

Projet de Recherche L3

Implementation of New Generation Blockchain's Proof of Concept

Alan Pulval-Dady
Yulin Shi
Zhe Wang

Encadrant :
Nour El Madhoun

Projet en collaboration avec
Daniel Maldonano-Ruiz
de l'EPN (École polytechnique nationale de Quito)



15 Mars 2022

Contents

1	Contexte	3
2	Objectif du projet	4
3	Besoins et contraintes	5
4	Préparation	6
5	Activités Réalisées	7
5.1	Recherche et Compréhension Générale	7
5.2	Implementation d'une Blockchain Traditionnelle en JAVA	10
5.2.1	Implémentation d'un Réseau	10
5.2.2	Implémentation des Noeuds de Bases	11
5.2.3	Implémentation Spécifique au PoW	12
5.2.4	Implémentation Spécifique au PoS	13
5.3	Nouvelle Proposition	14
6	Reste du Travail à Effectué	16
7	Conclusion	17

1 Contexte

La blockchain est un élément important de la nouvelle génération de technologies de l'information, un nouveau type de logiciel de base de données intégré aux réseaux distribués, à la cryptographie, aux contrats intelligents et à d'autres technologies. Grâce à la transparence des données, à la réduction des altérations et à la traçabilité, elle devrait résoudre les problèmes de confiance et de sécurité dans le cyberspace, promouvoir le changement de l'Internet, qui passe de la transmission d'informations à la transmission de valeurs, et reconstruire le système de l'industrie de l'information [1] .

Notre projet de recherche fait partie de la recherche Identités décentralisées basées sur les réseaux autonomes de nouvelle génération [7], qui vise à créer un réseau autonome dans lequel les utilisateurs peuvent stocker des informations sans compter sur des parties externes pour valider les informations stockées. Ce prototype est à la base de son intention de créer cette nouvelle génération de réseaux décentralisés, puisqu'il utilise les mêmes ressources informatiques que les entités de stockage et de validation.

2 Objectif du projet

Actuellement pour chaque type de blockchain tel que le Bitcoin ou encore Ethereum, il faut établir une infrastructure pour stocker les différentes informations. Cela a coup plus au moins élevé de calcul [8] pour tout d'abord exécuté les différents algorithmes de consensus tel que le proof-of-work [9], pour exemple la ville d'Ekibastouz au Kazakhstan recueille 50 000 machines de minage alimentées par une centrale à charbon, la consommation d'une telle ferme représenterait l'équivalent d'une consommation d'une ville d'environ 100 000 habitants, de plus il faut le matériels nécessaires au stockages de tout les blocs ayant pue être miné auparavant, le nombre de blocs pour le Bitcoin s'élève par exemple aujourd'hui à 726 882 blocs sachant qu'un bloc est miné toutes les 10 minutes.

Ce projet vise donc à réduire les coûts liés à la décentralisation des différents types de blockchain, en développant une nouvelle approche de la blockchain qui pourrait en stocker plusieurs non liées au sein de la même infrastructure. Ainsi, une seule infrastructure de calcul sera nécessaire pour de multiples blockchains réduisant immédiatement la consommation liée aux différentes fermes de minage créé pour chaque infrastructure, même si le nombre de blocs à stocker sera factuellement plus grand.

Dans cette vision, nous devons garantir différents types d'informations au sein d'un même blockchain, c'est-à-dire sans utiliser des ensembles différents pour les mettre en œuvre , et créer des algorithmes permettant de chercher les différentes informations dans la blockchain liées aux différentes infrastructures concaténées tout cela avec des temps d'exécution raisonnable.

3 Besoins et contraintes

- Nous devons d'abord réaliser une blockchain traditionnelle.
- Étudier les mécanismes de consensus proof-of-work(PoW) et proof-of-stake(PoS).
- Mettre en œuvre la conception du prototype de blockchain de prochaine génération, en modifiant l'en-tête du bloc pour permettre à deux blockchains différentes d'être stockées dans la même blockchain.
- Vérifier à quelle blockchain appartient un bloc avant de stocker un nouveau bloc.
- Comparer la performance du nouveau prototype de blockchain avec celle d'une blockchain traditionnelle.
- Il faut avoir des connaissances en cryptographie et en réseaux décentralisés.
- Les exigences de performance informatique élevées lors de la vérification du mécanisme de consensus PoW.

4 Préparation

Afin de mieux comprendre la structure et les principes des blockchains, nous avons étudié la cryptographie requise et recherché un grand nombre de références pertinentes. En particulier, nous avons approfondi les deux mécanismes de consensus des blockchains et compris leurs différences.

Parce que la blockchain est fondamentalement une base de données partagée, nous avons également appris la programmation concurrente afin de mettre en œuvre le PoW.

Les préparatifs plus détaillés que nous avons effectués sont présentés ci-dessous:

- Cryptographie
 - Protocole RSA
 - Fonctions de hachages
 - Signature de données
 - Chiffrement asymétrique
- Algorithmes de consensus de blockchain
 - Proof of Work
 - Proof of Stake
- Programmation concurrente

5 Activités Réalisées

5.1 Recherche et Compréhension Générale

En premier lieu, nous avons eu un cours d'introduction à la blockchain et à la cryptographie, ces cours nous ont permis d'acquérir les premiers outils nécessaires à la compréhension de notre projet.

Nous avons compris qu'une blockchain est un réseau dans lequel des utilisateurs échangent de la monnaie virtuelle. Ce réseau contient différents nœuds, les Lights Node qui gèrent les portefeuilles des différents utilisateurs et les Full Node qui stockent les différents échanges qui ont lieu au sein du réseau. Ces échanges sont gérés par des chaînes de blocs qui vont enregistrer toutes leurs informations. Les utilisateurs ayant pour fonction Full Node vont enregistrer toutes les transactions qui ont eu lieu depuis la création du réseau, ces transactions sont stockées dans les blocs de la chaîne de bloc. Un bloc est composé d'un en-tête, d'un corps et d'un pied. Les en-têtes contiennent les informations liées au bloc précédent, c'est-à-dire le hash des transactions et de l'en-tête du bloc précédent, et les pieds contiennent le hash des transactions du bloc courant.

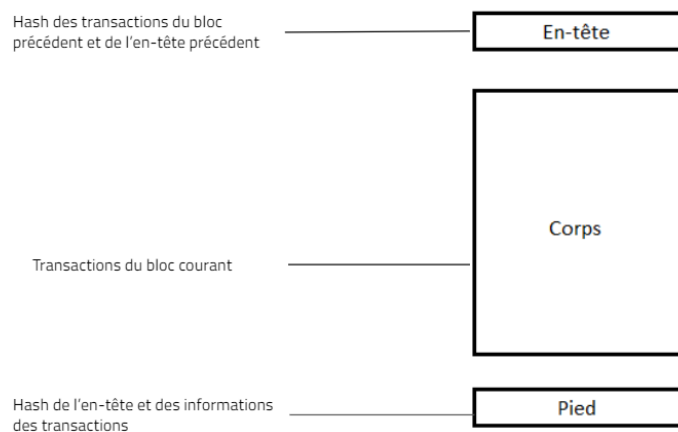


Figure 1: Schéma d'un bloc

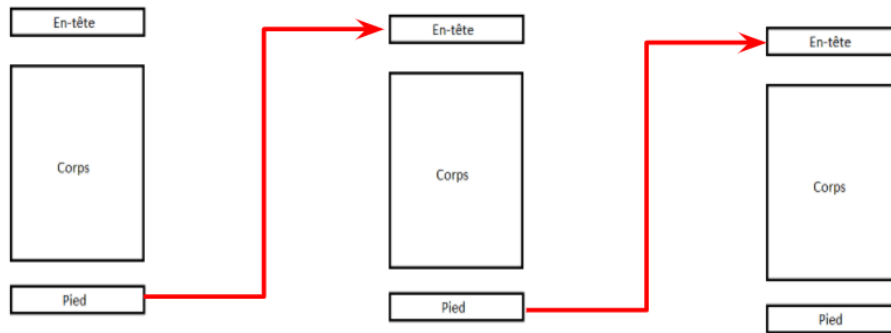


Figure 2: Schéma d'une chaîne de bloc

Via un algorithme de consensus les transactions sont vérifiées et validées par le réseau. Par exemple en appliquant le consensus proof-of-work [10], on considère une nouvelle catégorie de nœud qui est le mineur. Un mineur est un Full Node ayant pour fonction la création de bloc. Tous les mineurs minent de manière concurrente et le premier trouvant un bloc est récompensé par de la monnaie virtuelle. Lorsqu'un bloc est créé celui-ci est broadcasté à tout le réseau et est ajouté aux différentes chaînes de blocs.

Lorsque des blocs sont créés en même temps et broadcasté en même temps, les Full Node et Light Node crée des chaînes temporaires avec les différents blocs, lorsqu'une chaîne temporaire est assez grande alors, elle est validée et ajoutée à la suite de la chaîne de bloc de base.

Les échanges sont sécurisés par des outils cryptographiques permettant de garantir l'authenticité de chaque transaction. En effet, chaque utilisateur a pour s'authentifier une clef publique et une clef secrète, les utilisateurs lors d'un envoi d'une transaction aux réseaux signe celle-ci avec sa clef privée qui n'est pas révélé. Lors de la réception, les Full Node peuvent vérifier que la personne ayant créé la transaction est bien la bonne. Toujours dans l'exemple de l'utilisation du consensus proof-of-work, les mineurs doivent signer les blocs envoyés cela permet ainsi de vérifier l'authenticité des mineurs, si une transaction ou un bloc n'a pas une signature valide alors, elle est refusée.

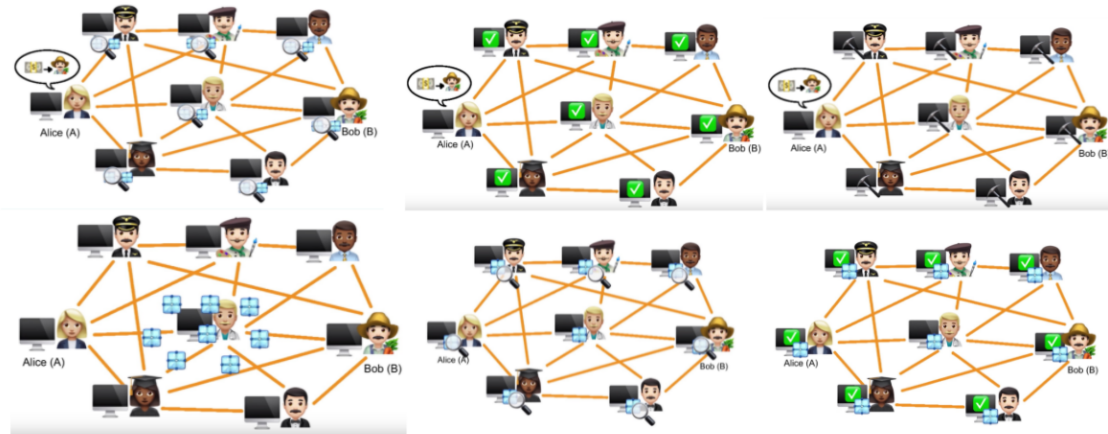


Figure 3: Schéma récapitulatif du fonctionnement d'une blockchain

Dans la figure ci-dessus, un utilisateur désire envoyer de l'argent à un autre utilisateur du réseau, sa requête est donc vérifiée par tous les utilisateurs. Si la transaction est bien signée et que l'utilisateur a assez de monnaie virtuelle alors les mineurs peuvent commencer à miner. Un des mineurs a été le plus rapide et va donc envoyé à tous le réseau le bloc avec la transaction, ce bloc est vérifié par tous le réseau puis s'il est validé, est ajouté à la chaîne de bloc.

Il existe un autre type de consensus que nous avons étudié, le proof-of-stake (PoS) [11]. Le proof-of-stake, ou "preuve d'enjeu", ou encore "*preuve de participation*" en français, est un système de validation des transactions effectuées au sein d'une blockchain.

Dans le cadre de la proof-of-stake, les utilisateurs de la blockchain qui souhaitent participer à la création de nouveaux blocs, et donc recevoir des récompenses, doivent posséder et "*mettre en jeu*" un certain nombre de jetons dans une cryptomonnaie utilisée sur le réseau. Le montant de cette "preuve d'enjeu" varie entre les blockchains et change au fur et à mesure de leur développement. Le système repose sur l'idée que plus un nœud d'une blockchain possède de jetons, plus il a intérêt à maintenir la sécurité du réseau auquel il participe. Par conséquent, plus un forgeur possède de jetons, plus il a de chances d'être sélectionné dans le cadre d'un système de proof-of-stake.

Le nœud qui a le droit de créer des blocs est appelé un validateur ou forgeur dans PoS. Un validateur ne peut pas retirer ses jetons lorsqu'il "*forge*" un bloc, cela garantit la sécurité de cet algorithme de consensus. Si des signes de triche sont constatés, ce dernier est dénoncé, perd les jetons qu'il a mis en jeu (ils sont généralement détruits ou transférés au dénonciateur) et son bloc est supprimé de la chaîne.

La probabilité que chaque vérificateur puisse obtenir le droit de vérifier et de créer un bloc varie en fonction de ses jetons détenue et le temps écoulé depuis que les jetons sont engagés. Plus un nœud investit de jetons, et puis plus la durée d'engagement des jetons est longue, plus la probabilité qu'il devienne un vérificateur est grande.

Le validateur doit valider toutes les transactions qu'il regroupe en blocs et diffuser le

bloc résultant sur le réseau, il recevra sa récompense lorsque les validateurs auront déterminé que le bloc est valide.

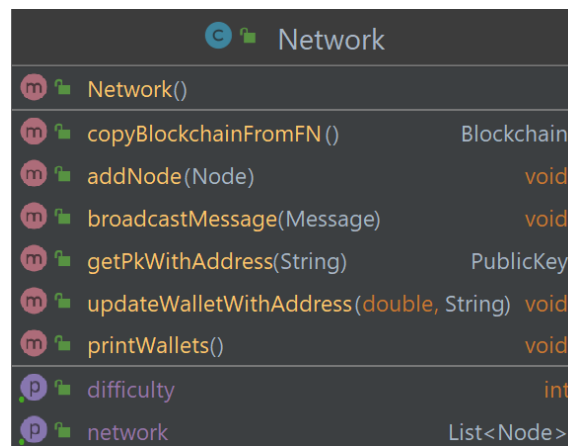
5.2 Implementation d'une Blockchain Traditionnelle en JAVA

Dans cette deuxième partie nous devons implémenter une blockchain traditionnelle en Java de deux manières, une avec le consensus proof-of-work et l'autre avec le consensus proof-of-stake.

Ces implémentations, nous permettront par la suite d'étudier les écarts avec une blockchain contenant plusieurs blockchains.

5.2.1 Implémentation d'un Réseau

Pour implémenter la blockchain, il nous a fallu tout d'abord définir un réseau.



Network	
Network()	
copyBlockchainFromFN()	Blockchain
addNode(Node)	void
broadcastMessage(Message)	void
getPkWithAddress(String)	PublicKey
updateWalletWithAddress(double, String)	void
printWallets()	void
difficulty	int
network	List<Node>

Figure 4: Implémentation d'un Réseau

Un réseau est une liste de Nœud représentant des utilisateurs de la blockchain. Ces utilisateurs peuvent broadcaster des messages via la fonction *broadCastMessage*. Une table assignant pour chaque utilisateur (identifié par le hash de leurs clé publique) leur clé publique, est aussi présente dans le réseau, cette table est utilisée lorsqu'un utilisateur veut envoyer de la monnaie virtuelle a un autre utilisateur du réseau. La fonction *updateWalletWithAddress* permet de mettre à jour les portefeuilles des utilisateurs si un bloc est considéré valide.

5.2.2 Implémentation des Nœuds de Bases

Nous avons par la suite défini des nœuds élémentaires pour la blockchain.

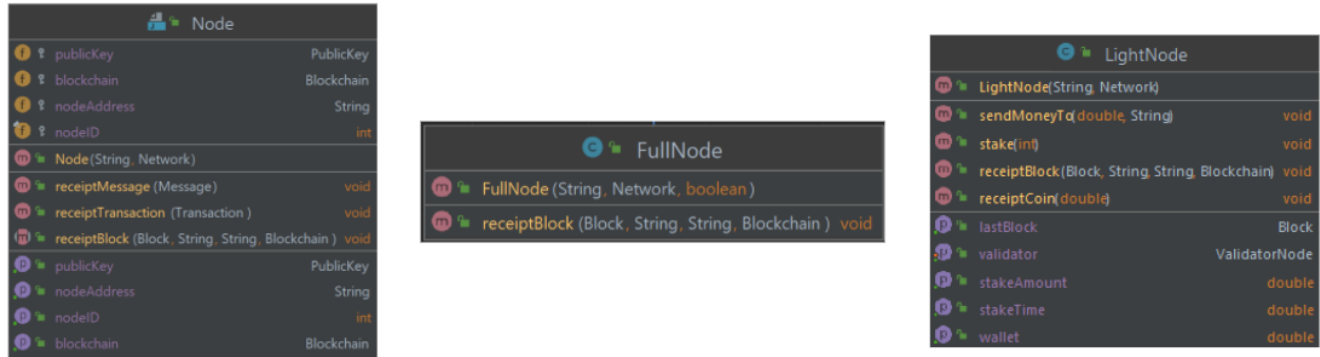


Figure 5: Implémentation d'un Réseau

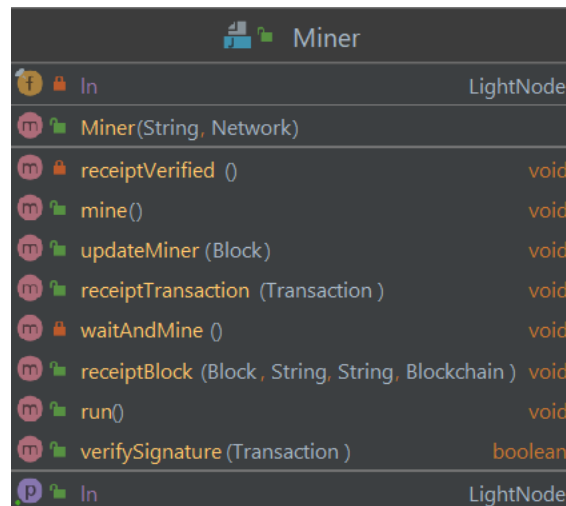
La classe Node sert de base aux différents types de Nœud possible, chaque Node possède une clef publique et privée, chaque nœud appartient à un réseau et chaque nœud a un identifiant. Les nœuds peuvent tous recevoir les messages, mais selon leurs classes, ils ont des comportements différents. Dans la suite, tous les nœuds héritent de la fonction Node.

La classe Full Node va pouvoir recevoir des blocks, les Full Node sont comme des serveurs, ils stockent tous les blocs depuis la création du réseau et vérifient la signature des blocs.

Les Light Node sont les portefeuilles des utilisateurs, ainsi l'attribut wallet représente l'argent virtuel possédé par chacun d'entre eux. Ils peuvent envoyer de l'argent via la fonction *sendMoneyTo* et en recevoir via *receiptCoin*. Les autres attributs (Stake, StakeTime) sont liés au consensus proof-of-stake que nous allons voir un peu plus bas.

5.2.3 Implémentation Spécifique au PoW

Comme vue précédemment le proof-of-work fonctionne avec des Mineurs qui sont censés "*miner*" les différents blocs de manière concurrente. L'opération de minage consiste à trouver une valeur de Hash plus petite que celle du bloc précédent dans la chaîne de bloc. Nous avons donc implémenté une classe Mineur.



Miner	
Ln	LightNode
Miner(String, Network)	
receiptVerified ()	void
mine()	void
updateMiner (Block)	void
receiptTransaction (Transaction)	void
waitAndMine ()	void
receiptBlock (Block , String, String, Blockchain)	void
run()	void
verifySignature (Transaction)	boolean
Ln	LightNode

Figure 6: Implémentation de Mineur

Un mineur peut miner un bloc avec la fonction `mine`, `updateMiner` indique qu'un bloc a été trouvé et qu'il faut par conséquent arrêter l'opération de minage, celui-ci est vérifié puis ajouté à la chaîne de bloc. Chaque Miner peut recevoir une transaction, les Miner stockent les transactions et toutes les 10 minutes commencent à miner un bloc via la fonction `mine`, la concurrence se fait via la fonction `run`. Chaque Miner à un Light Node pour stocker l'argent virtuelle gagné grâce à la création de bloc.

5.2.4 Implémentation Spécifique au PoS

Comme vue précédemment le proof-of-stake fonctionne à l'aide de Validateur qui sont sélectionnés selon leur monnaie virtuelle investie, c'est-à-dire la somme des investissements sur chaque Validateur.

ValidatorNode		
m	ValidatorNode (String, Network, LightNode)	
m	updateTransactionList (Block)	void
m	dellInvestor (String, double)	void
m	receiptBlock (Block, String, String, Blockchain)	void
m	addInvestor (String, double)	void
m	receiptTransaction (Transaction)	void
m	verifyTransaction (Transaction)	boolean
m	forgeBlock ()	void
m	getAndBroadcastReward (double)	void
m	addStake (double)	void
p	stakeAmount	double
p	investorList	Set<String>
p	stakeTime	long

Figure 7: Implémentation de ValidatorNode

Nous avons donc tout d'abord besoin d'une classe Validator Node. La classe Validator Node a donc une liste d'investisseurs ce qui nous permet d'avoir le Stake (la monnaie virtuelle total délégué au consensus) des Validator, une fonction *forgeBlock* nous permet de forger un block lorsque le Validateur est élu.

Nous avons aussi une deuxième classe qui va s'occuper de l'élection des Validateurs, cette classe devait pouvoir élire régulièrement un Validateur pour forger les blocs.

Validator		
m	Validator (Network)	
m	chooseValidator ()	void
m	validate ()	void
m	run ()	void

Figure 8: Implémentation de Validator

Cette classe se lance en arrière-plan pour choisir un validateur a un intervalle de temps régulier via la fonction *chooseValidator* et va indiquer au Validator Node élue de forger un bloc via la fonction *validate*.

5.3 Nouvelle Proposition

Afin d'atteindre l'objectif de créer et de maintenir un réseau décentralisé qui peut être utilisé comme une structure autonome où différents types de données peuvent être stockés, il est nécessaire de créer l'infrastructure logique sur laquelle ces différents types d'informations doivent être stockés. Nous proposons donc "Decentralised Identities based on Next Generation Autonomous Networks" (Identités décentralisées basées sur les réseaux autonomes de nouvelle génération)

Dans la figure 9, il faut voir la structure de la proposition, dans laquelle les transactions (en marron) sont incluses dans le hachage de l'en-tête du bloc de la blockchain (représenté par le champ de hachage 1, également en marron) comme dans la blockchain traditionnelle. Chaque blockchain traditionnelle possède également un autre champ de hachage, le hachage de l'en-tête du bloc précédent, pour réaliser la chaîne de blocs (représentée par le champ de hachage 3, en orange). Cette proposition ajoute un troisième champ de hachage (le champ de hachage 2, en vert et bleu) qui différencie les deux structures qui sont dans la nouvelle blockchain. La blockchain en bleu et la blockchain en vert sont deux blockchains distinctes qui fonctionnent traditionnellement comme deux structures différentes dans deux réseaux p2p différents. Avec la nouvelle configuration, ces deux blockchains existent dans le même réseau p2p mais restent indépendantes l'une de l'autre. Pour conserver la blockchain-inside-blockchain, tous les nœuds liés à la blockchain 1 (la bleue) lient leurs hashes d'en-tête les uns aux autres, de sorte que la blockchain 1 est uniquement liée à leurs blocs. Il en va de même pour la blockchain 2 (la verte). Mais aussi, pour maintenir un seul réseau, le champ de hachage 3 relie également le bloc à celui qui le précède immédiatement, quelle que soit la blockchain à laquelle il appartient. Ces relations sont représentées par des flèches bleues/vertes vers les blocs des blockchains 1 ou 2, et des flèches oranges pour relier les blocs aux blocs précédents.

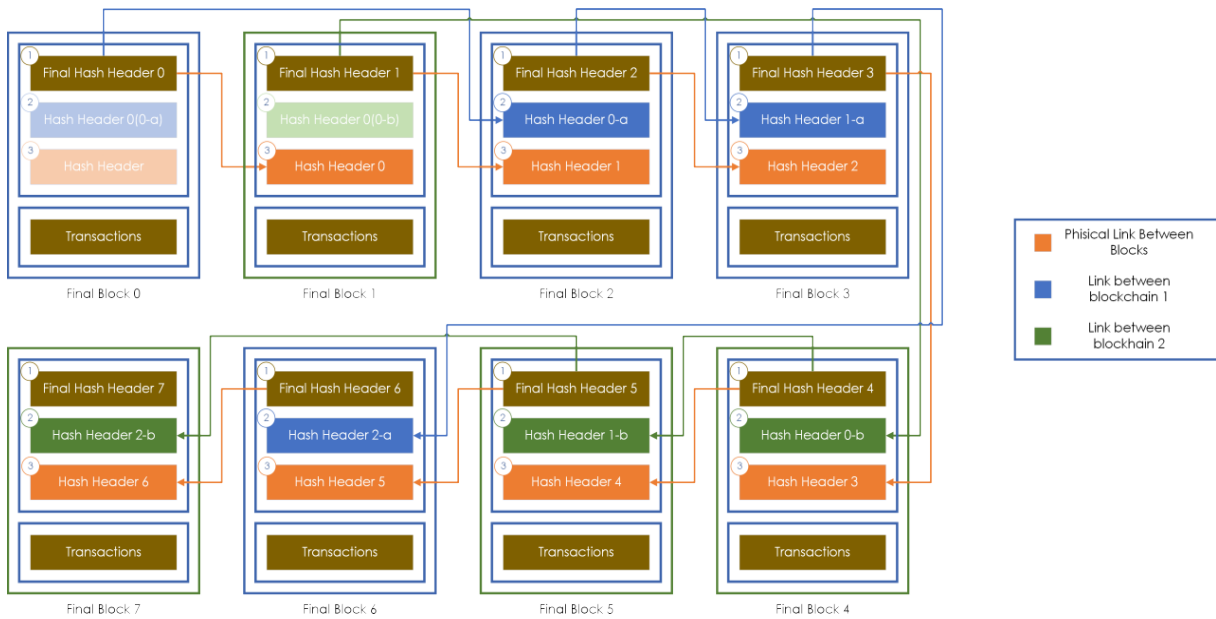


Figure 9: Schematics of new generation blockchain prototype

Figure 10 montre comment le protocole de consensus doit être modifié afin de lier les nouveaux blocs à leur blockchain correspondante ainsi qu'au bloc précédent dans la chaîne principale. Tout d'abord, les champs relatifs à la transaction du nouveau bloc sont remplis, y compris l'ID de la blockchain à laquelle le nouveau bloc appartient (1). Lorsque tous les champs sont remplis, le consensus demande l'ID de la blockchain, et le compare à l'ID du bloc précédent. Si la comparaison est positive, les valeurs du champ de hachage 1 du bloc précédent sont copiées dans les champs de hachage 2 et 3 du nouveau bloc (2) et le consensus peut continuer (6). Si la comparaison est négative, le consensus doit effectuer une recherche récursive des blocs précédents pour trouver la coïncidence de l'ID de la blockchain (3), lorsque la recherche obtient un résultat, le champ de hachage 1 du bloc trouvé est copié dans le champ de hachage 2 du nouveau bloc (4) et le champ de hachage 1 du bloc immédiatement précédent est copié dans le champ de hachage 3 du nouveau bloc (5). Après cela, le consensus peut continuer (6).

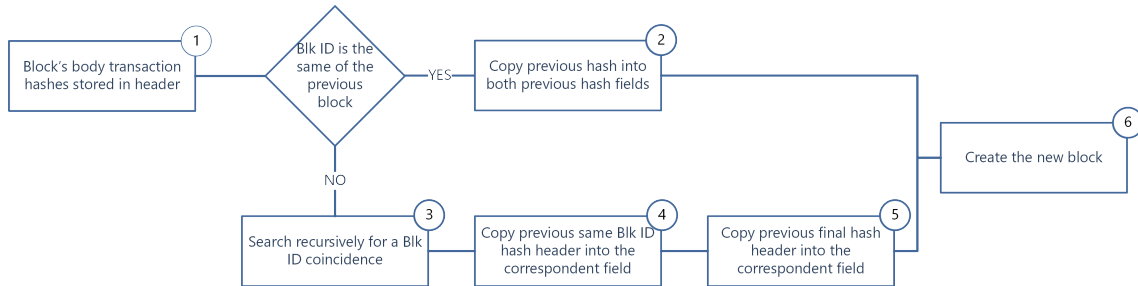


Figure 10: Creation of the new block flowchart

6 Reste du Travail à Effectué

- Recherche sur les différents types d'implémentations possible pour le nouveau type de blockchain
- Implémentation des différents prototypes
- Tests et calculs des différents écarts observé comparé à une blockchain traditionnelle
- Préparation en vue de l'oral des slides
- Entraînement a l'oral

7 Conclusion

Jusqu'à présent nous avons réussi à implémenter une blockchain traditionnelle en Java ainsi que les différents types de consensus. Lors de l'étude des différents algorithmes de consensus, nous avons rencontré des difficultés liées à leurs compréhensions et à leurs mise en place, en les résolvants nous avons pu acquérir une meilleure compréhension de la blockchain traditionnelle, ce qui va nous fournir une base solide pour implémenter un prototype de blockchain de nouvelle génération. Ce développement fait partie d'une recherche doctorale intitulée "Decentralised Identities based on Next Generation Autonomous Networks" (Identités décentralisées basées sur les réseaux autonomes de nouvelle génération) dans laquelle il est prévu de créer un blockchain où nous pouvons stocker plusieurs types de transactions de cryptomonnaie.

Pour nous, qui sommes novices en matière de projets de recherche, ce projet a exercé notre autonomie, notre capacité à trouver de la documentation et à résoudre des problèmes par nous-mêmes, et nous a permis de mieux comprendre la blockchain, une technologie de plus en plus utilisée.

References

- [1] M. Ahmed, I. Elahi, M. Abrar, U. Aslam, I. Khalid, and M. A. Habib. Understanding Blockchain. In Proceedings of the 3rd International Conference on Future Networks and Distributed Systems - ICFNDS '19, pages 1–8, New York, New York, USA, 2019. ACM Press.
- [2] D. Di Francesco Maesa and P. Mori. Blockchain 3.0 applications survey. *Journal of Parallel and Distributed Computing*, 138:99–114, apr 2020.
- [3] T. Hepp, F. Spaeh, A. Schoenhals, P. Ehret, and B. Gipp. Exploring Potentials and Challenges of Blockchain-based Public Key Infrastructures. In IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pages 847–852, Paris, France, France, apr 2019. IEEE.
- [4] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.
- [5] Shashank Motepalli, Hans-Arno. Reward Mechanism for Blockchains Using Evolutionary Game Theory
- [6] França, A. S. L., Neto, J. A., Gonçalves, R. F., Almeida, C. M. V. B. (2020). Proposing the use of blockchain to improve the solid waste management in small municipalities. *Journal of Cleaner Production*, 244, 118529.
- [7] Faqir, Y., Arroyo, J., Hassan, S. (2020, August). An overview of decentralized autonomous organizations on the blockchain. In Proceedings of the 16th international symposium on open collaboration (pp. 1-8).
- [8] Sedlmeir, J., Buhl, H. U., Fridgen, G., Keller, R. (2020). The energy consumption of blockchain technology: beyond myth. *Business Information Systems Engineering*, 62(6), 599-608.
- [9] Bach, L. M., Mihaljevic, B., Zagar, M. (2018, May). Comparative analysis of blockchain consensus algorithms. In 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) (pp. 1545-1550). Ieee.
- [10] Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H., Capkun, S. (2016, October). On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security (pp. 3-16).
- [11] King, S., Nadal, S. (2012). Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. self-published paper, August, 19(1).