République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A. Mira de Béjaïa
Faculté des Sciences Exactes
Département d'Informatique

Mémoire de fin de cycle

En vue de l'obtention de diplôme de Master en Informatique professionnel

Option : Génie Logiciel(GL).

Thème :

Conception et réalisation d'une blockchain, cas d'étude : gestion du dossier de santé électronique.

Encadré par :

Mme. YAICI Malika.

Réalisé par :

M^{elle}. Bazizi Sonia.
M^{elle}. Beldjoudi Chabha.

Devant le jury composé de :

Présidente : Mme. ALOUI Soraya
Examinateur : Mr. SAADI Mustapha

2019-2020.

Remerciements

Nous remercions ALLAH qui nous aide et nous donne la patience et le courage durant ces langues années d'étude.

Nos plus vifs remerciements vont à notre encadrant ***Mme. YAICI Malika*** *pour sa compréhension, sa disponibilité, son aide et ses conseils qui nous ont été utile pour l'achèvement de ce projet.*

Nous tennons à exprimer notre gratitude aux membres de jury pour avoir accepté de juger ce travail.

Un profond merci à toute personne qui a contribué de près ou de loin à la réalisation de ce projet.

Dédicaces

C'est avec profonde gratitude et sincères mots, que nous dédions ce modeste travail de fin d'étude à nos chers parents, qui ont sacrifié leurs vies pour notre réussite et nous ont éclairé le chemin par leurs conseils judicieux.

Nous espérons qu'un jour, nous pourrons leurs rendre un peu de ce qu'ils ont fait pour nous, que dieu leurs prête bonheur et longue vie.

Nous dédions aussi ce travail à nos frères et sœurs, nos familles, nos amis, tous nos professeurs qui nous ont enseigné et à tous ceux qui nous sont chers.

Table des matières

Table des matières i

Table des figures iii

Liste des tableaux v

Introduction 1

I Généralités 2

1 Introduction . . . 2
2 Historique . . . 2
3 Principe de la blockchain . . . 3
4 Les caractéristiques de la blockchain . . . 4
5 Les différentes technologies combinées dans la blockchain . . . 5
5.1 Généralités sur le réseau pair à pair . . . 5
5.2 Outils cryptographiques . . . 6
5.2.1 Fonction de hachage . . . 6
5.2.2 Identité . . . 6
5.2.3 Signature . . . 7
5.3 Le minage et les algorithmes de consensus . . . 7
5.3.1 Le minage . . . 7
5.3.2 Arbre de Merkle . . . 8
5.3.3 Les algorithmes de consensus . . . 9
6 Le fonctionnement de la blockchain . . . 11
7 Couches de la blockchain . . . 13
7.1 Couche Internet . . . 13
7.2 Couche de protocole Blockchain . . . 13
7.3 Couche d'application . . . 13
7.4 Couche Expérience utilisateur(UX) . . . 14
8 Types de blockchain . . . 14
8.1 Blockchain publique . . . 14
8.2 Blockchain privée . . . 14
8.3 Blockchain consortium . . . 15
8.4 Blockchain hybride . . . 15
9 Domaine d'application . . . 15
9.1 Paiement numérique et monnaie virtuelle . . . 15
9.2 Chaîne d'approvisionnement . . . 15
9.3 La santé . . . 16
9.4 L'assurance . . . 17
9.5 Internet des objets (IoT) . . . 17
10 Conclusion . . . 17

II Etude de l'existant **18**
1 Introduction . . . 18
2 Les systèmes basés sur des blockchains existantes . . . 18
2.1 Les systèmes basés sur la blockchain Ethereum . . . 18
2.1.1 Ancile . . . 18
2.1.2 MedRec . . . 21
2.1.3 MedChain . . . 22
2.1.4 Schéma de chiffrement consultable . . . 25
2.1.5 Système E-health basé sur MobileCloud . . . 26
2.2 Les système basés sur la blockchain Hyperledger Fabric . . . 28
2.2.1 Application pour Healthcare 4.0 . . . 28
2.3 Les systèmes basés sur les deux blockchains : Ethereum et HyperldgerFabric . . . 30
2.3.1 Medicalchain . . . 30
3 Les systèmes indépendants des blockchains existantes . . . 32
3.1 MeDShare . . . 32
4 La synthèse des articles étudiés . . . 36
5 Conclusion . . . 37

III Conception générale **38**
1 Introduction . . . 38
2 L'architecture générale de notre système . . . 39
3 Les acteurs du système et leurs rôles . . . 40
4 Les relations entre les acteurs (leurs communications) . . . 40
5 Le stockage . . . 40
5.1 La base de données . . . 40
5.2 La partie blockchain . . . 41
5.2.1 Les algorithmes de chiffrement . . . 41
5.2.2 L'algorithme de validation . . . 44
5.2.3 Transaction . . . 44
5.2.4 Contenu d'un bloc . . . 45
5.2.5 Blockchain . . . 46
5.3 L'application cliente . . . 47
6 Conclusion . . . 47

IV Conception détaillée et réalisation de l'application **48**
1 Introduction . . . 48
2 La partie analyse et conception . . . 48
2.1 Méthode d'analyse et de conception . . . 48
2.2 Le Processus Unifié (UP) . . . 49
2.3 Spécification des besoins . . . 49
2.3.1 Identification des acteurs du système . . . 49
2.3.2 Identification des besoins . . . 50
2.4 Expression des besoins . . . 52
2.4.1 Diagramme de cas d'utilisation . . . 52
2.4.2 Description des cas d'utilisations . . . 54
2.4.3 Diagramme de séquence . . . 64
2.4.4 Description des diagrammes de séquences . . . 64
2.4.5 Dictionnaire de données . . . 76
2.4.6 Diagramme de classes . . . 76

2.4.7 Modèle relationnel 77
3 La partie réalisation 78
3.1 L'environnement et les outils de développement 78
3.1.1 Les langages de programmation 78
3.1.2 Les bibliothèques 79
3.1.3 Les environnements de programmation 79
3.1.4 Utilité des outils pour notre application 80
3.2 La description de quelques interfaces 80
3.2.1 Inscription 81
3.2.2 Connexion 82
3.2.3 Profil médecin traitant 83
4 Conclusion 83

Conclusion et perspectives **84**

Bibliographie **86**

Table des figures

I.1 Exemple simplifié d'une blockchain [20]. 4
I.2 Arbre de Merkel. 9
I.3 Schéma explicatif de fonctionnement de la blockchain (bitcoin) [20]. 12
I.4 Les Couches de la blockchain[20]. 14

III.1 L'architecture de système PatientLedger. 39

IV.1 Description du Processus Unifié (UP) [39]. 49
IV.2 Diagramme de contexte statique de notre système. 50
IV.3 Diagramme de cas d'utilisation de notre système. 53
IV.4 Diagramme de séquence du cas «S'authentifier». 65
IV.5 Diagramme de séquence du cas « Créer un compte ». 66
IV.6 Diagramme de séquence du cas «Demander de rejoindre le réseau blockchain». . 67
IV.7 Diagramme de séquence du cas «Ajouter un patient/hôpital 1». 68
IV.8 Diagramme de séquence du cas «Ajouter un patient/hôpital 2». 69
IV.9 Diagramme de séquence du cas «Demander un accès». 70
IV.10Diagramme de séquence du cas «Accorder l'accès à un DSE». 71
IV.11Diagramme de séquence du cas «Révoquer l'accès à un DSE». 72
IV.12Diagramme de séquence du cas «Écrire dans un DSE». 73
IV.13Diagramme de séquence du cas «Consulter un DSE». 74
IV.14Diagramme de séquence du cas «Valider les blocs». 75
IV.15Diagramme de classe de notre système. 77
IV.16Choix de visiteur. 81
IV.17Page d'inscription (cas de patient). 81
IV.18Page de connexion1. 82
IV.19Page de connexion2. 82
IV.20Profil médecin traitant (liste de mes patients). 83

Liste des tableaux

IV.1 «S'authentifier». 54
IV.2 «Créer un compte». 55
IV.3 «Demander de rejoindre le réseau blockchain». 56
IV.4 «Ajouter un patient 1». 56
IV.5 «Ajouter un patient 2». 57
IV.6 «Ajouter un hôpital 1». 57
IV.7 «Ajouter un hôpital 2». 58
IV.8 «Demander l'accès à un DSE». 59
IV.9 «Accorder l'accès à un DSE». 60
IV.10«Révoquer l'accès à un DSE». 61
IV.11«Écrire dans un DSE». 62
IV.12«Consulter un DSE». 63
IV.13«Valider les blocs». 64
IV.14Dictionnaire de données 76

Introduction

Le dossier de santé électronique est à la fois, un outil de conservation des données personnelles relatives à la santé du patient, mais aussi un outil de travail du professionnel de santé qui sert de base à son travail diagnostic et thérapeutique. Ces dossiers de santé sont généralement stockés dans des bases de données traditionnelles gérées par des fournisseurs. Cela présente plusieurs problèmes de sécurité : la modification et la suppression d'un dossier de ces bases de données, on trouve aussi des données personnelles des patients volées par de nombreux hackers pour leur valeur sur le marché noir. C'est pour cette raison que la sécurité de ces dossiers devient un enjeu majeur pour la vie privée des patients et pour le secteur de la santé.

Parmi les solutions actuelles trouvées pour sécuriser les informations d'un patient et assurer la confidentialité et l'interopérabilité des données de santé, on trouve la blockchain. Grâce à sa nature décentralisée et son inaltérabilité, elle pourrait assurer l'intégrité des données de santé à travers l'ensemble des systèmes d'information.

Cette technologie est définie comme un registre distribué qui conserve un enregistrement permanent et immuable (infalsifiable) des données transactionnelles liées entre elles par une chaîne. Elle est une solution de gestion décentralisée des informations, qui pourrait renforcer la sécurité des données sensibles et offre une nouvelle manière de se faire confiance.

L'objectif de notre travail est d'élaborer un système en utilisant la technologie blockchain pour la gestion des dossiers de santé électronique des patients dans le but de remédier aux différentes failles de sécurité des applications de gestion dossiers de santé centralisées.

A cet effet, nous avons réparti notre mémoire en quatre chapitres principaux, le premier chapitre est consacré à la présentation détaillée de la blockchain et de quelques généralités. Le deuxième chapitre présente les principaux travaux de recherche des systèmes qui utilise la blockchain dans le domaine médical et une synthèse des points essentiels cités dans chaque travail traité. Dans le chapitre trois, nous avons abordé la conception de notre système d'une manière générale et les différentes parties qui le construit.

Enfin le quatrième chapitre est dédié à la conception à base UML et la réalisation de notre système, où nous allons présenter l'environnement de développement et les outils et langages de programmation utilisés dans notre application. Nous terminerons par une conclusion générale et des perspectives à notre travail.

Chapitre I

Généralités

1 Introduction

Dans ce premier chapitre nous allons introduire les généralités et les concepts de base de la technologie blockchain, dont un bref historique, son principe, ses caractéristiques, les différentes technologie combinées à la blockchain, son fonctionnement, les différentes couches qui la constituent, ces types et quelques cas d'utilisation.

2 Historique

La technologie de la blockchain est apparue dans les années 1991 quand les chercheurs Stuart Haber et W. Scott Stornetta [1] ont introduit une solution informatique, permettant l'horodatage [1] des documents numériques pour que ceux-ci ne soient jamais antidatés ou altérés. Leur système utilisait une blockchain sécurisée cryptographique pour stocker des documents horodatés.

En 1992, le protocole dit «arbre de Merkle» a été introduit au fonctionnement, pour rendre le système plus efficace en permettant à plusieurs documents d'être rassemblés en un seul bloc. Cependant, cette technologie tomba dans l'oubli, et le brevet expire en 2004, quatre ans avant la création du Bitcoin.

En 2004, l'informaticien et activiste cryptographique Hal Finney [1], lance un système appelé RPoW(«Reusable Proof Of Work» pour «Preuve de travail réutilisable»). Le système fonctionnait en recevant un jeton preuve du travail non échangeable basé sur le système Hashcash, celui-ci créait en retour un jeton possédant une signature RSA qui pouvait ensuite être transféré de personne en personne. Le RPoW a résolu le problème de la double dépense [2] en conservant un registre de la propriété des jetons, enregistré sur un serveur de confiance, conçu pour permettre à n'importe quel utilisateur à travers le monde de vérifier son exactitude et son intégrité en temps réel. On peut considérer le RPoW comme un premier prototype et une première étape dans l'histoire des crypto-monnaies.

1. L'horodatage est un mécanisme qui consiste à associer une date et une heure à un événement, une information ou une donnée informatique.

2. Double dépense : un bitcoin dépensé dans une transaction ne peut pas être dépensé une deuxième fois dans une transaction qui serait diffusée ultérieurement sur le réseau.

Le Bitcoin, l'idée cruciale de ce concept est de mettre en place un outil de paiement électronique, sous la forme de jetons (ou tokens), à l'aide d'un réseau décentralisé pair à pair (peer-to-peer), autrement dit de créer une monnaie de l'internet : le bitcoin[2].

Le réseau Bitcoin est basé sur l'algorithme de preuve de travail HashCash, mais au lieu d'utiliser une fonction informatique de confiance comme le RPoW, la protection contre la double dépense est assurée par un protocole pair à pair décentralisé afin de suivre et de vérifier les transactions. Les bitcoins sont "minés" en tant que récompense, en utilisant le mécanisme de preuve du travail, par des mineurs individuels et les transactions sont ensuite vérifiées et validées par les nœuds décentralisés dans le réseau [1].

Le 3 janvier 2009, le Bitcoin est né quand le premier bloc de Bitcoin est miné par Satoshi Nakamoto [1], le bloc offrait une récompense de 50 Bitcoins. Le premier destinataire de Bitcoin fut Hal Finney[1], qui a reçu 10 Bitcoins de la part de Satoshi Nakamoto dans la première transaction mondiale de Bitcoins, le 12 janvier 2009 [1].

L'Ethereum en 2013, le programmeur et co-fondateur du Bitcoin Magazine, Vitalik Buterin[1] déclara que le Bitcoin avait besoin d'un langage de script pour construire des applications décentralisées. Mais il n'a pas réussi à trouver un accord au sein de la communauté bitcoin, alors il lança le développement d'une nouvelle plate-forme informatique distribuée et basée sur la Blockchain : l'Ethereum, dotée d'une fonctionnalité de script appelée «smart contracts» (des contrats intelligents)[1].

Les smart contrats sont des programmes ou des scripts, utilisés pour faire une transaction si certaines conditions sont réunies. Les smart contrats sont écrits dans des langages de programmation spécifiques et compilés en bytecode, qui est une machine virtuelle «Turing-complet[3]» décentralisée (EVM pour Ethereum Virtual Machine) pouvant ensuite les lire et les exécuter. Les développeurs ont aussi la possibilité de créer et de publier des applications fonctionnant sur la blockchain Ethereum. Ces applications sont généralement appelées DApps (Applications décentralisées) et il en existe déjà des centaines, dont des plateformes de réseaux sociaux, des applications de paris sportifs ainsi que des échanges financiers [1].

La crypto-monnaie de l'Ethereum s'appelle l'Ether, elle peut être transférée entre des comptes et est utilisée pour payer les frais, engendrés par la puissance de calcul informatique consacrée à l'exécution des smart contrats [1].

3 Principe de la blockchain

Une blockchain, ou chaîne de blocs, est définie comme une base de données distribuée ou un registre distribué (ledger) qui conserve un enregistrement permanent et immuable (infalsifiable) des données transactionnelles liées entre elles par une chaîne.

Une blockchain est un système totalement décentralisé et basé sur un réseau pair à pair.

3. La machine de Turing est une idée théorique et abstraite qui peut simuler n'importe quel algorithme qui peut être construit logiquement. Cela signifie qu'une machine de Turing peut résoudre n'importe quel problème si elle peut être codée.

Chaque objet du réseau conserve une copie du registre afin d'éviter d'avoir un point unique de défaillance. Toutes les copies sont mises à jour et validées simultanément. Cette technologie peut être explorée dans de nombreux cas d'utilisation et utilisée comme un moyen sécurisé de gestion et protection de toutes sortes de données (monétaire ou pas).

Le registre est composé d'un ensemble de blocs. Chaque bloc contient deux parties. La première partie représente le corps du bloc. Il contient les transactions, appelées également faits (facts), que la base de données doit enregistrer. Ces faits peuvent être des transactions monétaires, des données médicales, des informations industrielles, des logs systèmes, etc. La deuxième partie est l'en-tête (header) du bloc. Ce dernier contient des informations concernant le bloc tel que l'horodatage (timestamp), l'hach des transactions, etc. Ainsi que le hachage du bloc précédent. De ce fait, l'ensemble des blocs existants forme une chaîne de blocs liés et ordonnés. Plus la chaîne est longue, plus il est difficile de la falsifier.

En effet, si un utilisateur malicieux veut modifier ou échanger une transaction sur un bloc :

1. Il doit modifier tous les blocs suivants, puisqu'ils sont liés par leurs hachs ;
2. Ensuite, il doit changer la version de la chaîne de blocs que chaque objet participant stocke [20].

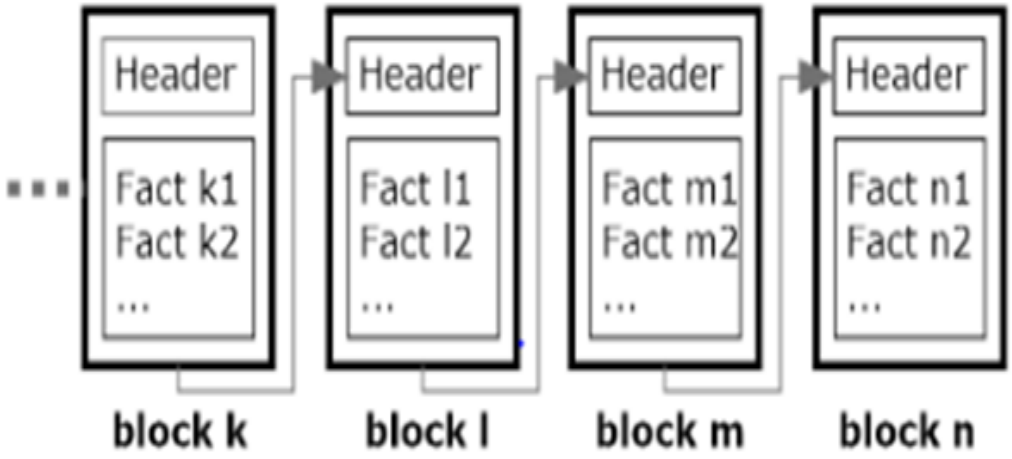

FIGURE I.1 – Exemple simplifié d'une blockchain [20].

4 Les caractéristiques de la blockchain

La technologie Blockchain se caractérise principalement de six éléments majeurs :

- **Elle est décentralisée :** la blockchain contient un système de bases de données décentralisé avec un contrôle en libre accès pour tous ceux qui sont connectés au réseau. Les données peuvent être consultées, surveillées, stockées et mises à jour sur plusieurs systèmes. Ces données ne sont pas toutes regroupées dans le serveur d'un intermédiaire central, mais au contraire « distribuées », c'est-à-dire hébergées chez chaque participant ;

il n'y a donc pas d'autorité unique pouvant approuver les transactions ou définir des règles spécifiques pour que les transactions soient acceptées. Cela signifie que la confiance est énorme, car tous les participants du réseau doivent parvenir à un consensus pour accepter les transactions [20].

- **Elle est transparente :** le registre est l'historique des transactions consultables en permanence tous les membres du réseau. Il est ainsi possible d'assurer la traçabilité intégrale d'un actif ou d'un produit ayant fait l'objet d'une transaction via une blockchain. Un participant intervient sous pseudonyme, mais toutes ses opérations sont traçables [28].

- **Elle est sécurisée :** la base de données peut uniquement être étendue et les enregistrements précédents ne peuvent pas être modifiés (le coût est très élevé si quelqu'un essaie de modifier les enregistrements précédents)[20].

- **Elle est immuable :** ces enregistrements sont dits immuables, une fois stockés, ils deviennent archivés pour toujours et ne peuvent pas être modifiés facilement sans le contrôle simultané de plus de 51 des nœuds du réseau. Le système cryptographique de validation garantit qu'il est quasiment impossible de réécrire une transaction une fois son bloc validé (personne n'a réussi à le faire depuis la création du Bitcoin) [20].

- **Autonome :** le système blockchain est indépendant et autonome, ce qui signifie que chaque nœud du système blockchain peut accéder aux données, les transférer, les stocker et les mettre à jour en toute sécurité, ce qui les rend fiables et exemptes de toute intervention externe [20].

- **Open source :** la technologie de la blockchain est formulée de manière à fournir un accès open source à toutes les personnes connectées au réseau. Cette polyvalence inimitable permet à quiconque non seulement de vérifier publiquement les enregistrements, mais également de développer diverses applications [20].

- **Anonyme :** lorsque le transfert de données à lieu entre nœuds, l'identité de l'individu reste anonyme, ce qui en fait un système plus sécurisé et fiable [20].

5 Les différentes technologies combinées dans la blockchain

La blockchain a combinée les technologies suivantes :

- Le réseau pair à pair ;
- La cryptographie ;
- Le consensus (mécanismes de validation de blocs).

5.1 Généralités sur le réseau pair à pair

En informatique, un réseau pair à pair est constitué d'un groupe d'appareils qui stockent et partagent collectivement des fichiers. Chaque participant (nœud) agit comme un pair individuel. Généralement, tous les nœuds ont une puissance égale et exécutent les mêmes tâches [16].

En substance, un système pair à pair est maintenu par un réseau distribué d'utilisateurs. Habituellement, ils n'ont pas d'administrateur ou de serveur central parce que chaque nœud

possède une copie des fichiers - agissant à la fois en tant que client et en tant que serveur pour d'autres nœuds. Ainsi, chaque nœud peut télécharger ou envoyer des fichiers à d'autres nœuds. C'est ce qui différencie les réseaux pair à pair des systèmes clients/serveurs plus traditionnels, dans lesquels les périphériques clients téléchargent des fichiers à partir d'un serveur centralisé [16].

Sur les réseaux pair à pair, lorsqu'un nœud agit en tant que client, il télécharge des fichiers à partir d'autres nœuds du réseau. Puis, lorsqu'il travaille en tant que serveur, il devient la source à partir de laquelle d'autres nœuds peuvent télécharger des fichiers. En pratique, les deux fonctions peuvent être exécutées en même temps [16].

Chaque nœud stocke, transmet et reçoit des fichiers, les réseaux pair à pair tendent à gagner en vitesse et en efficacité au fur et à mesure que leur communauté d'utilisateurs s'accroît. De plus, leur architecture distribuée rend les systèmes pair à pair très résistants aux cybers-attaques [4]. Contrairement aux modèles traditionnels, les réseaux pair à pair n'ont pas de point de défaillance unique [16].

5.2 Outils cryptographiques

Des outils cryptographiques sont utilisés pour garantir l'intégrité de la blockchain, l'identité du participant, l'authenticité des transactions et la confidentialité du contenu [4].

5.2.1 Fonction de hachage

Une fonction de hachage est une fonction qui va calculer une empreinte (ou signature) unique à partir des données fournies.
Cette fonction doit respecter les règles et propriétés suivantes :

- La longueur de l'empreinte doit être toujours la même.
- Il n'est pas possible de trouver les données originales à partir des empreintes : les fonctions de hachage ne fonctionnent que dans un seul sens.
- Il ne doit pas être possible de prédire une signature.
- Et enfin, évidemment pour des données différentes : les signatures doivent être différentes.

SHA256, SHA512, SHA1 et MD5 sont des exemples de fonctions de hachage [8].

La fonction de hachage est notamment utilisée pour chaîner les blocs entre eux. Ainsi, l'entête d'un bloc contient le «hach», résultat de l'application d'une fonction de hachage (hash function), du bloc précédent. Cet hach est un élément clé de l'intégrité de la blockchain. Ainsi, si un attaquant du système modifie le contenu d'un bloc bm, n'importe qui peut le détecter en calculant l'hach du bloc, et en comparant ce résultat avec l'hach stocké dans le bloc suivant bm+1 pour voir qu'il existe une incohérence [4].

5.2.2 Identité

Une identité sur la blockchain est définie par une adresse numérique, dérivée d'une paire de clés : une clé privée qui n'est connue que par son propriétaire et une clé publique destinée à être partagée avec d'autres utilisateurs. D'un point de vue technique, ces deux clés correspondent

4. Une cyber-attaque est une atteinte à des systèmes informatiques réalisée dans un but malveillant. Elle cible différents dispositifs informatiques.

à des nombres hexadécimaux et sont liées entre elles par une fonction mathématique. Ces clés sont utilisées pour l'identification et pour la signature des transactions. Perdre sa clé privée revient à perdre son identité sur la blockchain [4].

5.2.3 Signature

Lorsqu'un utilisateur enregistré souhaite ajouter une transaction à la blockchain, il l'envoie au réseau afin que les mineurs puissent la recevoir, la vérifier et l'ajouter à leur bloc. Mais pour être vérifiées, les transactions (qui sont des données) doivent être signées afin que le mineur puisse identifier incontestablement l'expéditeur. La signature est générée par une fonction cryptographique qui prend en entrée les données d'origine et la clé privée de l'expéditeur, et fournit en sortie une signature. Cette signature est vérifiable par toute personne utilisant une autre fonction qui vérifie la cohérence entre les données, la signature et la clé publique. Par conséquent, lors de l'envoi de données, l'expéditeur ajoute également la signature correspondante et sa clé publique [4].

5.3 Le minage et les algorithmes de consensus

5.3.1 Le minage

- **Définition :**

  Le minage dans les blockchains est l'un des éléments clés dans la décentralisation des blockchains, leur permettant de fonctionner de pair à pair, sans avoir besoin d'une autorité centrale tierce.

  Il s'agit d'un processus dans lequel les transactions entre utilisateurs sont validées et ajoutées au registre publique de la blockchain par les mineurs. Pour les mineurs les plus rapides, ces opérations de validation leurs permettent d'être récompensés.

- **Son fonctionnement :**

  Un mineur est un nœud du réseau qui collecte les transactions et qui les combine pour les organiser en blocs. Chaque fois que des transactions sont effectuées, les nœuds mineurs reçoivent et vérifient les transactions, les ajoutent au «pool» de mémoire (en informatique, un pool est un ensemble de données réutilisable) et commencent à les assembler dans un bloc de plusieurs transactions.

  La première étape du processus de minage d'un bloc consiste à hacher chaque transaction dans le pool de mémoire.

  Avant de commencer le processus, le nœud mineur ajoute une transaction par laquelle il s'envoie lui-même la récompense d'extraction. Cette transaction est appelée transaction «coinbase», c'est une transaction dans laquelle des tokens sont créés «à partir de rien» et, dans la plupart des cas, il s'agit de la première transaction d'un nouveau bloc.

  Une fois que chaque transaction est hachée, ces hachages sont ensuite organisés en ce que l'on appelle un arbre de Merkle ou arbre de hachage, ce qui signifie que les hachages sont organisés en paires, puis hachés à nouveau jusqu'à ce que «le sommet de l'arbre» soit atteint, ce que l'on appelle également un «root hash» ou encore une racine de Merkle.

Le root hash, le hachage du bloc précédent ainsi qu'un nombre aléatoire sont ensuite placés dans l'en-tête du bloc. L'en-tête de bloc est ensuite elle-même haché, produisant une sortie qui servira d'identifiant pour les blocs. L'identificateur de blocs doit être inférieur à une certaine valeur cible définie par le protocole. En d'autres termes, le hachage de l'en-tête de bloc doit commencer par un certain nombre de zéros.

Cette valeur cible «également appelée difficulté de hachage» s'échelonne, garantissant que le taux de création de nouveaux blocs reste proportionnel à la quantité de puissance de hachage du réseau.

Les mineurs continuent à hacher l'en-tête encore et encore en parcourant le nonce [5] jusqu'à ce qu'un des mineurs du réseau produise un hachage valide. Quand un hachage valide est trouvé, le nœud fondateur diffusera le bloc sur le réseau. Tous les autres nœuds vérifieront si le hachage est valide, puis ajouterons le bloc dans leur copie de la chaîne de blocs et passeront ensuite à l'extraction d'un bloc suivant.

Cependant, il arrive parfois que deux mineurs diffusent un bloc valide en même temps et que le réseau se retrouve avec deux blocs concurrents. Les mineurs commencent à miner le bloc suivant en fonction du bloc qu'ils ont reçu en premier. La compétition entre ces blocs se poursuivra jusqu'à ce qu'un nouveau bloc soit miné, et ce peu importe le bloc initial auquel il ajoute une suite. Le bloc qui est abandonné s'appelle un bloc orphelin ou un bloc périmé. Les mineurs de ce bloc reviendront ensuite au minage de la chaîne du bloc gagnant [21].

5.3.2 Arbre de Merkle

L'arbre de Merkle «Merkle Tree» est une méthode permettant de structurer des données en vue d'y accéder et d'en vérifier la véracité plus rapidement. Le nom vient du fait que cette méthode organise les données en les regroupant par deux, donnant ainsi la forme d'un arbre inversé. En effet, les transactions sont regroupées par groupes de deux, un hachage est ensuite appliqué à ce groupe. Les groupes sont regroupés par groupes de deux puis soumis au même procédé jusqu'au dernier hachage appelé la racine «Merkle Root» qui est ajoutée comme référence dans l'en-tête du bloc.

Cette organisation des transactions au sein d'un bloc va permettre aux mineurs lors de la vérification des nouvelles transactions, de remonter très rapidement vers la dernière transaction concernée et ainsi vérifier s'il existe bien une transaction d'un montant supérieur à celui qui doit être dépensé dans la nouvelle transaction. En d'autres termes, un arbre de Merkle permet d'identifier très rapidement les transactions qui appartiennent à un même membre. Si l'ensemble de ces transactions ou une seule suffit à couvrir la dépense réalisée par ce membre alors la transaction de ce membre sera autorisée [6].

5. Le nonce est un nombre arbitraire générée par le mineur pour modifier le hachage a fin de produire un hachage au dessous des difficultés cible.

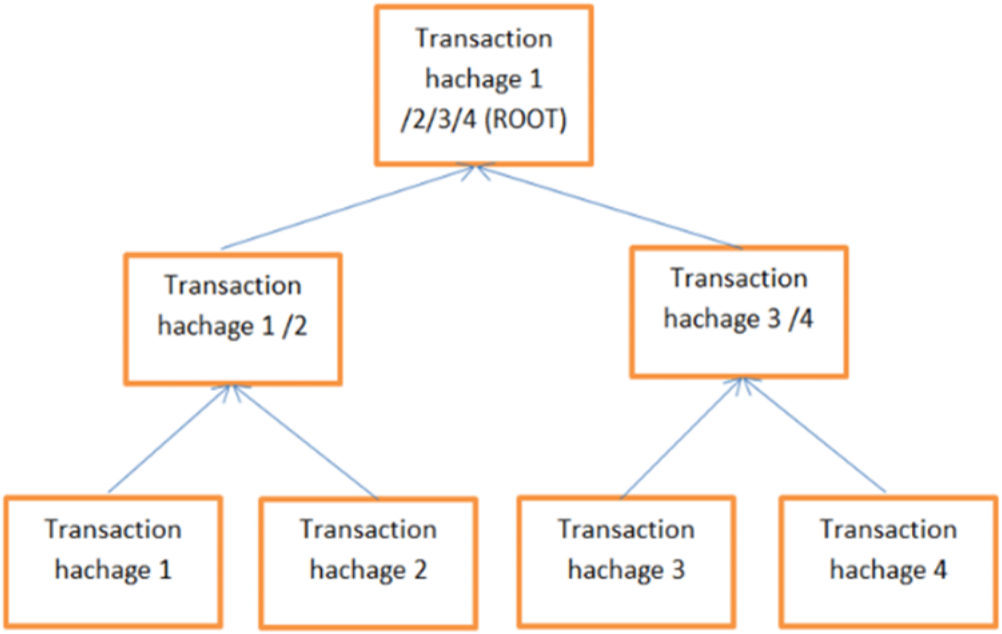

FIGURE I.2 – Arbre de Merkel.

5.3.3 Les algorithmes de consensus

Dans le contexte des crypto-monnaies, les algorithmes de consensus sont un élément crucial de chaque réseau blockchain, car ils sont responsables de maintenir l'intégrité et la sécurité de ces systèmes distribués. Le premier algorithme de consensus crypto-monnaie qui a été créé était le Proof of Work (PoW), conçu par Satoshi Nakamoto [17] et mise en œuvre sur Bitcoin comme un moyen de surmonter les erreurs byzantines [6] [17].

Un algorithme de consensus peut être défini comme le mécanisme par lequel un réseau blockchain parvient à un consensus. Les blockchains publiques sont construites en tant que systèmes distribués et, puisqu'ils ne dépendent pas d'une autorité centrale, les nœuds distribués doivent se mettre d'accord sur la validité des transactions. C'est là que les algorithmes de consensus entrent en jeu. Ils s'assurent que les règles du protocole soient respectées et que toutes les transactions aient lieu de manière fiable, de sorte que les monnaies (coins) ne puissent être dépensées qu'une seule fois [17].

Il existe plusieurs types d'algorithmes de consensus. Les implémentations les plus courantes sont PoW et PoS (Proof of Stake). Chaque solution présente des avantages et des inconvénients lorsqu'elle tente d'équilibrer la sécurité, la fonctionnalité et la scalabilité [17].

1. **Preuve de travail (PoW)**

   PoW était le premier algorithme de consensus à avoir été créé. Il est utilisé par Bitcoin et de nombreuses autres crypto-monnaies. L'algorithme de preuve de travail est une partie essentielle du processus de minage [17].

6. On dit que les erreurs sont byzantines lorsque le comportement d'un processus n'est pas celui spécifié par le protocole. Par exemple un processus qui tombe en panne est un cas particulier de processus byzantin.

Le minage via PoW implique de nombreuses tentatives de hachage, donc plus de puissance de calcul, ce qui signifie plus d'essais par seconde. En d'autres termes, les mineurs avec un taux de hachage élevé ont de meilleures chances de trouver une solution valable (alias hachage de bloc) pour le bloc suivant. L'algorithme de consensus PoW garantit que les mineurs ne peuvent valider un nouveau bloc de transactions et l'ajouter à la blockchain que si les nœuds distribués du réseau parviennent à un consensus et conviennent que le hachage de bloc fourni par le mineur est une preuve de travail valide [17].

2. **Preuve d'enjeu (PoS)**

L'algorithme de consensus PoS a été développé en 2011 en tant qu'alternative au PoW. Bien que le PoW et le POS partagent des objectifs similaires, ils présentent des différences et des particularités fondamentales. Surtout lors de la validation de nouveaux blocs [17].

En quelques mots, l'algorithme de consensus PoS remplace le minage PoW par un mécanisme dans lequel les blocs sont validés en fonction du nombre de coins verrouillés en enjeu par les participants. Le validateur de chaque bloc est déterminé par un investissement de la crypto-monnaie en elle-même et non par la quantité de puissance de calcul allouée. Chaque système PoS peut implémenter l'algorithme de différentes manières, mais en général, la blockchain est sécurisée par un processus d'élection pseudo-aléatoire prenant en compte la richesse du nœud et l'âge des coins (le temps pendant lequel les pièces sont verrouillées) auxquels s'ajoute un facteur de randomisation [17].

La blockchain Ethereum est actuellement basée sur un algorithme PoW, mais dans le futur, le protocole Casper [29] sera par la suite mis en place pour basculer le réseau d'un algorithme PoW vers un algorithme PoS afin d'augmenter sa scalabilité [17].

3. **Preuve d'autorité (PoA)**

Dans un consensus basé sur la PoA, les blocs et les transactions sont validés par des comptes approuvés à l'avance, que l'on appelle en anglais «validator». Le processus est automatique et mis à part le fait de vérifier que l'ordinateur n'est pas compromis, il n'y a rien d'autre à faire.

Pour devenir un validateur dans le consensus PoA, il faut que votre identité soit formellement vérifiée et affichée sur la blockchain. Car c'est votre identité et votre réputation qui sont mises en jeu, plutôt que votre puissance de calcul ou votre richesse.

Il y a donc 3 piliers sur lesquels ce consensus repose :
- Un moyen de certifier sans aucun doute possible l'identité d'une personne ;
- Un procédé suffisamment difficile à achever pour devenir validateur, afin que la perte de ce titre représente un problème majeur pour le validateur déchu ;
- Un procédé de sélection uniformisé pour tous les validateurs, afin que chacun des validateurs puisse faire confiance aux autres.

En créant un système de réputation lié à une identité, les validateurs sont incités à continuer de valider les transactions de la manière la plus efficace, honnête et transparente possible. S'ils ne le font pas, alors leur identité pourrait être associée à une réputation négative, ce qui leur ferait perdre ainsi leur rôle de validateur difficilement acquis.

Toutefois, ce système présente des inconvénients quelque peu extrêmes, dont le plus important est sa centralisation. En effet, si les validateurs doivent être choisis, c'est donc qu'une autorité centrale contrôle le réseau indirectement. Ce système de consensus est donc tout à fait adapté à des blockchains mises en place par des administrations.

Mais il n'est pas du tout adapté à des crypto-monnaies, pour lesquels la décentralisation est d'une importance capitale. En effet, il n'est impossible qu'une crypto-monnaie véritablement décentralisée souffre des mêmes problèmes que le système bancaire et monétaire mondial. C'est pour cette raison précise que la notion de décentralisation était au cœur des travaux de Satoshi Nakamoto [41].

6 Le fonctionnement de la blockchain

Pour une première approche du fonctionnement des blockchains, le plus facile est de raisonner avec une blockchain purement monétaire. On peut prendre l'exemple de Bitcoin, ou d'une blockchain avec des jetons "simples", pour laquelle une transaction se résume en fait à trois informations : qui donne quoi à qui [5].

1. Oussama souhaite envoyer 1 Bitcoin à Ilham sur le réseau Bitcoin, chacun possède une adresse publique, appelée «clé publique» (l'équivalent d'un RIB) : une suite d'environ 34 caractères contenant des chiffres et des lettres en minuscules ou majuscules (par exemple :13o7TCoNWbaqYp9g89w1gHrZ7GvvKftRsd, pour l'un et 16Xgsii16x4icN7yjQgXX648Lf4LxRsd19 pour l'autre) ;
2. Oussama signe la transaction avec sa clé privée : une autre suite de chiffres et de lettres, cette fois-ci confidentielle, qui autorise le versement de l'argent (les points 1 et 2 correspondent à l'étape 1 du schéma de la figure I.3) ;
3. La transaction est alors entrée à la suite d'autres transactions, dans ce qui est appelé un bloc (une grappe de plusieurs centaines, voire milliers, de transactions). Pour chaque transaction, différentes informations apparaissent et seront donc consultables par tous les membres du réseau :
   - Les clés publiques d'Oussama et d'Ilham [20] ;
   - Le nombre de Bitcoins transférés de Oussama à Ilham : 1 [20] ;
   - L'heure précise, et la date pendant laquelle l'opération a eu lieu (2019-07-12, 09 : 41 : 21) [20].
4. Chaque bloc, en plus de toutes ces transactions, contient un hach du bloc précédent : un nombre, qui correspond à l'unique résultat possible que l'on obtient quand on entre les informations du bloc précédent dans une fonction de hachage. Cela permet de noter dans chaque bloc le hach du bloc précédent. Les blocs sont donc enchaînés les uns aux autres (les points 3 et 4 correspondent à l'étape 2 du schéma de la figure I.3) [20] ;
5. Les membres du réseau qui le souhaitent vont alors entrer en compétition pour valider le bloc. Ils y sont incités par une récompense : celui qui aura le privilège d'effectuer cette validation aura automatiquement des bénéfices (par exemple 12,5 nouveaux Bitcoins pour le cas de bitcoin), qui seront générés pour l'occasion par le système. Pour gagner ce droit, tous les participants vont vérifier la véracité des informations contenues dans toutes les transactions du bloc : Est-ce que Oussama possède bien 1 Bitcoin au vu de toutes les transactions qu'il a déjà réalisées ? N'utilise-t-il bien ce Bitcoin qu'une seule fois ? Est-ce que les autres transactions sont également justes ? Mais ils vont aussi devoir résoudre un problème mathématique complexe, en utilisant les capacités de calcul de leur ordinateur (PoW pour le cas de bitcoin). Chaque ordinateur va tester des solutions jusqu'à trouver

la bonne. Le premier à réussir pourra valider le bloc et gagner la récompense. Le système a été conçu pour qu'une solution soit trouvée au bout de 10 minutes en moyenne (ce qui impose un léger délai à la sécurisation de la transaction)
(ce point 5 correspond à l'étape 3 du schéma de la figure I.3)[20] ;

6. Le bloc est alors validé. Il est ajouté à la blockchain qui est mise à jour sur les ordinateurs de chaque participant
(ce point correspond à l'étape 4 du schéma de la figure I.3) [20] ;
7. Les participants n'acceptent ce nouveau bloc que si les transactions qu'il contient sont valides. Si le bloc est valide, ils expriment leur accord en travaillant sur le bloc suivant selon le même processus, et y inscrivant l'hach du bloc validé. C'est pourquoi on parle de système de consensus [20].
8. Ilham a désormais reçu le Bitcoin envoyé par Oussama. En récompense du travail fourni, le mineur reçoit une certaine quantité de bitcoin créé pour l'occasion (pour le cas de la blockchain Bitcoin)
(les points 7 et 8 correspondent aux étapes 5 et 6 du schéma de la figure I.3) [20].

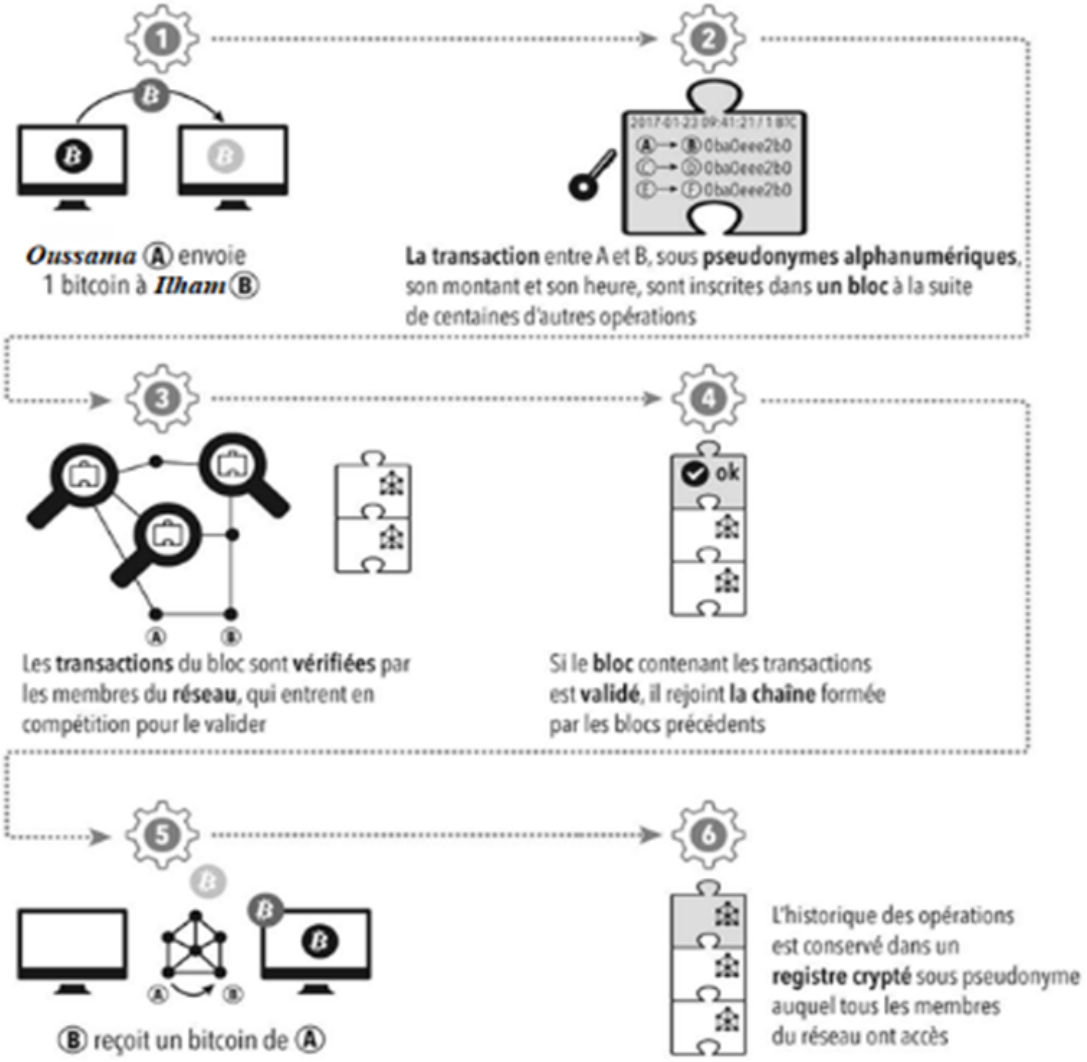

FIGURE I.3 – Schéma explicatif de fonctionnement de la blockchain (bitcoin) [20].

7 Couches de la blockchain

La blockchain contient différentes couches :

7.1 Couche Internet

Cette couche est la racine de toute la pile. C'est la combinaison de tous les réseaux tels que les appareils IoT, les smartphones, les ordinateurs, etc. Ils reposent tous sur la suite de protocoles Internet. Cette couche détermine la manière dont les informations doivent être transmises, adressées, mises en paquets, etc. La version initiale d'Internet connue sous le nom de Web Version 1 ou Web (WWW) a été introduite en 1989. Elle utilisait l'IoI (Internet of Information. Il a défini un cadre spécifique et les URLs vers d'autres ressources. Après plus de dix ans, la technologie a fait l'objet d'une autre mise à niveau et est devenue la version 2 du Web. Cette version a commencé à utiliser l'Internet d'Interactions, principalement utilisé sur les sites de médias sociaux et de commerce électronique. Avec cette nouvelle version, la connexion pair à pair s'est développée avec le temps et de nombreux groupes importants ont joué le rôle de support pour cela. Ces organisations ont profité de l'utilisation de publicités ou d'autres ressources d'abonnement. Cette nouvelle popularité a donné naissance à la dernière version Web 3. La plupart des gens considèrent cela comme Internet de valeur. La technologie blockchain est l'un des principaux initiateurs de cette nouvelle mise à niveau. Cette même technologie peut remplacer les applications de médias sociaux ou de covoiturage. Cela peut en effet faire du Web un endroit décentralisé après tout [20].

7.2 Couche de protocole Blockchain

Dans cette couche se trouvent les protocoles nécessaires à la blockchain. Les nœuds exécuteront des protocoles, ajouteront des données sur la blockchain et garantiront des transactions transparentes. Le protocole offre une quantité suffisante de commentaires à l'utilisateur [20].

7.3 Couche d'application

La couche application du système comprend toute autre fonctionnalité en dehors de simples crypto-monnaies. Les contrats intelligents et les applications décentralisées se trouvent dans cette couche. La couche permet le développement d'applications décentralisées fonctionnant au-delà de l'algorithme de consensus. Les contrats intelligents peuvent fournir des utilisations plus complexes. Les jetons sont absolument essentiels pour le réseau et pour le développement des DApp eux-mêmes [20].

7.4 Couche Expérience utilisateur(UX)

Cette couche donne accès à l'ensemble du réseau et aux différentes applications de la blockchain [20].

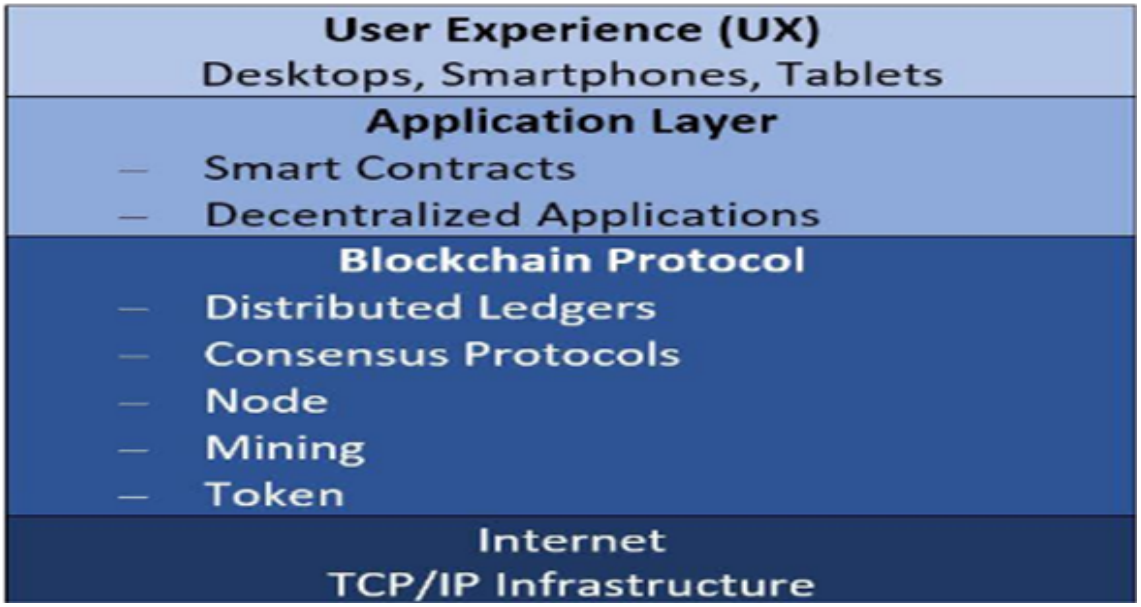

FIGURE I.4 – Les Couches de la blockchain[20].

8 Types de blockchain

Il existe principalement deux types de blockchains, blockchain privée et publique. Cependant, il existe également plusieurs variantes, telles que les blockchains consortium et hybride.

8.1 Blockchain publique

Une blockchain publique est un système de grand livre distribué non restrictif et sans permission. Toute personne ayant accès à Internet peut se connecter sur une plate-forme blockchain pour devenir un nœud autorisé et faire partie du réseau blockchain. Un nœud ou un utilisateur faisant partie de la blockchain publique est autorisé à accéder aux enregistrements actuels et anciens, à vérifier des transactions ou à effectuer une preuve de travail pour un bloc entrant et à effectuer une extraction. L'utilisation la plus élémentaire des blockchains publiques est l'extraction et l'échange de cryptomonnaies. Ainsi, les blockchains publiques les plus courants sont les blockchains Bitcoin et Ethereum. Les blockchains publiques sont généralement sécurisés si les utilisateurs respectent strictement les règles et les méthodes de sécurité [7].
Exemple : Bitcoin [22] et Ethereum [23] citées précédemment.

8.2 Blockchain privée

Une blockchain privée est une blockchain restrictive ou avec une autorisation, ne fonctionnant que dans un réseau fermé. Les blockchains privées sont généralement utilisées au sein d'une organisation ou d'une entreprise où seuls les membres sélectionnés sont membres d'un réseau de blockchain. Le niveau de sécurité, les autorisations et l'accessibilité sont entre les mains

de l'organisation qui contrôle. Ainsi, les blockchains privées ont une utilisation similaire à une blockchain publique, mais ont un réseau restreint et restrictif. Des réseaux de blockchain privées sont déployés pour le vote, la gestion de la chaîne d'approvisionnement, l'identité numérique, le domaine médical, etc. [7].
Exemples de blockchains privées : Hyperledger [24], Passcare [25].

8.3 Blockchain consortium

Un consortium blockchain est un type semi-décentralisé dans lequel plusieurs organisations gèrent un réseau de blockchain. Ceci est contraire à ce que nous avons vu dans une blockchain privée, gérée par une seule organisation. Plusieurs organisations peuvent agir en tant que nœud dans ce type de blockchain et échanger des informations ou procéder à une extraction. Les blockchains du consortium sont généralement utilisés par les banques, les organismes gouvernementaux, etc. [7].
Exemples de blockchain de consortium : B3i [26].

8.4 Blockchain hybride

Une blockchain hybride est une combinaison de la blockchain privée et publique. Elle utilise les fonctionnalités des deux types de blockchains, à savoir que l'on peut avoir un système basé sur une autorisation privée ainsi qu'un système publique sans autorisation. Avec un tel réseau hybride, les utilisateurs peuvent contrôler qui a accès à quelles données stockées dans la blockchain. Seule une partie sélectionnée de données ou d'enregistrements de la blockchain peut être rendue publique, le reste étant confidentiel sur le réseau privé. Une transaction dans un réseau privé d'une blockchain hybride est généralement vérifiée au sein de ce réseau. Mais les utilisateurs peuvent également la publier dans la blockchain publique pour la vérifier. Les blockchains publiques augmentent le hachage et impliquent plus de nœuds pour la vérification [7].

Aergo [27] est un exemple de blockchain hybride.

9 Domaine d'application

9.1 Paiement numérique et monnaie virtuelle

Pour ce qui est d'envoyer de l'argent dans le monde entier, la technologie blockchain s'est déjà révélée très efficace. Envoyer des crypto-monnaies à des amis, de la famille ou à d'autres personnes dans le monde coûte déjà moins cher et se fait plus rapidement que ce que proposent les banques centralisées et les solutions de paiement classiques [18].
Exemple de blockchain utilisé : bitcoin.

9.2 Chaîne d'approvisionnement

Une chaîne d'approvisionnement est un réseau de personnes et d'entreprises impliquées dans la création et la distribution d'un produit ou d'un service particulier - depuis les fournisseurs initiaux jusqu'aux utilisateurs finaux et aux clients. Un système de chaîne d'approvisionnement de base implique souvent les fournisseurs de produits alimentaires ou de matières premières, les fabricants (stade de la transformation), les entreprises de logistique et les détaillants finaux.

À l'heure actuelle, le système de gestion de la chaîne d'approvisionnement souffre d'un manque d'efficacité et de transparence et la plupart des réseaux rencontrent des difficultés lorsqu'ils tentent d'intégrer toutes les parties concernées. Idéalement, les produits et les matériaux, ainsi que l'argent et les données, doivent circuler de manière fluide au cours des différentes étapes de la chaîne. Cependant, le modèle actuel rend difficile le maintien d'une chaîne logistique cohérente et efficace, ce qui a un impact négatif non seulement sur la rentabilité des entreprises mais également sur le prix de détail final.

Certains des problèmes les plus urgents des chaînes d'approvisionnement peuvent être résolus grâce à l'utilisation de la technologie de la blockchain, qui offre de nouvelles méthodes d'enregistrement, de transmission et de partage des données [18].

Hyperledger est une plate-forme open source de développement du blockchain. Débutée en 2015 par la fondation Linux, elle est principalement développé en langage GO. C'est une solution qui regroupe de nombreux frameworks pour développer des smart contrats ainsi que des applications basées sur la blockchain [9].

La traçabilité des fruits de mer est un exemple des chaines d'approvisionnements. La blockchain utilisée dans ce cas est déployé sur la plateforme Hyperledger Sawtooth qui permet de construire, déployer et exécuter des registres distribués dédiés spécifiquement à la certification et à la traçabilité. Un restaurateur passe par quelques partenaires distributeurs de confiance, qui s'assurent que les produits respectent les normes d'hygiène et de maintien à température. Il est compliqué de faire appel à de nouveaux producteurs et la traçabilité des produits est coûteuse. L'idée est alors de placer des capteurs directement sur les lots de fruits de mer dès la récolte, dans le bateau du pêcheur. Ces capteurs permettent à la fois de donner une identité numérique au produit, un certificat de propriété, mais également de communiquer à travers la blockchain Sawtooth les différentes informations nécessaires au contrôle de la chaîne d'approvisionnement - position géographique, température, etc. [13].

9.3 La santé

Un registre des données médicales d'un patient en est un exemple.

Plusieurs acteurs de la santé s'intéressent de près à la création d'un registre patient distribué, pouvant par exemple s'appuyer sur une architecture blockchain. Actuellement, l'information n'est pas partagée entre les différents médecins, et le patient doit lui-même rapporter les comptes rendus de ses précédentes consultations auprès de chaque nouveau spécialiste. Une mission d'autant plus difficile pour un patient non averti ne maîtrisant pas le discours médical et n'ayant pas une idée précise du contenu de son dossier.

Pour lutter contre cette déperdition d'informations et favoriser une communication plus efficiente, certains projets commencent à voir le jour. C'est notamment le cas de Passcare. Passcare est un passeport médical connecté permettant au patient d'avoir un accès complet à ses données. Il regroupe l'ensemble des données médicales et permet de connecter tous les intervenants de santé à travers le monde. Traduisible en plusieurs langues, il est accessible partout à travers le monde sans limite géographique. Les données y sont protégées et anonymes, et contrôlées par le patient. La carte est dotée d'un QR code. En cas de problème, les différents professionnels peuvent avoir accès à des informations d'urgence permettant de sauver la vie du patient en

ayant son dossier. Passcare a déjà plus de 200 000 utilisateurs en France et a suscité l'intérêt de Google et IBM [14].

9.4 L'assurance

Le secteur des assurances à tôt-fait d'appréhender la blockchain en raison des nombreux avantages qu'elle apporte, notamment avec les contrats intelligents. Ces derniers sont matérialisés par des protocoles informatiques qui simplifient, contrôlent et réalisent la négociation ou l'exécution d'un contrat, ou rendent une clause contractuelle nulle de manière très sécurisée [10].

Dans ce secteur, des assureurs de premier plan comme Aegon, Allianz et Munich Re ont uni leurs forces pour former le consortium blockchain B3i[11].

B3i est une plateforme qui a été constituée en 2018, elle crée un meilleur secteur des assurances en développant des normes, des protocoles et une infrastructure de réseau afin d'éliminer les frictions liées au transfert de risques . Les actionnaires et les participants de B3i estiment que les nouvelles technologies peuvent offrir aux consommateurs finaux d'assurance un accès meilleur et plus rapide à l'assurance [12].

9.5 Internet des objets (IoT)

La technologie IoT (Internet des objets) comporte des réseaux d'appareils connectés embarqués, équipés de capteurs et de logiciels qui leur permettent de transférer des données. Ce nouveau domaine de l'innovation redéfinit progressivement un large éventail de secteurs, de la fabrication aux soins de santé. La capacité de recueillir des données en temps réel apporte un certain nombre d'avantages pour les entreprises, leur permettant d'automatiser les processus, de stimuler la productivité et d'améliorer l'expérience client.

Cependant, l'un des plus grands défis pour les personnes qui travaillent avec la technologie de l'IoT, c'est la sécurité, chaque appareil IoT agissant comme un point d'entrée potentiel pour les pirates. En cas de violation, des informations sensibles pourraient fuiter à très grande échelle ou les appareils IoT pourraient être exposés à un risque de détournement par des pirates. L'utilisation de la technologie blockchain comme base pour les appareils IoT réduit le risque de piratage informatique en diminuant les points d'entrée potentiels [3].

10 Conclusion

Au cours de ce chapitre, nous avons défini le principe de la blockchain, ses caractéristiques, les différentes technologies combinées dans la blockchain, ensuite les couches qui la constituent, puis nous avons donné les types de blockchain qui existent et enfin quelques champs d'applications.

Dans le chapitre suivant, nous allons présenter quelques travaux concernant des systèmes informatiques basés sur la technologie blockchain pour la gestion de dossier de santé électronique (DSE).

Chapitre II

Etude de l'existant

1 Introduction

Avant d'entamer la partie conception, nous avons fait une recherche bibliographique sur des travaux concernant des systèmes informatiques basés sur la technologie blockchain pour la gestion de dossier de santé électronique (DSE), où nous avons distingué deux grandes classes de ces travaux : une classe qui se base sur des blockchains déjà existantes, et une autre classe qui ne dépend d'aucune blockchain existante.

2 Les systèmes basés sur des blockchains existantes

Dans les articles [30], [33], [34], [35] et [36], les auteurs ont proposé des systèmes qui se basent sur la blockchain Ethereum, tant dit que dans l'article [32], c'est la blockchain Hyperledger Fabric qui a été utilisé. Et dans l'article [37], ils ont utilisé une double structure de la blockchain : Hyperledger Fabric et Ethereum à la fois.

Ces systèmes traités sont à accès contrôlé, les nœuds ont besoin d'autorisation d'une entité pour accéder au réseau blockchain. Ethereum et Hyperledger Fabric sont des plateformes open source qui permettent de créer des blockchains privées et autorisées en présentant des avantages comme un contrôle accru de la confidentialité. Ces plateformes ne se limitent pas aux créations des applications monétaires mais plutôt toutes type d'application. De plus, elles sont les plus utilisées pour déployer des smart contrat, ces derniers permettent de sécuriser l'accès aux DSEs. Les avantages que représentent ces deux plateformes, les rendent les plus utilisées.

2.1 Les systèmes basés sur la blockchain Ethereum

2.1.1 Ancile

Dans [30], les auteurs ont proposé une plate-forme basée sur la blockchain pour la gestion des DSEs : Ancile, une blockchain autorisée, «permissionned» en anglais, et qui a pour but d'atteindre quatre objectifs tout en répondant aux exigences de HIPAA «Health Insurance Portability and Accountability Act» : (1) donner la propriété et le contrôle final des DSEs au patient, (2) contrôler en toute sécurité qui peut accéder aux documents et suivre la façon dont les dossiers sont utilisés, (3) permettre le transfert sécurisé des dossiers et enfin (4) réduire la capacité des acteurs non autorisés.

La blockchain Ancile stocke les hachages des références de données lors de l'envoi des informations de lien de requête réelles dans une transaction privée via HTTPS. En plus de l'utilisation des transactions privées pour assurer la confidentialité, elle utilise des proxys pour le re-cryptage, qui permettent de stocker des clés et des dossiers cryptés directement sur la blockchain, facilitant leur transfert comme les ordonnances aux pharmacies. Cela garantit également que les utilisateurs n'ont pas besoin de stocker les clés localement, ce qui permet aux patients de supprimer les autorisations d'accès s'il le désire. Ancile n'inclut aucune forme d'incitation minière au-delà de la simple utilisation du système. Ils supposent que les fournisseurs et les gouvernements ont déjà une incitation pour sécuriser les informations médicales des patients. Elle utilise les fonctionnalités de contrat intelligent pour le contrôle d'accès et tenir compte des différents rôles des patients, des prestataires et des tiers sur la blockchain. Enfin, la structure de cette blockchain se base sur un algorithme de consensus plutôt que sur une preuve de travail. Cela permet d'avoir une validation accrue lors de l'ajout de nœuds au réseau ou de la suppression des utilisateurs nuisibles.

a. La cryptographie à Ancile

Il existe trois formes différentes de chiffrement et de stockage des données chiffrées qui sont effectué à l'aide du gestionnaire de chiffrement.

- La première forme de cryptage est à clé symétrique. Ancile utilise ce type sur les fichiers plus volumineux car il est efficace et élimine la nécessité de rechiffrer les fichiers plus tard.
- La deuxième forme de chiffrement est à clé publique. Ce chiffrement est utilisé pour protéger les informations lors de la distribution et pour indiquer clairement qui peut accéder aux DSE.
- La troisième forme de cryptage est le recryptage du proxy. Ce type résout le problème du transfert des enregistrements cryptés entre les nœuds sans partager de clés symétriques et cela à l'aide d'un proxy. Le proxy est responsable de la reconstruction d'un message crypté de telle sorte qu'un autre utilisateur puisse utiliser sa clé privée pour déchiffrer le document, même s'il n'était pas chiffré à l'origine avec le document associé à la clé publique.

Dans cet article, ils ont utilisé un schéma de proxy distribué de re-cryptage en aveugle, où des parties multiples (proxys) participent au processus de rechiffrement. Pour cela, un message est chiffré avec une clé publique principale, et la clé privée associée est ensuite distribuée par morceaux aux proxys. Ce faisant, les proxys peuvent rechiffrer le message sans pouvoir déchiffrer le message complet.

b. Processus de consensus

Ancile utilise l'algorithme de consensus QuorumChain qui s'appuie sur une méthode de vote majoritaire pour parvenir à un consensus. Cet algorithme est exécuté via un contrat intelligent qui suit les votes et les nœuds éligibles. Une fois qu'un nouveau bloc possible est trouvé, les nœuds autorisés peuvent alors voter sur le bloc suivant. Pour être ajouté à la blockchain, le bloc doit recevoir la majorité des votes et le nombre de votes reçus doit être supérieur au seuil. C'est pour éviter d'ajouter plusieurs blocs avec les mêmes transactions. Pour diminuer la probabilité que plusieurs nœuds mineurs créent des blocs identiques en même temps, l'algorithme QuorumChain implémente des sessions d'expiration, une durée pseudo-aléatoire qui doit s'écouler avant qu'un nœud mineur puisse créer

un bloc.

c. Les contrats intelligents

La plateforme proposée utilise six types de smart contrats :

- **Contrat de consensus (CCO) :** est un contrat global chargé de maintenir l'opération de minage de la blockchain, l'enregistrement des utilisateurs et certaines procédures de remplacement. Le CCO stocke les adresses Ethereum des nœuds avec autorisation de vote. Pour le minage, le CCO fonctionne en utilisant l'algorithme de consensus QuorumChain. Pour l'enregistrement des utilisateurs, le CCO est utilisé pour valider les nœuds qui demandent des niveaux de classification plus élevés lorsqu'ils sont ajoutés au système. Enfin, le CCO peut également être utilisé pour écraser les nœuds qui cessent d'exister ou qui ont été jugées préjudiciables au système.
- **Contrat de classification (CCL) :** est responsable de la classification des différents niveaux de nœuds dans le système en tant que patients, prestataires ou tierces parties. Avec ces informations, le CCL peut confirmer quels nœuds sont déjà enregistrés dans le système et éviter la double inscription. De plus, le CCO étant utilisé pendant le processus d'enregistrement des nœuds pour déterminer la classification des nœuds, le CCL peut être utilisé pour vérifier l'identité d'un nœud sans répéter le processus de consensus.
- **Contrat d'historique de service(CHS) :** conserve l'historique des relations des nœuds. Un nouveau CHS est créé pour chaque nœud pendant le processus d'enregistrement. Lorsque les nœuds interagissent avec Ancile, leur CHS leur fournit une liste complète de leurs relations en matière de soins de santé. Lorsqu'un prestataire souhaite établir une relation avec un patient, le CHS du patient demandera l'autorisation avant la mise à jour.
- **Contrat de propriété(CP) :** est responsable du suivi des dossiers que les fournisseurs stockent pour les patients. Un CP est créé lorsqu'une nouvelle relation est établie entre deux nœuds. De plus, le CP contient les informations nécessaires pour que le nœud patient trouve la base de données de DSE d'un fournisseur. Le CP répertorie ensuite chaque dossier patient avec un nom de fichier, un hachage de lien de requête, un hachage du dossier lui-même et une adresse vers un contrat d'autorisations (CA).
- **Contrat d'autorisations (CA) :** est spécifique à chaque dossier et est construit par le CP lorsqu'un nouveau dossier est ajouté au système. Il est conçu pour stocker les adresses Ethereum de tous les nœuds interagissant avec un dossier, un niveau d'accès, et une clé symétrique chiffré avec la clé publique de chaque nœud. Les différents niveaux d'accès sont les suivants :
  - Lecture : Un nœud a une clé symétrique générée pour lui lorsque le dossier est d'abord ajouté ou via le proxy du re-cryptage.
  - Transfert : Un nœud peut ajouter d'autres nœuds avec un accès en lecture. Lorsqu'un nœud reçoit ce niveau, des conditions spéciales peuvent également être ajoutées.
  - Propriétaire : Un nœud a le contrôle total du CA. Il peut ajouter d'autres nœuds de n'importe quel niveau d'accès, supprimez les nœuds du PC et modifiez les niveaux d'accès pour tous les nœuds existants.
  - Aveugle : Un nœud ne reçoit que l'adresse du CA.
- **Le contrat de rechiffrement (CR) :** est utilisé pour le rechiffrement du proxy. Dans Ancile, le rechiffrement du proxy sur la blockchain nécessite qu'un groupe défini de nœuds proxy reçoive une clé publique principale avec une clé privée partagée. Un CR doit être créé chaque fois qu'un nouvel ensemble de nœuds proxy est établi.

2.1.2 MedRec

Les auteurs de [33]ont proposé MedRec, un nouveau système de gestion des enregistrements décentralisé pour gérer les DSE, en utilisant la technologie blockchain. Leur système offre aux patients un journal complet et immuable et un accès facile à leurs informations médicales à travers les fournisseurs et les sites de traitement. Le contenu du bloc représente la propriété des données et les autorisations de visionnage partagées par les membres d'un réseau privé pair à pair (P2P).

Dans MedRec, via des contrats intelligents sur une blockchain Ethereum, qui permet d'automatiser et suivre certaines transitions d'état, les relations patient-fournisseur sont enregistrées, ce qui associe un dossier médical à des autorisations de visualisation et des instructions de récupération de données pour une exécution sur des bases de données externes.

Les fournisseurs peuvent ajouter un nouveau dossier associé à un patient particulier et les patients peuvent autoriser le partage des dossiers entre les fournisseurs. Dans les deux cas, la partie qui reçoit de nouvelles informations reçoit une notification automatisée et peut vérifier l'enregistrement proposé avant d'accepter ou de rejeter les données. Les auteurs gèrent la confirmation d'identité via la cryptographie à clé publique et utilisent une implémentation de type DNS qui mappe une forme d'identification déjà existant à l'adresse Ethereum de la personne.

Un algorithme de synchronisation gère l'échange de données hors blockchain «off-chain» entre une base de données de patients et une base de données de fournisseurs, après avoir référencé la blockchain pour confirmer les autorisations via leur serveur d'authentification de base de données.

a. Les contrats intelligents

Dans cet article, Ils utilisent les contrats intelligents d'Ethereum pour créer des représentations intelligentes des dossiers médicaux existants qui sont stockés dans des nœuds individuels sur le réseau. Ils construisent les contrats pour contenir des métadonnées sur le propriétaire de l'enregistrement, les autorisations et l'intégrité des données. Leur système de blockchain met en œuvre trois types de contrats :

- **Contrat de registre (CR) :** ce contrat global mappe l'identification des participants (chaines de caractères) à leur identité d'adresse Ethereum (équivalente à une clé publique). Le CR mappe également les identités à une adresse sur la blockchain, où un contrat spécial, appelé «Summary Contract», peut être trouvé.
- **Contrat de relation patient-fournisseur (RPF) :** un contrat de relation patient-fournisseur est émis entre deux nœuds du système lorsqu'un nœud stocke et gère les dossiers médicaux de l'autre. Le RPF définit un assortiment de pointeurs de données et d'autorisations d'accès associées qui identifient les enregistrements détenus par le prestataire de soins. Chaque pointeur se compose d'une chaîne de caractères représentant une requête qui, lorsqu'elle est exécutée dans la base de données du fournisseur, renvoie un sous-ensemble de données patient.
- **Contrat de sommaire(CS) :** ce contrat permet aux participants du système de localiser leurs antécédents médicaux. Il contient une liste de références aux contrats de relation patient-fournisseur (RPF), représentant tous les engagements passés et actuels du participant avec d'autres nœuds du système. Les patients, par exemple, auraient

leur CS rempli avec des références à tous les fournisseurs de soins qu'ils ont engagés.

b. L'opération de minage dans la blockchain MedRec

MedRec amène des chercheurs médicaux et des intervenants en soins de santé à être des mineurs dans le réseau. En retour, les bénéficiaires du réseau (les prestataires et les patients) débloquent l'accès à des données médicales agrégées et anonymisées en tant que récompenses pour le minage. Ils explorent cette idée dans leur prototype en implémentant une fonction spéciale dans le contrat RPF. Cela nécessite que les prestataires de soins attachent une requête de prime à toute transaction qu'ils envoient en mettant à jour le RPF. Lorsque le bloc contenant la transaction de mise à jour d'enregistrement est «miné», la fonction de minage ajoute automatiquement le mineur du bloc en tant que propriétaire de la prime. Le mineur peut ensuite la récupérer en envoyant simplement une demande pour cette prime au contrôleur d'accès à la base de données Gatekeeper du fournisseur.

c. Les composants de MedRec

MedRec est conçu autour de quatre composants logiciels pour chaque nœud :

- **Une bibliothèque Backend :** ils construisent plusieurs utilitaires, regroupés dans une bibliothèque backend, pour faciliter le fonctionnement du système. Cette bibliothèque résume les communications avec la blockchain et exporte une API d'appel de fonction.
- **Client Ethereum :** ce composant implémente toutes les fonctionnalités nécessaires pour rejoindre et participer au réseau de blockchain Ethereum. Cela gère un large éventail de tâches, telles que la connexion au réseau P2P, l'encodage et l'envoi de transactions et la conservation d'une copie locale vérifiée de la blockchain.
- **Base de données Gatekeeper :** elle implémente une interface d'accès «off-chain» à la base de données locale du nœud, régie par des autorisations stockées sur la blockchain. Le Gatekeeper exécute un serveur écoutant les requêtes des clients sur le réseau.
- **Gestionnaire de DSE :** tous les composants logiciels mentionnés précédemment sont reliés avec le gestionnaire de DSE et l'interface utilisateur. L'application affiche les données des bases de données SQLite locales pour les visualiser, et présente aux utilisateurs des notifications de mise à jour et des options de partage et de récupération de données.

Les auteurs ont fourni une implémentation prototype de ces composants qui s'intègrent à une base de données SQLite et est gérée via leur interface utilisateur Web. Une implémentation de ceux-ci peut être exécutée sur un PC local ou même un téléphone mobile.

2.1.3 MedChain

Dans [34], les auteurs ont présenté la conception d'un système qui utilise une blockchain pour la gestion des dossiers médicaux : MedChain. Il est conçu pour améliorer les systèmes actuels car il offre un accès interopérable, sûr et efficace pour les dossiers médicaux par les patients, les fournisseurs de soins de santé, et d'autres tiers, tout en gardant la vie privée des patients. MedChain emploie des contrats intelligents temporisés pour régir les opérations et le contrôle d'accès aux DSEs. Il adopte des techniques de cryptage avancées pour fournir une sécurité supplémentaire.

Le système proposé emploie des méthodes de hachage, à savoir SHA-256, pour assurer l'intégrité des données. MedChain conserve une valeur de l'hach du lien qui sera créé au cours de la création du dossier pour accéder au DSE dans la blockchain au lieu de stocker le lien lui-même. Pour accéder à un enregistrement, le lien de requête cryptée sera envoyé via HTTPS au participant associé qui a des droits d'accès. Par conséquent, la valeur de l'hach stockée dans la blockchain garantit qu'aucune modification n'a été faite en dehors de la blockchain lors du transfert. Et, pour plus de sécurité, MedChain enregistre le lien de requête, la clé et le DSE dans des endroits différents.

Le système attribue aux nœuds un degré qui indique la qualité d'un nœud fournisseur en ce qui concerne la quantité et la qualité de dossiers médicaux qu'il possède. Dans leur proposition, la qualité médicale des dossiers est définie comme ayant les attributs de lisibilité, de complétude, cohérence, exactitude et non-redondance. La qualité totale de tous les DSE pour tous les utilisateurs stockés dans la base de données d'un fournisseur détermine le degré du nœud. Les nœuds de fournisseurs avec des degrés moindres sont plus susceptibles d'être sélectionné pour créer le nouveau bloc. Le nœud avec le degré minimum sera classé comme nœud «créateur d'un bloc», tandis que les nœuds avec des degrés supérieurs au degré moyen du réseau seront considérés comme des «électeurs».

Ce système utilise un proxy pour résoudre le problème du transfert des messages cryptés entre les nœuds sans avoir besoin de partager la clé symétrique. Il est de la responsabilité du proxy de chiffrer un message d'une manière qui permet à un autre utilisateur de le déchiffrer via sa clé privée en dépit du fait que la clé publique associée n'est pas utilisée pour chiffrer le message.

Dans ce travail, MedChain adopte le schéma de recryptage distribué ElGamal avec aveuglement distribué «distributed blinding» où une clé publique maitresse est utilisée pour chiffrer un message et la clé privée associée est distribuée en morceaux à l'ensemble des proxys. Ainsi, l'ensemble des proxys ne sera pas en mesure de déchiffrer le message, au contraire ils peuvent seulement re-chiffrer le message.

a. Les composants logiciels du système proposé

Le système MedChain contient les composants logiciels suivants :

- **Gestionnaire de l'évaluation des dossiers médicaux :** Pour évaluer la quantité et la qualité des dossiers médicaux installés dans la base de données de chaque fournisseur, REM (Record Evaluation Manager), un outil basé sur le langage python, est installé sur les nœuds des fournisseurs pour effectuer cette évaluation durant la phase d'initialisation.
- **Le gestionnaire DB :** est une API écrite en GoLang, fournit un accès aux bases de données existantes des fournisseurs et est contrôlé par les informations des autorisations stockées dans la blockchain.
- **Le gestionnaire de chiffrement/déchiffrement :** Ce composant exécute les schémas de chiffrements et déchiffrements dans le système proposé. Les dossiers médicaux sont cryptés en utilisant un chiffrement à clé symétrique. Le gestionnaire génère en premier une clé symétrique pour chiffrer le dossier médical, puis re-chiffre cette clé avec les clés publiques du : nœud fournisseur, nœud patient, et l'ensemble des nœuds proxy. Pour distribuer en toute sécurité des informations entre les parties via HTTPS, un schéma de chiffrement à clé publique est utilisé.

- **Client Ethereum :** est le point d'accès au réseau Ethereum car il comprend toutes les fonctionnalités requises pour adhérer à ce réseau. Les auteurs ont utilisé une blockchain à permissions, donc seuls les nœuds autorisés utiliseront le client pour accéder à la blockchain. Pour la mise en œuvre de prototype proposé, le client GoEthereum est utilisé.
- **Interface DSE :** est l'interface Web qui est utilisée pour la gestion par les fournisseurs des DSE, la visualisation des DSE par les patients et la gestion des options de récupération, ainsi que le partage des données.

b. Les contrats intelligents

Les contrats intelligents proposés utilisent un ensemble de fonctions de connectivité et de synchronisation pour fournir le délai raisonnable pour effectuer des transactions et assurent ainsi une transaction autorisée est prévue. Tous les contrats ont un champ de date «T» qui est utilisé pour les fonctions de synchronisation. Dans le système proposé, les contrats intelligents utilisés sont :

- **Les contrats des nœuds de consensus(CNC) :** ce contrat est considéré comme un contrat global pour la préservation de l'opération de minage, l'enregistrement et la sélection des nœuds électeurs ainsi que le nœud du «créateur du bloc» du fournisseur qui générera le bloc suivant. Le CNC est responsable de déterminer les nœuds électeurs dans la blockchain, le CNC stocke le degré de chaque nœud fournisseur dans le système qui sera à son tour utilisé pour calculer le degré moyen du réseau et sélectionner les nœuds électeurs.
- **Les contrats de l'historique de l'intendance :** ce contrat maintient l'historique de l'intendance des nœuds de chaque participant dans le système où le dossier médical est stocké et géré par le nœud fournisseur du patient. Le SRHC localise l'historique des participants des dossiers médicaux en tenant une liste sommaire de l'intendance.
- **Les contrats des participants aux dossiers :** le but est de suivre tous les dossiers que les fournisseurs de santé stockent pour les patients et est généré lorsqu'une nouvelle relation d'intendance est établie entre deux nœuds.
- **Les contrats journaux :** ce contrat suit toutes les opérations effectuées sur les dossiers des patients afin de faciliter l'ajout / validation des blocs dans le réseau blockchain. Il répertorie les détails de la transaction dans un champ de données crypté du journal avec un champ d'état qui indique si la nouvelle entrée a été ajoutée à la blockchain.
- **Les contrats de contrôle d'accès (CCA) :** il inclut toutes les informations relatives aux autorisations qui sont spécifiques à chaque dossier. Il répertorie les adresses Ethereum pour tous les nœuds qui ont des autorisations d'accès. Ce contrat spécifie le niveau de cet accès (c'est-à-dire propriétaire, lecture et lecture aveugle) et la clé symétrique chiffrée avec la clé publique de chaque nœud.
- **Le contrat de re-cryptage du proxy (CReP) :** une clé publique principale ainsi qu'une clé privée partagée seront attribuées à l'ensemble des nœuds proxy. Le CRePsera automatiquement créé lors de l'établissement d'un nouvel ensemble de nœuds proxy. Chaque nœud proxy possède une paire de clés publiques/privées unique avec la clé publique connue par les autres.

2.1.4 Schéma de chiffrement consultable

Un système de cryptage consultable basé sur une blockchain pour les DSEs est proposé dans [35]. L'index des DSEs est construit par des expressions logiques complexes et stockées dans la blockchain, de sorte qu'un utilisateur peut utiliser les expressions pour rechercher l'index. Comme seul l'index est migré vers la blockchain pour faciliter la propagation, les propriétaires de données ont un contrôle total sur qui peuvent voir leurs données dans le DSE.

Dans ce système, seul l'index de recherche est ajouté à une blockchain Ethereum pour faciliter la distribution des DSEs, alors que les vrais DSEs sont stockés dans un serveur Cloud publique sous une forme cryptée. Lorsque les utilisateurs veulent accéder à ces dossiers de santé électronique, ils doivent s'authentifier auprès du propriétaire des données pour obtenir l'autorisation ainsi que la clé de déchiffrement.

a. Contrat intelligent

Le contrat intelligent Ethereum utilisé dans le schéma proposé est conçu pour retracer les récompenses monétaires, y compris les frais de transaction, entre les parties impliquées dans la transaction. Il garantit que le propriétaire des données est payé tant qu'il révèle la transaction, qui permet à d'autres utilisateurs de faire des recherches dans la base de données, et que les autres utilisateurs peuvent obtenir des résultats de recherche précis, s'ils effectuent les paiements requis. Le contrat intelligent, identifié par une adresse spéciale, se compose de code de script, un solde de devises et un espace de stockage pour les clés/valeurs.

b. Système de GAS

Dans Ethereum, le système de gas est introduit pour empêcher les programmes malveillants d'occuper des ressources informatiques. Dans Ethereum, chaque transaction a une consommation de gaz limitée et le système mettra fin à la transaction lorsque la limite de gas sera épuisée. Le gas est obtenu par l'échange de devises Ethereum et la consommation de gas est la source du revenu des travailleurs.

En utilisant le système de gas, un travailleur à Ethereum peut être financièrement compensé en résolvant avec succès un problème mathématique. Si la consommation de gas d'une transaction est supérieure au gas Limit, la transaction sera alors terminée et la consommation de gas sera remise au travailleur. Lorsqu'un utilisateur invoque une fonction de contrat intelligent, il doit déclarer qu'il y a suffisamment de gas dans le compte et être prêt à payer ce montant.

c. Les acteurs du système

Dans le système proposé, il existe trois entités, à savoir : les propriétaires de données, l'utilisateur et la blockchain.

- **Le propriétaire des données :** est une entité qui crée les DSEs, cette entité peut être humaine (par exemple, patient) ou une organisation (par exemple, un hôpital ou clinique). Le propriétaire de données construit ensuite l'index pour les DSEs respectifs et crée le contrat intelligent pour décrire la façon dont on peut rechercher dans l'index.

Une fois ceci terminé, le propriétaire des données envoie à la fois le contrat intelligent et l'index dans la blockchain. Après cela, le propriétaire des données encrypte les DSEs en utilisant un algorithme de chiffrement symétrique et les stocke sur le serveur Cloud.

- **L'utilisateur :** est une entité qui est autorisée par le propriétaire des données pour rechercher l'index pour obtenir les DSEs nécessaires. L'utilisateur peut être humain ou une organisation.
- **La blockchain :** est une entité qui stocke l'index et tous les contrats intelligents. Les utilisateurs autorisés recherchent dans la blockchain pour certains DSE spécifiques et le contrat intelligent devrait donner un résultat correct et immuable.

Lorsque l'utilisateur souhaite rechercher certains DSE, il s'authentifie d'abord auprès du propriétaire des données et obtient un jeton de recherche après autorisation. Ensuite, l'utilisateur utilisera le jeton pour demander la recherche du contrat intelligent. Le contrat intelligent utilisera le jeton pour rechercher dans la blockchain pour obtenir les résultats correspondants et les renvoyer à l'utilisateur.

2.1.5 Système E-health basé sur MobileCloud

Enfin dans [36], les auteurs ont proposé une nouvelle plateforme de partage de DSE qui combine la blockchain et le système de fichier interplanétaire décentralisé (IPFS) sur une plateforme de Cloud mobile. En particulier, ils ont conçu un mécanisme de contrôle d'accès fiable utilisant des contrats intelligents pour assurer le partage sécurisé des DSE entre différents patients et prestataires médicaux. Ils présentent une implémentation de prototype utilisant la blockchain Ethereum dans un scénario de partage de données réelles sur une application mobile avec le Cloud Amazon.

Ethereum a deux comptes différents : les comptes externes (EOA) et les comptes contractuels. Chaque compte est indexé par une adresse et défini par une paire de clés, une clé privée et une clé publique. Pour interagir avec la blockchain Ethereum, chaque utilisateur doit créer un compte pour devenir une entité dans le réseau.

a. Les composants du système

Les principales composantes du ce réseau sont présentés comme suit :

- **Gestionnaire DSE :** il est responsable de contrôler toutes les transactions des utilisateurs sur le réseau blockchain. La capacité de gestion du gestionnaire DSE est activée par des contrats intelligents grâce à des politiques d'utilisation strictes.
- **Administrateur :** il est utilisé pour gérer les transactions et les opérations sur le Cloud par les moyens d'ajout, la modification ou la révocation des autorisations d'accès. Il est responsable de déployer des contrats intelligents et leur mise à jour.
- **Contrats intelligents :** ils définissent toutes les opérations autorisées dans le système de contrôle d'accès. Les utilisateurs peuvent interagir avec les contrats intelligents via l'adresse du contrat et l'interface binaire d'application (ABI).
- **Stockage décentralisé :** puisqu'il est impossible de partager et de stocker de grandes données sur la blockchain, ils ont tiré parti d'un système de fichiers P2P décentralisé Inter Planetary File System (IPFS), une solution très prometteuse pour construire une plateforme de partage de fichiers dans le réseau de la blockchain.

b. La structure du bloc de données

Le bloc de DSE contient les parties suivantes :

- **Enregistrements de transactions :** les transactions de notre bloc sont organisées dans une structure basée sur l'arborescence Merkle où un nœud terminal représente une transaction d'accès aux données des utilisateurs mobiles.
- **En-tête de block :** l'en-tête de bloc contient les métadonnées suivantes pour vérifier le bloc de données.
  - Hach : le hachage SHA256 du bloc.
  - Hachage précédent : le hachage du bloc précédent qui est utilisé pour la validation du bloc.
  - MerkleRacine : une structure pour stocker un groupe de transactions dans chaque bloc.
  - Nonce : il fait référence à un nombre qui est généré par la preuve de travail sur les nœuds mineurs, afin de produire une valeur de hachage inférieure à un niveau de difficulté cible.
  - Estampillage : date de création du bloc et de la dernière transaction.

c. Le processus de minage

En plus de l'ajout de DSE au stockage dans le Cloud, les métadonnées de téléchargement des transactions (ID de zone, ID patient et valeur de hachage) sont également insérées dans le pool de transactions non validées. Les mineurs formeront périodiquement des blocs à partir de transactions dans le pool pour l'opération de minage. Le mineur le plus rapide qui vérifie le bloc de données enverra la signature à d'autres mineurs pour la validation. Si tous les mineurs parviennent à un accord, le bloc validé avec sa signature est ensuite ajouté à la blockchain dans un ordre chronologique. Enfin, tous les utilisateurs du réseau reçoivent ce bloc et synchronisent la copie de la blockchain via le client blockchain.

d. Contrat intelligent

Un contrat de partage de données est créé en premier lieu, et contrôlé par l'administrateur pour surveiller les opérations de transaction dans le réseau de blockchain. Soit les variables suivantes :

- **PK** : clé publique de l'utilisateur ;
- **userRole** : rôle d'utilisateur ;
- **Addr** : adresse du patient dans la blockchain.

Le contrat prévoit principalement les cinq fonctions suivantes.

- **AddUser (PK, UserRole)** : (exécuté par l'administrateur) cette fonction permet d'ajouter un nouvel utilisateur au contrat principal.
- **DeleteUser (PK, userRole)** : (exécuté par l'administrateur) il est utilisé pour supprimer des utilisateurs du réseau en fonction de la clé publique correspondante. Toutes les informations personnelles sont également supprimées du stockage Cloud.
- **PolicyList (PK)** : (exécuté par l'administrateur) une paire fournisseur de soins de santé/patient peut convenir d'une politique qui exprime leurs relations dans les services médicaux.

- **RetrieveEHRs(PK, Addr)** : (exécuté par le gestionnaire) il permet de récupérer les dossiers médicaux du Cloud. Un participant au réseau doit fournir l'adresse du patient (y compris l'ID patient et l'ID de zone) au contrat intelligent. Le contrat effectue une vérification puis envoie un message au gestionnaire de DSE pour extraire et renvoyer les données au demandeur.
- **Penalty(PK, action)** : (exécutée par l'administrateur) lors de la détection d'une demande non autorisée au système de DSE, le gestionnaire de DSE informera le contrat intelligent pour émettre une pénalité (sous forme de message d'avertissement) au demandeur.

2.2 Les système basés sur la blockchain Hyperledger Fabric

Cette catégorie représente des systèmes basés sur la plateforme Hyperledger Fabric. L'article [32] représente un exemple de cette catégorie.

2.2.1 Application pour Healthcare 4.0

Dans [32], les auteurs ont proposé un système de gestion de DSE basé sur la blockchain pour les applications de soins Healthcare 4.0 dérivée du concept d'Industrie 4.0 où des systèmes Hi-Tech et Hi-Touch sont introduits, utilisant le cloud computing, fog-computing et le edge computing, l'analyse de Big Data, l'IA et l'apprentissage automatique, pour construire des blockchain pour prendre en charge les accès en temps réel aux données cliniques des patients.

Avec l'avènement du Big Data, la taille et la complexité des dossiers de santé augmentent et ne sont toujours pas optimisées. La sécurité des dossiers de santé devient de plus en plus importante pour protéger les données contre les failles de sécurité et les activités criminelles. Ainsi la confidentialité des données des patients est essentielle à la bonne gestion des soins de santé.

La solution proposée utilise des plateformes et des outils pour mesurer les performances de ces systèmes de santé, par exemple Hyperledger Fabric, Composer, Docker Container, Hyperledger Caliper et le moteur de capture Wireshark. De plus, ils proposent un algorithme de politique de contrôle d'accès pour améliorer l'accessibilité des données entre les fournisseurs de soins de santé, en s'aidant de la simulation des environnements qui mettent en œuvre le système de partage des DSEs basé sur Hyperledger.

Le cadre proposé fournit une interaction sécurisée entre différents participants et organisations qui utilisent les mécanismes de consensus Crase Fault Tolerance (CFT) et Byzantine Fault Tolerance (BFT) qui ne nécessitent pas plus de coûts pour le minage.

a. Architecture du système proposé

Il y a quatre participants dans le système proposé : le patient, le clinicien, le laboratoire et l'administrateur du système. Aussi, divers actifs ou contrats intelligents sont définis.

Le flux de travail du système est simple à utiliser. Les participants s'enregistrent via l'application cliente, demandant un certificat d'inscription via un fournisseur de services d'adhésion à l'autorité de certification. Ensuite, l'autorité de certification délivre le certificat et la clé privée avec un nouvel ID pour inscrire le participant. Toutes les transactions

sont réparties sur le réseau de la blockchain hyperledger fabric. Les participants ont des rôles différents dans le système et ne peuvent accéder qu'aux dossiers auxquels ils ont été autorisés. Les patients peuvent ajouter des dossiers à l'aide de l'application cliente, qui invoque le chain code pour enchainer une transaction sur le réseau de la blockchain. Les transactions mises à jour sont distribuées sur le réseau ; cela garantit que chaque transaction sur le réseau est distribuée à chaque participant du système et que chaque transaction ne peut pas être modifiée ou supprimée par des utilisateurs non autorisés. Les transactions ne sont ajoutées qu'au hachage précédent avec un horodatage, de sorte que le réseau est entièrement sécurisé.

Les dossiers sont mis à jour et visibles par chaque utilisateur du réseau de la blockchain. Les prestataires, comme les cliniciens et le personnel de laboratoire, peuvent demander les données requises sur le réseau.

- **Module admin :**

  Le certificat d'inscription d'administrateur est demandé à l'autorité de certification. L'administrateur a un accès complet au système, y compris écrire, lire, mettre à jour et supprimer des participants. Si les cliniciens, les patients ou le personnel de laboratoire sont valides, l'administrateur peut délivrer un identifiant approprié à chaque participant pour permettre l'accès au réseau de la blockchain. Si le comportement du participant est jugé inapproprié, l'administrateur peut supprimer ce participant avec une remarque sur le réseau de blockchain hyperledger.

- **Module patient :**

  Le patient demande une clé privée pour se connecter à l'administration du réseau. Après avoir obtenu l'accès au réseau de la blockchain, le patient dispose de divers droits, tels que la lecture, l'écriture et la révocation des DSEs. Pour cette procédure le patient utilise son identifiant dans la blockchain. Si le patient a un nœud valide, les dossiers du patient, du clinicien et du personnel de laboratoire peuvent être consultés ou recherchés sur le réseau. Si les DES ne sont pas disponibles, alors le système devrait informer le clinicien que les antécédents médicaux de ce patient ne sont pas trouvés. Après cela, le patient peut retirer l'accès au membre du personnel de laboratoire ou au clinicien du réseau une fois le traitement terminé ou si le patient ne souhaite pas que ses données soient partagées.

- **Module clinicien :**

  Dans l'étape de saisie, le clinicien demande une clé à l'administrateur du réseau pour activer la connexion. Dans la phase de sortie, le clinicien est autorisé à accéder aux transactions d'hyperledger du clinicien. Le nœud doit être un nœud valide. Si l'identifiant du clinicien appartient au réseau blockchain, le dossier médical des patients est alors attribué au clinicien. Le clinicien peut alors lire et mettre à jour le DSE autorisé dans lesystème. Si le clinicien n'a pas accès aux identifiants des patients, alors il peut écrire des enregistrements dans le réseau d'hyperledger. Un clinicien peut également rechercher des cliniciens et du personnel de laboratoire disponibles sur le réseau.

- **Module personnel du laboratoire :**

  Le personnel du laboratoire demande la clé privée à l'administrateur du réseau. Si le nœud s'avère valide, l'accès est alors accordé sur le réseau d'hyperledger. Le fonctionnement de nœud de laboratoire est similaire au nœud clinicien. Le nœud de laboratoire peut lire les dossiers médicaux et rédiger un rapport sur les résultats des tests des patients, tels que les rapports de sang ou d'immunité, etc. Ce nœud peut également rechercher tout le personnel de laboratoire et les cliniciens disponibles sur le réseau.

2.3 Les systèmes basés sur les deux blockchains : Ethereum et HyperldgerFabric

Contrairement aux systèmes des classes précédentes qui se basent sur une seule blockchain, dans cette classe ils ont proposé un système qui utilise deux blockchain Ethereum et HyperldgerFabric : Medicalchain.

2.3.1 Medicalchain

Medicalchain [37] est un système décentralisé qui utilise la technologie de la blockchain pour créer un DSE axé sur l'utilisateur tout en conservant une seule et unique version réelle des données de l'utilisateur.

Medicalchain permet à l'utilisateur de donner l'accès à leurs données personnelles de santé aux professionnels de la santé. Ensuite elle enregistre les interactions avec ces données dans un cadre vérifiable, transparent et sécurisé sur la blockchain. Enfin, Medicalchain est une plateforme que d'autres peuvent utiliser pour construire des applications qui complètent et améliorent l'expérience de l'utilisateur.

Medicalchain est construit en utilisant une double structure de la blockchain. La première blockchain contrôle l'accès aux dossiers de santé et est construit sur l'architecture Hyperledger Fabric basée sur les autorisations ce qui permet de varier les niveaux d'accès ; contrôle des utilisateurs qui peuvent voir leurs dossiers, combien ils voient et pour combien de temps. La deuxième blockchain est alimentée par un Token ERC20 sur Ethereum et contient toutes les applications et services pour la plateforme.

a. Définitions et autorisations des participants

Avec la présence de différents acteurs, la gestion de l'identité et l'accès aux données sont essentiels à la solution de Medicalchain. Ils ont proposé quelques exemples de permission de lecture/écriture (L/E) :

- **Praticien**
  - Accès L/E sur les DSEs autorisés ;
  - Demande d'autorisation pour un autre praticien/ institution pour obtenir un accès en L/E.
- **Patient**
  - Lire son DSE ;

- Autorisation d'un praticien/institution à L/E dans le DSE ou une partie de leur DSE ;
- Révoquer l'autorisation de Praticiens/Institutions ;
- Autorisation du parent le plus proche/ personne à contacter en cas d'urgence à Lire/accepter la permission ;
- Inscrire certains attributs dans son DSE (le poids, la température corporelle, la tension artérielle, le taux de glycémie, etc).

- **Institution de recherche**
  - Lire les DSEs autorisés.

b. Transaction

Toute interaction avec les dossiers de santé est enregistrée comme une transaction sur le réseau. Les transactions ne sont visibles que par les participants associés à la transaction. Voici des exemples de la manière dont les transactions se déroulent sur Medicalchain.

- **Le patient accorde l'accès**
  - Le patient A accorde l'accès au DSE au praticien B.
  - L'identité du praticien B est ajoutée à l'actif autorisé du patient A sur le registre
  - L'identité du patient A est ajoutée à l'actif autorisé du praticien B sur le registre
  - La clé symétrique du DSE est décryptée avec la clé privée du patient A
  - La clé symétrique est ensuite cryptée avec la clé publique du praticien B
- **Révocation de l'accès par le patient**
  - Le patient A révoque l'accès du praticien B
  - L'identifiant du praticien B est retirée de l'actif autorisé du patient A
  - L'identifiant du patient A est retirée de l'actif autorisé du praticien B
  - La clé privée du patient A est utilisée pour décrypter la clé symétrique du DSE qui est utilisée pour décrypter le DES.
  - Le DSE est crypté avec une nouvelle clé symétrique
  - La nouvelle clé symétrique est cryptée avec la clé publique du patient A et les clés publiques de tous les autres qui ont une autorisation
- **Praticien référant d'un patient**
  - Le praticien A met à jour les autorisations pour permettre au praticien B d'accéder au DSE du patient P
  - Chaincode (smart contract) vérifie que le praticien A à l'autorisation d'accéder au DSE
  - Le praticien A utilise sa clé privée pour décrypter la clé symétrique du DSE
  - La clé publique du praticien B est utilisée pour crypter la clé symétrique
  - L'identifiant du praticien B est ajoutée à l'actif autorisé du patient P
  - L'identifiant du patient P est ajoutée à l'actif autorisé du praticien B

c. Le cryptage

Pour garantir le respect de la vie privée, les dossiers médicaux sont cryptés en utilisant la cryptographie à clé symétrique. Le dossier est crypté et stocké dans un entrepôt de données au sein de la juridiction réglementaire appropriée.
Chaque fois qu'une entité a l'autorisation d'accéder au dossier du patient :
- Le dossier est décrypté avec la clé privée du propriétaire ;
- La clé symétrique est cryptée avec la clé publique de l'utilisateur autorisé .

Si un utilisateur autorisé demande un accès au dossier alors :
- La clé privée de l'utilisateur demandeur est utilisée pour décrypter la clé symétrique du DSE ;
- La clé symétrique décryptée est utilisée pour décrypter le DSE du patient ;

Dans le cas où l'accès d'un participant est supprimé à partir d'un dossier médical :
- La clé symétrique est décryptée avec la clé privée du propriétaire du DSE ;
- Le DSE est décrypté à l'aide de la clé symétrique ;
- Le dossier est recrypté avec une nouvelle clé symétrique ;
- La clé symétrique est cryptée avec toutes les clés publiques des autres utilisateurs autorisés.

3 Les systèmes indépendants des blockchains existantes

Dans cette classe les auteurs ont proposé, des systèmes avec leurs propres blockchains. Les systèmes de cette classe sont rares, car leurs créateurs sont généralement des professionnels de santé, et qui n'ont peut-être pas une large expérience dans le domaine du développement des blockchains. L'avantage des systèmes de cette classe est que leurs créateurs sont libres de personnaliser les concepts de leurs propres blockchains. Un exemple de ces systèmes est présenté dans [31] : MedShare.

3.1 MeDShare

MeDShare est un système qui permet le partage des données médicales entre les dépositaires de «Big Data» médicales dans un environnement sans confiance. Le système est basé sur la blockchain et fournit la provenance, l'audit et le contrôle des données médicales partagées dans des référentiels Cloud entre les entités «Big Data». Ce système repose sur des contrats intelligents pour surveiller efficacement le comportement des données en toute transparence.

a. **Le mécanisme de partage dans MedShare**

Le système proposé se compose de quatre couches principales, à savoir :

- **Couche utilisateur :** comprennent toutes les différentes classifications d'utilisateurs dont l'intention est d'accéder aux données du système à des fins de recherche ou à d'autres fins utiles.
- **Couche de requête de données :** se compose d'ensemble de structures de requête qui accèdent, traitent, transfèrent ou répondent aux requêtes posées sur le système. Ces requêtes peuvent être des demandes d'accès aux données à partir de l'infrastructure de base de données existante. Cette couche s'interface directement avec la couche de structuration des données et de provenance et possède des mécanismes, mis en œuvre pour interpréter et traduire les actions entre la couche de structuration des données et de provenance et l'environnement extérieur (hors du système). Les utilisateurs interagissent directement avec la couche de requête pour les demandes de données. Les composants de la couche de requête de données sont :

  - Système d'interrogation : est responsable du traitement de la demande dans un format souhaité par la couche de structuration des données et de provenance.

- Déclencheur : est une entité responsable de la traduction des actions vers et depuis l'environnement de contrat intelligent, car les contrats intelligents ne peuvent pas fonctionner pleinement en dehors de l'environnement de la blockchain.

- **Couche de structuration et de provenance des données :** se compose de composants individuels qui aident à traiter la demande d'accès aux données à partir de la couche d'infrastructure de base de données existante. La couche effectue en outre des calculs sur les données demandées et des données d'étiquette avec des fonctionnalités qui surveillent chaque action effectuée sur les données. Des algorithmes et des structures sont implémentés dans cette couche pour signaler les actions qui sont stockées en toute sécurité dans une base de données et indexées de manière appropriée, déclenchant une action sur les données surveillées si nécessaire. Les résultats de chaque action réalisée sont diffusés sur un réseau immuable pour garantir un audit sans confiance et équitable. Enfin, la couche a la responsabilité d'authentifier toutes les demandes et actions relatives à l'accès aux données de l'ensemble du système. Les différentes entités de la couche de structuration des données et de provenance :

  - Authentificateur : est chargé de vérifier la légitimité des demandes envoyées par les demandeurs au système des propriétaires de données.
  - Nœuds de traitement et de consensus : traitent les formulaires créés pour les requêtes qui sont ensuite mis en blocs et diffusés dans la blockchain. De plus, le nœud de consensus est chargé de créer des packages contenant les données demandées et le contrat intelligent à remettre au demandeur.
  - Contrats intelligents : sont des fonctions activées et exécutées à la réception d'une action. Les contrats intelligents générés sont intégrés avec des clés cryptographiques permettant aux contrats de crypter les rapports générés à partir de l'activation des actions. Le rôle principal d'un contrat intelligent est d'identifier les actions effectuées sur les données envoyées et enregistrer les données dans une base de données. Enfin, les contrats intelligents révoquent l'accès aux données compromises avec des actions non autorisées sur les données par le demandeur.
  - Base de données autorisée de contrat intelligent : est une entité de stockage de rapports de violation de données et une entité d'action qui est activée à la réception d'un rapport de violation par les nœuds de traitement et de consensus pour tous les rapports reçus par les contrats intelligents du demandeur.
  - Blockchain Network : le réseau blockchain est composé de blocs individuels diffusés dans un réseau et enchaînés ensemble selon une méthode chronologique. Le rôle principal du réseau blockchain est de maintenir une base de données immuable répartie chronologiquement sur la livraison et la demande de données du système. La blockchain maintient également un bloc latéral pour les actions individuelles concernant des données spécifiques signalées par les contrats intelligents dans le système du propriétaire des données.

- **Couche d'infrastructure de base de données existante :** se compose de systèmes de base de données déjà établis mis en œuvre par des parties individuelles pour accomplir des tâches spécifiques. Ces systèmes de base de données ne sont accessibles que par le personnel autorisé de ces sociétés car ils hébergent des informations sensibles qui nécessitent des mécanismes sécurisés pour protéger de manière adéquate ces données sensibles. Pour accéder aux données de ces bases de données, les ensembles de données demandés sont passés par des ensembles de calculs pour désensibiliser les données avant qu'elles ne soient partagées.

b. **Réseau blockchain**

La blockchain est structuré sur plusieurs threads, identifiées uniquement à l'aide de l'identité du demandeur. Les blocs parents sont enchainés aux blocs latéraux pour conserver un journal contigu de rapports développé. Chaque bloc parent représente différentes instances de traitement d'une demande et ceux-ci sont indexés et mis à jour par les contrats intelligents dans un bloc latéral. Un bloc est développé à partir d'un formulaire créé lors d'une requête. Le nœud de traitement et de consensus est responsable de la maintenance des blocs et de leur ajout au réseau de blockchain et sont les seules entités ayant un accès direct au réseau de la blockchain. Tout en surveillant les blocs parents avec des blocs latéraux, les nœuds alertent le système sur les violations de l'utilisation des données.

c. **Contrats intelligents**

Dans ce travail, ils ont utilisé des contrats intelligents pour signaler les actions (lire, écrire, supprimer, dupliquer, déplacer et copier) effectuées par un demandeur sur des données demandées au système d'un propriétaire de données. Cela permet aux propriétaires de données d'obtenir une assurance et un contrôle complets sur la provenance des données, car toute la ligne de vie des données envoyées serait surveillée dans un environnement contrôlé où le propriétaire des données n'a besoin d'aucune assurance de la confiance du demandeur.

Le suivi des actions est véhiculé dans des scripts de contrat intelligents par une fonction getAction. La sensibilité des données est classée en deux niveaux qui sont : haut et bas. L'avantage de les spécifier dans les contrats intelligents est de créer un moyen efficace de traitement et de nœuds de consensus pour faire correspondre, traiter et vérifier des blocs spécifiques. Les commentaires sont générés sous forme d'instructions pour décrire les actions qui ont été effectuées sur les données. Ce sont généralement des commentaires de violation et d'exemption.

La fonction accessControl signifie les autorisations définies par le propriétaire des données qui seraient exécutées conjointement avec la base de données autorisée du contrat intelligent. En cas de violation du contrat de données, l'accès aux données est révoqué et en attente d'examen par le propriétaire des données, qui a le choix de demander un nouvel accès ou de récupérer les données auprès du demandeur.

d. **Structure du bloc parent**

La structure du bloc proposée est composée :

- D'un format qui l'identifie d'une façon unique
- La taille du bloc.
- Des en-têtes de bloc qui sont hachés avec sha256() comme dans les en-têtes bitcoin. L'en-tête de bloc contient six composants :

  – la version des données qui indique les règles de validation à suivre pour un type de données particulier.
  – Hachage de bloc précédent

- Le hachage de la racine de l'arbre de merkle.
- Un horodatage de la date de création du bloc.
- Une difficulté cible qui est une valeur qui indique comment le traitement est réalisé par les nœuds de traitement et de consensus. Pour rendre le traitement difficile pour les nœuds malveillants mais efficace et résoluble par les nœuds de consensus.
- Un nonce qui est un nombre arbitraire généré par les nœuds de consensus pour modifier le hachage d'en-tête afin de produire un hachage en dessous de la difficulté cible.

Le bloc possède aussi :

- Un compteur d'actions dont la fonction est d'enregistrer le nombre total d'actions de violation applicables sur les données consultées dans le bloc entier.
- Avant le compteur d'actions on trouve horodatages et les données.
  - Les horodatages sont constitués du temps de réception de la demande (TTR), du temps nécessaire pour traiter la demande (TTP) et du temps d'envoi du package au demandeur (TTS)
  - La partie donnée est constituée de l'identité du propriétaire des données (OID), identité du demandeur (RID), sensibilité des données (Dsens), but de la demande de données (DRP), identité du nœud de traitement (NID) et signature du nœud de traitement (Nsig).
- Enfin, une structure qui définit le bloc entier : le temps de verrouillage du bloc. Il s'agit d'un horodatage qui enregistre la dernière entrée de transaction ainsi que la fermeture d'un bloc. Le temps de verrouillage du bloc signifie généralement l'heure à laquelle le bloc entre dans la blockchain.

e. Structure du bloc latéral

Un bloc latéral est constitué d'un format déduit en concaténant un ID généré par un mineur à la partie ID du bloc parent.

f. La validation d'une demande

Pour une demande valide, l'authentificateur transmet la demande aux nœuds de traitement et de consensus où le traitement des demandes en formulaires est terminé. Le formulaire généré contient un hachage de l'horodatage de la réception de la demande et un hachage de l'ID du demandeur. Le but de la demande de données est marqué sur le formulaire, puis transmis à l'infrastructure de base de données existante. L'infrastructure de base de données existante reçoit le formulaire, récupère les données et envoie les données récupérées à la couche de structuration et de provenance des données. Ceci est reçu par le nœud de traitement et de consensus et un hachage de cet horodatage est concaténé à un enregistrement existant d'un horodatage haché pour la demande. Les nœuds de traitement et de consensus envoient une demande au centre de contrats intelligent pour ajouter des ensembles de règles aux données demandées. Un contrat intelligent est généré et marqué sur le formulaire qui contient les données indexées pour former une sorte de contiguïté avec le bloc associé.

g. Le cryptage

Pour le système proposé, il est nécessaire d'adopter des primitives cryptographiques pour le transfert sécurisé des résultats des requêtes entre les parties de partage de don-

nées non fiables. Une description des primitives et des clés les mieux adaptées au système est donnée. Les clés comprennent :

- La clé privée du demandeur, générée par un demandeur et utilisée numériquement pour signer les demandes d'accès aux données.
- La clé publique du demandeur, générée par un demandeur et envoyée à l'authentificateur des propriétaires de données pour être utilisée comme vérification de l'identité du demandeur pour l'accès aux données.
- La clé de contrat d'authentificateur, paire de clés générée par l'authentificateur et attachée au contrat intelligent dans un package utilisé pour chiffrer les rapports que le système du demandeur laisse au système du propriétaire des données.

4 La synthèse des articles étudiés

Ces articles de recherche traitent la mise en œuvre de la blockchain dans le domaine de santé. Leur objectif principal est d'introduire des systèmes basés sur la blockchain qui établirait des contrôles d'accès aux données médicales et permettraient aux patients une plus grande surveillance de leurs informations médicales personnelles.

- Pour ce qu'est du stockage des dossiers de santé des patients, il diffère d'un travail à un autre tel que, dans les articles [35] et [36], le stockage se fait dans un serveur Cloud publique sous une forme cryptée. Dans les articles [33] et [37], ils stockent les DSEs directement sur la blockchain. Et dans les articles [30] et [34], les DSEs des patients sont stockés dans des bases de données des fournisseurs tout en intégrant la blockchain.

- Pour garantir la sécurité et respecter la vie privée, les dossiers médicaux sont cryptés. Les systèmes proposés dans [32], [36] utilisent le chiffrement symétrique. D'autre part dans les articles [31] et [33], les auteurs ont utilisé le chiffrement asymétrique. Par contre le système proposé dans [37], utilise à la fois le chiffrement symétrique et asymétrique. Aussi, le système proposé dans [30] se base en plus de chiffrement symétrique et asymétrique, sur le recryptage du proxy. Cependant dans l'article [34], ils ont employé le schéma de recryptage distribué ElGamal avec aveuglement distribué «distributed blinding». Tant dit que dans l'article [35] leur système utilise le chiffrement symétrique indexé (SSE).

- Les auteurs des articles [31] et [36] ont proposé une structure du bloc complète. Dans les articles [35] et [37] ils ont donné la structure de DSE de patient.

- L'ajout des transactions, la validation des blocs et leurs chainages dans la blockchain nécessitent un algorithme de consensus, tel que dans l'article [34], les auteurs ont utilisé un nouveau mécanisme incitatif intégré à l'algorithme de consensus POA (Proof Of Autority), dans [35] ils ont utilisé le même algorithme qui est utilisé dans la blockchain ethereum et dans [36] ils ont utilisé l'algorithme de consensus POC (Proof Of Concept). Ces algorithmes s'exécutent par des entités qu'on appelle mineurs et un exemple de choix de ces entités est proposé dans l'article [33], leur système amène des chercheurs médicaux et des intervenants en soins de santé à être des mineurs dans le réseau blockchain. Un autre algorithme est proposé dans l'article [30], QuorumChain qui s'appuie sur une méthode de vote majoritaire pour parvenir à un consensus. Aussi le cadre proposé dans [32] fournit une interaction sécurisée entre différents participants et organisations qui utilisent les mécanismes de consensus Crase FaultTolerance (CFT) et Byzantine FaultTolerance

(BFT) qui ne nécessitent pas plus de coûts pour le minage.

- Pour interaction avec la blockchain, une application cliente est obligatoire. Dans les articles proposés, ils ont utilisé des applications conviviales pour les participants. Dans les articles [30], [33], [34] et [37] ils ont utilisé le client ethereum aussi appelé ethereum-Go. D'autre part, le système proposé dans [36] a utilisé une application mobile.

- Les systèmes proposés ont intégré des contrats intelligents. Cependant, chaque système les utilise pour atteindre des objectifs différents mais le contrôle d'accès reste un objectif commun entre ces systèmes. Le système proposé dans [30] utilise aussi ces contrats pour maintenir l'opération de minage, classifier des différents niveaux de nœuds, conserver l'historique des relations des nœuds, suivre les dossiers que les fournisseurs stockent pour les patients et pour le rechiffrement du proxy. Et Dans [31], ils sont aussi utilisés pour identifier les actions effectuées sur les données envoyées et enregistrer les données dans une base de données. Dans les articles [33] et [35] ils ont utilisé des contrats intelligents Ethereum, pour créer des représentations intelligentes des DSEs existants dans le premier et pour retracer les récompenses monétaires dans le deuxième. Le système proposé dans [36], les utilise aussi pour assurer le partage sécurisé des DSEs entre différents patients et prestataires médicaux.

- Les réseaux blockchain intégré dans les systèmes proposés comprend un ensemble d'acteurs dont le patient reste l'acteur principal dans tous ces réseaux. Le système proposé dans l'article [30], comprend le patient, fournisseur et compagnie d'assurance. Et dans [32], Il y a quatre acteurs dans le système proposé : le patient, le clinicien, le laboratoire et l'administrateur du système. Dans [37], ils ont définie 3 acteurs : le patient, le patricien et l'institution de recherche. Alors que dans [33] et [34], ils ont définie : le patient, le fournisseur et le prestataire de santé. Les systèmes proposés dans [31], [35] et [36], comprennent les propriétaires de données qui peuvent être humaine (par exemple, patient) ou une organisation (par exemple, un hôpital ou clinique) et l'utilisateur de ces données.

5 Conclusion

Dans ce chapitre, nous avons présenté une synthèse des points essentiels cités dans chaque article traité. A partir de cette synthèse, nous avons tiré les concepts de base d'une blockchain dans le domaine du partage de dossiers de santé électronique. Dans le chapitre suivant, nous allons concevoir notre système d'une manière générale et présenter ses différentes parties.

Chapitre III

Conception générale

1 Introduction

Dans ce chapitre, nous avons cherché à concevoir un système basé sur la technologie blockchain pour la gestion des DSEs qui pourrait donner le contrôle final des DSEs aux patients et garantir la vie privée de ces derniers.

Problématique : Les DSE sont généralement gérés par des fournisseurs individuels, ce qui signifie que tous les enregistrements privés sont stockés dans des bases de données gérées par le fournisseur responsable de la création du document. Cela présente plusieurs problèmes de sécurité, de confidentialité et de contrôle qui doivent être résolus [30].

Aussi, étant donné que les fournisseurs sont seuls responsables de la gestion des enregistrements, l'intégrité des données peut être difficile à confirmer dans le cas où une entité malveillante altère la copie unique de l'enregistrement. Cela signifie également que si un enregistrement est supprimé d'une base de données de fournisseurs, les informations peuvent être définitivement perdues. Cette faille pourrait avoir des conséquences extrêmes sur la santé des patients [30]. Quelle solution pourrait être proposée pour résoudre les problèmes posés ci-dessus ?

Notre proposition : pour résoudre les problèmes décrits ci-dessus et répondre aux objectifs de confidentialité et de sécurité, nous allons conceptualiser un système en utilisant la technologie blockchain. De la synthèse des articles étudiés (chapitre précédent) cette technologie a été prouvée et validée pour le partage et la protection de données sensibles. Ceci est accompli en répliquant toutes les données sur la blockchain sur tous les nœuds du système. Du fait de cette redondance, une blockchain est facilement vérifiable et n'a pas de point de défaillance unique.

Dans les travaux proposés, ils ont utilisé plusieurs méthodes pour le stockage des DSEs, dans le Cloud, dans des BDD chez les fournisseurs et d'autres directement sur la blockchain. De même, nous allons stocker les DSEs directement sur la blockchain. Comme dans l'article [37], notre système va utiliser à la fois le chiffrement symétrique et asymétrique pour garantir l'intégrité des données. La validation des blocs est appliquée de différentes façons dans les travaux vus, il existe ceux qui utilisent des preuves de validation (PoA, PoC) et un autre qui utilise un algorithme de consensus qui s'appuie sur une méthode de vote, alors que dans notre cas, nous allons utiliser la preuve de travail (PoW). Pour une meilleure confidentialité des données sensibles, un paramètre visibilité accompagnera ces données pour préciser quelles données sont accessibles par quels acteurs.

2 L'architecture générale de notre système

Notre système (appelé PatientLedger) est caractérisé par un réseau blockchain privée, chaque nœud de ce réseau possède une copie de la blockchain. Le médecin traitant est le nœud principal dans le réseau, c'est le propriétaire de la blockchain et sa fonctionnalité principale est d'ajouter des patients au réseau, les nœuds patients à leurs tour autorisent les nœuds tiers d'accéder à leurs DSE (ajouter ces nœuds tierces au réseau blockchain). Chaque acteur doit avoir un compte dans notre système pour pouvoir interagir avec la blockchain (ajoutent des données à la blockchain) via un site web (application client). Les données personnelles des acteurs sont stockées dans une BDD classique du serveur et les données médicales (qui sont le DSE) dans la blockchain, cette séparation a pour but de garder l'anonymat des patients car les transactions stockées dans la blockchain sont anonymes. La figure III.1 représente l'architecture de notre système :

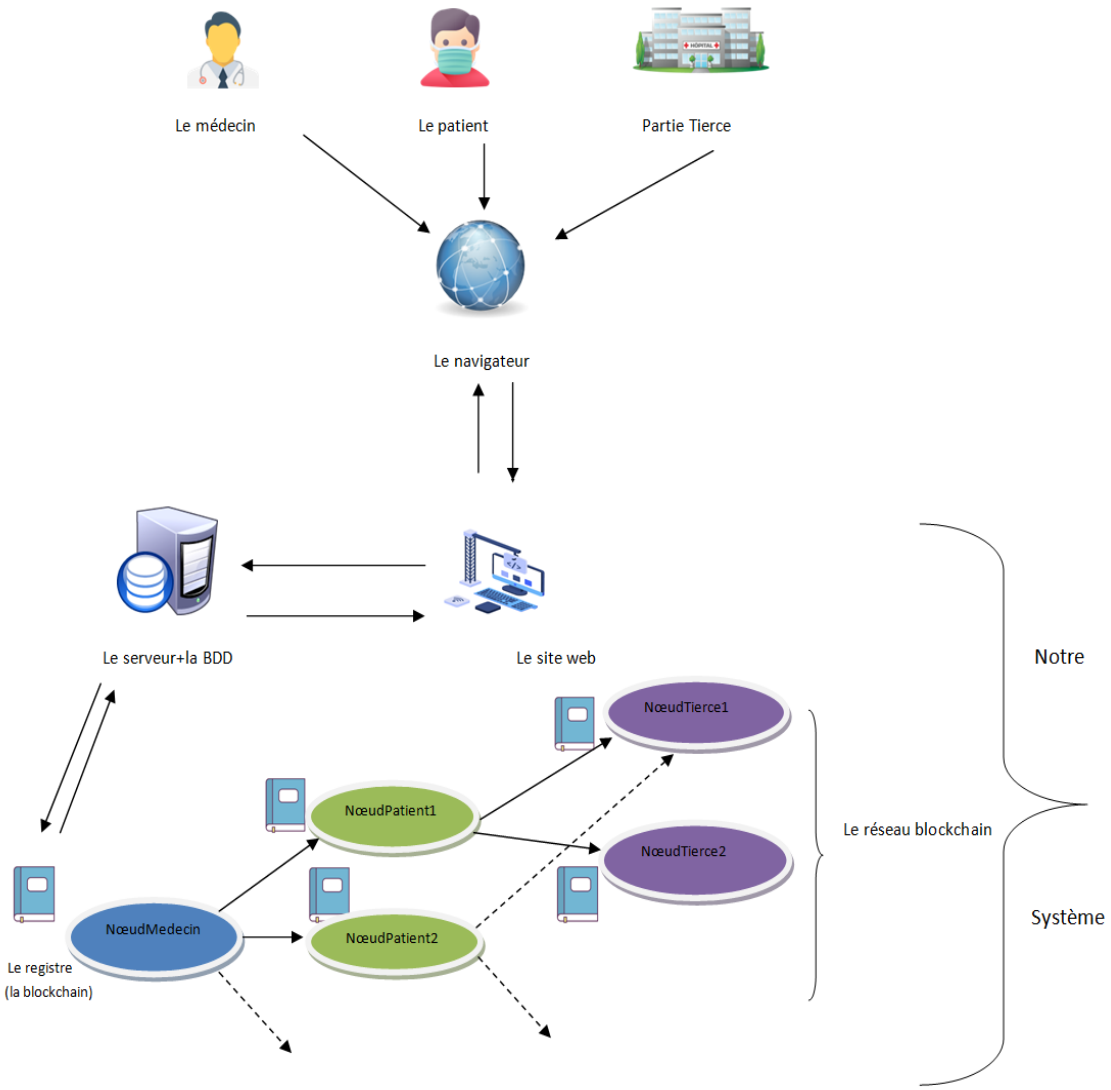

FIGURE III.1 – L'architecture de système PatientLedger.

3 Les acteurs du système et leurs rôles

Comme décrit dans la figure III.1, notre système comprend trois types d'acteurs qui sont :

1. **Les médecins traitants :** notre système se compose de plusieurs médecins traitants, chaque médecin peut ajouter des DSEs pour leurs patients. Ces médecins ont les rôles suivants :
   - Ajouter des patients au réseau blockchain ;
   - Ajouter des informations médicales aux DSEs de leurs patients ;
   - Consulter ou récupérer des données des DSEs de leurs patients ;
   - Effectuer l'opération de validation de blocs (c'est le rôle mineur).
2. **Les patients :** notre système se compose de plusieurs patients qui se soignent chez les médecins traitants de ce réseau, dont ils ne peuvent rejoindre ce dernier qu'après avoir eu une autorisation auprès de leurs médecins traitants. Ces patients ont les rôles suivants :
   - Gérer l'accès (accorder ou révoquer) des parties tierces à leurs DSEs ;
   - Ajouter des informations médicales à leurs DSEs : Poids, taille, tension artérielle, etc ;
   - Consulter ou récupérer des données de leurs DSEs.
3. **Les nœuds tiers :** les patients ont besoin en plus de leurs médecins traitants, d'autres professionnels de santé tels que l'hôpital, laboratoire d'analyses et le pharmacien, etc. Ces derniers, ont pour but d'ajouter des informations médicales ou de consulter les DSEs. De ce fait, nous avons introduit d'autres nœuds de type nœud tiers à notre réseau. Ces derniers ont les rôles suivants :
   - Ajouter des informations médicales aux DSEs des patients auxquels ils ont accès ;
   - Consulter ou récupérer des données des DSEs auxquels ils ont accès.

4 Les relations entre les acteurs (leurs communications)

D'après la figure III.1, des relations sont définies entre les acteurs de notre système et qui sont :

- **Relation patient/médecin traitant :** après qu'un médecin traitant ajoute les patients qui soignent au réseau blockchain, il aura un accès total (en lecture et en écriture) à leurs DSEs stockés dans la blockchain.
- **Relation patient/parties tierces :** les parties tierces ne peuvent accéder aux DSEs des patients qu'après avoir un accès auprès de ces derniers. Les patients peuvent à leur tour révoquer cet accès si c'est nécessaire.

5 Le stockage

Pour ce qui est du stockage, nous avons séparé celui des données personnelles et comptes utilisateur de celui des données médicales :

5.1 La base de données

Nous avons utilisé une base de données relationnelle du côté serveur de notre site web, pour stocker les informations personnelles des utilisateurs tel que leurs nom, prénom, date de naissance, adresse, et leurs comptes : login et mot de passe.

5.2 La partie blockchain

Ce qui concerne le stockage des données médicales des patients, nous avons créé une blockchain pour les sauvegarder en tant que transactions chiffrées, horodatées et inaltérable. Ces transactions sont ensuite organisées dans des blocs enchainés en respectant des règles prédéfinies. Pour la création de cette blockchain nous nous sommes basés sur les points suivants :

5.2.1 Les algorithmes de chiffrement

Dans notre système, nous avons utilisé deux formes de chiffrement pour garantir la sécurité et respecter la vie privée des utilisateurs :

a. Chiffrement symétrique

Le chiffrement symétrique (ou chiffrement à clé secrète ou symétrique) consiste à utiliser la même clé pour le chiffrement et le déchiffrement. Ce type de chiffrement consiste à appliquer une opération (algorithme) sur les données à chiffrer à l'aide de la clé symétrique, afin de les rendre inintelligibles. Nous l'avons utilisé pour garantir l'authenticité des nœuds au réseau blockchain, et chiffrer les données partagées entre ces nœuds. Les algorithmes suivants expliquent son utilisation :

a.1. L'inscription à la blockchain

Chaque acteur déjà inscrit au serveur du système, doit être aussi inscrit à la blockchain pour être identifié et devenir membre dans cette dernière. L'algorithme suivant représente le processus d'inscription à la blockchain :

- chaine : le type chaine de caractères ;
- Donnee_chiffrees : ce sont les données résultantes de chiffrement et qui vont être stockées dans la blockchain ;
- Cle_privee : la clé privée générée pour un acteur ;
- Cle_publique : la clé publique générée pour un acteur.

```
Algorithme inscription_BC
    Déclaration
    Donnee_chiffrees: chaine;
    Type Tab_cles = Tableau [Cle_privee, Cle_publique] de chaine ;
    Debut
        Tab_cles <- Generer_les_cles() ;
        Donnee_chiffrer <- Chiffrer_sym(Cle_publique,Cle_privee) ;
        Stocker(Donnee_chiffrees);
        Ecrire (Vous êtes un membre dans la blockchain, voici vos clés ', Tab_cles) ;
    Fin.
```

a.2. La connexion à la blockchain

Pour qu'un acteur puisse accéder à la blockchain, il doit s'authentifier en utilisant sa clé privée. L'algorithme suivant représente le processus de connexion à la blockchain :

- chaine : type chaine de caractères ;

- Donnee_dechiffrees : ce sont les données résultantes de déchiffrement des données chiffrées ;
- Donnee_chiffrees : ce sont les données chiffrées récupérées de la blockchain ;
- Cle_privee : la clé privée d'un acteur.

```
Algorithme connexion_BC
Déclaration
Donnee_dechiffrees, Donnee_chiffrees, Cle_privee : chaine ;
I,n : integer ;
Trouve : Boolean ;
Debut
  Lire(Cle_privee);
  Récupérer(Donnee_chiffrees);
  I <- 0 ; trouve <- false ; n <- nombre_membre() ;
  Tantque( i< n et trouve =false)
    Donnee_dechiffrer <- Dechiffrer_sym(Donnee_chiffrees,Cle_privee) ;
    Si (Donnee_dechiffrer = false)
      I++ ;
    Sinon
      Trouve <- true ;
    Finsi.
  Fintantque.
  Si (trouve = false)
     Ecrire ('Vous n'êtes pas un membre de la blockchain') ;
  Sinon
    Ecrire ('Vous êtes bien authentifie') ;
  Finsi.
Fin.
```

a.3. L'écriture dans un DSE

L'algorithme suivant représente comment le cryptage symétrique est utilisé dans le processus d'écriture dans la blockchain (l'ajout des données à la blockchain) :

- chaine : type chaine de caractères ;
- Donnee_medicale : c'est les données introduites par un acteur ;
- Cle_symetrique : la clé symétrique associé à un DSE d'un patient ;
- Donnee_medicale_chiff : c'est les données résultantes de chiffrement des Donnee_medicale par la clé symétrique.

```
Algorithme ecriture_DSE
Déclaration
   Donnee_medicale, Donnee_medicale_chiff, Cle_symetrique : chaine;
Debut
   Lire( Cle_symetrique);
   Lire (Donnee_ medicale);
   Donnee_medicale_chiff <- Chiffrer_sym(Donnee_medicale, Cle_symetrique);
  Stocker(Donnee_medicale_chiff) ;
Fin.
```

a.4. La Lecture d'un DSE

Aussi, pour lire (consulter) les données stockées dans la blockchain, ces dernières doivent d'être déchiffrées avec l'algorithme symétrique et cela après les avoir récupérés depuis la blockchain. Cette utilisation est décrite dans l'algorithme suivant :

- chaine : type chaine de caractères ;
- Cle_symetrique :la clé symétrique associée à un DSE d'un patient ;
- Donnee_medicale_chiff : c'est les données médicales récupérer depuis la bockchain ou elles étaient stockées chiffrées.
- Donnee_medicale : c'est les données médicales d'origines, elles résultent de déchiffrement des données chiffrées en utilisant la clé symétrique ;

```
Algorithme lecture_DSE
Déclaration
    Donnee_medicale, Donnee_medicale_chiff, Cle_symetrique : chaine;
Debut
    Lire( Cle_symetrique);
    Récupérer (Donnee_medicale_chiff);
    Donnee_medicale <- Dechiffrer_sym(Donnee_medicale_chiff, Cle_symetrique);
    Ecriture(Donnee_medicale_chiff) ;
Fin.
```

b. Chiffrement asymétrique

Le chiffrement asymétrique (ou chiffrement à clés publiques) consiste à utiliser une clé publique pour le chiffrement et une clé privée pour le déchiffrement. D'autre part, nous avons opté pour ce type de chiffrement pour vérifier la signature des transactions et leur origine et aussi pour gérer l'accès aux DSEs donné aux parties tierces.

b.1. Donner accès à un DSE

L'algorithme de chiffrement asymétrique est utilisé dans le processus «donner accès» comme suit :

- chaine : type chaine de caractères ;
- Cle_publique_pt : la clé publique de partie tierce ;
- Cle_symetrique : la clé symétrique associée à un DSE d'un patient ;
- Accorder : c'est la repense de patient à la demande de la partie tierce.

```
Algorithme donnee_acces
  Déclaration
      Cle_publique_pt,Cle_symetrique : chaine ;
      Accorder_acces : Boolean ;
  Debut
      Lire(Accorder_acces );
      Lire(Cle_publique_pt);
      Lire( Cle_symetrique);
      Si (Accorder_acces = true)
          Donnee_chiffrer <- Chiffrer_asym(Cle_symetrique ,Cle_publique_pt) ;
          Stocker(Donnee_chiffrees);
      Finsi.
  Fin.
```

5.2.2 L'algorithme de validation

Comme toute blockchain, un algorithme de validation est nécessaire pour valider les blocs avant qu'ils soient enchainés à la suite d'autres blocs. Pour notre cas, les nœuds de validation sont les médecins traitant (mineurs). Pour cela, nous avons utilisé l'algorithme de preuve de travail (POW) qui est une méthode où le mineur hache l'ensemble des données suivant : la transaction, le hach de bloc précédent et un nombre aléatoire qui sera incrémenté jusqu'à trouver un hach valide qui commence par trois zéros. La fonction qui suit, calcule le hach valide pour un bloc :

- chaine : type chaine de caractères ;
- Hach_bloc : est l'identificateur de bloc ;
- Nonce :est un nombre entier généré par le mineur pour trouver un hach valide pour le bloc.

```
Fonction PoW (donnee chaine) : chaine ;
Déclaration
   hach_bloc: chaine;
   Nonce : integer ;
Debut
    Nonce <- 0 ; hach_bloc <- hacher (donnee+ Nonce) ;
       Tantque (hach_bloc ne commence pas 3 zéros)
          Nonce ++ ;
          hach_bloc <- hacher (donnee+Nonce) ;
        Fintantque.
       Retourner (hach_bloc) ;
Fin.
```

5.2.3 Transaction

Tout ajout au DSE est enregistré comme une transaction sur la blockchain. Elles ne sont visibles que par les nœuds qui ont accès. La transaction ne doit pas inclure la clé privée de

l'émetteur mais son identité est prouvée par une signature générée par le chiffrement de cette transaction avec sa clé privée. Dans notre système, la structure générale de la transaction est définie comme suit :

- chaine : type chaine de caractères ;
- From : la clé publique de l'émetteur ;
- To : la clé publique de récepteur ;
- Data : les données médicales à ajouter ;
- Horodatage : la date de l'envoie de la transaction ;
- Visible : de type booléen, c'est la visibilité des données médicales pour le patient, lorsque ses données ajoutées par des professionnels de santé sont très sensibles et qu'elles ne doivent pas êtres accessibles par le patient, dans ce cas l'attribut visible sera à faux.
- Cle_privee : la clé privée de l'émetteur,
- Cle_symetrique : la clé symétrique associée à un DSE d'un patient,
- transaction : est un tableau de chaine de caractère.

```
Fonction Transaction (From chaine,To chaine,Data chaine,Typedata chaine,Visible Boolean,
Horodatage chaine,Cle_privee chaine, Cle_symetrique chaine) : transaction ;
Déclaration
  Type transaction = tableau [1..6] de chaine ;
   Signature : chaine ;
Début
  transaction[1] <- From ;
  transaction[2] <- To ;
  transaction[3] <- data ;
  transaction[4] <- Typedata;
  transaction[5] <- Horodatage;
  Signature < - chiffrer_asym(From+To+Typedata+Visible+Horodatage,Cle_privee) ;
  transaction[6] <- Signature;
  Retourner(transaction) ;
Fin ;
```

5.2.4 Contenu d'un bloc

C'est une entité contenant les transactions effectuées par les nœuds du réseau. Dans notre cas, chaque bloc contient une seule transaction et un en tête qui contient :

- Index : la position du bloc dans la chaine ;
- Hachprece : le hachage de bloc précédent ;
- Horodatage : un horodatage de la date de création du bloc ;
- Nonce : est un nombre arbitraire généré par le mineur pour modifier le hachage a fin de produire un hachage au-dessous des difficultés cible ;
- Hach : est l'identificateur de bloc, c'est le hach valide qui résulte de l'application d'une fonction de hachage sur les informations de bloc précédentes en lui ajoutant le nonce trouvé.

La fonction suivante permet de créer un nouveau bloc en respectant la structure décrite ci-dessus, elle est exécutée par le mineur pour chaque transaction valide :

- chaine : type chaine de caractères.

```
Fonction Bloc (Transaction trasnaction,Hachprece chaine,index integer,Horodatage chaine) : bloc ;
Déclaration
  Type bloc = tableau [1..3] de chaine ;
   Hach : chaine ;
Début
  bloc[1] <- Transaction ;
  bloc[2] <- To ;
  Hach<- PoW(Transaction+Hachprece+index+Horodatage) ;
  bloc[3] <- Hach;
  Retourner(bloc) ;
Fin ;
```

5.2.5 Blockchain

La blockchain est une structure linéaire de blocs valides, commençant par un premier bloc considéré comme valide par défaut, ces blocs sont chaînés selon des règles précises qui sont :

1. L'index du bloc doit suivre l'index du bloc précédent.
2. Le Hachprece correspond bien au hach du bloc précédent.
3. Le hach du bloc est valide.

Si chaque bloc de cette blockchain respecte les trois critères définis ci-dessus, la blockchain est alors valide. L'algorithme qui suit décrit la structure de la blockchain :

- chaine : type chaine de caractères ;
- Blockchain : un tableau des blocs.

```
Algorithme Blockchain
 Déclaration
Type Blockchain= Tableau [1..n] de chaine ;
 Form,To,Data, TypeData ,Horodatage_T,Cle_privee, Cle_symetrique,Hachprece,Index,Horodatage_B:  chaine ;
Visible : Boolean ;
Type  transaction= tableau [1..6]  de chaine ;
Type  bloc= tableau [1..3]  de chaine ;
Debut
    transaction<- Transaction(From,To,Data,TypeData, Horodatage_T,visible, Cle_privee, Cle_symetrique) ;
   Si(longueur(Blocs[]=0))
    bloc<-Bloc(Transaction,null,1,Horodatage_B) ;
    Blockchain[1]<-bloc ;
   Sinon
    Si (( hach (bloc)commence  par 3zéro) et (index=index(Bloc[index-1])+1  )et (hachprece=hach(Bloc[index-1]))
     bloc<-Bloc(Transaction, Hachprece, index, Horodatage_B ) ;
     Blockchain[index]<-bloc;
    Finsi ;
  Finsi ;
Fin.
```

5.3 L'application cliente

Pour interagir avec la blockchain, nous avons choisi de créer un site web qui est simplement accessible depuis un moteur de recherche. Via ce site, chaque acteur crée un compte dans notre système afin d'ajouter des données médicales aux DSEs auxquels ils ont accès.De plus, ce site facilite l'opération de validation des blocs pour le nœud mineur, qui est une opération essentielle mais compliquée dans la blockchain.

6 Conclusion

Dans ce chapitre, nous avons présenté l'architecture de notre système basée sur la technologie blockchain d'une manière générale : les modules du système, ses acteurs, leurs rôles, les relations entre eux et la structure de chaque entité qui forme la blockchain. Dans le chapitre qui suit, on va se concentrer sur la réalisation d'un mini réseau blockchain qui se composera d'un médecin traitant, un patient (ou plus) et un hôpital (ou plus) qui joue le rôle de la partie tierce.

Chapitre IV

Conception détaillée et réalisation de l'application

1 Introduction

Nous allons entamer ce chapitre par l'analyse et la conception de notre système, ensuite nous allons passer à sa réalisation. Dans la première partie, nous allons définir le rôle de chaque acteur qui interagit avec le système. Nous allons aussi modéliser leurs rôles sous forme de diagramme de cas d'utilisation, puis nous modéliserons les cas d'utilisation sous forme de diagrammes de séquence. Nous finirons cette partie par le diagramme de classes et le modèle relationnel.

Et dans la deuxième partie qui est consacrée à la réalisation et la mise en œuvre de notre application basée sur la technologie blockchain pour la gestion de dossier de santé électronique, nous présenterons en premier lieu l'environnement et les outils de développement utilisés à savoir les langages de programmation, les bibliothèques et les environnements de programmation. Nous finirons cette deuxième partie par une présentation de quelques interfaces de notre application.

2 La partie analyse et conception

2.1 Méthode d'analyse et de conception

UML (Unified Modeling Langage) se définit comme un langage de modélisation graphique et textuelle destiné à comprendre et décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue.

Il comporte plusieurs diagrammes, structurels et comportementaux, qui permettent de représenter respectivement des vues statiques et dynamiques d'un système [39]. Les diagrammes d'UML qu'on va utiliser dans notre conception :

- Le diagramme de contexte statique ;
- Le diagramme de cas d'utilisation ;
- Le diagramme de séquence ;
- Le diagramme de classe.

2.2 Le Processus Unifié (UP)

Un processus unifié est un processus de développement logiciel construit sur UML, il est itératif et incrémental, centré sur l'architecture, conduit par les cas d'utilisation et piloté par les risques. La démarche UP se résume dans les étapes suivantes :

- Expression des besoins ;
- Analyse ;
- Conception ;
- Implémentation ;
- Test [38].

La figure IV.1 décrit les différentes étapes de ce processus :

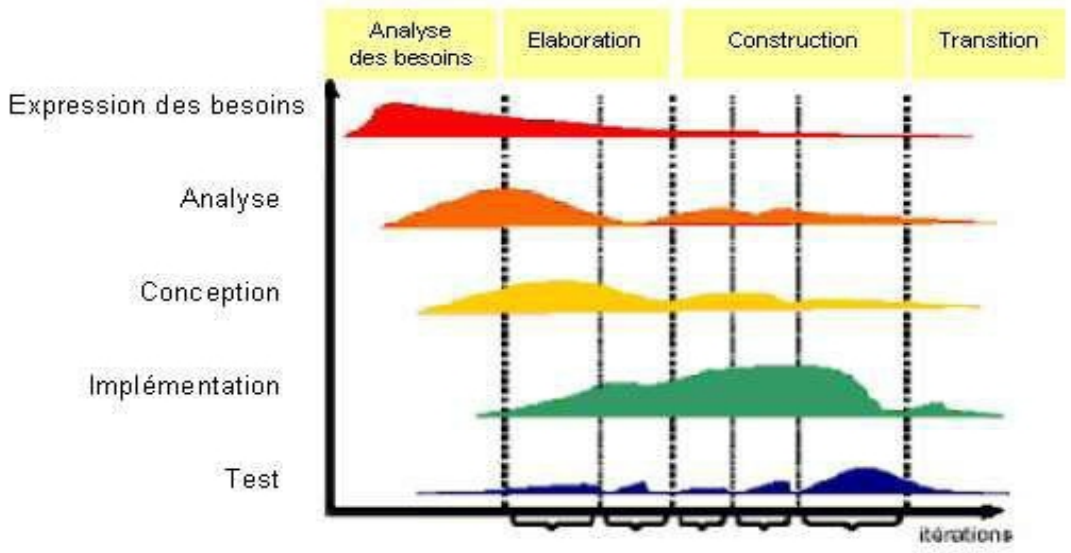

FIGURE IV.1 – Description du Processus Unifié (UP) [39].

2.3 Spécification des besoins

2.3.1 Identification des acteurs du système

a. Définition d'acteur

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié, autrement dit 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 [40].

Dans notre cas, nous avons identifié principalement trois acteurs en interaction avec notre système :

- **Le médecin traitant :** c'est l'acteur principal de notre système. Il est chargé d'ajouter des patients au réseau blockchain et de leur ajouter des informations médicales à leurs DSEs stockés dans la blockchain.

- **Le patient :** c'est l'acteur suivi par ce médecin traitant. Il est chargé de gérer l'accès des autres parties tierces (hôpital) à son DSE.
- **Nœud tiers :** dans notre cas nous avons pris l'hôpital comme exemple de nœuds tiers. C'est l'acteur qui ajoute des nouvelles informations médicales à un DSE d'un patient après avoir eu un accès de la part de ce dernier.

b. Définition de diagramme de contexte statique

Le diagramme de contexte se situe au début du processus d'analyse, son objectif est simple il doit représenter le système à modéliser, en général sous la forme d'une boite noire et les différents acteurs qui interagissent ce système [41]. Notre système est en interactions avec un médecin traitant, au moins un patient et possible de plusieurs nœuds tiers (hôpital, cabinet dentaire, laboratoire d'analyses médicales etc.). La figure IV.2 représente le diagramme de contexte statique du notre système :

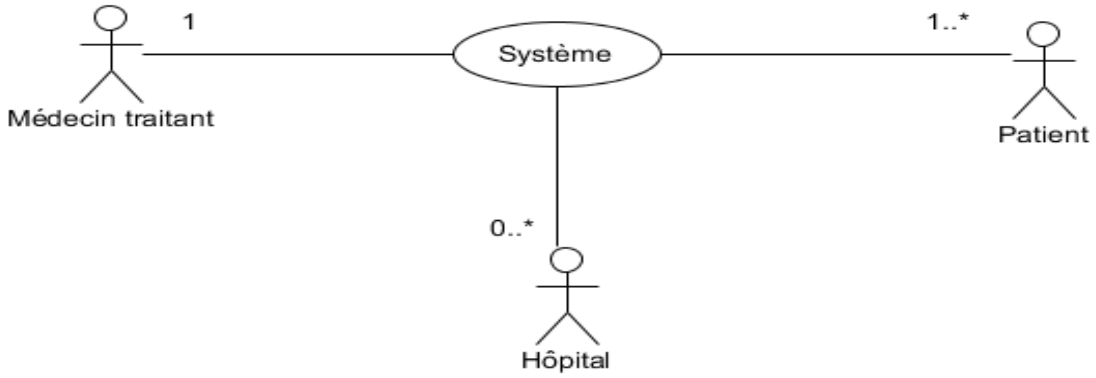

FIGURE IV.2 – Diagramme de contexte statique de notre système.

2.3.2 Identification des besoins

a. Les besoins fonctionnels

Notre système doit offrir les fonctionnalités suivantes :

- **Créer un compte :** chaque utilisateur doit créer un compte via l'application web, ce dernier est stocké dans la BDD du serveur.
- **Demander de participer au réseau blockchain :** cette fonctionnalité concerne le patient/l'hôpital. Après avoir créé un compte il doit demander auprès du médecin traitant/patient de rejoindre le réseau blockchain.
- **Ajouter un patient au réseau blockchain :** cette fonctionnalité est une réponse du médecin traitant à la demande d'un patient ou bien une réponse du patient à la demande d'un hôpital. Cette opération se traduit par les trois transactions suivantes qui seront stockées dans la blockchain :

- Les transactions d'inscription et d'accès sont ajoutées à des blocs et ces derniers sont validés (trouver un hach bloc valide) directement à la blockchain sans passer par la file d'attente.

cas 1 : si le demandeur est un patient alors

1. La donnée1 = la clé publique de patient + (la clé publique de patient) chiffrée avec la clé privée du patient en utilisant un algorithme symétrique.
   - La transaction_d'inscription = la donnée1 + (la donnée1) signée avec la clé privée du patient.
2. La donnée2 = la clé publique de patient + (la clé symétrique de patient) chiffrée avec la clé publique du patient en utilisant un algorithme asymétrique + accorder à vrai.
   - La transaction_d'accès = la donnée2 + (la donnée2)signée avec la clé privée du patient.
3. La donnée3 = la clé publique de patient + la clé publique de médecin traitant + (la clé symétrique de patient) chiffrée avec la clé publique du médecin traitant en utilisant un algorithme asymétrique + accorder à vrai.
   - La transaction_d'accès = la donnée3 + (la donnée3)signée avec la clé privée du patient.

cas 2 : si le demandeur est l'hôpital alors

1. La donnée1 = la clé publique de l'hôpital + (la clé publique de hôpital) chiffrée avec la clé privée du l'hôpital en utilisant un algorithme symétrique.
   - La transaction_d'inscription = la donnée1 + (la donnée1) signée avec la clé privée de l'hôpital.
2. La donnée2 = la clé publique de patient+ la clé publique de l'hôpital + (la clé symétrique de patient) chiffrée avec la clé publique du l'hôpital en utilisant un algorithme asymétrique + accorder à vrai.
   - La transaction_d'accès = la donnée2 + (la donnée2)signée avec la clé privée du patient.

- **Demander l'accès à un DSE :** cette fonctionnalité concerne l'hôpital. Après avoir créé un compte et joint le réseau blockchain, l'hôpital envoie une demande d'accès à un patient précis pour pouvoir ajouter et consulter son DSE.
- **Accorder l'accès à un DSE :** cette fonctionnalité est une réponse du patient à la demande d'accès de nœud hôpital. Cette opération se traduit par la transaction suivante qui sera stockée dans la blockchain :
  1. La donnée = la clé publique de patient + la clé publique de l'hôpital + (la clé symétrique de patient) chiffrée avec la clé publique du l'hôpital en utilisant un algorithme asymétrique + accorder à vrai.
     - La transaction_d'accès = la donnée2 + (la donnée2)signée avec la clé privée du patient.
- **Révoquer l'accès à un DSE :** cette fonctionnalité est effectuée par le patient, il révoque un accès à un nœud hôpital au moment où il découvre que ce dernier n'est pas un nœud de confiance ou bien il n'est plus un client de cet hôpital. Cette opération se traduit par la transaction suivante qui sera stockée dans la blockchain :
  1. La donnée = la clé publique du patient + la clé publique de l'hôpital + (clé symétrique=null) chiffrée avec la clé publique de l'hôpital en utilisant un algorithme asymétrique + accorder à faux.
     - La transaction_d'accès = la donnée + (la donnée)signée avec la clé privée du patient.
- **Écrire dans un DSE :** cette fonctionnalité est effectuée par chaque acteur qui possède un accès au DSE de patient concerné. Cette opération se traduit par la transaction suivante qui sera envoyée à la file d'attente des transactions non encore validées :

1. La donnée = l'adresse publique de nœud émetteur(Soit le médecin traitant, le patient propriétaire de DES ou bien un nœud tiers qui possède un accès à ce DSE) + l'adresse publique de patient concerné + (les informations médicales) chiffrée avec la clé symétrique associe au DSE+ type de la donnée + l'horodatage + la visibilité pour le patient.
   - La transaction_DSE = la donnée + (la donnée) signée avec la clé privée de l'émetteur.

- **Consulter un DSE :** cette fonctionnalité est effectuée par chaque acteur qui possède un accès au DSE de patient concerné.
- **Valider les blocs :** c'est la fonctionnalité de mineur qui est le médecin traitant, la validation est un processus qui se déroule comme suit :
  1. Récupérer les transactions à valider de la file d'attente.
  2. Valider les transactions : vérifier la signature des transactions.
  3. Pour chaque transaction valide, le mineur va créer un nouveau bloc en respectant la structure définit dans le chapitre précédent.
  4. En fin, le bloc valide va être ajouté à la blockchain.

a. Les besoins non fonctionnels

Notre système doit répondre aux exigences suivantes :

- **L'authentification :** l'accès des utilisateurs à notre système est géré par une double authentification, la première se fait avec un login et un mot de passe pour accéder au serveur du système et la deuxième avec une clé privée pour accéder à la blockchain.
- **La sécurité :**
  1. Le critère "authentification" assure déjà une partie de la sécurité ;
  2. Les données médicales des patients sont protégées par des fonctions cryptographiques.
  3. L'utilisation de la technologie blockchain pour le stockage des DSE assure un grand niveau de sécurité.
  4. Les transactions stockées dans la blockchain sont anonymes, cela protège la vie privée des utilisateurs.
- **L'ergonomie et l'utilisabilité :** les interfaces doivent être conviviales et ergonomiques. Elles doivent être simples, lisibles, confortables à l'œil et faciles à utiliser.
- **Maintenabilité :** le code doit être compréhensible afin d'assurer son évolution et son extensibilité pour répondre aux besoins des utilisateurs.
- **Performance :** un logiciel doit être avant tout performant c'est à dire à travers ses fonctionnalités, répond aux exigences des utilisateurs d'une manière optimale.

2.4 Expression des besoins

2.4.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation décrit les grandes fonctions d'un système du point de vue des acteurs, mais n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation. Bien que de nombreux diagrammes d'UML permettent de décrire un cas, il est recommandé de rédiger une description textuelle, car c'est une forme souple qui convient dans bien des situations [40].

La figure IV.3 représente le diagramme de cas d'utilisation du notre système :

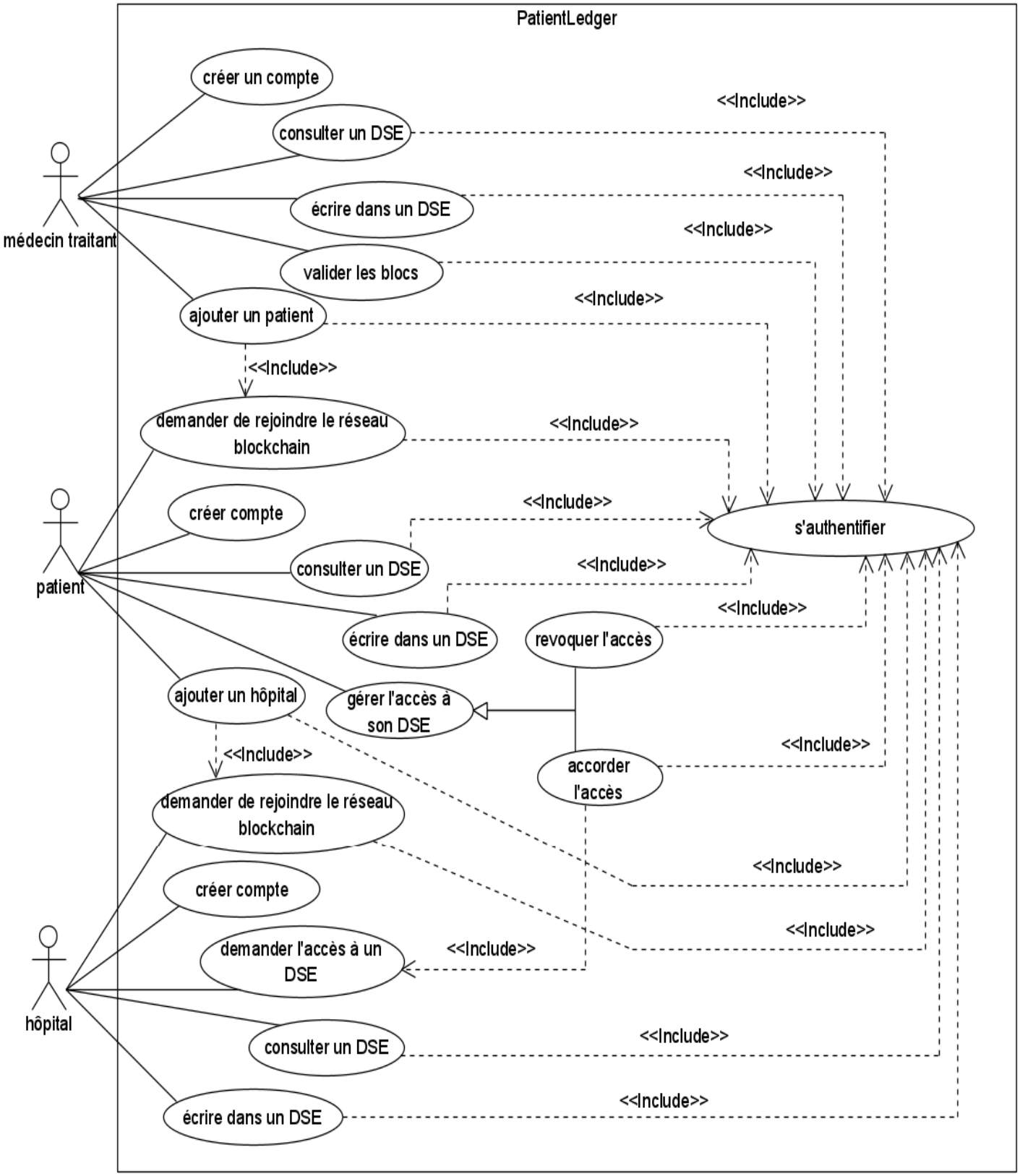

FIGURE IV.3 – Diagramme de cas d'utilisation de notre système.

2.4.2 Description des cas d'utilisations

a. Description du cas «S'authentifier»

Titre	S'authentifier.
Acteur	Médecin traitant, patient et hôpital.
Description	Ce cas d'utilisation permet aux utilisateurs de se connecter à notre système, en commençant par se connecter au serveur puis à la blockchain.
Pré-condition	- Application lancée.
Scénario nominal	1. L'utilisateur demande de se connecter ; 2. Le système lui renvoie le formulaire de connexion ; 3. L'utilisateur introduit son login et son mot de passe puis il le valide ; 4. Le système vérifie la saisie ; 5. Le système vérifie l'existence de l'utilisateur dans la BDD ; 6. Le système lui renvoie un formulaire de connexion à la blockchain ; 7. L'utilisateur saisie sa clé privée puis il le valide ; 8. Le système vérifie la saisie ; 9. Le système vérifie l'existence de l'utilisateur dans la blockchain ; 10. Le système lui renvoie la page appropriée.
Scénario d'exception	- Erreur de saisie, l'enchainement démarre au point (4/8) ; (5/9) Le système affiche un message d'erreur et propose une nouvelle saisie, alors le scénario nominal reprend au point (3/7) ; - L'utilisateur n'existe pas dans la BDD du serveur/blockchain, l'enchainement démarre au point (5/9) (6/10) Le système affiche un message d'erreur.
Post-condition	L'utilisateur est authentifié, il accède à son profil.

TABLE IV.1 – «S'authentifier».

b. Description du cas «Créer un compte»

Titre	Créer un compte.
Acteur	Médecin traitant, patient et hôpital.
Description	Ce cas d'utilisation permet aux utilisateurs de créer un compte dans notre système.
Pré-condition	- Application lancée.
Scénario nominal	1. l'utilisateur demande de s'inscrire ; 2. Le système lui renvoie le formulaire d'inscription ; 3. L'utilisateur remplit le formulaire puis il le valide ; 4. Le système vérifie la saisie ; 5. Le système vérifie l'existence de l'utilisateur dans la BDD de serveur ; 6. Le système ajoute l'utilisateur à la BDD de serveur ; - Si utilisateur = le médecin traitant 7. Le système génère une transaction d'inscription pour le médecin ; 8. Le système ajoute la transaction à la blockchain.
Scénario d'exception	- Erreur de saisie, l'enchainement démarre au point (4) ; (5) Le système affiche un message d'erreur et propose une nouvelle saisie, alors le scénario nominal reprend au point (3) ; - L'utilisateur existe déjà dans la BDD du serveur, l'enchainement démarre au point (5) ; (6) Le système affiche un message d'erreur.
Post-condition	- Pour le médecin traitant, il accède à son profil. - Pour le patient et l'hôpital, ils seront redirigés vers l'interface où demander de rejoindre le réseau blockchain.

TABLE IV.2 – «Créer un compte».

c. Description du cas d'utilisation «Demander de rejoindre le réseau blockchain»

Dans ce cas, un demandeur (qui peut être un patient ou un hôpital) demande auprès d'un offreur (qui peut être un médecin traitant ou un patient) de l'ajouter au réseau blockchain.

Titre	Demander de rejoindre le réseau blockchain .
Acteur	Patient et hôpital.
Description	Ce cas d'utilisation permet au patient/à l'hôpital de demander auprès de médecin traitant/patient de l'ajouter au réseau Blockchain.
Pré-condition	- Application lancée ; - Le patient/l'hôpital doit être authentifié au serveur.
Scénario nominal	1. Le patient/l'hôpital demande de rejoindre le réseau blockchain ; 2. Le système lui renvoie un formulaire pour saisir la clé publique de médecin traitant/patient ; 3. Le patient/l'hôpital remplit le formulaire puis il le valide ; 4. Le système vérifie la saisie ; 5. Le système vérifie l'existence de médecin traitant/patient ; 6. Le système génère une notification à envoyer pour l'offreur ; 7. Le système affiche un message de confirmation de l'envoi de notification pour le patient/l'hôpital.
Scénario d'exception	- Erreur de saisie, l'enchainement démarre au point (4) (5) Le système affiche un message d'erreur et propose une nouvelle saisie, alors le scénario nominal reprend au point (3) ; - médecin traitant/patient n'existe pas, l'enchainement démarre au point (5) (6) Le système affiche un message d'erreur.
Post-condition	- Le médecin traitant/patient reçoit une notification.

TABLE IV.3 – «Demander de rejoindre le réseau blockchain».

d. Description du cas d'utilisation «Ajouter un patient»

Ce cas d'utilisation se déroule en deux scénarios, le premier du coté médecin traitant (offreur) et l'autre du coté patient (demandeur).

d.1. Ajouter un patient du coté médecin traitant :

Titre	Ajouter un patient du coté médecin traitant.
Acteur	Médecin traitant.
Description	Ce cas d'utilisation permet au médecin traitant d'ajouter un patient demandeur au réseau blockchain.
Pré-condition	- Application lancée ; - Le médecin traitant doit être authentifié ; - Le médecin traitant a reçu la notification de patient demandeur ;
Scénario nominal	1. Le médecin traitant demande la liste de ses demandes ; 2. Le système lui affiche la liste des demandes ; 3. Le médecin traitant accepte la demande d'ajout de ce patient au réseau blockchain ; 4. Le système affiche un message de confirmation.
Post-condition	- La demande du patient est acceptée.

TABLE IV.4 – «Ajouter un patient 1».

d.2. Ajouter un patient du coté patient :

Titre	Ajouter un patient du coté patient.
Acteur	Patient.
Description	Ce cas d'utilisation permet au médecin traitant d'ajouter un patient demandeur au réseau blockchain.
Pré-condition	- Application lancée ; - Le patient doit être authentifié au serveur ; - Le médecin traitant a accepté la demande de ce patient.
Scénario nominal	1. Le système génère une paire de clés (publique et privée) et une clé symétrique pour ce patient ; 2. Le système génère une transaction d'inscription pour le patient demandeur ; 3. Le système ajoute la transaction à la blockchain ; 4. Le système génère une transaction d'accès pour le patient ; 5. Le système génère une transaction d'accès pour le médecin ; 6. Le système ajoute les deux transactions d'accès à la blockchain ; 7. Le système envoie un message de confirmation au patient demandeur en lui affichant ses deux clés.
Post-condition	- Le patient est devenu un membre de réseau blockchain, il accède à son profil.

TABLE IV.5 – «Ajouter un patient 2».

e. Description du cas d'utilisation «Ajouter un hôpital»

Aussi, ce cas d'utilisation se déroule en deux scénarios, le premier du coté patient (offreur) et l'autre du coté hôpital (demandeur).

e.1. Ajouter un hôpital du coté patient :

Titre	Ajouter un hôpital du coté patient.
Acteur	Patient.
Description	Ce cas d'utilisation permet au patient d'ajouter l'hôpital demandeur au réseau blockchain.
Pré-condition	- Application lancée ; - Le patient doit être authentifié à la blockchain ; - Le patient a reçu une notification de l'hôpital demandeur.
Scénario nominal	1. Le patient demande la liste de ses demandes ; 2. Le système lui affiche la liste des demandes ; 3. Le patient accepte la demande d'ajout de ce patient au réseau blockchain ; 4. Le système affiche un message de confirmation.
Post-condition	- La demande de l'hôpital est acceptée.

TABLE IV.6 – «Ajouter un hôpital 1».

e.2. Ajouter un hôpital du coté hôpital :

Titre	Ajouter un hôpital du coté hôpital.
Acteur	Hôpital.
Description	Ce cas d'utilisation permet au patient d'ajouter l'hôpital demandeur au réseau blockchain.
Pré-condition	- Application lancée ; - L'hôpital doit être authentifié au serveur ; - Le patient a accepté la demande de ce patient.
Scénario nominal	1. Le système génère une paire de clés (publique et privée) et une clé symétrique pour cet hôpital ; 2. Le système génère une transaction d'inscription pour l'hôpital demandeur ; 3. Le système ajoute la transaction à la blockchain ; 4. Le système génère une transaction d'accès pour l'hôpital ; 5. Le système ajoute les deux transactions d'accès à la blockchain ; 6. Le système envoie un message de confirmation à l'hôpital demandeur en lui affichant ses deux clés.
Post-condition	- L'hôpital est devenu un membre de réseau blockchain, il accède à son profil.

TABLE IV.7 – «Ajouter un hôpital 2».

f. Description du cas d'utilisation «Demander l'accès à un DSE»

Titre	Demander l'accès à un DSE.
Acteur	Hôpital.
Description	Ce cas d'utilisation permet à l'hôpital de demander auprès d'un patient l'accès à son DSE.
Pré-condition	- Application lancée ; - L'hôpital doit être authentifié à la blockchain.
Scénario nominal	1. L'hôpital demande un accès à un DSE d'un patient ; 2. Le système lui renvoie un formulaire pour spécifier quel patient ; 3. L'hôpital saisit la clé publique de patient à qu'il veut demander un accès puis il le valide ; 4. Le système vérifie la saisie ; 5. le système vérifie l'existence de ce patient ; 6. Le système vérifie que cet hôpital ne possède pas déjà un accès au DSE de ce patient ; 7. Le système génère une notification à envoyer pour le patient ; 8. Le système affiche un message de confirmation pour l'hôpital.
Scénario d'exception	- Erreur de saisie, l'enchainement démarre au point (4) ; (5) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (3) ; - Le patient n'existe pas, l'enchainement démarre au point (5) ; 6) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (3) ; - L'hôpital possède déjà l'accès au DSE de ce patient, l'enchainement démarre au point (6) ; 7) Le système lui affiche un message d'erreur et lui propose d'écrire ou de consulter le DSE de ce patient.
Post-condition	Le patient reçoit une notification.

TABLE IV.8 – «Demander l'accès à un DSE».

g. Description du cas d'utilisation « Accorder l'accès à un DSE»

Titre	Accorder l'accès à un DSE.
Acteur	Le patient.
Description	Ce cas d'utilisation permet au patient d'accorder à l'hôpital l'accès à son DSE stocké dans la blockchain.
Pré-condition	- Application lancée ; - Le patient doit être authentifié à la blockchain ; - Le patient a reçu la notification de l'hôpital.
Scénario nominal	1. Le patient demande la liste des notifications reçus ; 2. le système affiche la liste de ses notification ; - Si le patient accepte la demande : 3. le patient accepte la demande d'accès à son DSE ; 4. Le système génère une transaction d'accès pour cet hôpital en utilisant la clé symétrique du patient ; 5. Le système ajoute la transaction à la blockchain ; - Si le patient rejette la demande : 3. le patient rejette la demande d'accès à son DSE ; 4. Le système supprime la demande rejetée de la BDD ; 6. le système affiche un message de confirmation.
Post-condition	- L'hôpital a un accès au DSE de ce patient.

TABLE IV.9 – «Accorder l'accès à un DSE».

h. Description du cas «Révoquer l'accès à un DSE»

Titre	Révoquer l'accès.
Acteur	Le patient.
Description	Ce cas d'utilisation permet au patient de révoquer à l'hôpital l'accès à son DSE stocké dans la blockchain.
Pré-condition	- Application lancée ; - Le patient doit être authentifié à la blockchain.
Scénario nominal	1. Le patient demande une révocation d'accès à son DSE ; 2. Le système lui renvoie le formulaire de révocation d'accès à son DSE ; 3. Le patient saisit la clé publique de l'hôpital à qui il veut révoquer cet accès puis il le valide ; 4. Le système vérifie la saisie ; 5. Le système vérifie que cet hôpital possède un accès au DSE de ce patient ; 6. Le système met à jour l'accès dans la BDD ; 7. Le système génère une transaction de révocation d'accès ; 8. Le système ajoute la transaction à la blockchain ; 9. Le système affiche un message de confirmation au patient que la révocation est faite avec succès.
Scénario d'exception	- Erreur de saisie, l'enchainement démarre au point (4) ; (5) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (3). - L'hôpital n'existe pas, l'enchainement démarre au point (5) ; 6) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (3) ;
Post-condition	- L'hôpital ne possède pas l'accès au DSE de ce patient.

TABLE IV.10 – «Révoquer l'accès à un DSE».

i. Description du cas « Écrire dans un DSE »

Titre	Écrire dans un DSE.
Acteur	Médecin traitant, patient et hôpital.
Description	Ce cas d'utilisation permet aux utilisateurs d'ajouter des informations pour un patient précis à la blockchain.
Pré-condition	- Application lancée ; - L'utilisateur doit être authentifié à la blockchain.
Scénario nominal	I. Si l'utilisateur est le médecin traitant : 1. Le Médecin traitant demande la liste de ses patients ; 2. Le système lui renvoie la liste des patients ; 3. Le médecin traitant choisit le patient à qui il veut ajouter des données ; II. Si l'utilisateur est l'hôpital : 1. L'hôpital demande d'ajouter des données médicales ; 2. Le système lui renvoie un formulaire pour spécifier quel patient ; 3. L'hôpital saisit la clé publique de patient à qui il veut ajouter des données ; 4. Le système vérifie la saisie ; 5. Le système vérifie est-ce que cet hôpital possède un accès au DSE de patient spécifié ; III. Si l'utilisateur est le patient : 1. Le patient demande d'ajouter ses données médicales ; - Les étapes suivantes sont communes entre les cas précédents : a. Le système lui renvoie la page d'ajout au DSE ; b. L'utilisateur introduit les données médicales puis il les valide ; c. Le système génère une transaction d'ajout de données médicales ; d. Le système envoie la transaction à la file d'attente des transactions non encore validées ;. e. Le système renvoie un message de confirmation à l'utilisateur.
Scénario d'exception	- Erreur de saisie, l'enchainement démarre au point (II.4) ; (II.5) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (II.3) ; -Le patient n'existe pas, l'enchainement démarre au point (II.5) (II.6) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (II.4) ; - L'hôpital ne possède pas l'accès au DSE de ce patient, l'enchainement démarre au point (II.5) ; (a) Le système lui affiche un message d'erreur et lui propose de demander un accès au DSE de ce patient.
Post-condition	Les données médicales sont ajoutées à la blockchain.

TABLE IV.11 – «Écrire dans un DSE».

j. Description du cas « Consulter un DSE »

Titre	Consulter un DSE.
Acteur	Médecin traitant, patient et hôpital.
Description	Ce cas d'utilisation permet aux utilisateurs de consulter le DSE d'un patient précis.
Pré-condition	- Application lancée ; - L'utilisateur doit être authentifié à la blockchain.
Scénario nominal	I. Si l'utilisateur est le médecin traitant : 1. Le Médecin traitant demande la liste de ses patients ; 2. Le système lui renvoie la liste des patients ; 3. Le médecin traitant choisit le patient à consulter son DSE ; II. Si l'utilisateur est l'hôpital : 1. L'hôpital demande de consulter un DSE d'un patient ; 2. Le système lui renvoie un formulaire pour spécifier quel patient ; 3. L'hôpital saisit la clé publique de patient à consulter son DSE puis il le valide ; 4. Le système vérifie la saisie ; 5. Le système vérifie que cet hôpital possède un accès au DSE de patient spécifié ; III. Si l'utilisateur est le patient : 1. Le patient demande de consulter son DSE ; - Les étapes suivantes sont communes entre les cas précédents : a. Le système récupère toutes les données médicales qui concernent le patient spécifié depuis la blockchain ; b.Le système renvoie la page de consultation de DSE à l'utilisateur.
Scénario d'exception	-Erreur de saisie, l'enchainement démarre au point (II.4) ; (II.5) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (II.3) ; -Le patient n'existe pas, l'enchainement démarre au point (II.5) (II.6) Le système lui affiche un message d'erreur et lui propose une nouvelle saisie, alors le scénario nominal reprend au point (II.4) ; - L'hôpital ne possède pas l'accès au DSE de ce patient, l'enchainement démarre au point (II.5) ; (a) Le système lui affiche un message d'erreur et lui propose de demander un accès au DSE de ce patient.
Post-condition	Les données médicales de patient spécifié sont affichées pour l'utilisateur.

TABLE IV.12 – «Consulter un DSE».

k. Description du cas «valider les blocs»

Titre	valider les blocs.
Acteur	Médecin traitant.
Description	Ce cas d'utilisation permet au médecin traitant d'exécuter un processus de validation des blocs.
Pré-condition	- Application lancée ; - Le médecin traitant doit être authentifié à la blockchain.
Scénario nominal	1. Le médecin traitant demande de valider les blocs ; 2. Le système récupère la liste de transactions à valider depuis la file d'attente ; - Pour chaque transaction : 3. Le système vérifie la validité de la transaction ; 4. Le système récupère le bloc précédent de la blockchain ; 5. Le système crée un nouveau bloc avec la transaction validée ; 6. Le système ajoute le bloc valide à la blockchain ; 7. Le système affiche un message de confirmation pour le médecin traitant.
Scénario d'exception	- Dans l'étape(2) : si la liste des transactions récupérée est vide, le système lui affiche un message en lui indiquant qu'il n'a aucune transaction non encore validée. -si la transaction n'est pas valide, l'enchainement démarre au point (3) (4) Le système lui affiche un message en lui indiquant que la transaction n'est pas valide.
Post-condition	les blocs sont ajoutés à toutes les copies de la blockchain.

TABLE IV.13 – «Valider les blocs».

2.4.3 Diagramme de séquence

Le diagramme de séquence précise les échanges de messages (déclenchant des événements) entre acteurs et objets (ou entre objets et objets) de manière chronologique, l'évolution du temps se lisant de haut en bas [42].

2.4.4 Description des diagrammes de séquences

a. Diagramme de séquence du cas « S'authentifier »

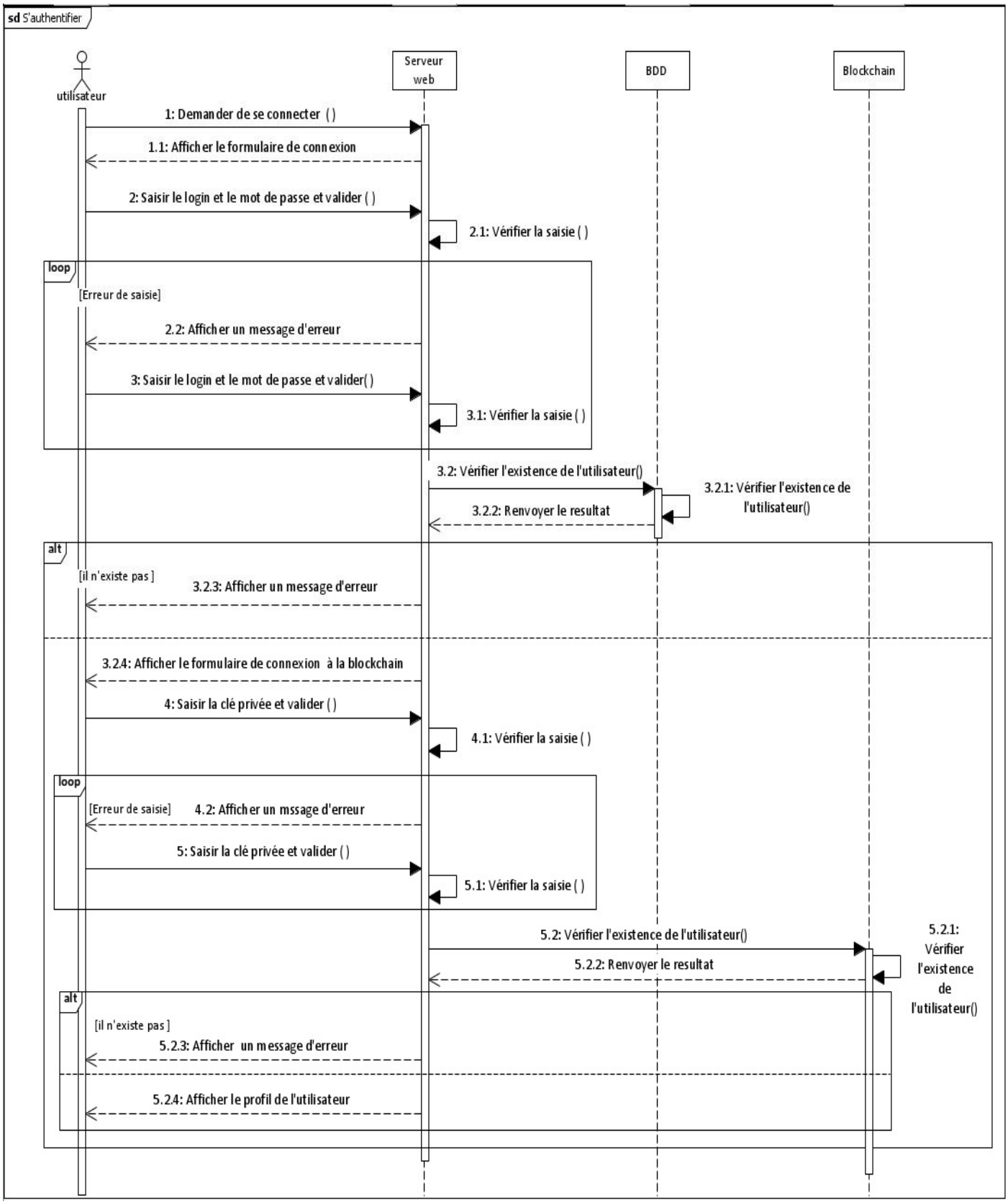

FIGURE IV.4 – Diagramme de séquence du cas «S'authentifier».

b. Diagramme de séquence du cas «Créer un compte»

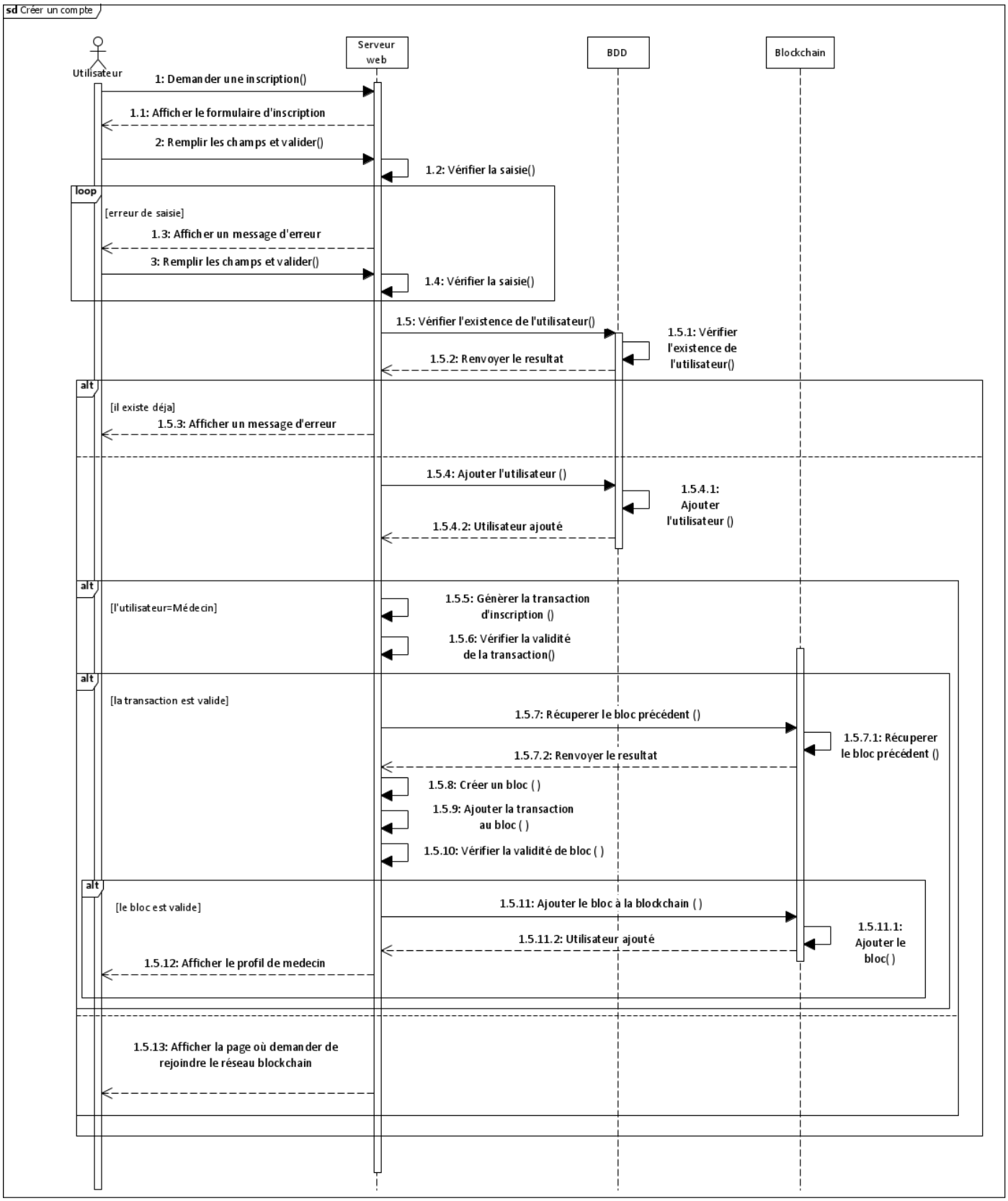

FIGURE IV.5 – Diagramme de séquence du cas « Créer un compte ».

c. Diagramme de séquence du cas «Demander de rejoindre le réseau blockchain»

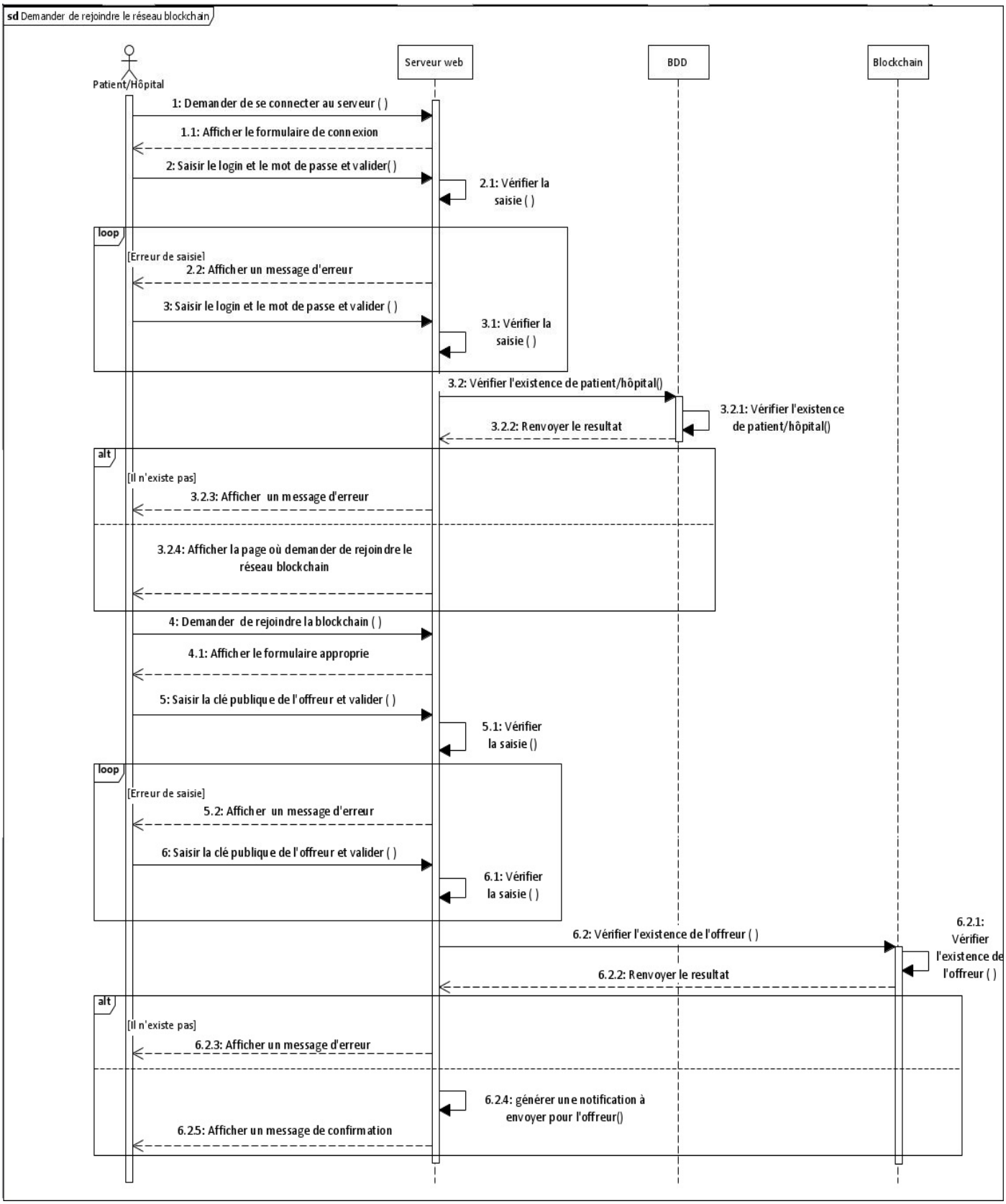

FIGURE IV.6 – Diagramme de séquence du cas «Demander de rejoindre le réseau blockchain».

d. Diagrammes de séquence du cas «Ajouter un patient/hôpital»

Ce cas d'utilisation se déroule en deux scénarios, le premier du coté médecin traitant ou patient (offreur) et l'autre du coté patient ou hôpital (demandeur).

d.1. Diagrammes de séquence du cas «Ajouter un patient/hôpital 1»

Ce diagramme de séquence représente les interactions de l'offreur avec le système. Après que l'offreur reçoit la demande de rejoindre le réseau blockchain de la part du demandeur, il peut soit l'accepter soit de la rejeter.

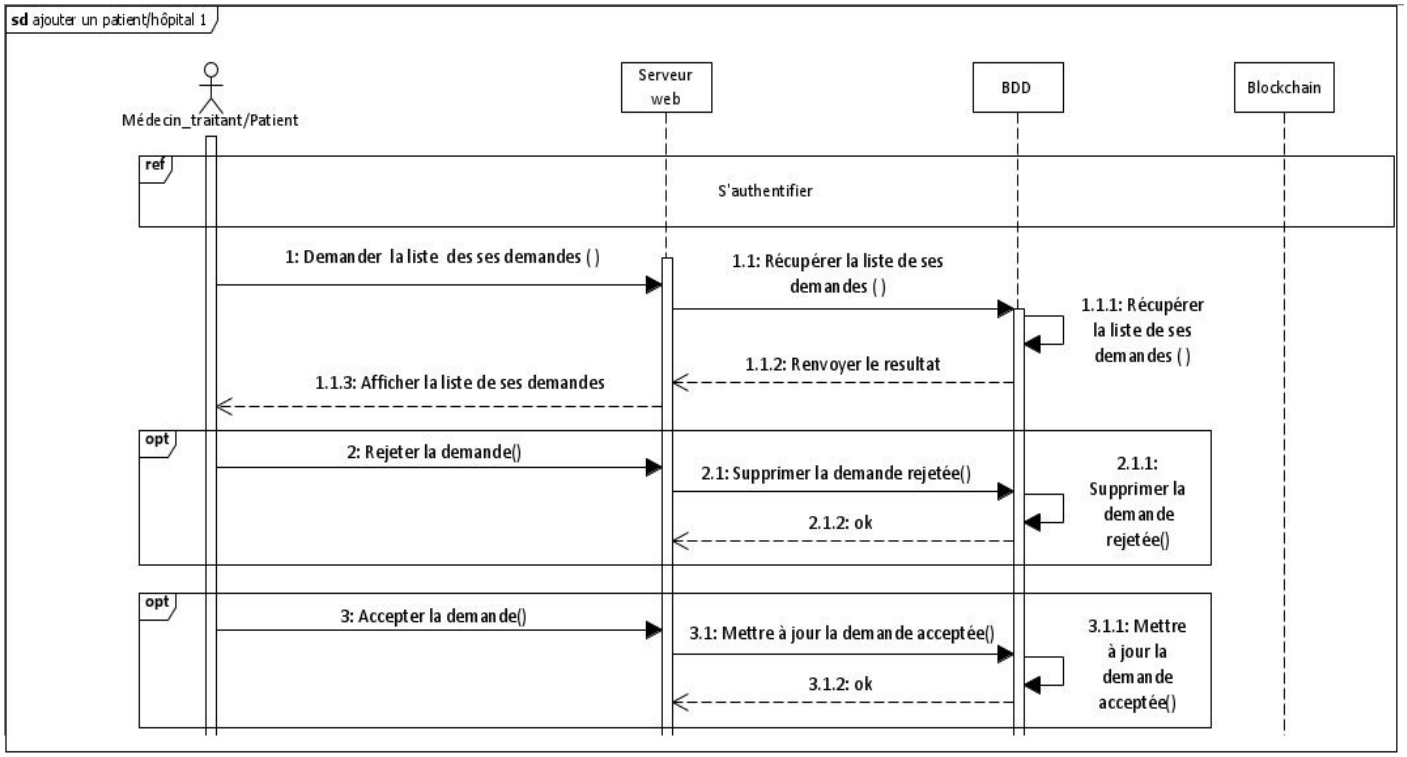

FIGURE IV.7 – Diagramme de séquence du cas «Ajouter un patient/hôpital 1».

d.2. Diagramme de séquence du cas «Ajouter un patient/hôpital 2»

Ce diagramme de séquence représente les interactions de demandeur avec le système après que sa demande de rejoindre le réseau blockchain a été acceptée.

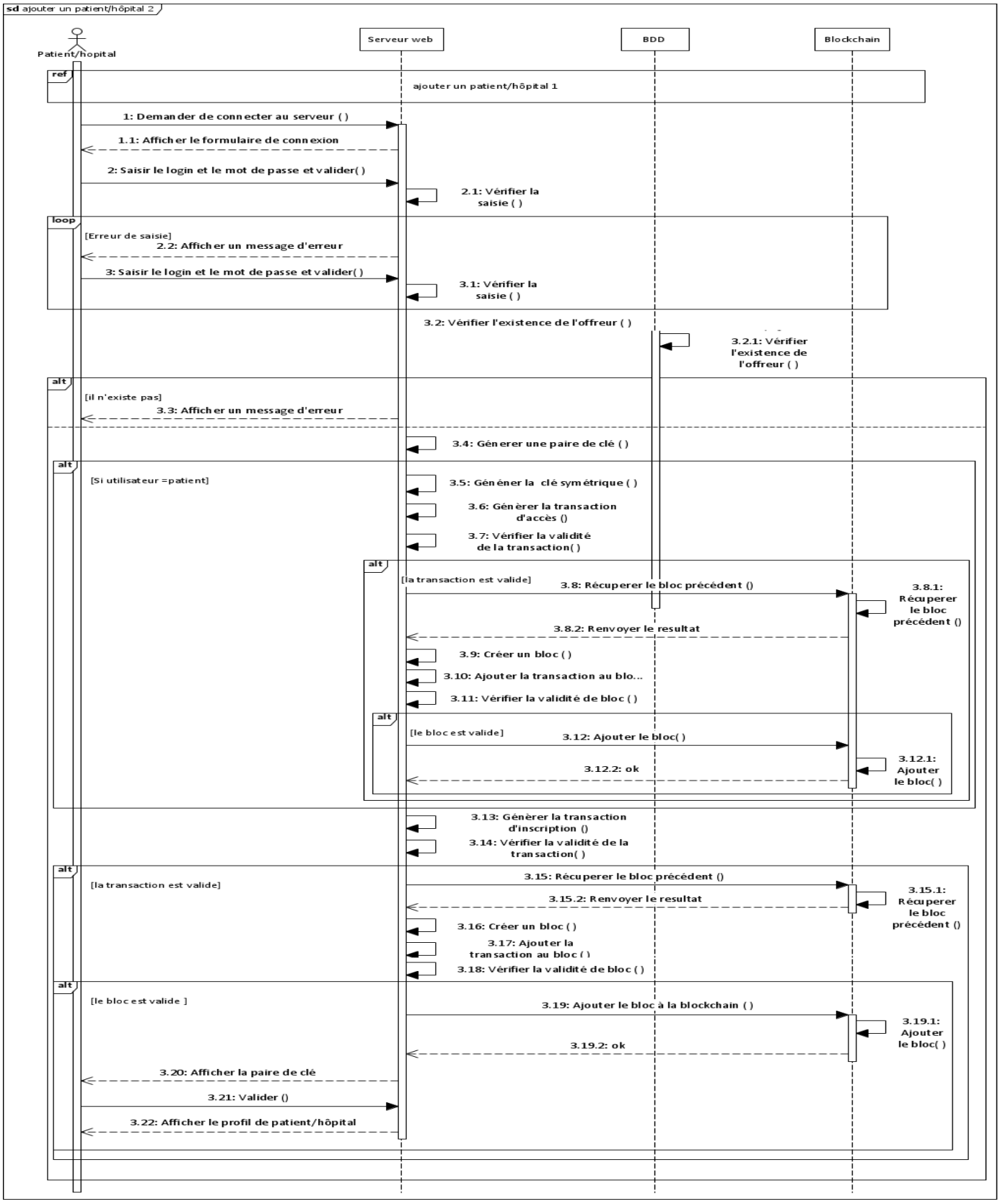

FIGURE IV.8 – Diagramme de séquence du cas «Ajouter un patient/hôpital 2».

e. Diagramme de séquence du cas «Demander un accès»

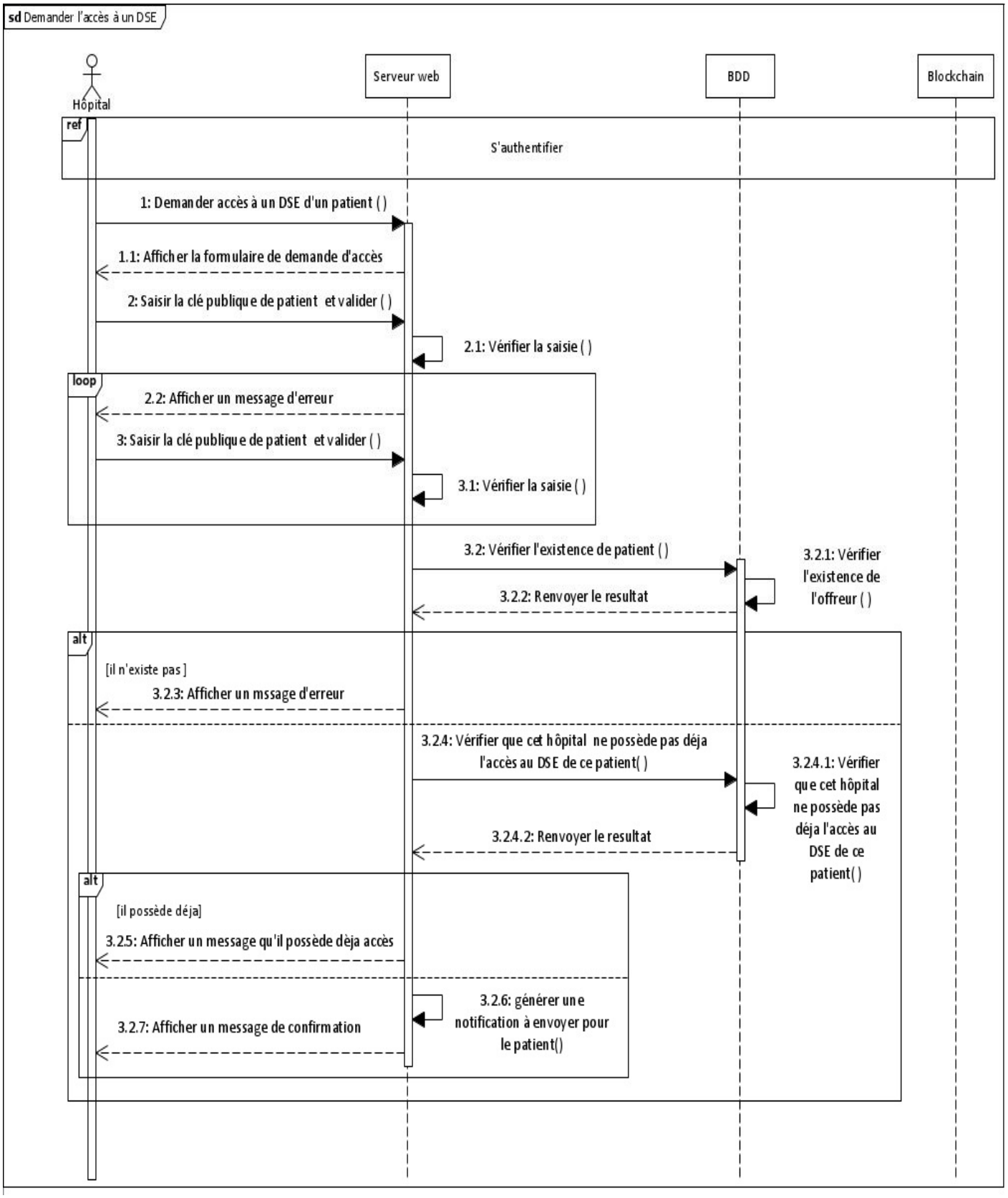

FIGURE IV.9 – Diagramme de séquence du cas «Demander un accès».

f. Diagramme de séquence du cas «Accorder l'accès à un DSE»

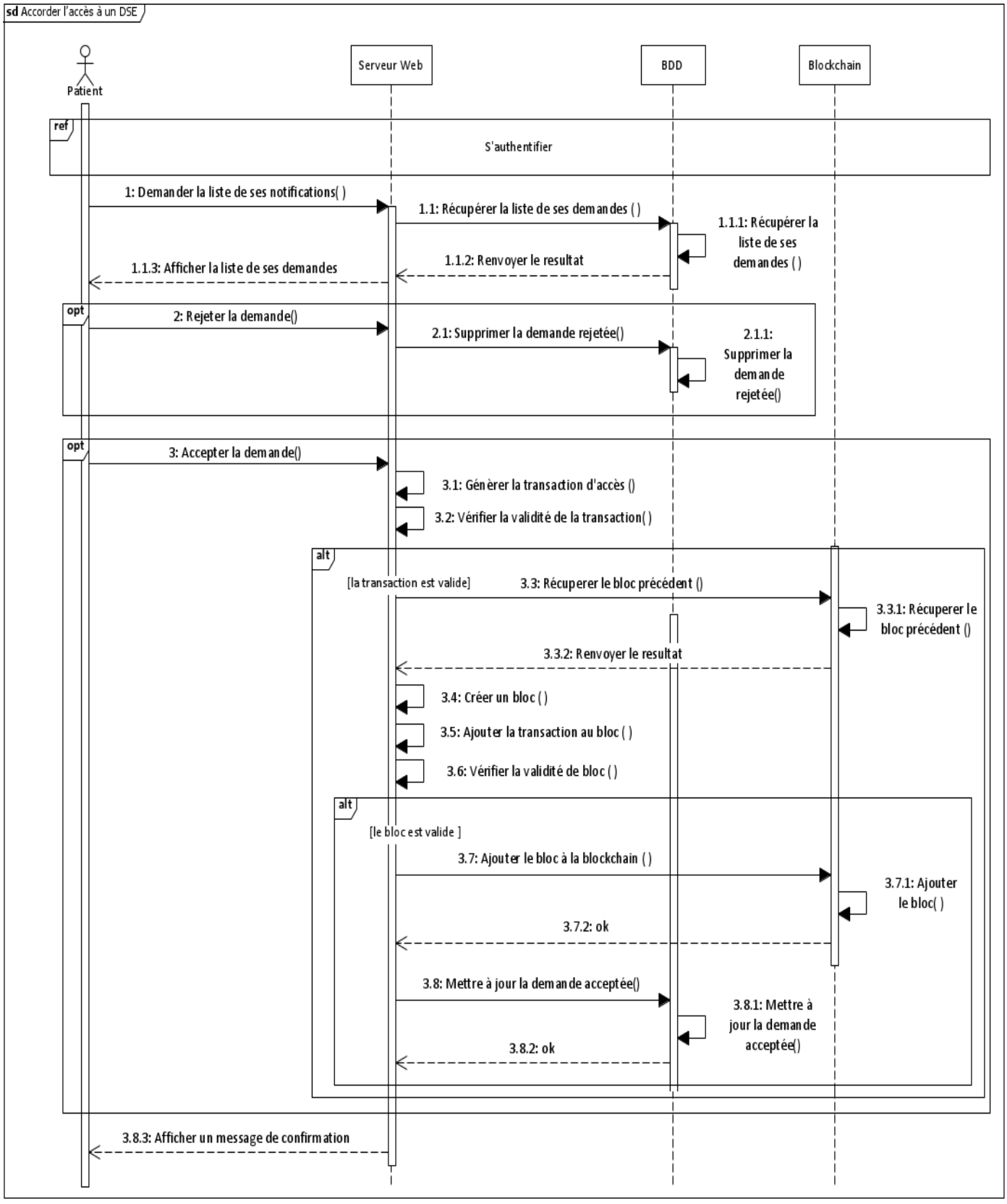

FIGURE IV.10 – Diagramme de séquence du cas «Accorder l'accès à un DSE».

g. Diagramme de séquence du cas « Révoquer l'accès à un DSE »

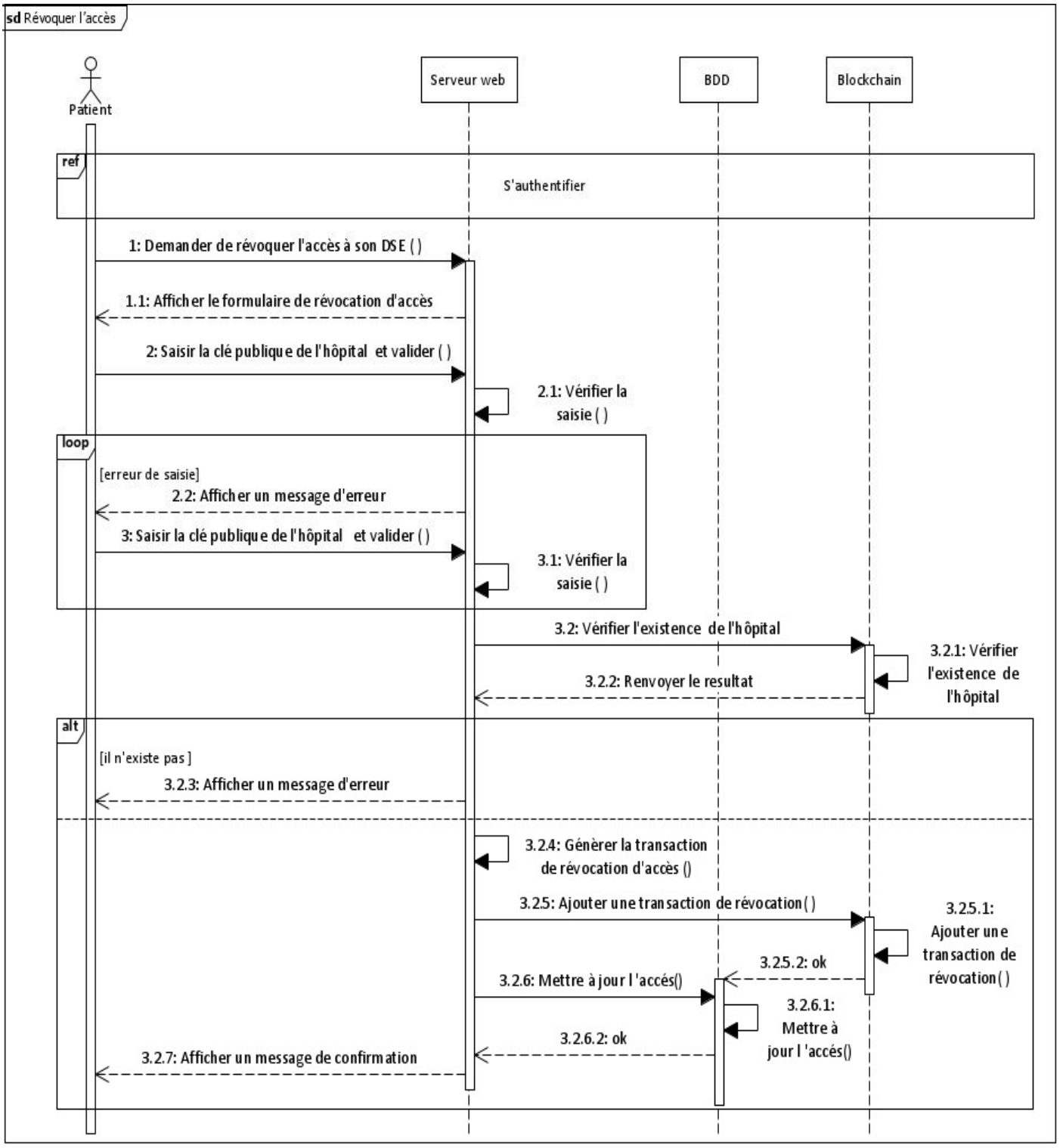

FIGURE IV.11 – Diagramme de séquence du cas «Révoquer l'accès à un DSE».

h. Diagramme de séquence du cas «Écrire dans un DSE»

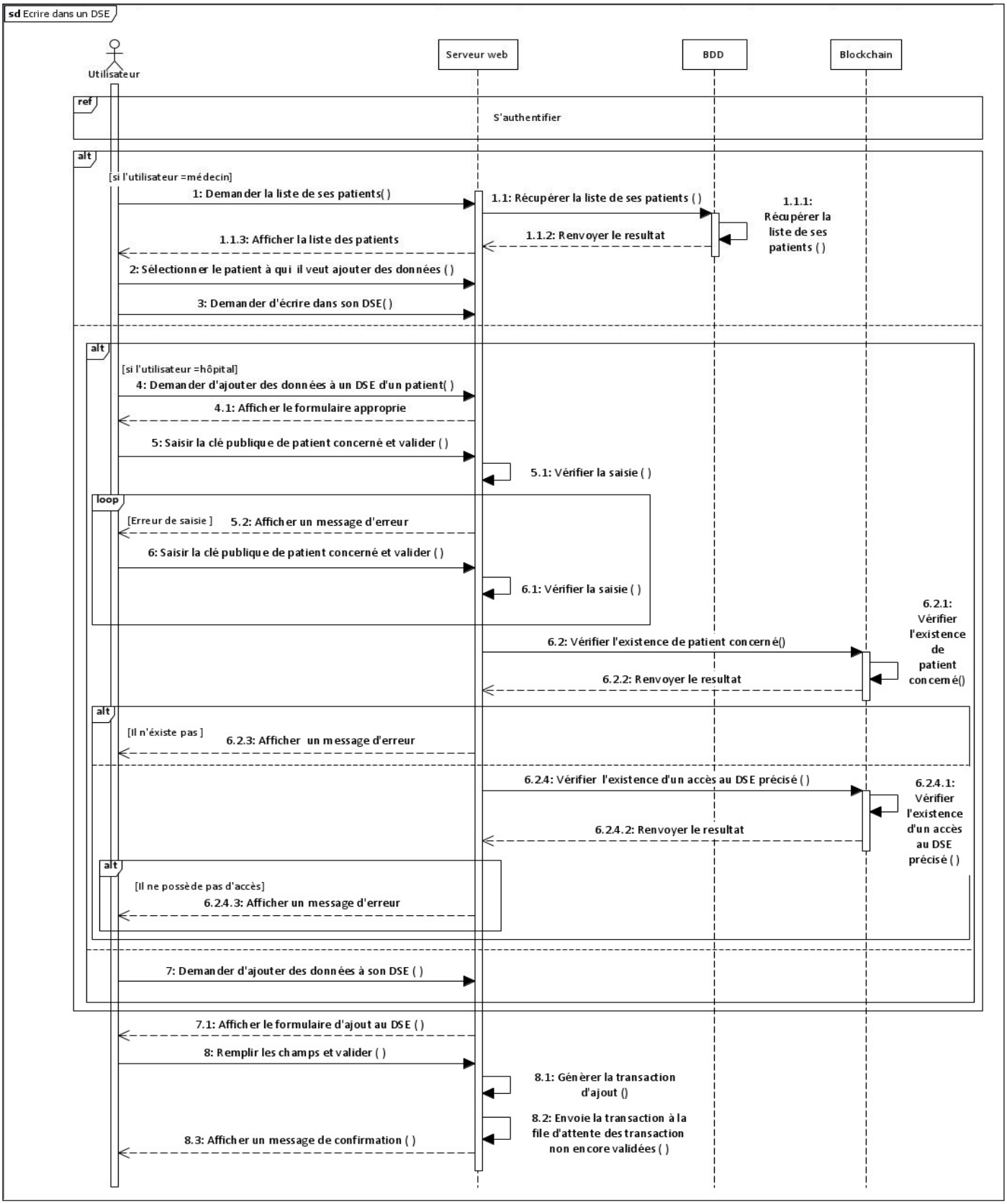

FIGURE IV.12 – Diagramme de séquence du cas «Écrire dans un DSE».

i. Diagramme de séquence du cas «Consulter un DSE»

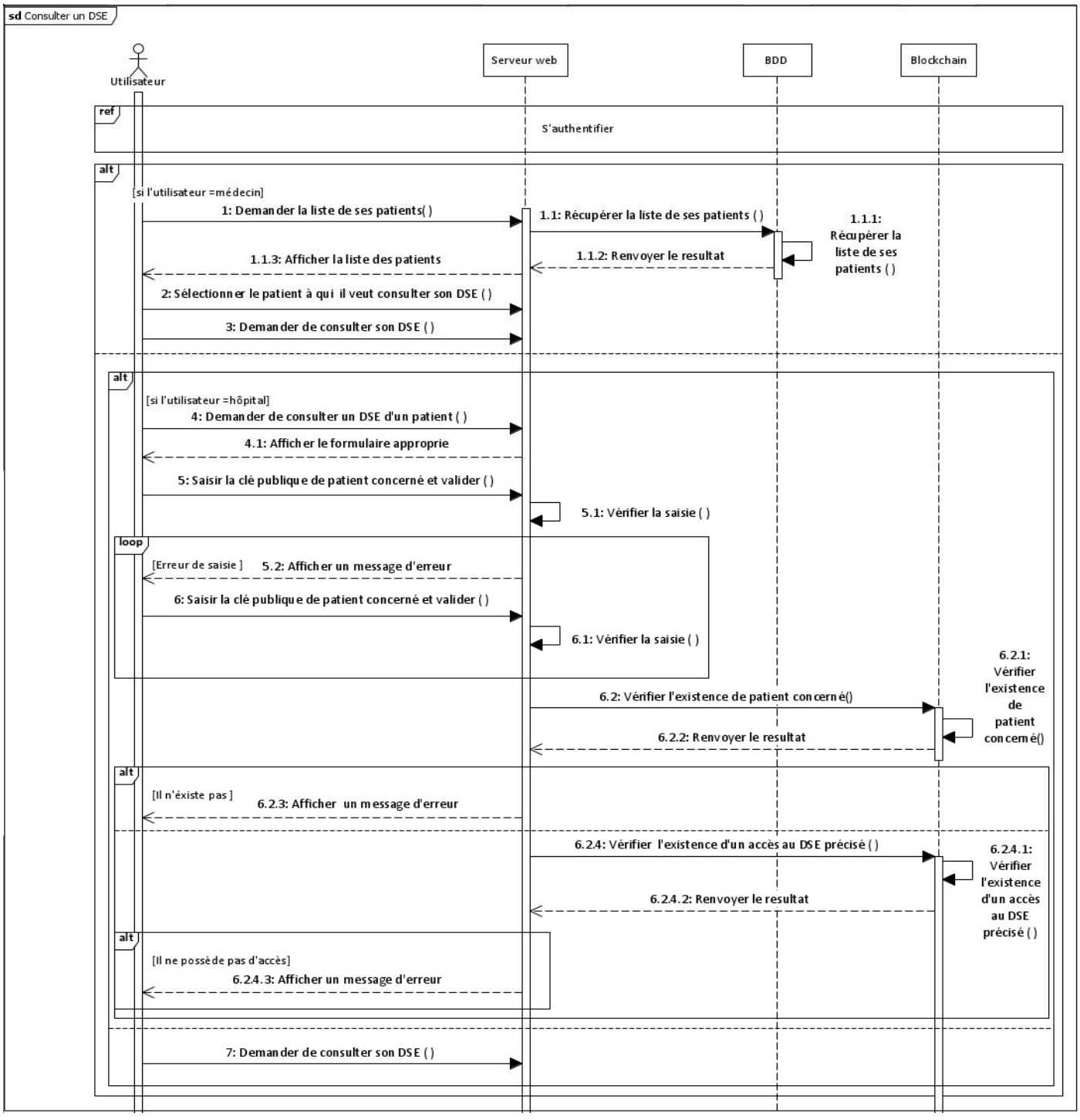

FIGURE IV.13 – Diagramme de séquence du cas «Consulter un DSE».

j. Diagramme de séquence du cas «Valider les blocs»

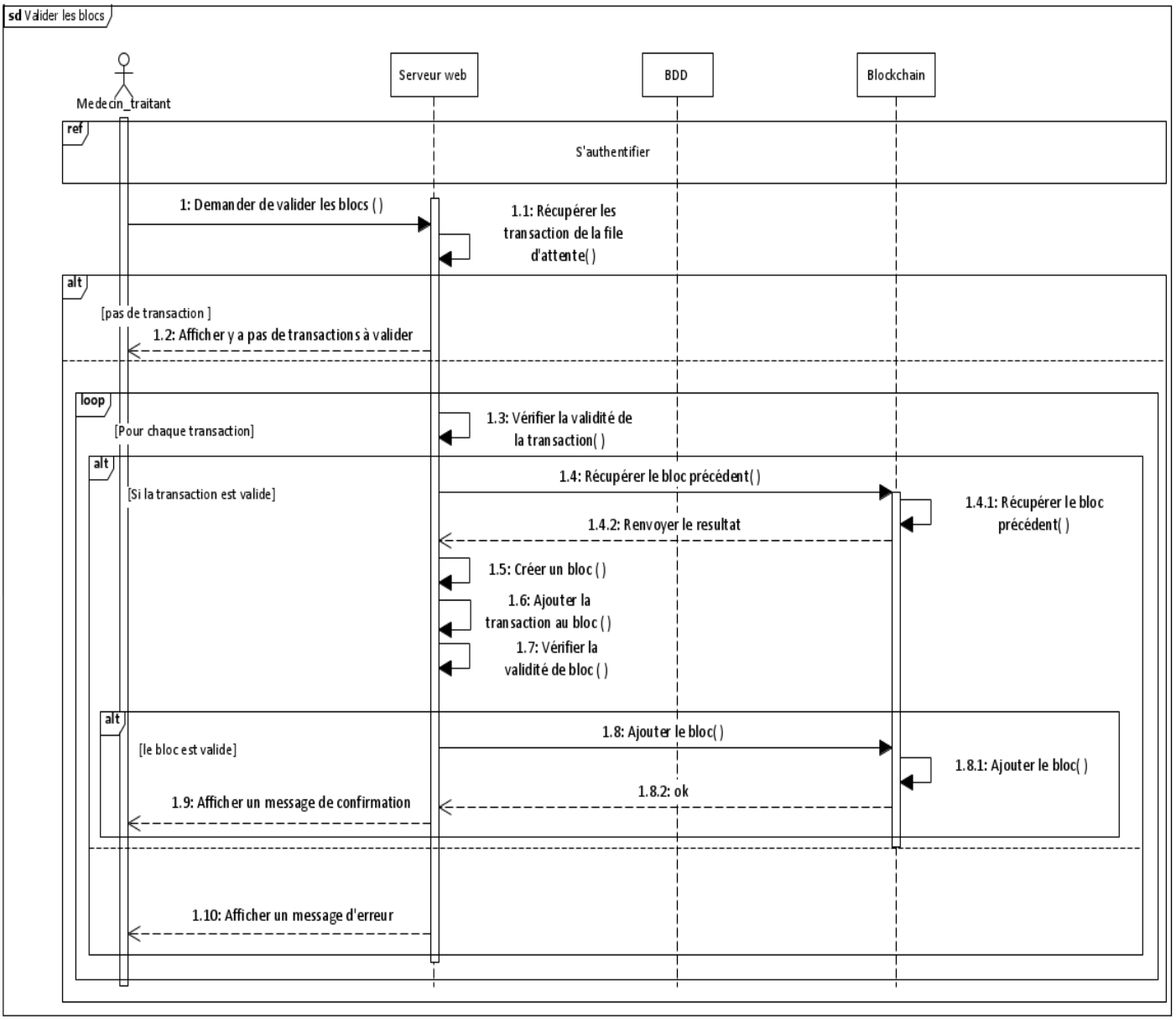

FIGURE IV.14 – Diagramme de séquence du cas «Valider les blocs».

2.4.5 Dictionnaire de données

Pour la spécification des champs et attributs des tables de notre diagramme de classes, nous proposons le dictionnaire de données suivant :

Classe	Attribut	Signification	Type	Taille
Utilisateur	idUt	Identifiant de l'utilisateur	integer	10
	nom	Nom de l'utilisateur	string	40
	prenom	Le prénom de l'utilisateur	string	40
	sexe	Le sexe de l'utilisateur	string	10
	email	L'émail de l'utilisateur	string	50
	tel	Numéro de téléphone de l'utilisateur	string	30
	adresse	L'adresse de l'utilisateur	string	100
	login	Le login de l'utilisateur	string	30
	motdepasse	Mot de passe de l'utilisateur	string	30
	clePublique	La clé publique de l'utilisateur	string	255
	idmed	Identifiant de médecin traitant	integer	10
Patient	dateNais	La date de naissance de patient	date	/
Medecin_traitant	specialite	La spécialité du médecin traitant	string	30
Hopital	nomHop	Nom de l'hôpital	string	40
	email	L'émail de l'hôpital	string	50
	telHop	Numéro de téléphone de l'hôpital	string	30
	adresseHop	L'adresse de l'hôpital	string	100
	loginHop	Le login de l'hôpital	string	30
	motdepasseHop	Mot de passe de l'hôpital	string	30
	clePubliqueHop	La clé publique de l'hôpital	string	255
Autoriser	idpatient	Identifiant de patient	integer	10
	idHop	Identifiant de l'hôpital	integer	10
	accorder	Accorder l'accès à un DSE d'un patient	booleen	/

TABLE IV.14 – Dictionnaire de données

2.4.6 Diagramme de classes

Le diagramme de classes constitue l'un des pivots essentiels de la modélisation avec UML. En effet, ce diagramme permet de donner la représentation statique du système à développer. Cette représentation est centrée sur les concepts de classe et d'association. Chaque classe se décrit par les données et les traitements dont elle est responsable pour elle-même et vis-à-vis d'autres classes. Les traitements sont matérialisés par des opérations. Le détail des traitements n'est pas représenté directement dans le diagramme de classes [43].

La figure IV.15 représente le diagramme de classes de notre système :

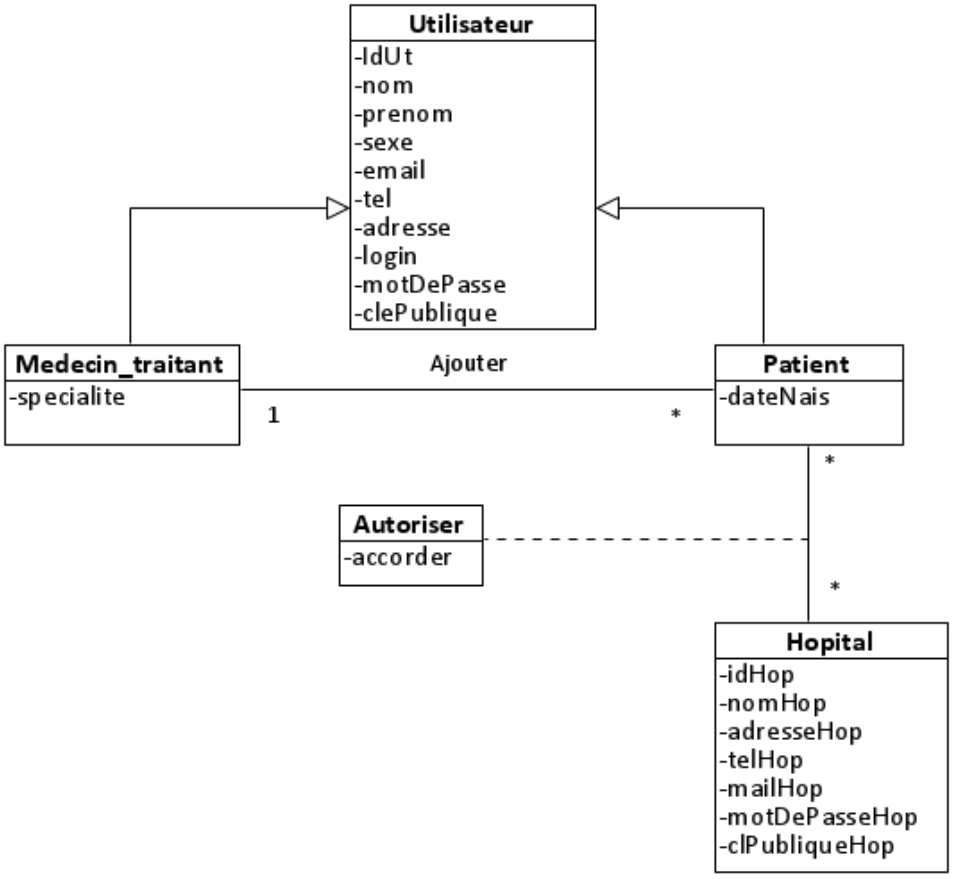

FIGURE IV.15 – Diagramme de classe de notre système.

2.4.7 Modèle relationnel

Le modèle relationnel est basé sur une organisation des données sous forme de tables. La manipulation des données se fait selon le concept mathématique de relation de la théorie des ensembles «l'algèbre relationnelle». Elle est constituée d'un ensemble d'opérations formelles sur les relations. Les opérations relationnelles permettent de créer une nouvelle relation (table) à partir d'opérations élémentaires sur d'autres tables [44].

Pour pouvoir passer à l'implémentation d'une base de données relationnelle, nous avons appliqué les règles de passages sur notre diagramme de classes ce qui a donné le modèle relationnel suivant :

- Medecin_traitant (#idUt, nom, prenom, sexe, email, tel, adresse, login, motDePasse, clePublique, specialite) ;
- Patient (#idUt, nom, prenom, sexe, email, tel, adresse, photo, login, motDePasse, clePublique, dateNais,#idmed) ;
- Hopital (idHop, nomHop, adresseHop, telHop, mailHop, loginHop, motDePasseHop, clePubliqueHop) ;
- Autoriser (#idHop, #idpatient, accorder).

3 La partie réalisation

3.1 L'environnement et les outils de développement

Dans cette partie nous allons énumérer les différents langages, bibliothèques et logiciels utilisés pour la mise en œuvre de notre application. Le choix de ces derniers repose sur leurs simplicités d'utilisation et leurs manières de fournir des résultats correspondant à nos besoins.

3.1.1 Les langages de programmation

a. PHP (HyperText Preprocessor)

Hypertext Preprocessor, est un langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. C'est un langage impératif orienté objet.

Il a permis de créer un grand nombre de sites web célèbres, comme Facebook, Wikipédia, etc. Il est considéré comme une des bases de la création de sites web dits dynamiques mais également des applications web [45].

b. JavaScript

JavaScript est un langage (plus précisément orienté objet) de programmation qui permet d'implémenter des mécanismes complexes sur une page web. C'est un langage principalement côté client qui permet de rendre dynamiques et interactives les pages web [46].

c. HTML (HyperText MarkupLanguage)

HTML est un langage informatique créer des pages web pour l'Internet. Ce langage permet de réaliser de l'hypertexte à base d'une structure de balise.

Par ailleurs, HTML n'est pas un langage de programmation proprement dit comme C, C++, etc. mais plutôt un langage qui permet de mettre en forme du contenu et de naviguer d'une page à une autre dans nos modules par l'intermédiaire d'hyperliens [47].

e. CSS (Cascading Style Sheets)

Le terme CSS est l'acronyme en anglais de Cascading Style Sheets qui peut se traduire par «feuille de style en cascade». Le CSS est un langage informatique utilisé sur l'internet pour mettre en forme les fichiers HTML ou XML [48].

f. JSON (JavaScript Object Notation – Notation Objet issue de JavaScript)

JSON est un format léger d'échange de données. Il est facile à lire ou à écrire pour des humains. Il est aisément analysable ou générable par des machines. Il est basé sur un sous-ensemble du langage de programmation JavaScript.

JSON est un format texte complètement indépendant de tout langage, mais les conventions qu'il utilise seront familières à tout programmeur habitué aux langages descendants du C, comme par exemple : C lui-même, Java, JavaScript, Perl, Python, PHP et bien d'autres. Ces propriétés font de JSON un langage d'échange de données idéales. JSON se base sur deux structures :

- Une collection de couples nom/valeur.
- Une liste de valeurs ordonnée [49].

3.1.2 Les bibliothèques

a. Bootstrap

Bootstrap est une bibliothèque (framework) gratuite, basée sur HTML, CSS et JavaScript. Elle est créée par deux développeurs, Mark Otto et Jacob Thornton, pour construire les interfaces d'application (ou site) web compatibles avec tous les équipements avec différentes tailles d'écran (PC, tablette, smartphone)[50].

b. OpenSSL

OpenSSL est une boîte à outils robuste, de qualité commerciale et complète pour les protocoles Transport Layer Security (TLS) et Secure Sockets Layer (SSL). C'est également une bibliothèque de cryptographie à usage général. Cette bibliothèque offre une commande en ligne (openssl) permettant de :

- la création de clés RSA, DSA (signature)
- la création de certificats X509
- le calcul d'empreintes (MD5, SHA, RIPEMD160, ...)
- le chiffrement et déchiffrement (DES, IDEA, RC2, RC4, Blowfish, ...)
- etc [51].

OpenSSL est concédé sous une licence de style Apache, ce qui signifie essentiellement la liberté de l'obtenir et de l'utiliser à des fins commerciales et non commerciales sous réserve de quelques conditions de licence simples.

3.1.3 Les environnements de programmation

a. Visual Studio Code

Visual Studio Code est éditeur de texte open source, gratuit et multiplateforme (Windows, Mac et Linux), développé par Microsoft. Principalement conçut pour le développement d'applications avec JavaScript, TypeScript et Node.js, l'éditeur peut s'adapter à d'autres types de langages grâce à un système d'extension bien fourni[52].

b. Wamp Server

WampServer est une plateforme de développement Web sous Windows, permettant de faire fonctionner localement des scripts PHP. WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL [53].

c. MySQL

MySQL est un système de gestion de base de données relationnelle (SGBDR) parmi les plus populaires au monde. Il est distribué sous double licence, une licence publique générale GNU et une propriétaire selon l'utilisation qui en est faites. La première version de MySQL est apparue en 1995 et l'outil est régulièrement entretenu [54].

b. PhpMyAdmin

phpMyAdmin est un logiciel libre écrit en PHP qui a pour mission de s'occuper de l'administration d'un serveur de base de données MySQL ou MariaDB. Vous pouvez utiliser phpMyAdmin pour réaliser la plupart des tâches d'administration, ceci incluant la création de base de données, l'exécution de demandes, et l'ajout de comptes utilisateur [55].

d. Apache

Apache est l'un des serveurs les plus répondus sur internet et il est basé sur le protocole http. L'objectif est de fournir, à un maximum de personne et d'organisation, une plateforme solide pour réaliser des testes et des applications de productions, l'un des avantages est son fonctionnement sur de nombreux système d'exploitation dont Microsoft, Linux, etc [56].

3.1.4 Utilité des outils pour notre application

- Pour implémenter la partie interfaces de notre site web, nous avons utilisé les langages HTML, CSS et JS et pour rendre notre site adaptatif à tous les types des équipements, nous avons utilisé aussi la bibliothèque Bootstrap.
- Pour implémenter la partie processus de notre application, nous avons utilisé le langage PHP.
- Nous avons utilisé le format JSON pour structurer les transactions, les blocs et la blockchain.
- Nous avons utilisé la boite à outil OpenSSL pour le cryptage et la génération des clés pour chaque acteur.

3.2 La description de quelques interfaces

Dans cette partie nous allons présenter à l'aide des interfaces, les services qu'offre notre application.

3.2.1 Inscription

Si l'utilisateur qui visite le site web est invité (c'est-à-dire ne possède pas de compte), donc il doit s'inscrire sur le site. En cliquant sur le lien "S'inscrire" il trouvera un modal (figure IV.16) pour préciser quel type visiteur est-il (médecin traitant, patient ou hôpital), puis un formulaire approprié va apparaître dans la page d'inscription (figure IV.17), où l'utilisateur introduit toutes ses informations personnelles.

FIGURE IV.16 – Choix de visiteur.

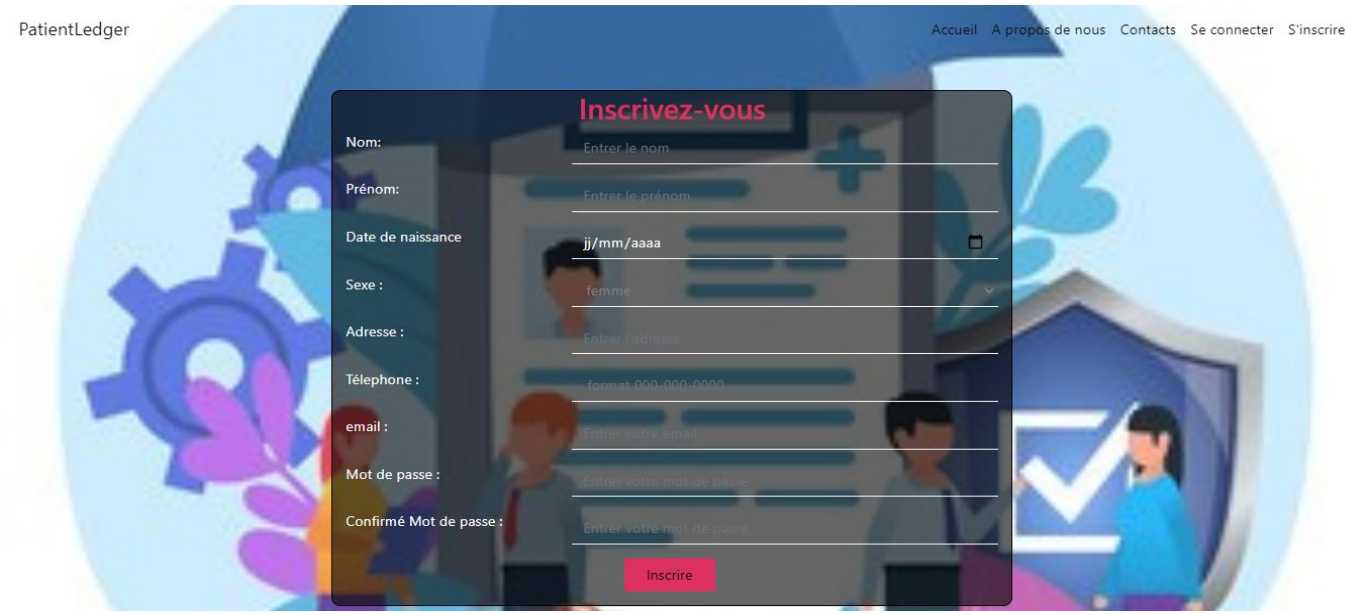

FIGURE IV.17 – Page d'inscription (cas de patient).

3.2.2 Connexion

La connexion à notre application se fait en deux étapes, la connexion au serveur puis à la blockchain. Dans la page de connexion1 (figure IV.18), l'utilisateur introduit son login et mot de passe pour s'authentifier au serveur. Et s'il est correctement authentifié la page connexion2 (figure IV.19) apparait ou il va introduire sa clé privée pour qu'il soit authentifié à la blockchain s'il fait déjà partie de réseau, sinon il trouve un choix à demander de rejoindre le réseau blockchain auprès de médecin traitant s'il est un patient ou auprès d'un patient s'il est un hôpital.

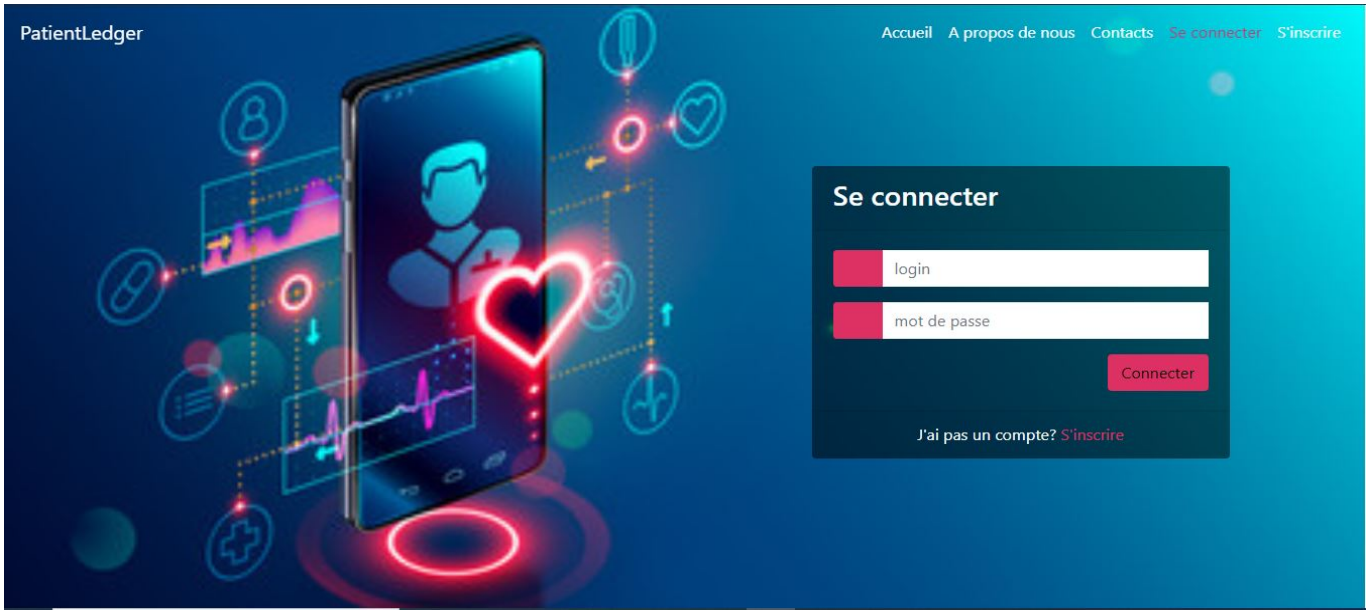

FIGURE IV.18 – Page de connexion1.

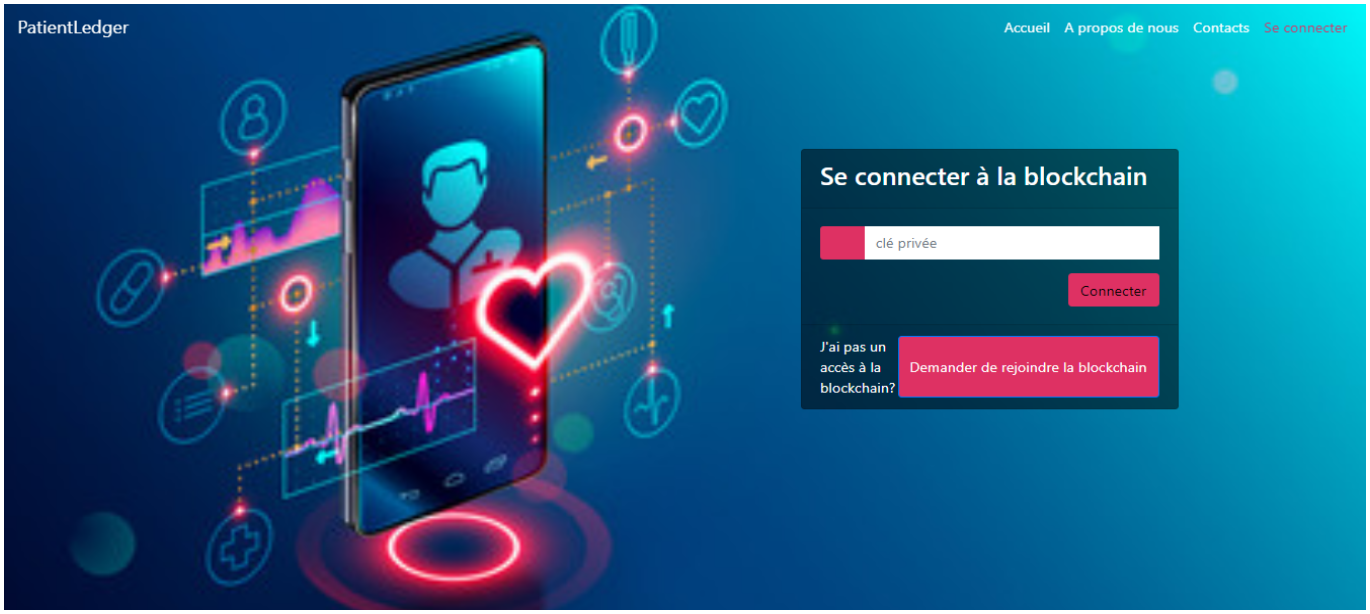

FIGURE IV.19 – Page de connexion2.

3.2.3 Profil médecin traitant

C'est la page qui apparait au médecin traitant bien authentifié, où il peut télécharger ses deux clés (la clé publique et la clé privée) associées à son nœud blockchain, il trouve toutes les demandes de ses patients et il a le choix de les accepter ou les rejeter, il trouve aussi la liste de ses patients à qui il peut ajouter des données médicales à leurs DSE stocké dans la blockchain ou bien de consulter ces derniers (ses données personnelles comme le poids, ses consultations, ses radios et ses ordonnances). Comme il peut exécuter le processus de validation avec un simple clic.

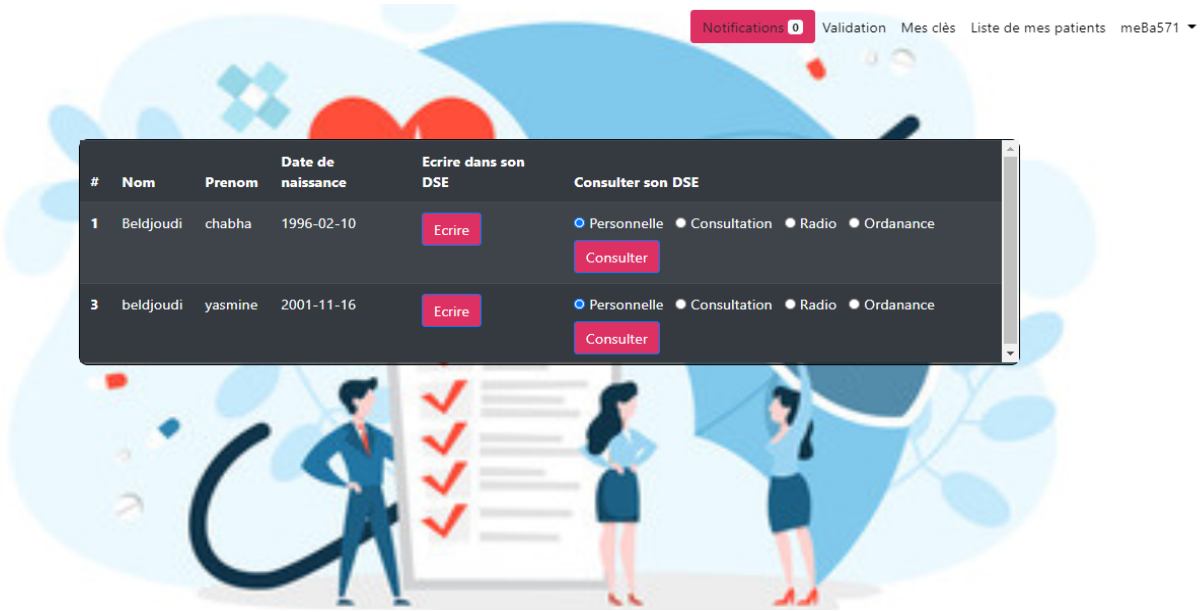

FIGURE IV.20 – Profil médecin traitant (liste de mes patients).

4 Conclusion

Ce chapitre a été consacré pour la phase de conception et la phase de réalisation de notre application. Dans la première phase nous avons décrit d'une façon détaillée la modélisation de notre système en utilisant le langage de modélisation «UML», ainsi que le processus unifié, qui nous a aidés à faire une conception détaillée avec les différents diagrammes. Et dans la deuxième phase, nous avons présenté des outils de développement et les langages utilisés, ainsi que quelques interfaces de notre application.

Conclusion et perspectives

L'objectif de notre travail était de réaliser un système de gestion des dossiers de santé électroniques des patients en utilisant la technologie blockchain. Pour cela, nous avons réalisé un système qui se compose d'un site web pour la gestion de DSE des patients et une blockchain pour le stockage des données médicales des patients.

Notre travail a débuté par les généralités et les concepts de base de la technologie blockchain, dont nous avons cité les points suivants : un bref historique, son principe, ses caractéristiques, les différentes technologies combinées à la blockchain, son fonctionnement, les différentes couches qui la constituent, ces types et quelques cas d'utilisation.

Ensuite, nous avons étudié un ensemble de travaux de recherches sur les systèmes qui utilisent les blockchains pour la gestion des DSE des patients où nous avons extrait les concepts de base de la réalisation d'une blockchain dans le domaine du partage de dossiers de santé électronique.

Puis, nous avons conçu notre système d'une manière générale et définis ses différentes parties. Enfin, nous avons consacré une partie pour la phase de conception et la phase de réalisation de notre application appelée PatientLedger. Dans la première phase nous avons décrit d'une façon détaillée la modélisation de notre système en utilisant le langage de modélisation «UML», ainsi que le processus unifié, qui nous a aidés à faire une conception détaillée avec quelques diagrammes. Et dans la deuxième phase, nous avons présenté la réalisation et la mise en œuvre de notre système.

Dans PatientLedger, l'accès des utilisateurs à leurs profils est géré par une double authentification, la première se fait avec un login et un mot de passe pour être authentifié au serveur du système et la deuxième avec une clé privée pour être authentifié à la blockchain. Nous avons conçu ce système pour qu'il stocke les données personnelles des utilisateurs dans la base de données de serveur et les données médicales des patients sous forme cryptés dans une blockchain que nous avons créé. Dans notre système, le patient est l'acteur qui contrôle l'accès à son DSE soit en lecture soit en écriture pour les parties tierces.

Ce projet a été très bénéfique pour nous car il nous a permis de renforcer et enrichir nos connaissances théoriques dans le domaine de la blockchain et les crypto-monais, ainsi que dans le domaine de la conception. Il nous a encore donné l'occasion d'apprendre quelques principes de la cryptographie et les appliquer dans notre réalisation et aussi de mieux maîtriser le langage de programmation PHP.

Donc, pour la raison de la situation sanitaire que nous vivons actuellement et pour le manque de moyens de communications et de recherche, nous n'avons pas pu réaliser certains points dans notre application. Donc telle qu'elle est actuellement, elle offre un minimum de services et reste toujours perfectible.

Ces contraintes nous ont permis de tracer une ligne de perspectives, nous évoquons :

- Réaliser un réseau pair à pair, pour la distribution de la blockchain entre les nœuds de réseau.
- Remplacer la saisie des clés des membres (clé publique et clé privée) de la blockchain par un code à scanner (un code QR ou en utilisant la carte chifa).
- La validation des blocs par plusieurs acteurs et non pas par un seul (le médecin traitant).
- Traiter le cas de changement de médecin traitant, car dans le système que nous avons réalisé la blockchain est dédié pour chaque médecin traitant, donc il faut que le nouveau médecin traitant aura tous les blocs attachés à ce patient.

Bibliographie

[1] https ://www.binance.vision/fr/blockchain/history-of-blockchain, consulter le 13/01/2020.

[2] http ://www.senat.fr/rap/r17-584/r17-584_mono.html ?fbclid=IwAR2GDH_ns2S_9RNs-FYZ-toS4HeFoCriRII1TyizuxFg12sit4ZnQViGw9E#toc2, consulter le 14/01/2020.

[3] https ://www.intel.com/content/www/us/en/financial-services-it/article/the-benefits-of-blockchain-iot-biot.html, consulter le 19/03/2020.

[4] https ://www.systematic-paris-region.org/fr/les-fondamentaux-de-la-blockchain/, consulter le 01/02/2020.

[5] Blockchain France, La Blockchain décryptée Les clefs d'une révolution, Publié par l'Observatoire Netexplo www.netexplo.org, 264 Rue du Faubourg Saint-Honoré – 75008 Paris, Mai 2016,https ://www.fg2a.com/wp-content/uploads/2017/01/La-blockchain-decryptee-le-s-clefs-dune-revolution.pdf, consulter le 30/01/2020.

[6] https ://www.cryptoast.fr/liste-differents-consensus-crypto-monnaies-blockchain, consulter le 01/02/2020.

[7] https ://www.data-flair.training/blogs/blockchain-tutorial, consulter le 14/03/2020.

[8] https ://www.culture-informatique.net/cest-quoi-hachage/, consulter le 30/01/2020.

[9] https ://www.members.loria.fr/lnussbaum/ptasrall2019/rapport_blockchain.pdf, consulter le 19/03/2020.

[10] https ://www.silicon.fr/brandvoice/les-assurances-et-la-blockchain-quest-ce-que-cela-peut-donner, consulter le 19/03/2020.

[11] https ://www.ionos.fr/digitalguide/web-marketing/vendre-sur-internet/blockchain/, consulter le 19/03/2020.

[12] https ://www.b3i.tech/who-we-are.html, consulter le 19/03/2020.

[13] https ://www.bitconseil.fr/registres-distribues-supply-chain/, consulter le 19/03/2020.

[14] https ://www.bitconseil.fr/blockchain-sante-presentation-cas-usage, consulter le 19/03/2020.

[15] https ://www.aergo.io/developer/tech/blockchain, consulter le 19/03/2020.

[16] https ://www.binance.vision/fr/blockchain/peer-to-peer-networks-explained, consulter le 04/02/2020.

[17] https ://www.binance.vision/fr/blockchain/what-is-a-blockchain-consensus-algorithm, consulter le 05/02/2020.

[18] https ://www.binance.vision/fr/blockchain/blockchain-use-cases, consulter le 19/03/2020.

[19] https ://www.academy.binance.com/fr/blockchain/difference-between-blockchain-and-bitcoin, consulter le 19/03/2020.

[20] A. O. Ayadi, 2019, _Etat_de_l'art_de_la_Blockchain dans : Analyse et étude de la sécurité des données médicales dans l'Internet des objets à partir d'une approche

technologique Blockchain, mémoire de master Professional en Informatique, Réseaux et Systèmes Distribués, université Constantine 2 – Abdelhamid Mehri, chapitre 3, https ://www.researchgate.net/publication/335174496_CHAPITRE_III_Etat_de _l'art_de_la_Blockchain, consulter le 26/03/2020.

[21] https ://www.binance.vision/fr/blockchain/what-is-cryptocurrency-mining, consulter le 06/02/2020.

[22] https ://www.bitcoin.com/, consulter le 14/03/2020.

[23] https ://www.ethereum.org/fr/ , consulter le 14/03/2020.

[24] https ://www.hyperledger.org/, consulter le 14/03/2020.

[25] https ://www.passcare.com/, consulter le 14/03/2020.

[26] https ://www.b3i.tech/home.html, consulter le 14/03/2020.

[27] https ://www.aergo.io/, consulter le 14/03/2020.

[28] https ://www.strategie.gouv.fr/sites/strategie.gouv.fr/files/atoms/files/fs-rapport-blockchain-21-juin-2018.pdf, consulter le 26/03/2020.

[29] https ://www.coinrivet.com/fr/what-is-the-ethereum-casper-protocol/, consulter le 05/02/2020.

[30] G. G. Daghera, J. Mohlerb, M. Milojkovicc, P. B. Marella, Ancile : Privacy-preserving framework for access control and interoperabilityof electronic health records using blockchain technology, Sustainable Cities and Society, Vol. 39, pp. 283–297, 2018.

[31] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, et M. Guizani, MeDShare : Trust-Less Medical Data SharingAmong Cloud Service Providers via Blockchain, IEEE Access, vol. 5, pp. 14757-14767, 2017.

[32] S.Tanwar, K.Parekh , R. Evans Blockchain-based electronic healthcare record system for healthcare 4.0 applications, Journal of Information Security and Applications, vol.50 Art, N 102407, 2020.

[33] A.Ekblaw, A. Azaria, J. D. Halamka, A.Lippman, A Case Study for Blockchain in Healthcare :"MedRec" prototype for electronic health records and medical research data, White paper,2016, https ://www.healthit.gov/sites/default/files/5-56-onc_blockchainchallenge_mitwhitepaper.pdf, consulter le 20/05/2020.

[34] E.-Y. Daraghmi , Y.-A. daraghmi, et S.-M. Yuan, MedChain : A Design of Blockchain-BasedSystem for Medical Records Access and Permissions Management, IEEE Access, vol.7 pp. 164595-164613, 2019.

[35] L. Chen, W.-K.Lee, C.-C.Chang , K.-K. R. Choo ,N. Zhang, Blockchain based searchable encryption for electronic health recordsharing, Future Generation Computer Systems,vol. 95 pp. 420–429, 2019.

[36] D. C. NGUYEN, P. N. PATHIRANA,M. DING, et A. SENEVIRATNE, Blockchain for Secure EHRs Sharing of MobileCloud Based E-Health Systems, IEEE Access, vol.7, pp. 66792-66806, 2019.

[37] A. Albeyatti, Medicalchain, White paper 2.1, 2018, https ://www.medicalchain.com/Medicalchain-Whitepaper-EN.pdf, consulter le 20/05/2020.

[38] P. ROQUES et F.VALLÉE. UML en action.

[39] P. ROQUES et F.VALLÉE. Les cahiers du programmeur UML2 modélisé une application web.

[40] L. Audibert. UML2 De la apprentissage à la pratique.

[41] B. Claude, Le langage UML 2.0 Diagramme de Contexte, Université de Nantes.

[42] O. Sigaud. Introduction à la modélisation orientée objets.

[43] J. GABY et D. GABY. UML2 Analyse et conception.

[44] https ://www.web.maths.unsw.edu.au/∼lafaye/CCM/relation/relintro.htm ?, consulter le 11/08/2020.

[45] S. ROHAUT, cour PHP.

[46] https ://www.developer.mozilla.org/fr/docs/Learn/JavaScript/First_steps/What_is_JavaScript/, consulter le 11/08/2020.

[47] http ://www.glossaire.infowebmaster.fr/html/, consulter le 11/08/2020.

[48] https ://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203277-css-cascading-style-sheets-definition-traduction/, consulter le 11/08/2020.

[49] https ://www.json.org/json-fr.html, consulter le 11/08/2020.

[50] https ://www.getbootstrap.com/, consulter le 11/08/2020.

[51] https ://www.fil.univ-lille1.fr/∼wegrzyno/portail/PAC/Doc/TP-Certificats/tp-certif001.html/, consulter le 11/08/2020.

[52] https ://www.code.visualstudio.com/docs, consulter le 11/08/2020.

[53] https ://www.wampserver.com/2011/11/11/presentation-wampserver/, consulter le 11/08/2020.

[54] https ://www.sql.sh/sgbd/mysql, consulter le 11/08/2020.

[55] https ://www.docs.phpmyadmin.net/fr/latest/intro.html, consulter le 11/08/2020.

[56] https ://www.httpd.apache.org/, consulter le 11/08/2020.

Résumé

Ce travail a pour but de réaliser un système basé sur la technologie de la blockchain pour la gestion des DSEs qui pourrait donner le contrôle final des DSEs aux patients et garantir la vie privée de ces derniers et sécuriser leurs informations médicales. Notre système est muni des fonctionnalités nécessaires et adéquates aux besoins des patients.

Avant de mettre en œuvre notre solution nous avons fait une étude sur des systèmes existants, et pour le concevoir nous avons utilisé un processus de développement appelé Processus Unifié (UP), qui se base sur UML comme langage de modélisation conçut pour fournir une méthode normalisée pour la conception.

La réalisation de l'application s'est faite sous l'éditeur Visuel Studio Code en utilisant les langages de programmation PHP et JavaScript et le Framework Bootstrap.

Mots-clés : Dossier de Santé Electronique, Blockchain, PHP, JavaScript, CSS, Bootstrap, OpenSSL, JSON.

Abstract

This work aims to achieve a system based on blockchain technology for the management of EHRs that could give final control of EHRs to patients and guarantee their privacy and secure their medical information. Our system has the necessary and adequate features to the patient needs.

Before implementing our solution we did a study of existing systems, and to design it, we used a developmental process called Unified Process (UP), which is based on UML as a modeling language designed to provide a standardized method for design.

The application was made in Visual Studio Code editor using the programming languages PHP, JavaScript and the Bootstrap Framework.

Keywords : Electronic Health Record, Blockchain, UML, UP, PHP, JavaScript, CSS, Bootstrap, OpenSSL, JSON.