectivement 2, 3, 4 et 5) déterminent la colonne. La sélection de la ligne se faisant sur deux bits, il y a 4 possibilités (0,1,2,3). La sélection de la colonne se faisant sur 4 bits, il y a 16 possibilités (0 à 15). Grâce à cette information, la fonction de sélection "sélectionne" une valeur codée sur 4 bits.
Voici la première fonction de substitution, représentée par une matrice de 4 par 16 :

S_1

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Soit $D_0$1 égal à 101110. Les premiers et derniers bits donnent 10, c'est-à-dire 2 en binaire. Les bits 2,3,4 et 5 donnent 0111, soit 7 en binaire. Le résultat de la fonction de sélection est donc la valeur situé à la ligne n°2, dans la colonne n°7. Il s'agit de la valeur 11, soit en binaire 111.
Chacun des 8 blocs de 6 bits est passé dans la fonction de sélection correspondante, ce qui donne en sortie 8 valeurs de 4 bits chacune. Voici les autres fonctions de sélection :

S_2

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5

0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
1	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
1	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Chaque bloc de 6 bits est ainsi substitué en un bloc de 4 bits. Ces bits sont regroupés pour former un bloc de 32 bits.

permutation

Le bloc de 32 bits obtenu est enfin soumis à une permutation P dont voici la table :

P

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

OU Exclusif L'ensemble de ces résultats en sortie de P est soumis à un OU Exclusif avec le G_0 de départ (comme indiqué sur le premier schéma) pour donner D_1, tandis que le D_0 initial donne G_1.

Itération

L'ensemble des étapes précédentes (rondes) est réitéré 16 fois.

Permutation initiale inverse

A la fin des itérations, les deux blocs G_{16} et D_{16} sont "recollés, puis soumis à la permutation initiale inverse :

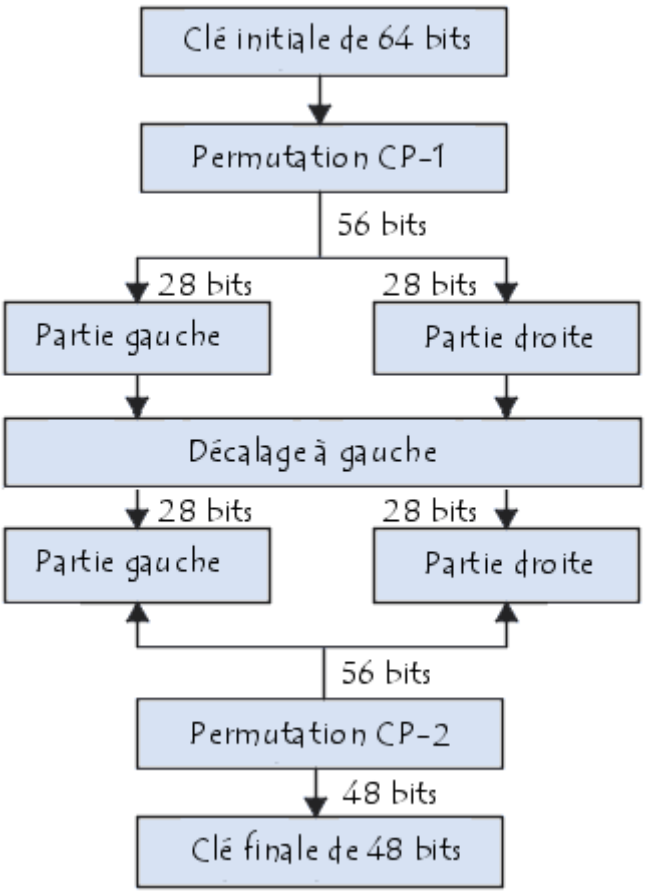
PI-1								
	40	8	48	16	56	24	64	32
	39	7	47	15	55	23	63	31
	38	6	46	14	54	22	62	30
	37	5	45	13	53	21	61	29
	36	4	44	12	52	20	60	28
	35	3	43	11	51	19	59	27
	34	2	42	10	50	18	58	26
	33	1	41	9	49	17	57	25

Le résultat en sortie est un texte codé de 64 bits !

Génération des clés :

Etant donné que l'algorithme du DES présenté ci-dessus est public, toute la sécurité repose sur la complexité des clés de chiffrement.

L'algorithme ci-dessous montre comment obtenir à partir d'une clé de 64 bits (composé de 64 caractères alphanumériques quelconques) 8 clés diversifiées de 48 bits chacune servant dans l'algorithme du DES :



Dans un premier temps les bits de parité de la clé sont éliminés afin d'obtenir une clé d'une longueur utile de 56 bits.

La première étape consiste en une permutation notée CP-1 dont la matrice est présentée ci-dessous :

CP-1

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

Cette matrice peut en fait s'écrire sous la forme de deux matrice Gi et Di (pour gauche et droite) composées chacune de 28 bits :

G_i							
	57	49	41	33	25	17	9
	1	58	50	42	34	26	18
	10	2	59	51	43	35	27
	19	11	3	60	52	44	36

D_i							
	63	55	47	39	31	23	15
	7	62	54	46	38	30	22
	14	6	61	53	45	37	29
	21	13	5	28	20	12	4

On note G_0 et D_0 le résultat de cette première permutation.
Ces deux blocs subissent ensuite une rotation à gauche, de telles façons que les bits en seconde position prennent la première position, ceux en troisième position la seconde, ...
Les bits en première position passent en dernière position.
Les 2 blocs de 28 bits sont ensuite regroupés en un bloc de 56 bits. Celui-ci passe par une permutation, notée CP-2, fournissant en sortie un bloc de 48 bits, représentant la clé K_i.

CP-2												
	14	17	11	24	1	5	3	28	15	6	21	10
	23	19	12	4	26	8	16	7	27	20	13	2
	41	52	31	37	47	55	30	40	51	45	33	48
	44	49	39	56	34	53	46	42	50	36	29	32

Des itérations de l'algorithme permettent de donner les 16 clés K_1 à K_{16} utilisées dans l'algortihme du DES.

LS															
	1	2	4	6	8	10	12	14	15	17	19	21	23	25	27	28

2.7 Algorithme AES :

2.7.1 Présentation générale de l'algorithme AES

L'AES (Advanced Encryption Standard) est, comme son nom l'indique, un standard de cryptage symétrique destiné à remplacer le DES (Data Encryption Standard) qui est devenu trop faible au regard des attaques actuelles. Historiquement, le développement de l'AES a été instigué par le NIST (National Institue of Standards and Technology). Il est également approuvé par la NSA (National Security Agency) pour l'encryptions des informations dites très sensibles.[15].

Cet algorithme suit les spécifications suivantes :
• L'AES est un standard, donc libre d'utilisation, sans restriction d'usage ni brevet.
• C'est un algorithme de type symétrique
• C'est un algorithme de chiffrement par blocs

Il supporte différentes combinaisons [longueur de clé]-[longueur de bloc] : 128-128, 192-128 et 256-128 bits (en fait, l'AES supporte également des tailles de blocs variables, mais cela n'est pas retenu dans le standard)[15].

• En termes décimaux, ces différentes tailles possibles signifient concrètement que :
• 3.4 x 1038 clés de 128-bit possibles
• 6.2 x 1057 clés de 192-bit possibles 1.1 x 1077 clés de 256-bit possibles

Pour avoir un ordre d'idée, les clés DES ont une longueur de 56 bits (64 bits au total dont 8 pour les contrôles de parité), ce qui signifie qu'il y a approximativement 7.2 x 1016 clés différentes possibles. Cela nous donne un ordre de 1021 fois plus de clés 128 bits pour l'AES que de clés 56 bits pour le DES. En supposant que l'on puisse construire une machine qui pourrait cracker une clé DES en une seconde (donc qui puisse calculer 255 clés par seconde), alors cela prendrait environ 149 mille milliards d'années pour cracker une clé AES[15].

2.7.2 Caractéristiques et points forts de l'AES :

Le choix de cet algorithme répond à de nombreux critères plus généraux dont nous pouvons citer les suivants :
• Sécurité ou l'effort nécessaire pour une éventuelle cryptanalyse
• Puissance de calcul qui entraine une grande rapidité de traitement
• Besoins en ressources et mémoire très faibles
• Flexibilité d'implémentation, cela inclut une grande variété de plateformes et d'applications ainsi que des tailles de clés et de blocs supplémentaires Compatibilité hardware et software, il est possible d'implémenter l'AES aussi bien sous forme logicielle que matérielle
• Simplicité, le design de l'AES est relativement simple

Si l'on se réfère à ces critères, on constate que l'AES est également un candidat particulièrement approprié pour les implémentations embarquées qui suivent des règles beaucoup plus strictes en matière de ressources, puissance de calcul, taille mémoire, etc... C'est sans doute cela qui a poussé le monde de la 3G (3ème génération de mobiles) à adopter cet

algorithme pour son schéma d'authentification[15].

2.7.3 Principe de fonctionnement de l'AES :

L'AES procède par blocs de 128 bits, avec une clé de 128 bits également. Chaque bloc subit une séquence de 5 transformations répétées 10 fois(voir figure 2.5)[15] :
La figure 2.5 présente Schéma fonctionnement de l'algorithme AES.

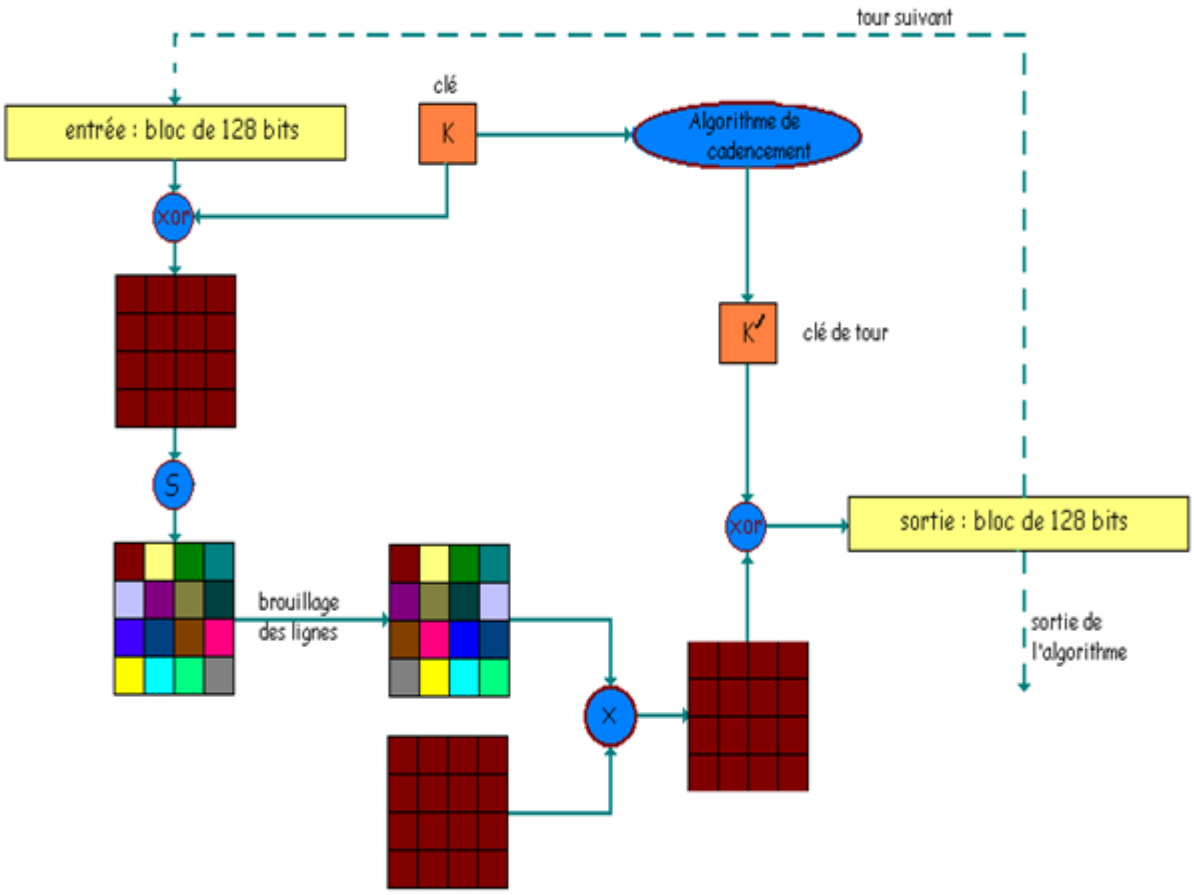


FIGURE 2.5 – Schéma fonctionnement de l'algorithme AES.

• Addition de la clé secrète (par un ou exclusif).

• Transformation non linéaire d'octets : les 128 bits sont répartis en 16 blocs de 8 bits (8 bits=un octet), eux-même dispatchés dans un tableau 4×4. Chaque octet est transformé par une fonction non linéaire S. S peut être simplement vue comme une substitution sur les entiers compris entre 1 et 256. En particulier, elle peut être implantée sur ordinateur par un simple tableau.

• Décalage de lignes : les 3 dernières lignes sont décalées cycliquement vers la gauche : la 2ème ligne est décalée d'une colonne, la 3ème ligne de 2 colonnes, et la 4ème ligne de 3 colonnes.

• Brouillage des colonnes : Chaque colonne est transformée par combinaisons linéaires des différents éléments de la colonne (ce qui revient à multiplier la matrice 4×4 par une autre matrice 4×4). Les calculs sur les octets de 8 bits sont réalisés dans le corps à 2^8 éléments.

• Addition de la clé de tour : A chaque tour, une clé de tour est générée à partir de la clé secrète par un sous-algorithme (dit de cadencement). Cette clé de tour est ajoutée par un ou exclusif au dernier bloc obtenu.

2.8 Conclusion

La cryptographie étant un sujet très vaste, nous nous sommes contenté de présenter ses principaux concepts et ses algorithmes les plus connus et utilisés en occurrences le DES et de l'AES, en passant par le fameux RSA. A ce bref survol des principaux concepts et bases de fonctionnement des applications de messagerie électronique d'une part et ceux de la cryptographie d'autre part, nous somme, enfin, prés à entamer les phases de conception et de réalisation de notre messagerie sécurisée.

3 Conception

3.1 Introduction

La phase de conception est la première étape dans la réalisation d'un projet, elle doit d'écrire de manière non ambiguë le fonctionnement futur du système, afin d'en faciliter la réalisation. Pour cela, différentes méthodes existent permettant de formaliser les étapes préliminaires du développement. Dans ce chapitre, nous présentons les objectifs de notre application, ce qui nous amène à identifier les possibilités du système et les besoins des utilisateurs que nous essayons de projeter dans des diagrammes de cas d'utilisations globaux et détaillés.

3.2 Spécification des besoins

L'application envisagée doit satisfaire les besoins fonctionnels qui seront exécutés par le système et les besoins non fonctionnels qui perfectionnent la qualité logicielle du système[1].

Les besoins fonctionnels

Les besoins fonctionnels ou besoins métiers représentent les actions que le système doit exécuter, il ne devient opérationnel que s'il les satisfait. Cette application doit couvrir principalement les besoins fonctionnels suivants[1] :

- Enregistrement des utilisateurs.
- Authentification des utilisateurs.
- Envoyer et recevoir des messages textes.
- Exécute un service de fond afin d'obtenir des messages, même lorsque l'application est fermée.
- Utiliser une zone de notification lorsqu'un nouveau message ou une nouvelle invitation est reçue.
- Gestion d'amis (Ajout des amis par nom d'utilisateur, affichage de la liste des amis avec leurs statut(en ligne/hors ligne), accepter ou refuser des invitations.
- Se déconnecter (quitter l'application).
- Sécurité d'échange des messages

Les besoins non fonctionnels

Ce sont des exigences qui ne concernent pas spécifiquement le comportement du système mais plutôt identifient des contraintes internes et externes du système. Les principaux besoins non fonctionnels de notre application se résument comme suit :

- Le code doit être clair pour permettre des futures évolutions ou améliorations.
- L'ergonomie : l'application offre une interface conviviale et facile à utiliser.
- La sécurité : l'application doit respecter la confidentialité des données.
- Garantir l'intégrité et la cohérence des données à chaque mise à jour et à chaque insertion.

3.3 Le langage UML

UML (Unfied Modeling Language), se définit comme un langage de modélisation graphique et textuel destiné à comprendre et à définir des besoins, spécifique et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue. UML modélise l'ensemble des données et des traitements en élaborant des différents diagrammes[2].
En clair, il ne faut pas designer UML en tant que méthode (Il y manque la démarche) mais plutôt comme une boite d'outils qui sert à améliorer les méthodes de travail.

3.3.1 L'intérêt d'UML

UML est un langage semi-formel et normalisé[2] :

Gain de précision.

Gagne de stabilité.

Encourage l'utilisation d'outils.

UML est un support de communication performant :

Il cadre l'analyse

Il facilite la compréhension de représentations abstraites complexes.

Son caractère polyvalent et sa souplesse en font un langage universel.

3.3.2 Définition d'un modèle

Un modèle est une vue subjective mais pertinente de la réalité. Un modèle dénit une frontière entre la réalité et la perspective de l'observateur. Ce n'est pas "la réalité", mais une vue très subjective de la réalité. Bien qu'un modèle ne représente pas une réalité absolue, un modèle reflète des aspects importants de la réalité, il en donne donc une vue juste et pertinente[2].

3.3.3 Les différents types diagrammes d'UML

La figure 3.1 représente les différentes catégories de diagrammes UML.

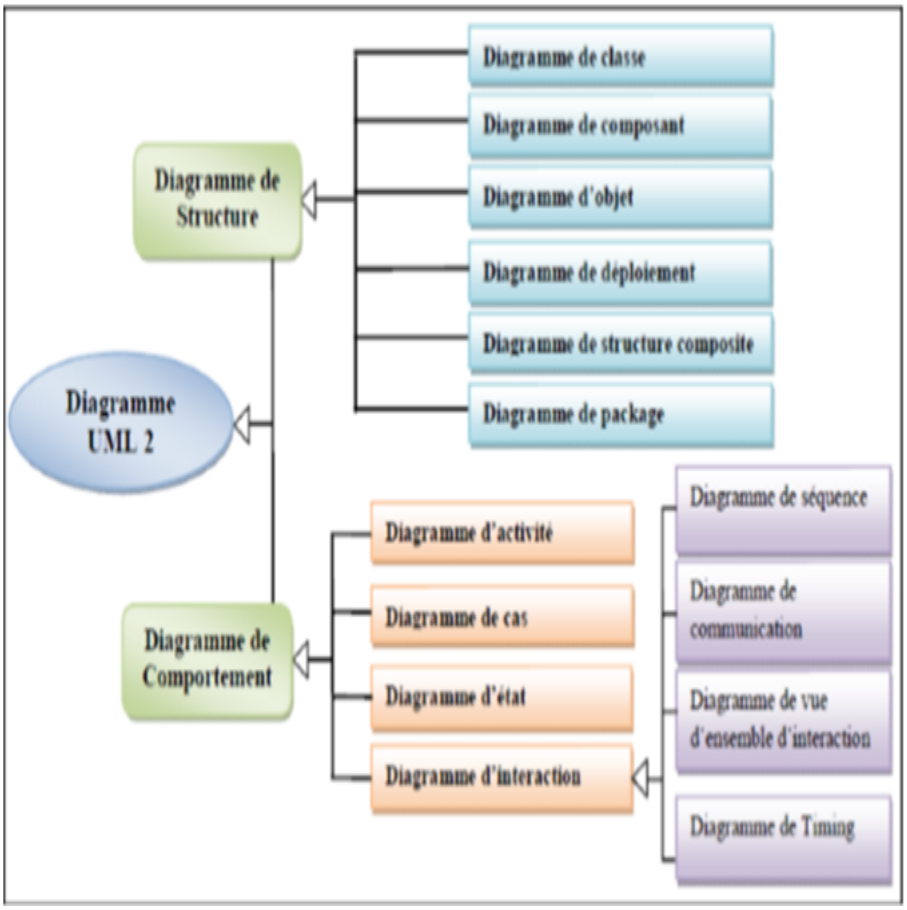


FIGURE 3.1 – les diagrammes UML.

3.3.4 Identification des acteurs

Un acteur représente un rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d'utilisation[2].

Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages susceptibles d'être porteurs de données.[2].

Dans notre cas, nous avons un acteur qui est l' Utilisateur ayant un accès au système via un contrôle d'accès (login et mot de passe). Les opérations qu'il peut effectuer sont :

- Envoyer des invitations par nom d'utilisateur.
- Afficher la liste des amis avec leurs statues (en ligne / hors ligne).
- Envoyer et recevoir des messages texte.
- Accepter ou refuser des invitations.
- Se déconnecter.

3.3.5 Diagramme de contexte du système à réaliser

La figure 3.2 ci-dessous montre l'interaction entre l'utilisateur et le système que nous allons mettre en place :

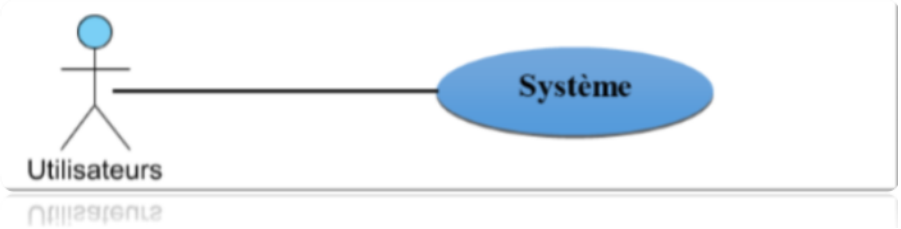


FIGURE 3.2 – Diagramme de contexte du système à réaliser.

3.3.6 Les Diagrammes des cas d'utilisation

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. C'est le premier diagramme du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre[3].

3.3.7 Outil de modilisation (StarUML) :

Le produit **StarUML**(figure 3.3) est un outil de modélisation UML (Unified Modeling Language) open source qui vient se substituer aux outils payants tels que " IBM Rational Rose " ou " Borland Together " et qui a fait l'objet de mon étude sur les logiciels de " conception - implémentation ", étude portant sur l'utilisation d'un logiciel ou d'un plug-in " Eclipse " open source pouvant faire de la modélisation UML ainsi que de l'implémentation Java (avec rétro-ingénierie).

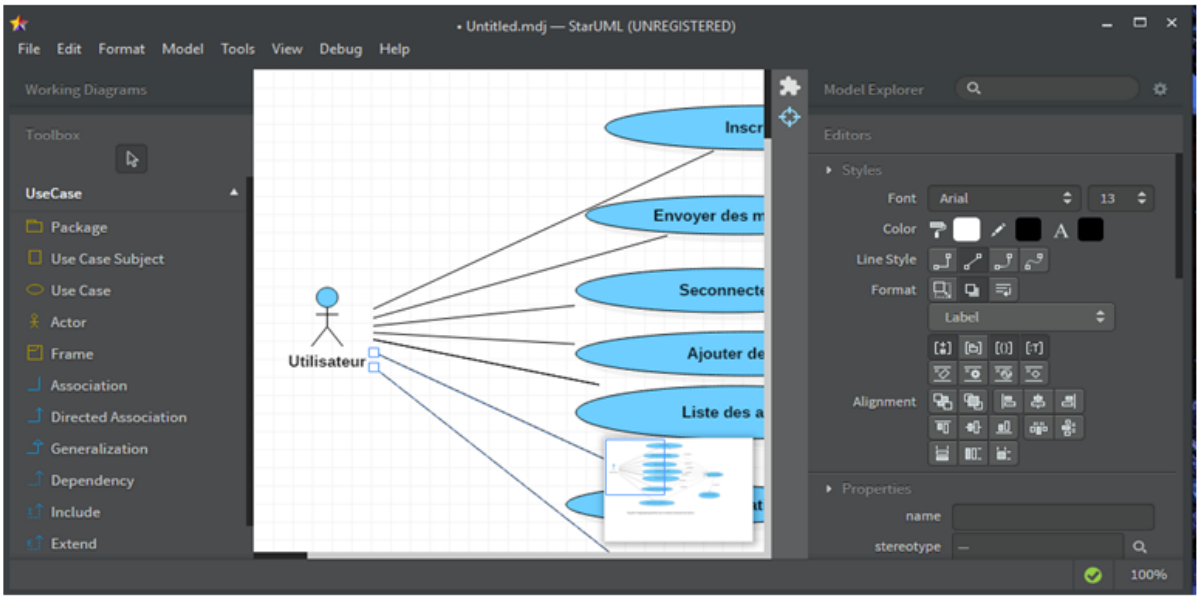


FIGURE 3.3 – StarUML Modélisation .

Ce logiciel open source est simple d'utilisation et intuitif pour un débutant. " StarUML " est simple d'installation et consomme peu de ressources système (mémoire 200 Mo et utilisation CPU faible). Il possède également une documentation (en anglais et en coréen -)

Cet outil respecte l'approche MDA (Model Driven Architecture) permettant de générer du code à partir de la modélisation UML (diagramme de classes). " StarUML " peut générer du code C, C++, Java et PHP 5 grâce aux divers plugins le composant.

Seule remarque, avant la génération du code, il faut au préalable avoir inclus le profil adéquat (ex : Java) dans les modèles. Pour cela, on utilise le menu " Model " puis " Profiles " puis on choisit le profil que l'on veut. Il existe un profil pour C, C++, Java et EJB (Enterprise JavaBeans).

" StarUML " peut également générer de façon automatique des diagrammes de classes grâce aux 26 patrons de conception qu'il connaît (3 patterns pour EJB et 23 patterns pour GOF (Gang of Four - ensemble de patrons de conception réutilisables, minimisation des interactions entre les diverses classes) présent dans le repository). Pour cela, il faut utiliser le menu " Apply Patterns " et choisir le pattern que l'on désire utiliser[17].

Le diagramme global des cas d'utilisations

La figure 3.4 présente le diagramme global des cas d'utilisations associés aux acteurs.

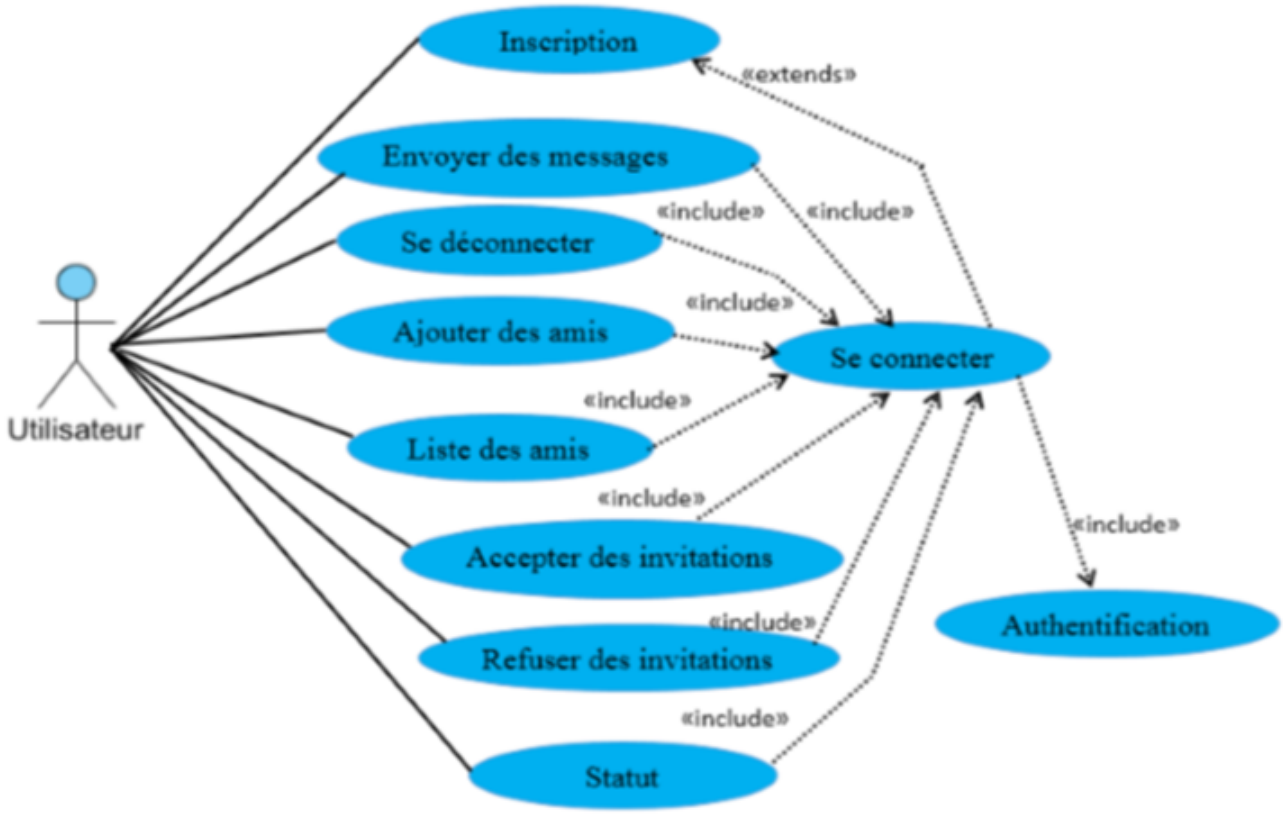


FIGURE 3.4 – Diagramme global des cas d'utilisations associés aux acteurs.

Les différents cas d'utilisation

L'étude de cas d'utilisation a pour objectif de déterminer ce que chaque utilisateur attend du système. La détermination du besoin est basée sur la représentation de l'interaction entre l'acteur et le système.

À chaque cas d'utilisation doit être associée une description textuelle des interactions entre l'acteur et le système et les actions que le système doit réaliser en vue de produire les résultats attendus par les acteurs. Pour exprimer les cas d'utilisations de notre système[3], nous avons choisi le formalisme de description décrit par le tableau représenté dans la figure 3.5.

Numéro du cas d'utilisation	Nom du cas d'utilisation.
Résumé	But du cas d'utilisation.
Acteur	Acteurs participants au cas d'utilisation.
Précondition	Condition qui doit être remplie avant le début du cas d'utilisation.
Scénario nominal	Séquence d'actions normales associées au cas d'utilisation.
Alternative **Exception**	Séquence d'actions alternatives pouvant conduire également à un succès. Séquences d'actions conduisant à un échec.

FIGURE 3.5 – formalisme de description des cas d'utilisation.

Le cas d'utilisation "Authentification"

La figure 3.6 présente un tableau descriptif du cas d'utilisation "Authentification" .

Cas d'utilisation N° 1	**Authentification**
Résumé	Vérification de l'identité des utilisateurs (Login et mot de passe).
Acteurs	Utilisateur.
Précondition	L'utilisateur doit avoir un compte.
Scénario nominal	[début] · Demande de connexion ; · Le système affiche le formulaire d'authentification (demande le nom d'utilisateur et le mot de passe ; · L'utilisateur saisit son login et son mot de passe puis valide ; · Le système vérifie la conformité des informations fournies **A1** ; · Le système donne l'accès à l'interface correspondante ; [fin]
Alternative A1	Dans le cas où les informations fournies sont incomplètes ou incorrectes le système réaffiche le formulaire d'authentification et attend que l'utilisateur ressaisisse ses informations ;

FIGURE 3.6 – Description du cas d'utilisation " Authentification ".

Remarque

A1 (alternative) est une étiquette utilisée dans le cas où les informations saisies sont incomplètes ou incorrectes pour que le système réaffiche le formulaire d'authentification.

Diagramme du cas d'utilisation "Authentification"

La figure 3.7 present le diagramme cas d'utilisation " Authentification .

FIGURE 3.7 – cas d'utilisation "Authentification" .

Le cas d'utilisation "Ajouter un ami"

La figure 3.8 présente la formalisme de description des cas d'utilisation "Ajouter un ami".

Cas d'utilisation N° 2	**Ajout un ami**
Résumé	L'utilisateur a le privilège d'ajouter des nouveaux amis.
Acteurs	Utilisateur.
Précondition	Authentification.
Scénario nominal	[début] · Authentification ; · L'utilisateur demande l'interface d'ajout d'un ami ; · Le système affiche l'interface ; · L'utilisateur saisie le pseudo de l'ami a inviter ; · Confirmer l'action ; [fin]

FIGURE 3.8 – Description du cas d'utilisation "Ajouter un ami".

Diagramme du cas d'utilisation "Ajouter un ami"

La figure 3.9 présente le diagramme cas d'utilisation "Ajouter un ami"

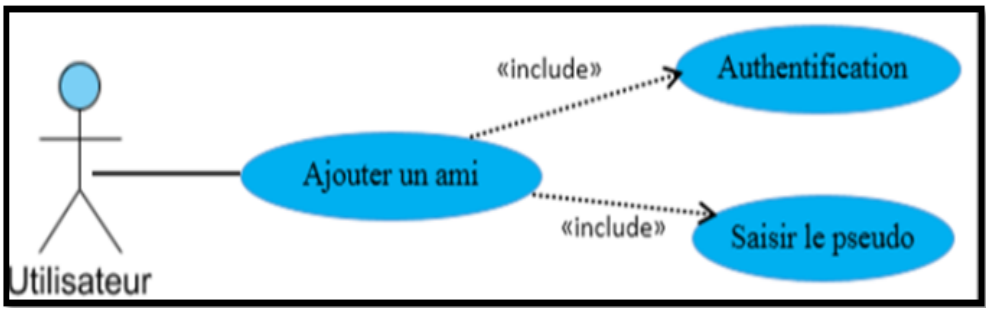


FIGURE 3.9 – Le cas d'utilisation "Ajouter un ami".

Le cas d'utilisation Inscription/Enregistrement

La figure 3.10 présente la formalisme de description des cas "Inscription/Enregistrement".

Cas d'utilisation N° 3	**Inscription / Enregistrement**
Résumé	Permet à l'utilisateur de s'inscrire.
Acteurs	Utilisateur.
Précondition	Aucune.
Scénario nominal	[début] · Demande de formulaire d'inscription ; · Le système affiche le formulaire d'inscription (demande le nom d'utilisateur (pseudonyme), email, mot de passe ; · L'utilisateur saisit ses informations puis valide ; · Le système vérifie la conformité des informations fournies **A1** ; [fin]
Alternative A1	Dans le cas où les informations fournies sont incomplètes ou incorrectes le système réaffiche le formulaire d'inscription et attend que l'utilisateur ressaisisse ses informations ;

FIGURE 3.10 – Description du cas d'utilisation "Inscription/Enregistrement".

Diagramme du cas d'utilisation " Inscription "

La figure 3.11 présente le diagramme cas d'utilisation "Inscription"

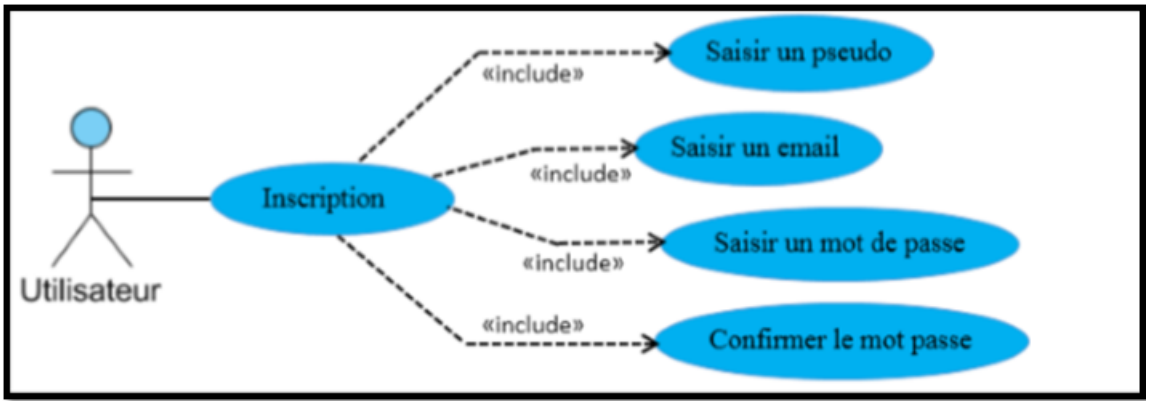


FIGURE 3.11 – Le cas d'utilisation "Inscription" .

Le cas d'utilisation "Déconnexion"

La figure 3.12 présente la formalisme de description des cas "Déconnexion".

Cas d'utilisation N° 4	**Déconnexion.**
Résumé	Permet à l'utilisateur de quitter son compte.
Acteurs	Utilisateur.
Précondition	Authentification.
Scénario nominal	[début] · Accès à l'application. · Accès au menu de l'application. · Cliquer sur le bouton « Déconnexion ». · L'application renvoi automatiquement la page d'accueil. [fin]

FIGURE 3.12 – Description du cas d'utilisation "Déconnexion".

Diagramme du cas d'utilisation "Déconnexion".

La figure 3.13 présente le diagramme cas d'utilisation " Déconnexion "

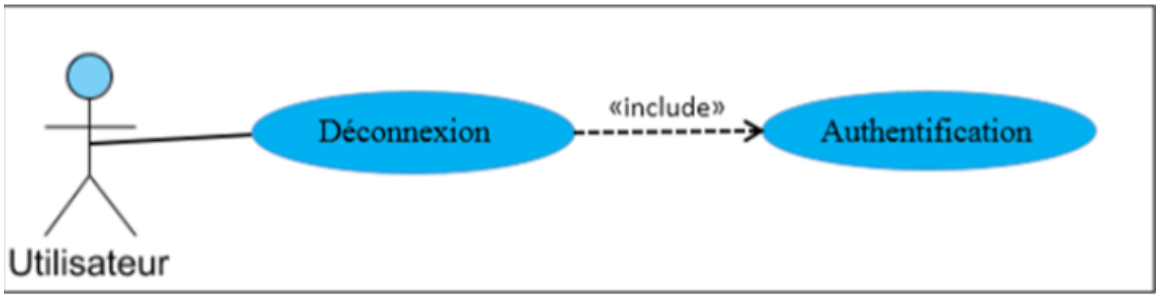


FIGURE 3.13 – Le cas d'utilisation " Déconnexion ".

Le cas d'utilisation "Envoyer des messages"

La figure 3.14 présente la formalisme de description des cas "Envoyer des messages".

Cas d'utilisation N° 5	**Envoyer des messages**
Résumé	Permet à l'utilisateur d'envoyer des messages à ses amis qui s'affichent dans une liste.
Acteurs	Utilisateur.
Précondition	Authentification, le récepteur doit être dans la liste des amis.
Scénario nominal	[début] · L'utilisateur choisir l'ami dans la liste ; · Le système affiche le formulaire des messages ; · L'utilisateur saisie son message puis il clique sur le bouton envoyer ; [fin]

FIGURE 3.14 – Description du cas d'utilisation "Envoyer des messages".

Diagramme du cas d'utilisation " Envoyer des messages "

La figure 3.15 présente le diagramme cas d'utilisation "Envoyer des messages"

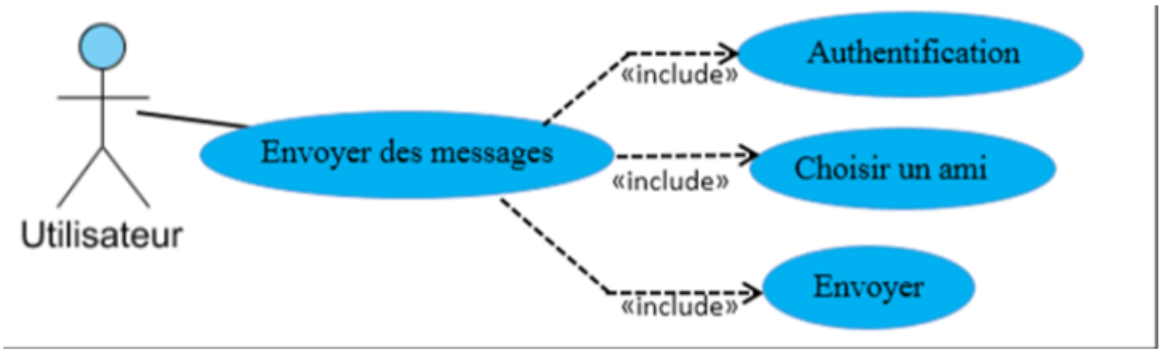


FIGURE 3.15 – Le cas d'utilisation "Envoyer des messages".

3.4 Diagrammes de séquence

3.4.1 Définition d'un diagramme de séquence

Un diagramme de séquences est un diagramme d'interaction qui expose en détail la façon dont les opérations sont effectuées : quels messages sont envoyés et quand ils le sont.

Les diagrammes de séquences sont organisés en fonction du temps qui s'écoule au fur et à mesure que nous parcourons la page.

Les objets impliqués dans l'opération sont répertoriés de gauche à droite en fonction du moment où ils prennent part dans la séquence[3].

3.4.2 Diagramme de séquence du cas d'utilisation Inscription

Le diagramme de séquence suivant illustre les interactions nécessaire pour l'inscription de l'utilisateur.
La figure 3.16 présente le Diagramme de séquence du cas d'utilisation Inscription

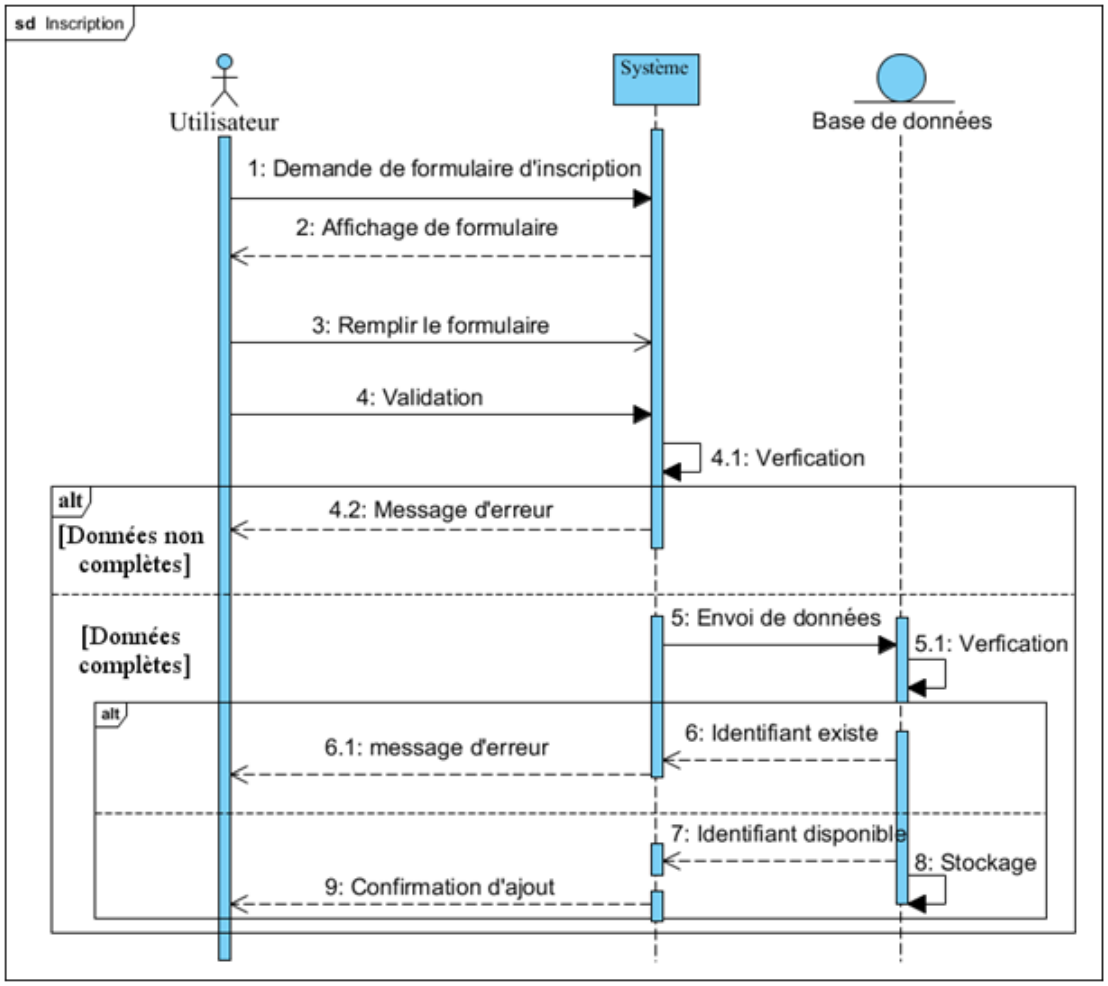


FIGURE 3.16 – Diagramme de séquence du cas d'utilisation "Inscription".

3.4.3 Diagramme de séquence du cas d'utilisation Authentification

Le diagramme de séquence suivant illustre les interactions nécessaire pour l'authentification de l'utilisateur.
La figure 3.17 présente le Diagramme de séquence du cas d'utilisation Authentification

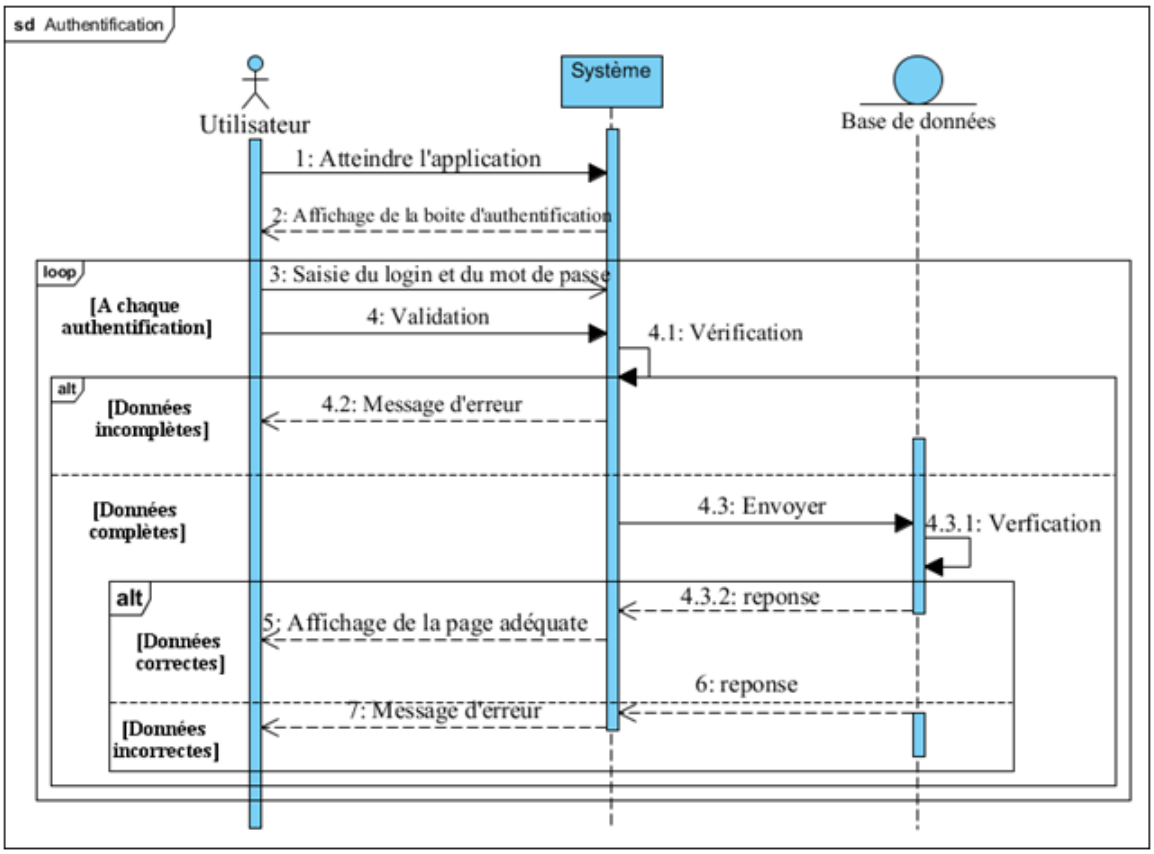


FIGURE 3.17 – Diagramme de séquence du cas d'utilisation Authentification.

3.4.4 Diagramme de séquence du cas d'utilisation "Ajout des amis" :

Le diagramme de séquence suivant illustre les interactions nécessaire pour ajouter des amis

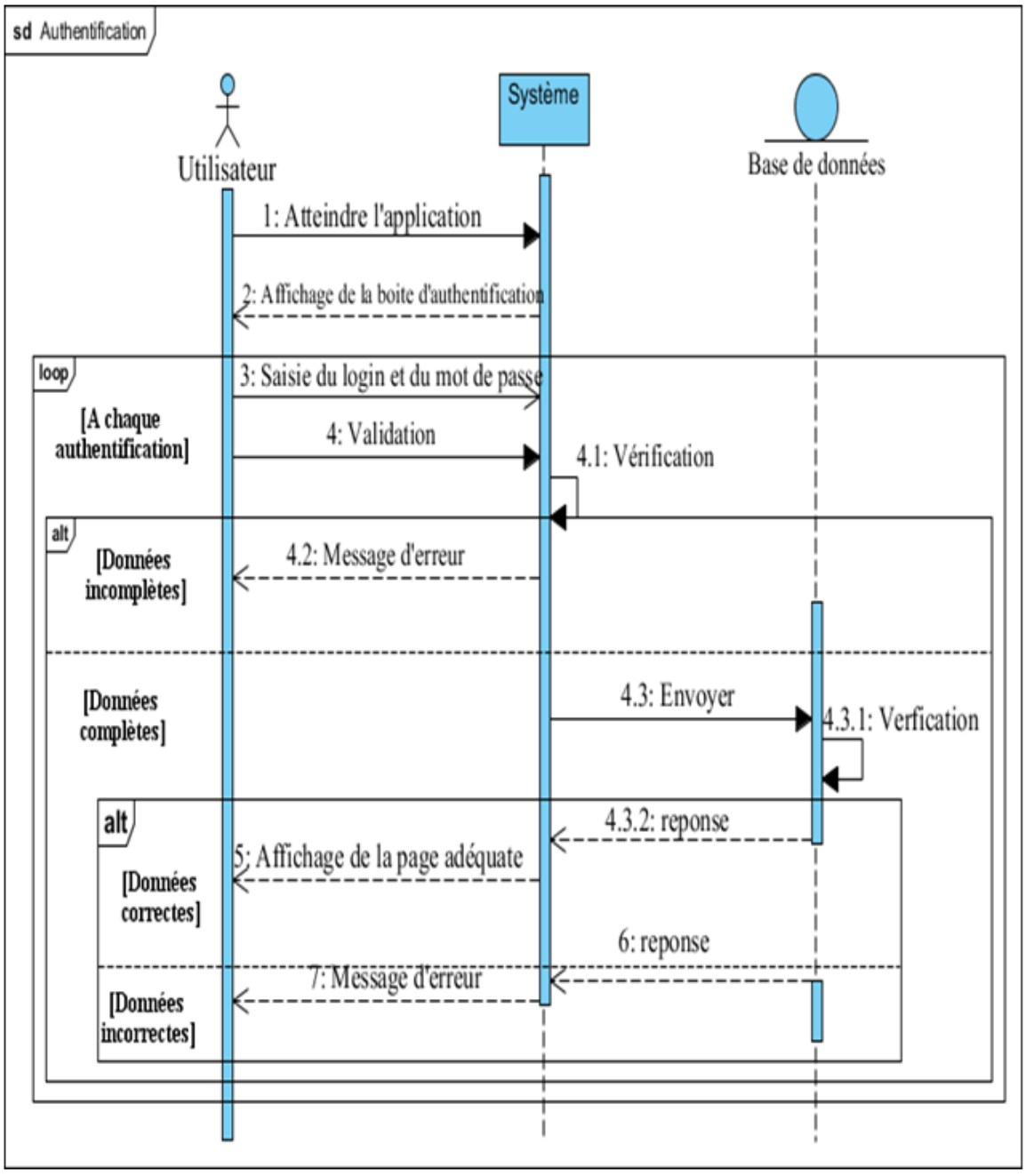


FIGURE 3.18 – Diagramme de séquence du cas d'utilisation "Ajout des amis".

3.4.5 Diagramme de séquence du cas d'utilisation Envoyer des messages

Le diagramme de séquence suivant illustre les interactions nécessaire pour l'envoie des messages.

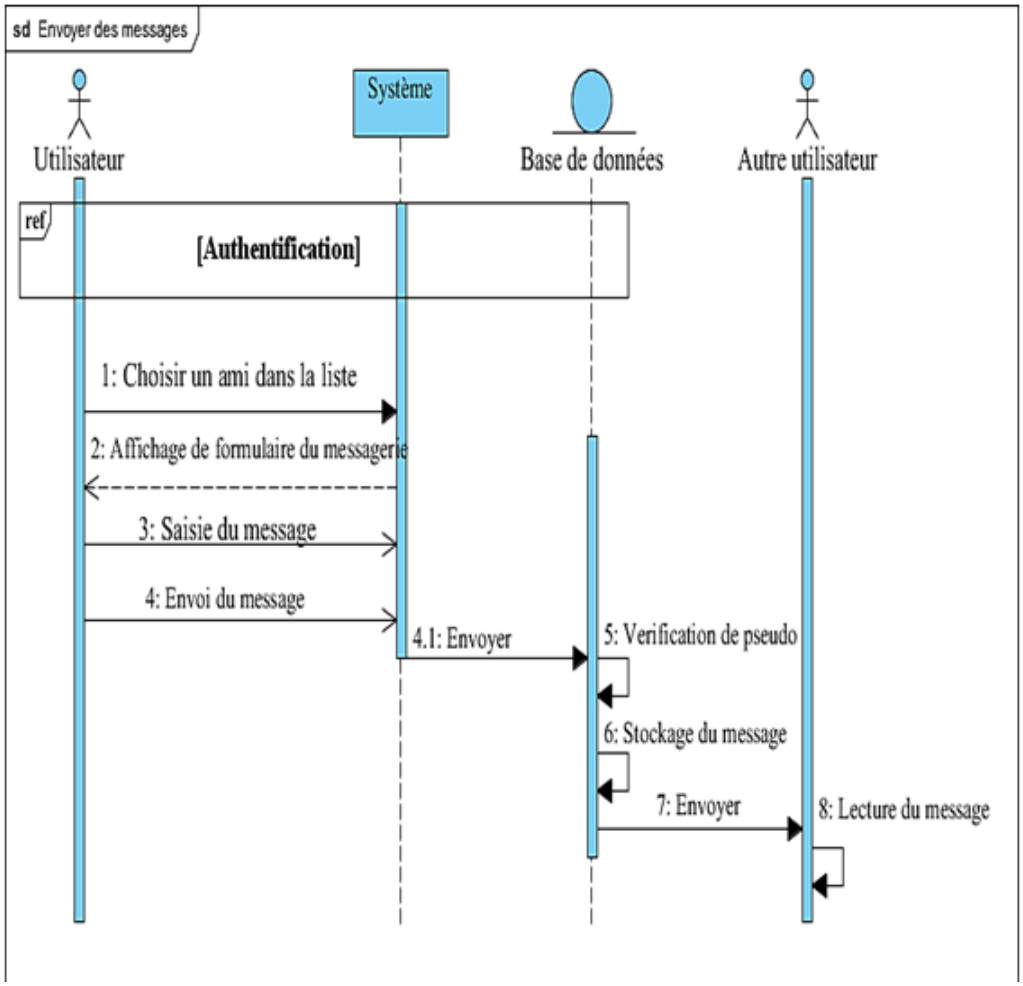


FIGURE 3.19 – Diagramme de séquence du cas d'utilisation envoyer des messages.

3.4.6 Diagramme de séquence du cas d'utilisation Déconnexion

Le diagramme de séquence suivant illustre les interactions nécessaire pour la déconnexion.

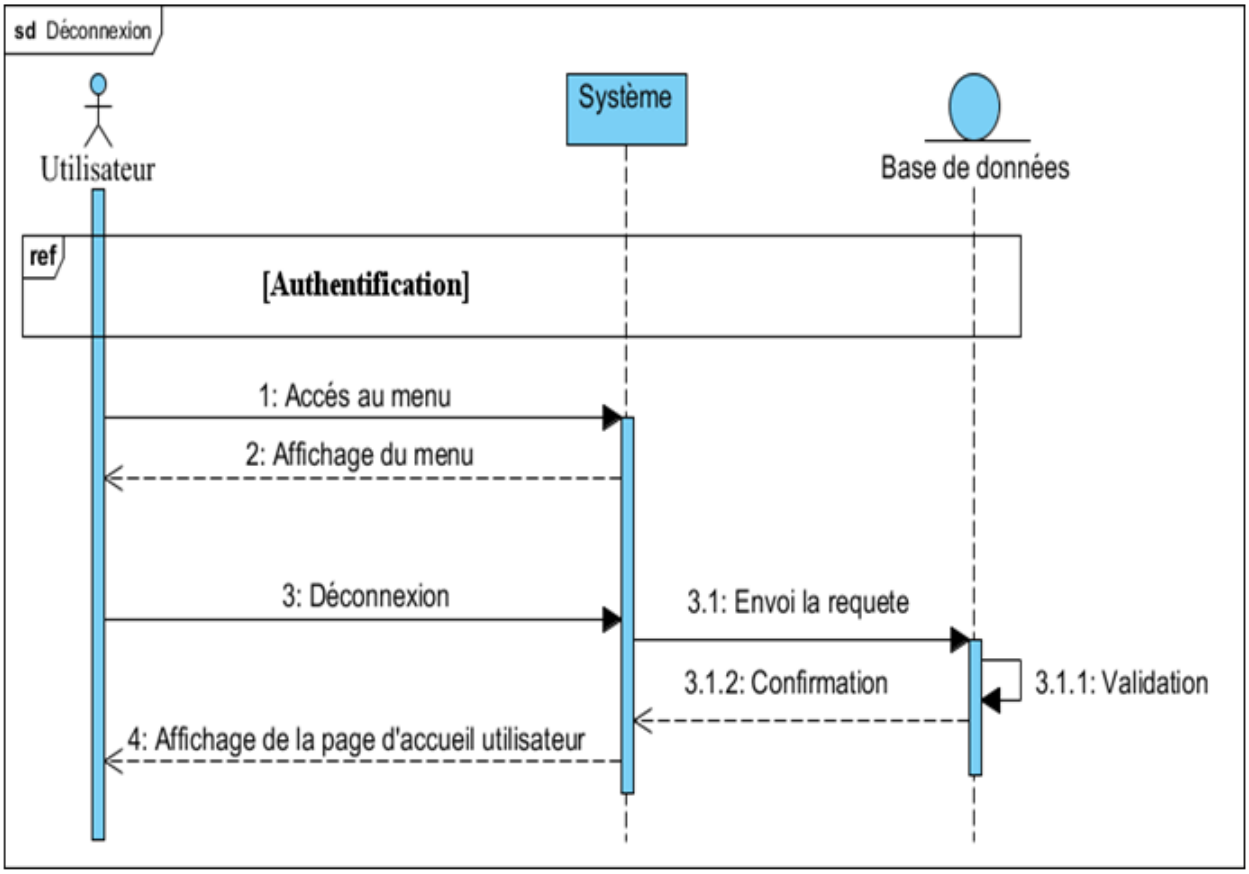


FIGURE 3.20 – Diagramme de séquence du cas d'utilisation Déconnexion.

3.5 Diagramme de classe

3.5.1 Définition

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il est le seul obligatoire lors d'une telle modélisation. Il montre la structure interne. Il permet de fournir une représentation abstraite des objets du système qui vont interagir pour réaliser les cas d'utilisation. Il est important de noter qu'un même objet peut très bien intervenir dans la réalisation de plusieurs cas d'utilisation. Les cas d'utilisation ne réalisent donc pas une partition des classes du diagramme de classes. Un diagramme de classes n'est donc pas adapté (sauf cas particulier) pour détailler, décomposer, ou illustrer la réalisation d'un cas d'utilisation particulier.

Il s'agit d'une vue statique, car on ne tient pas compte du facteur temporel dans le comportement du système. Le diagramme de classes modélise les concepts du domaine d'application ainsi que les concepts internes créés de toutes pièces dans le cadre de l'implémentation d'une application. Chaque langage de Programmation orienté objet donne un moyen spéci que d'implémenter le paradigme objet (pointeurs ou pas, héritage multiple ou pas, etc.), mais le diagramme de classes permet de modéliser les classes du système et leurs relations indépendamment d'un langage de programmation particulier[3].

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

3.5.2 Diagramme de classe de l'application :

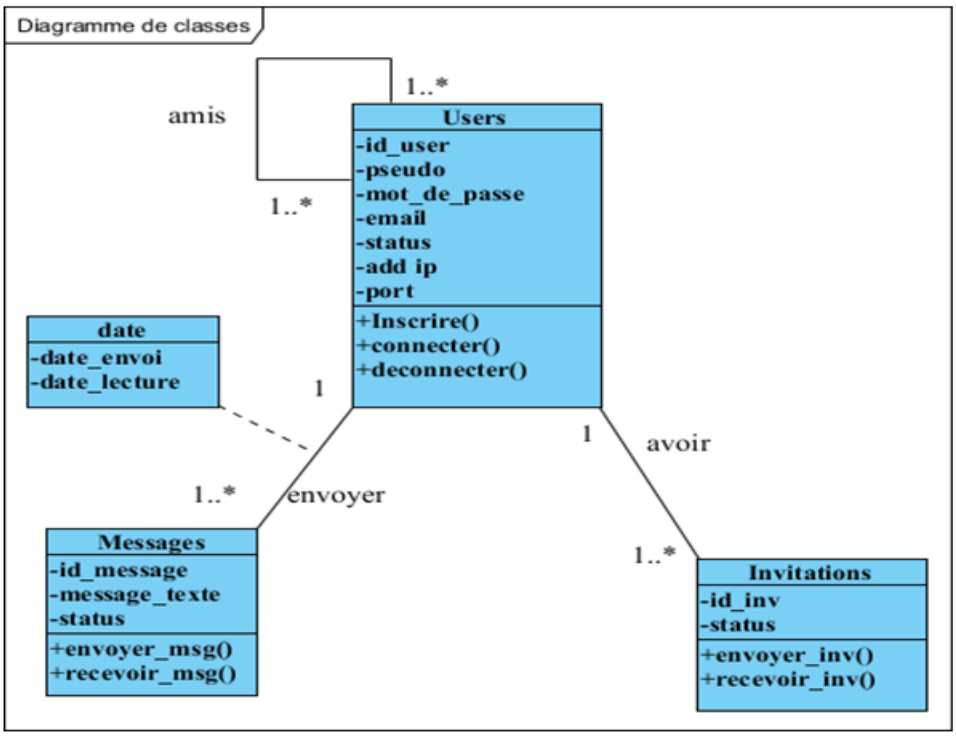


FIGURE 3.21 – Diagramme de classe.

3.5.3 Règle de dérivation du modèle relationnel à partir d'un modèle de classes

Règle 1 : Transformation des classe : Chaque entité devient une relation[3].

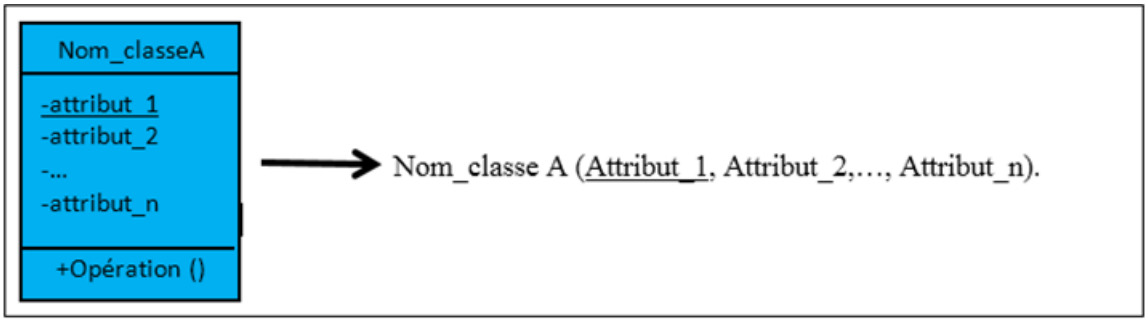


FIGURE 3.22 – Passage d'une classe a une relation.

Règle 2 : Association un-a-plusieurs : Il faut ajouter un attribut de type clé étrangère dans la relation ls de l'association. L'attribut porte le nom de la clé primaire de la relation père de l'association[3].

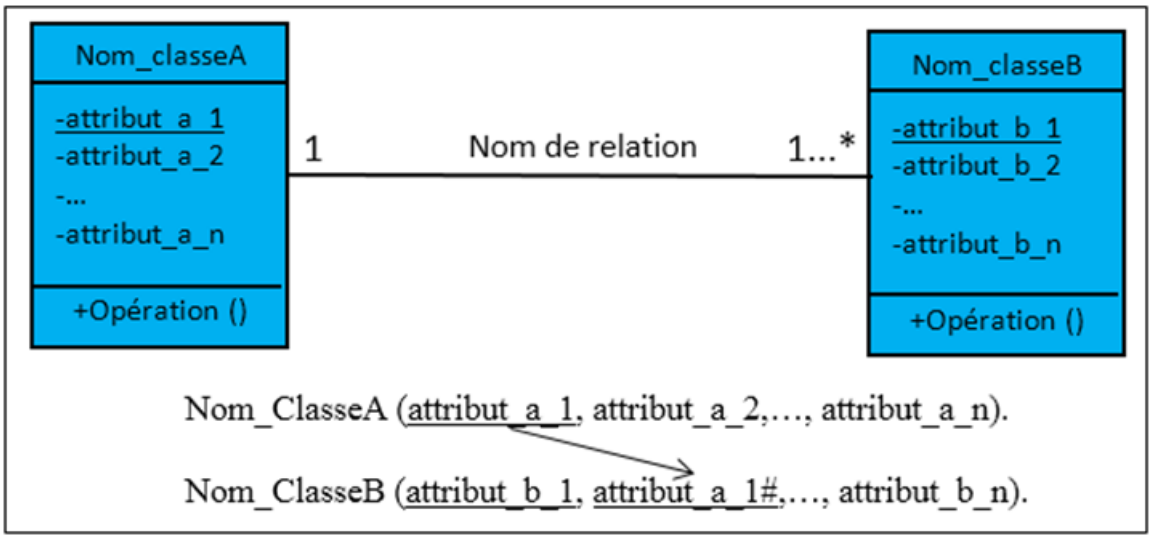


FIGURE 3.23 – Passage un a plusieurs.

Règle 3 : Association un-a-un : Il faut ajouter un attribut clé étrangère dans la relation dérivée de l'entité ayant la cardinalité minimale égale à zéro. Dans le cas de UML, il faut ajouter un attribut 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 l'entité (classe) connectée à l'association[3].

L'application des règles de passage énumérées précédemment, nous permet d'avoir le modèle relationnel de la base de données du notre application à mettre en oeuvre

• Users (id_user, pseudo, mot_de_passe, email, status, add_ip, port) ; status = En ligne / Hors ligne.

• Amis (id_user1#, id_user2#, status); status = En ligne / Hors ligne.

• Messages (id_message, id_user1#, id_user2#, message_texte, sta- tus) ; Status = lu / non lu

• Invitations(id_inv,id_user1#,id_user2#,status);status(Accepter / Refuser) ; Id_user1 = émetteur, id_user2 = récepteur

3.6 Conclusion

Ce chapitre a été consacré à l'étude conceptuelle de notre projet. Nous avons défini, en premier lieux les besoins et identifié toutes les entités composant notre système. Actions et interactions de ces entités ont été décrites à l'aide des diagrammes d'UML, que nous avons aussi présenté. Les diagrammes obtenus et les règles de modélisation, relatifs à l'UML, ont été utilisés comme plateforme théorique pour l'implémentation de notre application.

4 l'implémentation

4.1 Introduction

A ce stade du processus, les cas d'utilisation sont terminés, le problème a été analysé en profondeur ; nous avons défini une conception mieux appropriée aux besoins de l'application ; Nous pouvons donc entamer la phase de réalisation et de mise en œuvre de notre application de messagerie instantanée sécurisé ,nous allons présenter, dans ce chapitre, les outils de développement adoptés à savoir : l'environnement utilisé qui est NetBeans IDE, Sceen builder ainsi que les langages de programmation (JAVA.FX), et nous allons présenter le système de gestion de base de données Apache Derby (Apache Derby serveur),ainsi que le langage de manipulation de bases de données SQL, et nous allons présenter les interfaces principales de l'application.

4.2 Outils de développement

4.2.1 NetBeans IDE

NetBeans IDE(figure 4.1) est un environnement de développement intégré (EDI),permet également de supporter différents autres langages, comme java, C, C++, JavaScript, XML, Groovy, PHP et HTML de façon native ainsi que bien d'autres (comme Python ou Ruby) par l'ajout de greffons. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refacturant, éditeur graphique d'interfaces et de pages Web).

NetBeans constitue par ailleurs une plate forme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)).

Principaux langages supportés

Il supporte principalement les langages suivants :

- Java (Java SE , Java ME , Java FX , Java EE), Javadoc
- Groovy et Grails
- PHP
- JavaScript
- C, C++, Fortran. Netbeans ne requiert pas l'utilisation d'un compilateur particulier. ? noter le support des bibliothèques

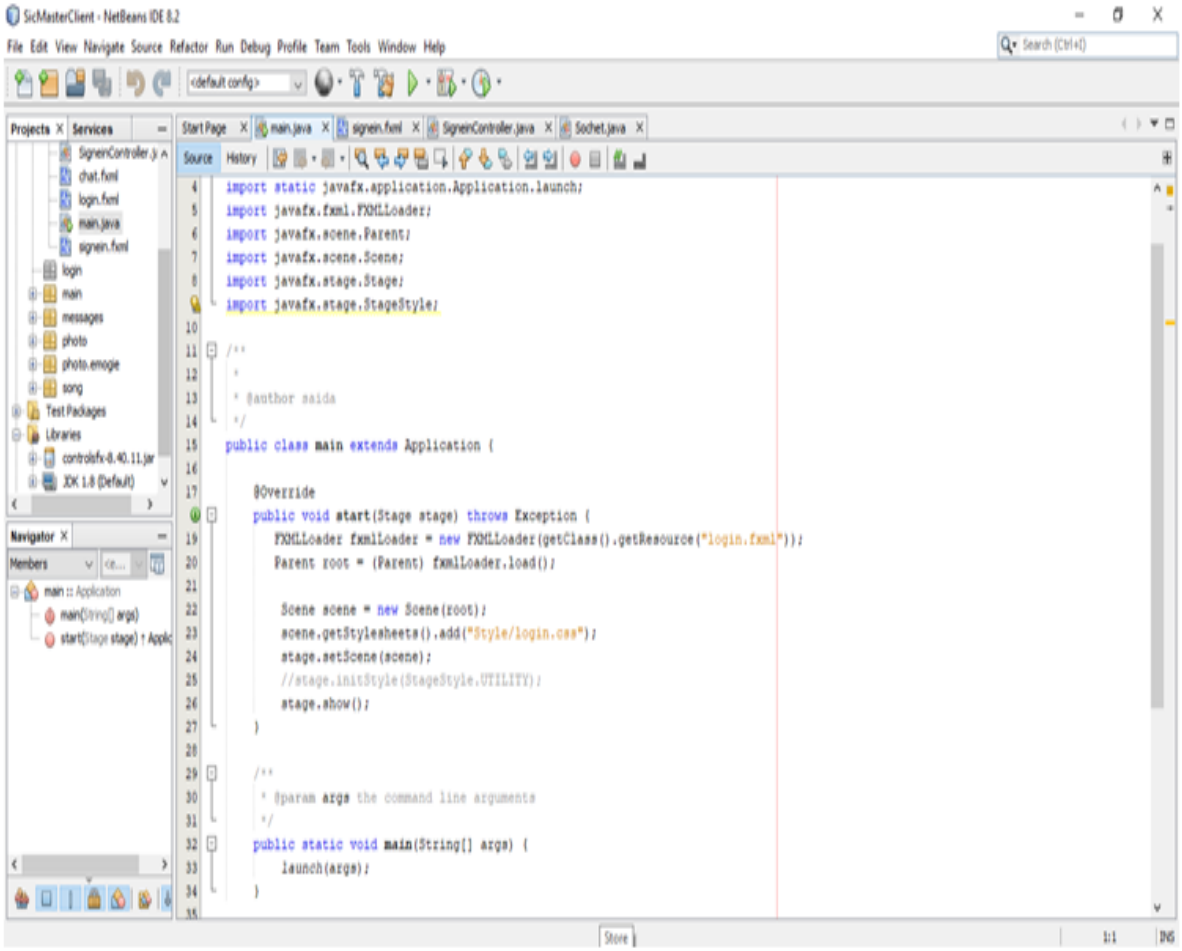


FIGURE 4.1 – Fenêtre de programmation Sur Netbeans.

- Python (via un greffon développé par la communauté)
- HTML, XHTML, XML , CSS

4.2.2 Langage de programmation(Java) :

Java est un langage de programmation et une plate-forme informatique qui ont été créés par Sun Microsystems en 1995. Beaucoup d'applications et de sites Web ne fonctionnent pas si Java n'est pas installé et leur nombre ne cesse de croître chaque jour. Java est rapide, sécurisé et fiable. Des ordinateurs portables aux centres de données, des consoles de jeux aux superordinateurs scientifiques, des téléphones portables à Internet, la technologie Java est présente sur tous les fronts[19].

C'est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être très largement utilisé pour le développement d'applications de types très disparates. Java est notamment largement utilisée pour le développement d'applications d'entreprises et mobiles[19].

Java est un langage interprété, ce qui signifie qu'un programme compilé n'est pas directement exécutable par le système d'exploitation mais il doit être interprété par un

autre programme, qu'on appelle interpréteur comme illustré dans la figure 4.2.

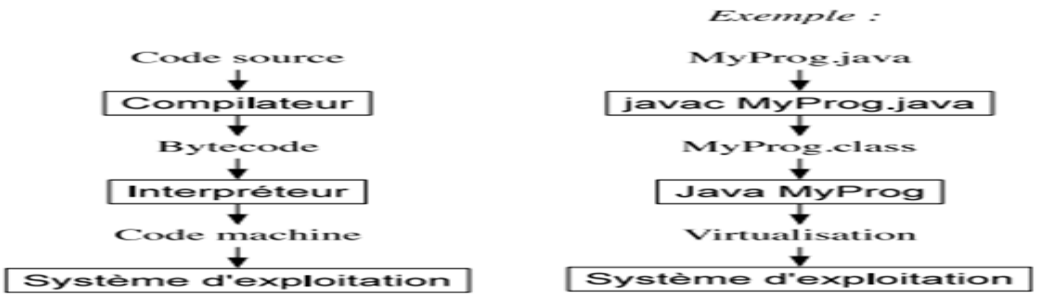


FIGURE 4.2 – architecture exécutable Code java.

4.2.3 JavaFX :

JavaFX(voir figure 4.3 est une bibliothèque graphique intégrée dans le Java Run Time(JRE) et le ...(JDK) de Java.

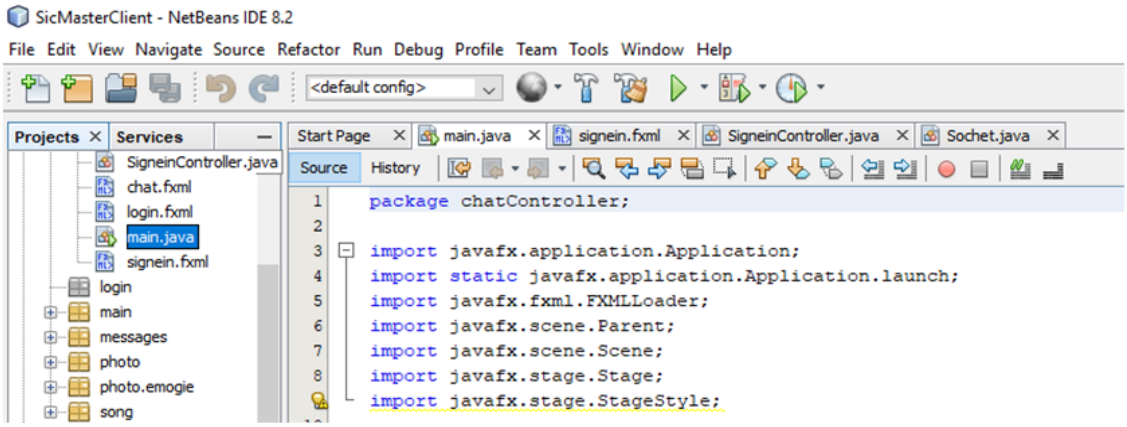


FIGURE 4.3 – projet JavaFX Main.

Oracle la décrit comme "The Rich Client Platform", c'est-à-dire qu'elle permet de réaliser des interfaces graphiques évoluées et modernes grâce à de nombreuses fonctionnalités, telles que les animations, les effets, la 3D, l'audio, la vidéo, etc. Elle a de plus l'avantage d'être dans le langage Java, qui permet de réaliser des architectures avec des paradigmes objet, et aussi de pouvoir utiliser le typage statique.

4.2.4 Scene Builder :

JavaFX Scene Builder (Scene Builder) vous permet de concevoir rapidement des interfaces utilisateur d'application JavaFX en faisant glisser un composant de l'interface utilisateur d'une bibliothèque de composants de l'interface utilisateur et en le déposant dans une zone d'affichage du contenu. Le code FXML de la mise en page de l'interface utilisateur que vous créez dans l'outil est automatiquement généré en arrière-plan[21].

Scene Builder peut être utilisé comme un outil de conception autonome, mais il peut également être utilisé avec des IDE Java pour que vous puissiez utiliser l'IDE pour écrire, construire et exécuter le code source du contrôleur que vous utilisez avec l'interface utilisateur de votre application. Bien que Scene Builder soit plus étroitement intégré à l'EDI NetBeans, il est également intégré aux autres EDI Java décrits dans ce document. L'intégration vous permet d'ouvrir un document FXML à l'aide de Scene Builder, d'exécuter les exemples Scene Builder et de générer un modèle pour le fichier source du contrôleur[21]. La figure 4.4 présente Utilisation JavaFX Scene Builder .

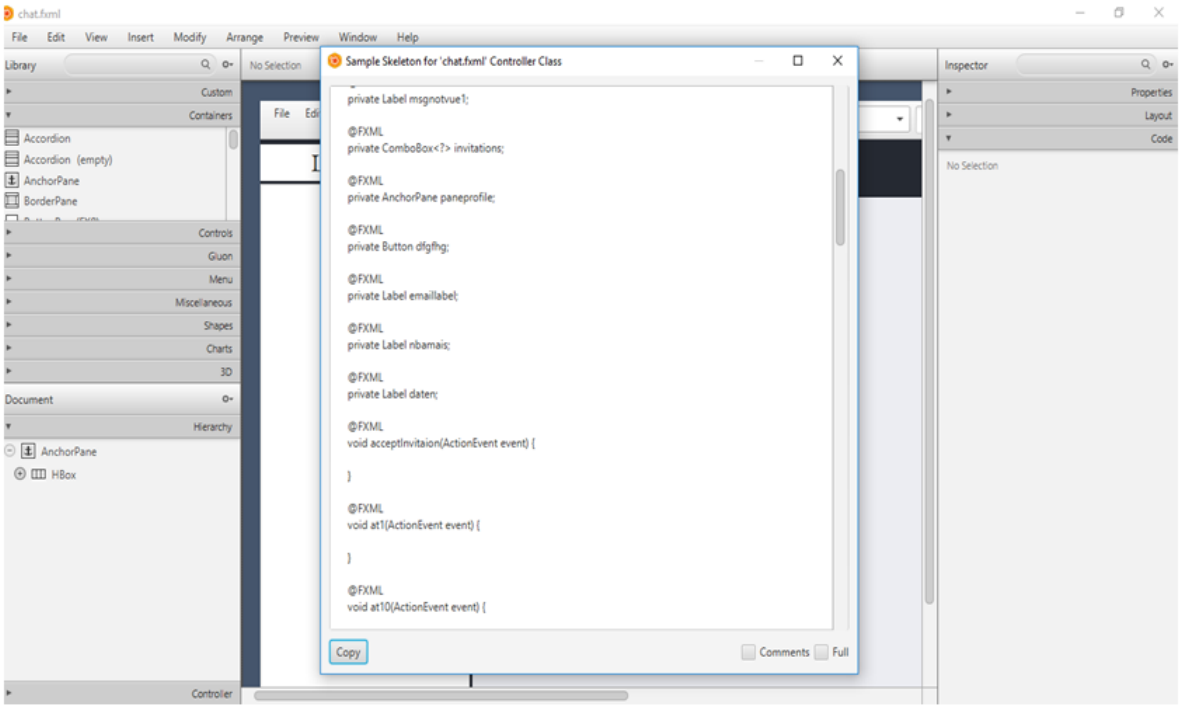


FIGURE 4.4 – Utilisation JavaFX Scene Builder.

4.2.5 Serveur Apache derby :

Apache Derby est une base de données relationnelle open-source entièrement développée en Java par la fondation Apache[22].

Derby a la particularité de pouvoir être utilisé comme gestionnaire de base de données embarqué dans une application Java. Ce qui rend inutile l'installation et la maintenance d'un serveur de base de données autonome. A l'inverse Derby supporte aussi un mode de fonctionnement client-serveur[22].

Le driver embarqué Derby :

(Derby Embedded Driver), voir figure 4.5 permet la connexion simultanée de plusieurs utilisateurs et l'accès concurrent à plusieurs bases. Néanmoins, dans cette configuration, un seul driver - et donc une seule application - peut accéder à une même base à un instant donné[22].

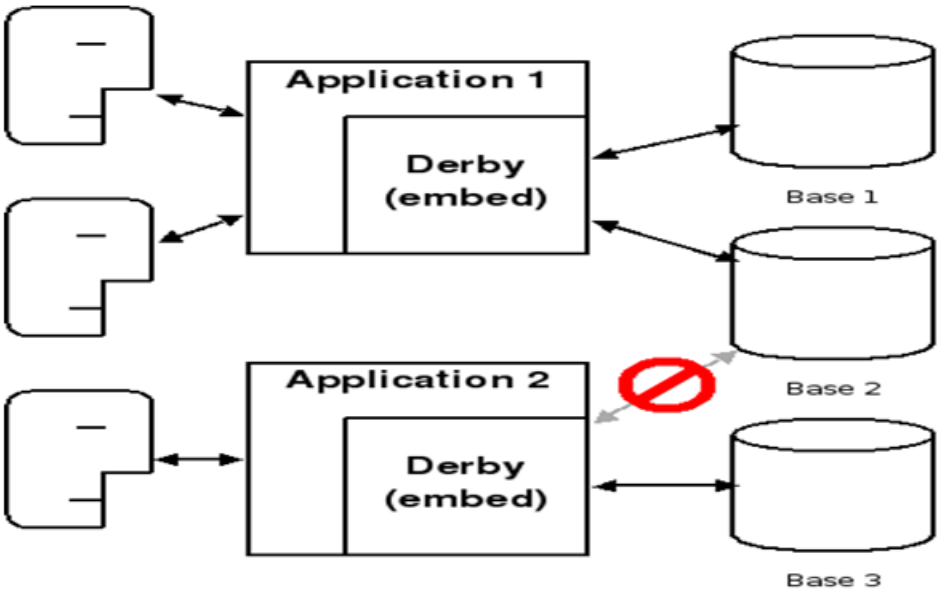


FIGURE 4.5 – Le driver embarqué Derby.

Si nous avons choisi ce SGBD c'est plus pour des raison de performance et de fonctionnalité ouverte. Apatch derby est beaucoup moins complexe à installer et à administrer que d'autre système.

4.3 Implémentation de notre base de données

Notre base des données a été crée à l'aide d'Apache Derby. Et est constituée de quatre tables comme illustré dans la figure 4.6.

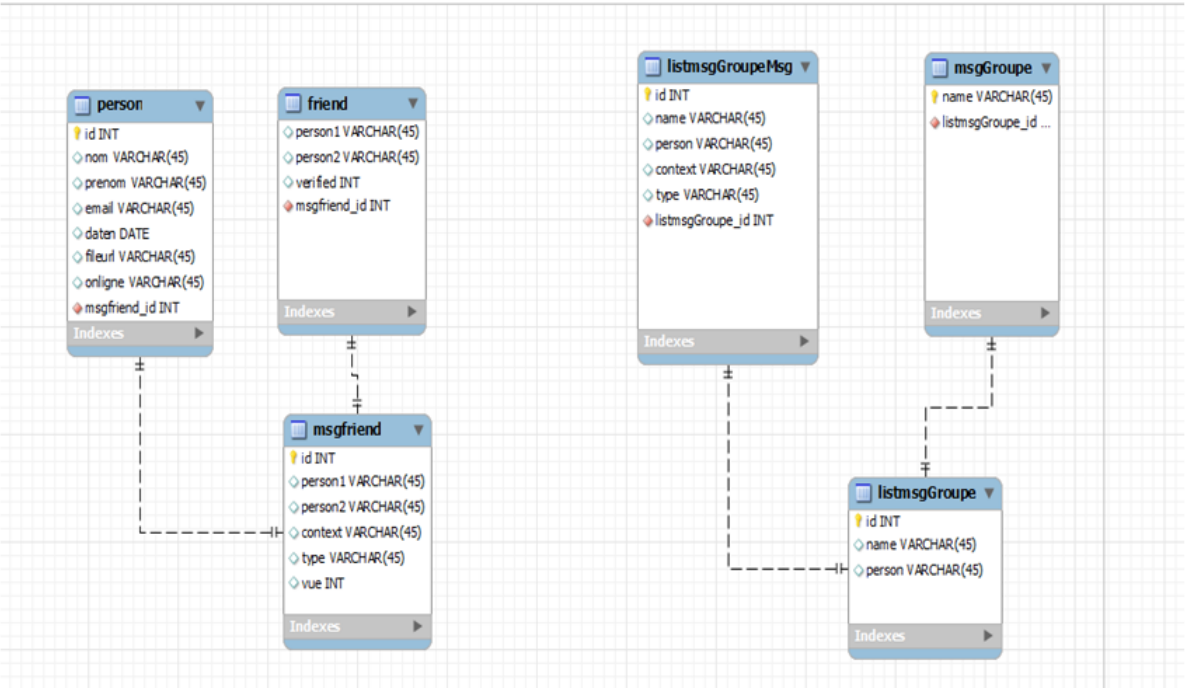


FIGURE 4.6 – les tables de Basse de donnée.

La base de données constitue souvent un applicatif critique. Sa sécurité, et le respect de bonnes pratiques, s'avèrent d'autant plus indispensables que l'ouverture du système d'information sur le réseau. A fin de sécuriser notre base de donnée nous avons procédé à la crypter dés sa création on utilisant Derby qui permet de configurer une base de données pour le cryptage dés sa création.

4.4 Principales interfaces

Dans ce qui suit, nous allons présenter quelques interfaces de notre application

• **Interface d'accueil**

La fenêtre d'accueil, qui est la première interface d'interaction avec l'utilisateur, permet à un utilisateur déjà inscrit d'accéder aux différentes fonctionnalités de l'application après une étape d'authentification. Un nouveau utilisateur peut également y accéder après une étape d'inscription.

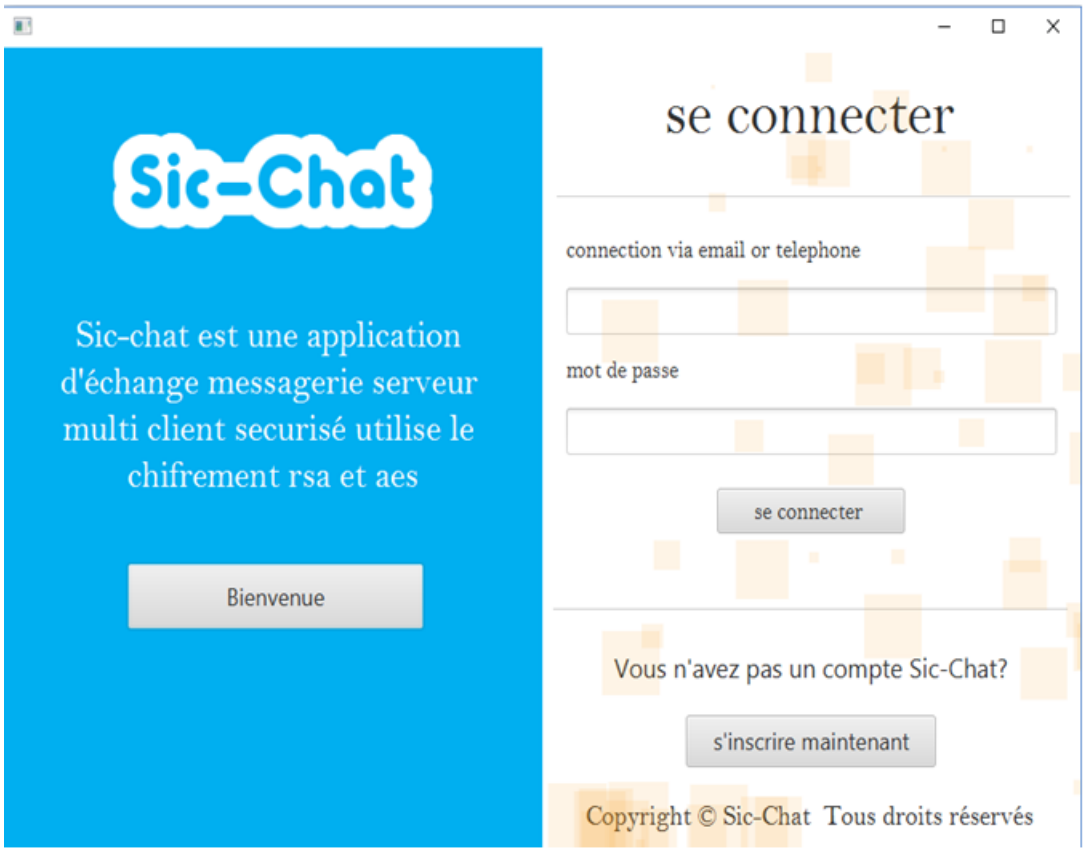


FIGURE 4.7 – Interface d'accueil d'application.

• **Interface d'administration**

Cette interface permet au superviseur d'avoir un contrôle total sur l'application, lui permettant ainsi d'accéder a toutes ses fonctionnalités

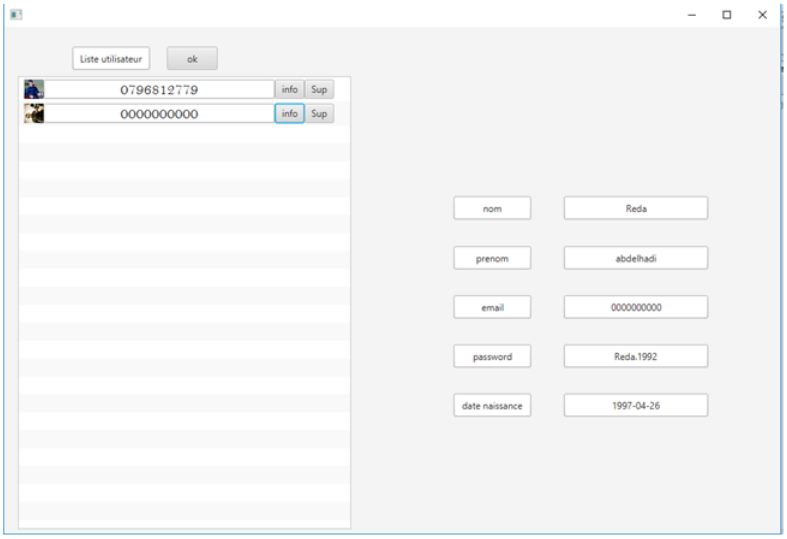


FIGURE 4.8 – Interface d'administration site.

• **Interface gestion de l'inscription et login**

Cette interface permet au superviseur de gérer les différentes sessions des utilisateurs.

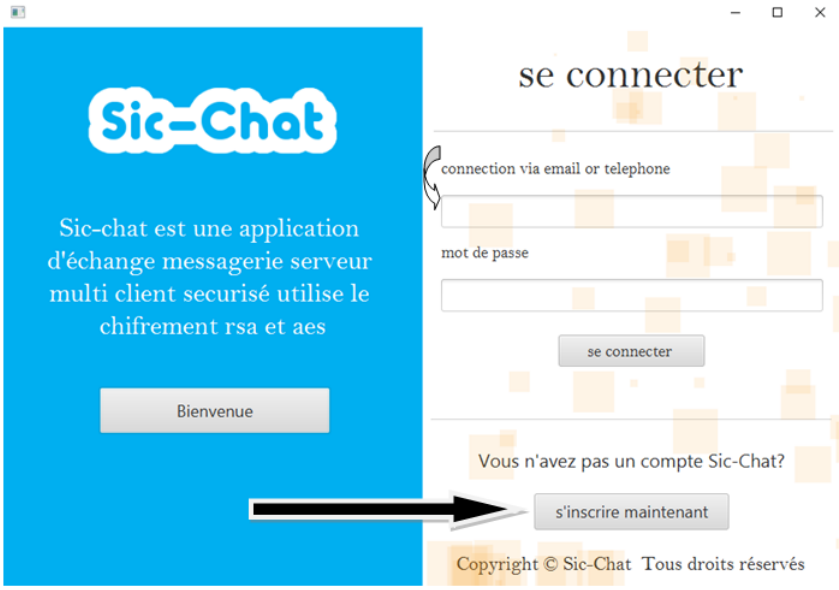


FIGURE 4.9 – Interface gestion de l'insecription et login.

- **Interface gestion gestion de l'inscription**

Cette fenêtre permet à un nouveaux utilisateur de s'inscrire. Il doit pour cela remplir les différents champs d'informations exigées pour l'inscription.

FIGURE 4.10 – Interface gestion de l'insecription.

4.5 Quelque exemple de code source

4.5.1 Code l'inscription et login :

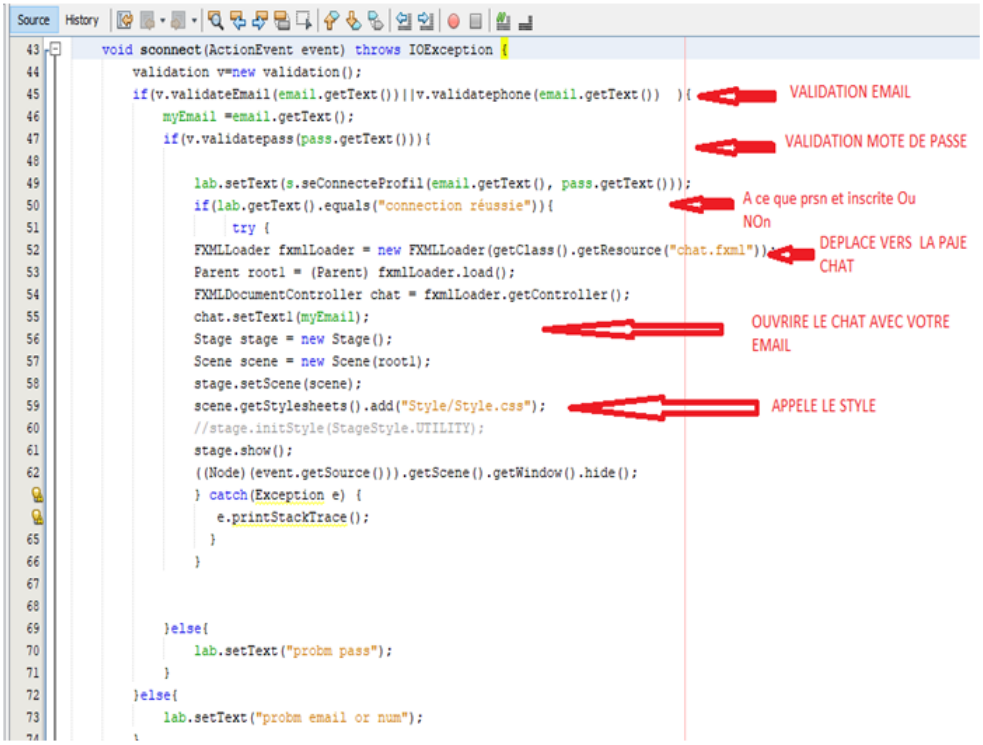


FIGURE 4.11 – code source login.

ce tableau si présente notre base de données

4.5.2 Code l'inscription

FIGURE 4.12 – code source Insecription.

4.5.3 La Méthode Threed de communication avec le serveur

Cette communication, dont le code source est donnés dans la figure 4.13, est assurée par le biais des sockets. Il est à noter que le serveur est muliti-client ce qui permet à plusieurs clients de se connecter à lui en "parallèle". Pour cela on a créer trois threads :

FIGURE 4.13 – Code source des threeds de connexion.

• Le thread principal : Il va attendre les connexions en écoutant le port ouvert pour le serveur. A chaque connexion il va lancer un nouveau thread de type client.

• **Le thread client :** Chargé d'envoyer, dés son arrivée, un message au client concernés. à chaque connexion d'un nouveau client, un tread client est crée (voir figure 4.14

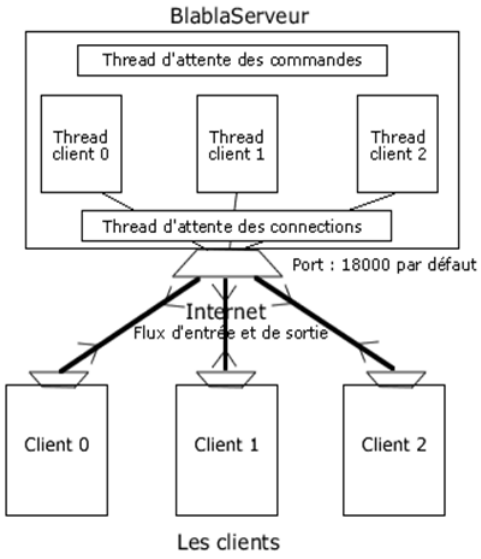


FIGURE 4.14 – threads clients

• **Le thread des commandes** : attend que l'administrateur du serveur tape des commandes dans la console pour faire une action (quitter le serveur par exemple).

4.6 Conclusion et perspectives :

Dans ce chapitre, nous avons décrit brièvement le processus de réalisation de notre application en spécifiant l'environnement et les outils de développement utilisés. Aussi nous avons présenté quelques fenêtres graphiques de notre application.
Lors de la réalisation de cette application nous avons acquit une expérience considérable notamment en matière de programmation Java. et nous aimerions bien d'élargir notre savoir et notre savoir faire pour implémenter une version mobile de cette application on la faisons migré vers Androïde.

5 Conclusion Générale

Au terme de ce mémoire, nous avons pu exploiter nos connaissances théoriques et pratiques pour mettre en place un système de messagerie sécurisée qui permet aux différents utilisateurs d'échanger messages et différents documents entre eux.

Dans un premier temps, nous avons défini et analysé les besoins de système. Ensuite, nous avons entamé la phase de conception qui a permis de structurer et définir les besoins attendus du système en utilisant le langage de modélisation UML. Enfin, nous avons abordé la réalisation en utilisant les outils d'implémentation appropriés.

Ce projet a été pour nous une opportunité de se lancer dans le fascinant domaine de la sécurité des services réseaux et la cryptographie

Bibliographie

[1] Laurent Piechocki/ Frédéric Di Gallo " Cours UML" édition 2007-2008.

[2] Dominique Vauquier "Vers un monde lisible" , édition 2008-2009.

[3] Chantal Morley, Jean Hugues, Bernard le Blanc " UML pour l'analyse d'un système d'information " édition 2007

[4] Conception et réalisation d'une application de messagerie instantanée sous Android " Université A/Mira de Béjaïa Faculté des Sciences Exactes Département d'informatique " Promotion 2014-2015 Réaliser par : Mr Lahcene KEDJAR ,Mr Mokhtar OUMAKHLOUF.

[5] Sécuriser les échanges numériques "L'UNIVERSITé DE LILLE 2 " F ?VRIER 2013 - UNJF . Promotion 2010 Réaliser par HASSAN BEZZAZI, MA ?TRE DE CONF ?RENCES.

[6] Cryptographie et Sécurité informatique . Notes de cours provisoires " Université de Liège Faculté des Sciences Appliquées " 2009 - 2010 Renaud Dumont.

[7] Développement d'une application pour l'échange des messages sécurisés. : " Université Abou Bakr Belkaid- Tlemcen " Année universitaire : 2014-2015 Réalisé par Kebir Bahia,Rahmouni Samia.

[8] https ://www.ehealth.fgov.be/file/view/ac7becbc5bd0904bfd7b6044c1f93d34 ?filename=faq-consentement-eclaire.pd 24/01/2018 /

[9] www.ofppt.info/wp-content/uploads/2014/08/Les-types-dattaques-informatique.pdf / 18/02/2018.

[10] www.nouvelobs.com/rue89/rue89-surveillance/20160408.RUE2624/quelle-messagerie-est-la-plus-sure.html / 22/02/2018.

[11] www.lemondeinformatique.fr/actualites/lire-les-5-meilleures-apps-de-messagerie-chiffree-65936.html / 23/02/2018.

[12] www.whatsapp.com/ ?l=fr 23/02/2018.

[13] www.journaldugeek.com/2016/03/11/fwire-messagerie-ultra-securisee / 23/02/2016.

[14] www.openclassrooms.com/courses/l-algorithme-rsa/crypter-et-decrypter / 26/02/2018.

[15] www.bibmath.net/crypto/index.php ?action=affichequoi=moderne/aes 28/02/2018.

[16] www.commentcamarche.com/contents/204-introduction-au-chiffrement-avec-des 03/03/2018.

[17] www.staruml.io/ 14/03/2018.

[18] www.netbeans.org/projects/www/downloads/ 15/03/2018.

[19] www.oracle.com/technetwork/java/index.html

[20] www.openclassrooms.com/courses/apprenez-a-programmer-en-java/installer-les-outils-de-developpement / 15/03/2018.

[21] www.openclassrooms.com/courses/les-applications-web-avec-javafx/presentation-de-l'interface-graphique-en-javafx .

[22] www.docs.oracle.com/javafx/2/overview/jfxpub-overview.htm 02/04/2018.

[23] www.gluonhq.com/products/scene-builder 06/04/2018.

[24] www.chicoree.fr/w/Premiers-pas-avec-Apache-Derby / 16/04/2018.

[25] www.jmdoudoux.fr/java/dej/chap-exceptions.htm / 20/04/2018 .

[26] Bouche.S Butel.A Chamon.E Gonel.Jf Bergeroun.R,Mertin.M. Securité de la messagerie. clusif. 2005.

[27] Goumidi Djamal Eddine, Fonction logistique et standard chaotique pour le chiffrement des images satellitaires, Mémoire présenté pour l'obtention du diplôme de Magister (école doctorale) en Electronique, Spécialité : Télécommunications spatiales, Université Mentouri de Constantine (UMC), 2010