

# BlocNote

Modèle de chaîne de blocs pour les élèves de la maturité  
gymnasiale.

PROJET INDIVIDUEL DANS LE CADRE DE LA FORMATION GYMINF

MARIE DI MARCO

Juin 2023

**Supervisé par :**

Dr. Pierre-Alain CHÉRIX  
et Prof. Dr. Didier BUCHS  
Université de Genève

# Remerciements

Je tiens d'abord à remercier le superviseur de ce travail, M. Pierre-Alain Chérix et Prof. Buchs pour leurs remarques pertinentes et constructives. Je remercie la direction du Collège Rousseau pour les aménagements qui m'ont permis de mener ce projet à bien, et l'opportunité de le mettre en pratique lors du projet d'informatique de troisième année, et à mes collègues qui ont bien voulu servir de cobayes pour tester l'activité. Merci à Florian Dubath, les auteurs de BlockSim [11] et les auteurs de Bloxxgame [27] pour avoir pris le temps de m'expliquer leurs versions de la chaîne de blocs et avoir répondu à mes questions. Merci à tous ceux qui mettent gratuitement à disposition des ressources en logiciel libre sur internet. Je remercie mes enfants qui ne cessent de me rappeler qu'il y a une vie en dehors de l'école.

# Abstract

This work summarises common knowledge on blockchains like Bitcoin and Ethereum in order to build the BlocNote model, a blockchain aimed at teaching K-12 level students the basics of blockchains through an instructional game. The teaching activities are mostly unplugged, but are also supported by Python programs allowing to simulate a BlocNote blockchain, verify the students inputs during the activities and visualise the blockchain.

**Keywords :** blockchain, register, PoW, PoS, smart contract, NFT, instructional game, college students, unplugged activity, Python simulation program, Python visualization program, Python interactive program for checking blocks validity

Dans ce travail, les principaux éléments des chaînes de blocs telles que Bitcoin et Ethereum sont résumées afin de construire le modèle BlocNote, une chaîne de bloc visant à enseigner les bases des chaînes de blocs au niveau de la maturité gymnasiale. Les activités-élèves consistent en un jeu pédagogique déconnecté, qui est aussi soutenu par des programmes Python permettant de simuler une chaîne de blocs BlocNote, vérifier les blocs créés par les élèves lors des activités et visualiser les chaînes de blocs.

**Keywords :** chaîne de blocs, registre, preuve de travail, preuve d'enjeu, contrat intelligent, jeton non-fongible, jeu pédagogique, élèves de la maturité gymnasiale, activité déconnectée, programme de simulation en Python, programme interactif de vérification des blocs en Python, programme de visualisation de la chaîne de blocs en Python

# Table des matières

<b>1. Chaînes de blocs</b>	<b>3</b>
1.1. Chaîne de blocs . . . . .	4
1.1.1. Définitions . . . . .	4
1.1.2. Définition pour les élèves . . . . .	6
1.1.3. Représentation . . . . .	6
1.2. Fonctionnement . . . . .	9
1.2.1. Exemple d'achat avec cryptomonnaie . . . . .	11
1.2.2. Réseau . . . . .	11
1.2.3. Bloc . . . . .	13
1.2.4. Noeud . . . . .	15
1.2.5. Jeton (token) . . . . .	17
1.2.6. Adresse, clés privée et publique, et portefeuille électronique (wallet)	18
1.3. Consensus . . . . .	19
1.3.1. Preuve de travail (proof of work, PoW) . . . . .	20
1.3.2. Preuve d'enjeu (proof of stake, PoS) . . . . .	22
1.3.3. Inattaquable (?) . . . . .	23
1.3.4. Immobile, inaltérable . . . . .	24
1.4. Transactions . . . . .	27
1.4.1. Mempool . . . . .	28
1.4.2. Confirmation, finalisation et exécution . . . . .	28
1.4.3. Registre . . . . .	30
1.5. Contrat intelligent (smart contract, SC) . . . . .	35
1.5.1. Fonctionnement . . . . .	36
1.5.2. Jeton non-fongible (non-fungible token, NFT) . . . . .	38
1.5.3. SC NFT . . . . .	38
1.5.4. SC PoS . . . . .	39
<b>2. Transposition didactique</b>	<b>40</b>
2.1. BlocNote . . . . .	40
2.1.1. Valeurs . . . . .	40

2.1.2. Modèle . . . . .	42
2.2. Mécanismes de la chaîne de blocs illustrés avec BlocNote . . . . .	48
2.2.1. Exemple d'achat avec cryptomonnaie . . . . .	48
2.2.2. Double-dépense . . . . .	50
2.2.3. S'attribuer la récompense d'un copain . . . . .	50
2.2.4. S'attribuer les jetons d'une transaction dont on n'est pas le destinataire . . . . .	52
2.2.5. Création d'un NFT . . . . .	53
2.2.6. Achat et vente d'un NFT . . . . .	54
2.3. Modalités . . . . .	58
2.3.1. Jeu pédagogique (instructional games) . . . . .	60
2.3.2. Revue des jeux pédagogiques . . . . .	61
2.3.3. Activité déconnectée... ou pas . . . . .	67
2.4. Découpage des activités . . . . .	68
<b>3. Progammes Python</b>	<b>70</b>
3.1. Présentation générale . . . . .	70
3.1.1. Niveau . . . . .	71
3.1.2. Principes . . . . .	71
3.1.3. Architecture . . . . .	72
3.2. Description des programmes . . . . .	72
3.2.1. bchain_rousseau.py . . . . .	72
3.2.2. bchain_interac.py . . . . .	76
3.2.3. bchain_sim.py . . . . .	77
3.2.4. bchain_visualization.py . . . . .	77
3.3. Activités in silico . . . . .	78
3.3.1. Par où commencer? . . . . .	78
3.3.2. Support aux activités . . . . .	79
3.4. Mais bon, ma foi, c'est comme ça. . . . .	81
<b>4. Conclusion</b>	<b>82</b>
<b>A. Acronymes</b>	<b>85</b>
<b>B. Activités élèves</b>	<b>86</b>
B.1. Activité 1 : enchaînement de blocs et preuve de travail . . . . .	86
B.2. Activité 2 : consensus . . . . .	86
B.3. Activité 3 : transactions . . . . .	86
B.4. Activité 4 : contrats intelligents et NFTs . . . . .	86
B.5. Activité 5 : preuve d'enjeu . . . . .	86

<b>C. License of the Documentation</b>	<b>87</b>
--	-----------

# 1

## Chaînes de blocs

---

<b>1.1. Chaîne de blocs</b>	<b>4</b>
1.1.1. Définitions	4
1.1.2. Définition pour les élèves	6
1.1.3. Représentation	6
<b>1.2. Fonctionnement</b>	<b>9</b>
1.2.1. Exemple d'achat avec cryptomonnaie	11
1.2.2. Réseau	11
1.2.3. Bloc	13
1.2.4. Noeud	15
1.2.5. Jeton (token)	17
1.2.6. Adresse, clés privée et publique, et portefeuille électronique (wallet)	18
<b>1.3. Consensus</b>	<b>19</b>
1.3.1. Preuve de travail (proof of work, PoW)	20
1.3.2. Preuve d'enjeu (proof of stake, PoS)	22
1.3.3. Inattaquable (?)	23
1.3.4. Immuable, inaltérable	24
<b>1.4. Transactions</b>	<b>27</b>
1.4.1. Mempool	28
1.4.2. Confirmation, finalisation et exécution	28
1.4.3. Registre	30
<b>1.5. Contrat intelligent (smart contract, SC)</b>	<b>35</b>
1.5.1. Fonctionnement	36
1.5.2. Jeton non-fongible (non-fungible token, NFT)	38
1.5.3. SC NFT	38
1.5.4. SC PoS	39

---

Ce chapitre résume les principaux mécanismes et composants des chaînes de blocs dans le but de justifier les choix du modèle de BlocNote et poser une base pour le développement des activités destinées aux élèves.

## 1.1. Chaîne de blocs

La première section est consacrée aux définitions données par les différents acteurs du domaine des chaînes de blocs, afin de cerner une définition adaptée aux élèves du collège.

### 1.1.1. Définitions

Les définitions du dictionnaire Larousse [43] et du Merriem-Webster [40] se rejoignent, et le présent travail les reflète au mieux :

1. Technologie de stockage et de transmission de l'information, transparente et décentralisée, qui permet de valider et sécuriser n'importe quel échange de données.
2. En particulier. Base de données libre d'accès dans laquelle sont stockées chronologiquement, sous forme de blocs non modifiables liés les uns aux autres, les transactions successives effectuées entre ses utilisateurs depuis sa création. (Recommandation officielle : bloc de chaînes.)

De Filippi [33] la définit comme un cadastre décentralisé et certifié incorruptible.

Bitcoin se définit comme "un grand livre comptable partagé et public, incluant toutes les transactions confirmées", ou encore "un système de paiement pair à pair fonctionnant sans autorité centrale" [50].

Ethereum se définit comme "un réseau distribué d'ordinateurs (noeuds) exécutant le logiciel qui peut vérifier les blocs et les données de transaction" [2].

Le Ministère de l'économie, des finances et de la souveraineté industrielle et numérique [57] donne une définition très basique :

Pour définir la blockchain, le mathématicien Jean-Paul Delahaye donne l'image d' « un très grand cahier, que tout le monde peut lire librement et gratuitement, sur lequel tout le monde peut écrire, mais qui est impossible à effacer et indestructible ».

Même si cette définition n'est pas satisfaisante d'un point de vue intellectuel, elle fournit une première image mentale qui ne risque pas de nuire aux apprentissages ultérieurs.

La définition donnée par Depierre et al. dans le "Lexique de la blockchain" [52] est plus complète, et quoique trop complexe pour les élèves, elle retient des éléments dont il faut tenir compte dans notre modèle :

Technologie de stockage et de transmission de transactions, sécurisée par des méthodes de cryptographie asymétrique, fonctionnant sans organe central de contrôle et dont l'ensemble des informations, le registre, constitué par des blocs de transactions successifs de taille fixe<sup>1</sup>, est répliqué (c'est-à-dire recopié) de manière décentralisée et de pair-à-pair, sur base volontaire, sur un

---

<sup>1</sup>NDLR : dans Bitcoin et Ethereum, les blocs ne sont pas de taille fixe.

grand nombre de nœuds actifs sur internet ou sur toute autre forme de réseau informatique.

Par extension, une blockchain (littéralement, une chaîne de blocs) représente également l'outil de preuve de l'échange des informations auquel on peut accéder dans le simple but de vérifier la validité ou l'existence d'une transaction. Une blockchain peut donc être assimilée à un grand livre ou à un registre ouvert et accessible (on parle alors de blockchain publique, avec ou sans système de permission) ou confidentiel (dès lors une blockchain privée, également avec ou sans système de permission), pseudonymisé (ou parfois anonymisé), immuable et infalsifiable.

La définition donnée par le lexique de Blockchain France [14] : "technologies permettant de stocker et d'échanger de la valeur sur internet sans intermédiaire centralisé" porte à confusion, car les chaînes de blocs ne contiennent aucune valeur réelle, mais que des informations représentant des transactions, sans aucune garantie que ces transactions aient lieu. Cette définition est cependant répandue dans le monde financier, sur internet et dans l'imaginaire populaire. Il est essentiel que les élèves comprennent que la monnaie créée par les chaînes de blocs n'a aucune attaché dans le monde réel, et que sa valeur repose entièrement sur la confiance de ses utilisateurs.

Selon le NIST [54]<sup>2</sup>,

La chaîne de blocs représente un nouveau paradigme pour les interactions numériques et constituent la technologie à la base de la plupart des cryptomonnaies. Une chaîne de blocs est un registre collaboratif et inaltérable, qui tient à jour les données transactionnelles. Ces données transactionnelles sont groupées en blocs. Un bloc est connecté au bloc précédent par le biais d'un identifiant unique basé sur les données contenues par le bloc précédent. Par conséquent, si les données sont changées dans un bloc, son identifiant unique change, ce qui se voit dans tous les blocs subséquents, fournissant une preuve d'altération. Cet effet domino permet à tous les utilisateurs de la chaîne de blocs de savoir si les données d'un bloc ont été altérées. Parce que le réseau d'une chaîne de blocs est difficilement altérable, il fournit une méthode résiliente de tenue de compte collaborative.

Selon [17]<sup>3</sup>,

une chaîne de blocs est un registre distribué pour enregistrer des transactions, maintenu par plusieurs noeuds sans autorité centrale grâce à un protocole cryptographique distribué. Tous les noeuds valident l'information à insérer dans la chaîne de blocs, et le protocole de consensus garantit que les noeuds s'entendent sur un ordre unique dans lequel les informations sont insérées.

Les chaînes de blocs jouent le rôle d'un tiers de confiance, sûr et sécuritaire, qui maintient l'état global, médiatisant les échanges et produisant des calculs fiables.

---

<sup>2</sup>Traduction libre par l'auteure de ce travail.

<sup>3</sup>Traduction libre par l'auteure de ce travail.

## Chaîne de blocs, base de données et registre

Selon Wikipedia, "une base de données permet de stocker et retrouver des données structurées ou brutes, en rapport à un thème ou une activité, de nature plus ou moins différente, plus ou moins reliées entre elles", ce qui pourrait s'appliquer aux chaînes de blocs. Cependant, le dispositif d'une base de données comporte par définition un système de gestion de base de données, soit un "logiciel moteur qui manipule la base de données et dirige l'accès à son contenu", des logiciels applicatifs et des règles d'accès et d'utilisation des données ; or ces derniers éléments ne se retrouvent pas dans les chaînes de blocs. En effet, les données d'une chaîne de blocs, soit les transactions entre les participants de la chaîne, sont analysées et traitées à l'extérieur de la chaîne de blocs. On ne peut donc pas dire qu'une chaîne de blocs soit une base de données.

Selon [33], une chaîne de blocs n'est pas une base de données car elle est administrée de manière collective, mais Cachin [17] définit la chaîne de blocs comme une "base de données distribuée maintenant une liste d'enregistrements en croissance constante, contrôlée par des entités qui peuvent ne pas avoir confiance entre elles".

Du point de vue de l'informatique, le registre d'une chaîne de blocs s'apparente aux registres comme le DNS (Domain Name Server) et le RIR (registre international régional), qui gèrent et/ou distribuent des noms de domaine et les adresses IP [9], à la différence près que le registre d'une chaîne de blocs est immuable.

La chaîne de blocs est un type de registre distribué (distributed ledger), mais pas tous les registres distribués sont des chaînes de blocs. Selon [33], la chaîne de blocs constitue un registre de référencement plutôt qu'une plateforme d'archivage, car sa maintenance est coûteuse. En effet, le registre du Bitcoin avait une taille de plus de 200 GB en 2020. Celui de Ethereum est beaucoup moins volumineux car les blocs qui ne sont pas finalisés sont éliminés, il n'est donc pas nécessaire de télécharger le registre en entier.

### 1.1.2. Définition pour les élèves

La définition de chaîne de blocs qui sera retenue pour les élèves de maturité gymnasiale dans le cadre des activités BlocNote est la suivante :

Registre transparent et décentralisé, libre d'accès aux et opéré par les utilisateurs de l'intranet du collège, dans lequel sont stockées toutes les transactions effectuées entre ses utilisateurs depuis sa création, sous forme de blocs inaltérables liés les uns aux autres.

On n'attend pas des élèves qu'ils comprennent la définition à sa seule lecture, mais plutôt qu'ils l'apprivoisent et la maîtrisent au fil des activités mises en place.

### 1.1.3. Représentation

Toute chaîne de blocs commence par un bloc-genèse ne contenant aucune transaction et ne donnant lieu à aucune récompense. Le processus d'enchaînement d'un nouveau bloc dépend de l'algorithme de consensus de la chaîne de blocs, détaillé à la section 1.3. Dans tous les cas, un nouveau bloc peut être enchaîné à n'importe quel bloc déjà présent dans la chaîne.

Les chaînes de blocs ne fournissent qu'un long registre de numéros de blocs, leurs hachages, les transactions qu'ils contiennent, et les hachages de ces transactions, ainsi que la PoW le cas échéant. La table 1.1 montre un exemple simplifié de registre sans les hachages.

BlocID	PoW	Tx1	Tx2	Tx3
0	0	0	0	0
40	0b101000	04100	0	0
90	0b1011010	09100	0	0
590	0b1001001110	05100	98071	95010
5590	0b1010111010110	05100	59021	59011
940	0b1110101100	09100	48081	0
890	0b1101111010	08100	94041	94011
440	0b110111000	04100	0	0
8940	0b10001011101100	08100	84030	94050
88940	0b10101101101101100	08100	48071	98031
5440	0b1010101000000	05100	45040	0
4440	0b1000101011000	04100	49011	45010

TABLE 1.1. – Exemple simplifié de registre d'une chaîne de blocs.

D'un point de vue mathématique [9], étant donné qu'un bloc peut être enchaîné à n'importe quel bloc, et qu'on peut remonter les blocs-parents jusqu'au bloc-genèse, la chaîne de bloc est un graphe connexe et acyclique, ou plus spécifiquement un arbre enraciné orienté. En effet, un graphe représente un ensemble de noeuds reliés ou non entre eux par des arêtes ; la position dans l'espace des noeuds n'a pas d'importance, la seule chose qui compte est la relation entre les sommets. Un arbre enraciné est simplement un arbre dont un sommet porte le statut particulier de racine, qui correspond au bloc-genèse dans le cas d'une chaîne de blocs. Un arbre orienté est un arbre où les arêtes ont une direction ; dans le cas d'une chaîne de blocs, on peut remonter les blocs-parents jusqu'au bloc-genèse à partir de n'importe quel bloc.

Afin de visualiser les liens entre les blocs, une représentation graphique linéaire telle qu'illustrée dans la figure 1.1 et dans les jeux pédagogiques discutés à la section 2.3.2. est le plus souvent utilisée. Ce type de représentation ne se retrouve que dans les documents de vulgarisation, mais n'existe pas dans la chaîne de blocs.

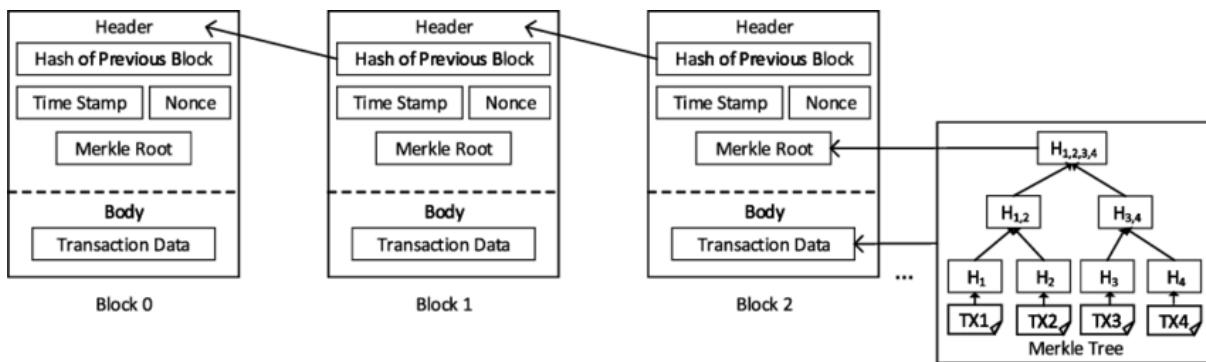


FIGURE 1.1. – Représentation graphique classique d'une chaîne de blocs [21].

Or une représentation linéaire laisse penser que le prochain bloc doit nécessairement être placé "à droite" du dernier bloc d'une chaîne, et que les chaînes respectent un certain ordre. La visualisation de BlocNote est en étoile (figure 1.2), montrant qu'un bloc peut

être enchaîné à n'importe quel bloc-parent, n'importe où dans la chaîne, et qu'aucune fourche ni bloc n'a de priorité sur une autre ; des fourches peuvent donc partir dans toutes les directions.

Par exemple, la chaîne de blocs montrée à la figure 1.2 représente la même chaîne de blocs que le registre de la table 1.1 ; la figure 3.2 montre la même chaîne de blocs, mais cette fois les blocs contiennent les transactions. Cette chaîne comporte cinq fourches : 0-40-440-4440, 0-40-440-5440, 0-40-940-8940-88940, 0-90-590-5590 et 0-90-890.

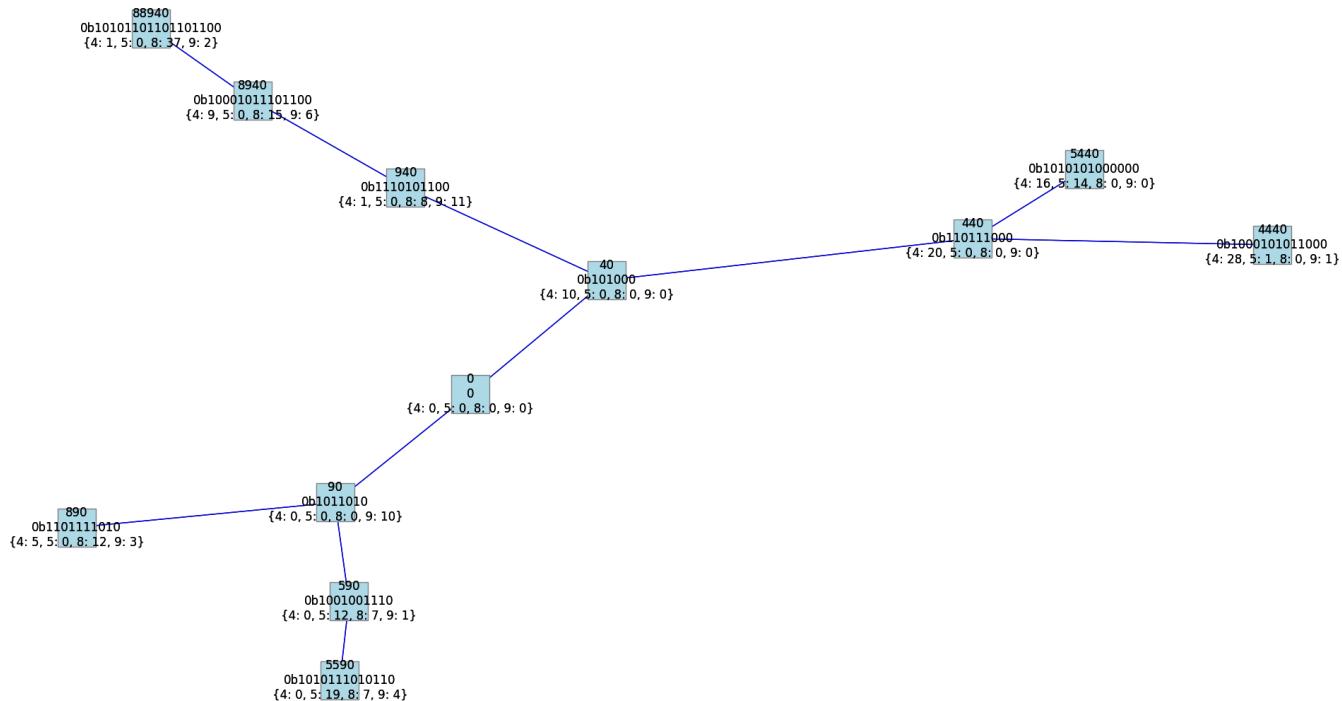


FIGURE 1.2. – Représentation graphique d'une chaîne de blocs BlocNote : chaque bloc contient le numéro du bloc, la PoW et les soldes des noeuds.

Notons qu'il existe deux types de fourches :

**fourche** résultant de la situation où plusieurs blocs ont la même profondeur ; ces fourches sont accidentelles et temporaires, et sont résolues grâce au protocole de consensus (voir section 1.3).

**fourche dure** ("hard fork") est due à une changement dans le protocole (par exemple, si une nouvelle fonctionnalité est introduite, pour rétrograder la chaîne de blocs suite à une attaque, en cas de désaccord parmi les utilisateurs, pour colmater un bug, ...) ; ces fourches sont intentionnelles et permanentes.

Le registre (table 1.1) montre les blocs par ordre chronologique d'enchaînement, alors que la représentation graphique (figure 1.2) montre l'ordre logique. Dans l'exemple de la table 1.1 et de la figure 1.2, logiquement, le bloc 940 suit le bloc 40, mais chronologiquement, il a été enchaîné après le bloc 5590, qui se trouve pourtant à une distance plus grande du bloc-genèse.

Nous verrons à la section 1.3 que seuls les blocs contenus dans la chaîne la plus longue sont conservés, tous les autres ne sont pas tenus en comptes. Par exemple, les seules transactions qui seront exécutées dans la chaîne de blocs de la table 1.1 ou de la figure 1.2 sont celles contenues dans les blocs 0-40-940-8940-88940.

## 1.2. Fonctionnement

En principe, n’importe qui peut participer à une chaîne de blocs publique et sans permission (permissionless) comme Bitcoin et Ethereum : il suffit d’installer le logiciel et le registre de la chaîne de blocs sur un ordinateur. C’était le cas dans les balbutiements de cette technologie, dans les années 2000. En réalité, une immense puissance de calcul est nécessaire pour calculer la PoW et un espace de stockage gigantesque pour le registre Bitcoin. Dans le cas d’Ethereum, le logiciel étant devenu extrêmement complexe par les SCs et la nécessité de surveiller en permanence la chaîne de blocs font que les tâches de validateur sont confiées à des serveurs spécialisés et à la pointe.

Les principaux acteurs d’une chaîne de blocs sont les suivants :

- les utilisateurs passent des transactions sur les plateformes d’échange de leur choix ;
- les plateformes d’échange font le lien entre les utilisateurs et les chaînes de blocs ;
- les noeuds gèrent les transactions que les plateformes d’échange leur soumettent, surveillent et maintiennent la chaîne de blocs ;
- les mineurs ou les validateurs insèrent dans la chaîne de blocs des nouveaux blocs contenant les transactions soumises par les utilisateurs.

Notons que les rôles sont diffus : un utilisateur peut lui-même insérer un bloc et des transactions sur la chaîne de blocs, et une plateforme d’échange peut créer sa propre chaîne de blocs (comme Binance [74] par exemple).

Les principaux composants d’une chaîne de blocs sont les suivants :

- les jetons (souvent appelés cryptomonnaie), dont la quantité représente l’importance de leur propriétaire dans la chaîne de blocs ;
- les blocs, qui pérennisent les transactions dans la chaîne ;
- les transactions, qui créent les jetons ou transfèrent la propriété d’un jeton ;
- le registre, qui consiste en la liste des transactions confirmées.

### Plateformes d’échange

Un utilisateur lambda fait le plus souvent affaire à une plateforme d’échange, et le fonctionnement d’une chaîne de blocs reste complètement opaque : il utilise la cryptomonnaie de la même façon qu’une monnaie officielle, et paye des commissions et des frais comme il le ferait avec un compte dans une banque classique.

Selon [59], les plateformes d’échange les plus performantes sont Kraken, Gemini et Crypto.com ; la figure 1.3 montre les plateformes d’échange selon leurs volumes, et la figure 1.4 montre les meilleures plateformes selon leurs caractéristiques : aucune commission (public.com), la mieux adaptée aux débutants (coinbase), la plus sécuritaire (Crypto.com) et la moins chère (Binance).

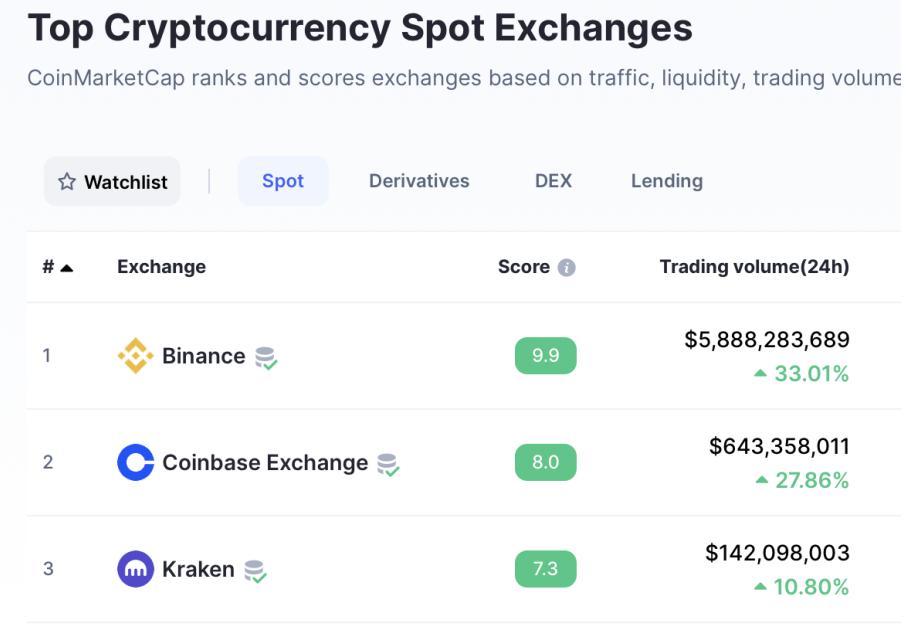


FIGURE 1.3. – Plateformes d'échange selon leurs volumes [6].



FIGURE 1.4. – Meilleures plateformes d'échange selon [65].

### 1.2.1. Exemple d'achat avec cryptomonnaie

Afin d'aider le lecteur à former un modèle mental de l'utilisation de la cryptomonnaie dans le vrai monde, la situation où un utilisateur vend un objet, et un autre utilisateur souhaite l'acheter, est illustrée dans les figures 2.1, 2.2 et 2.6 :

1. l'acheteur insère une transaction signifiant qu'il souhaite transférer une partie de ses jetons à l'adresse du vendeur, le plus souvent par le biais d'une plateforme d'échange (figure 2.1) ;
2. la plateforme d'échange soumet la transaction sur le réseau de la chaîne de blocs (figure 2.2) ;
3. un noeud-mineur ou un noeud-validateur inclue la transaction dans un bloc, puis insère le bloc dans la chaîne de blocs (figure 2.6) ;
4. si le bloc fait partie de la chaîne la plus longue, la transaction est finalisée (figure 2.6) ;

Si tout va bien, l'utilisateur-vendeur et l'utilisateur-acheteur respectent les conditions de la transaction, et l'acheteur récupère le bien mis en vente. En effet, la chaîne de bloc garantit le transfert des jetons de l'acheteur au vendeur, mais ne garantit pas l'exécution de la transaction dans le vrai monde.

Notons que si les utilisateurs étaient des initiés, ils auraient pu se passer des services des plateformes d'échange, du noeud et du mineur ou validateur, et ainsi insérer la transaction directement dans la chaîne de blocs sans devoir payer les commissions impliquées dans l'utilisation des services.

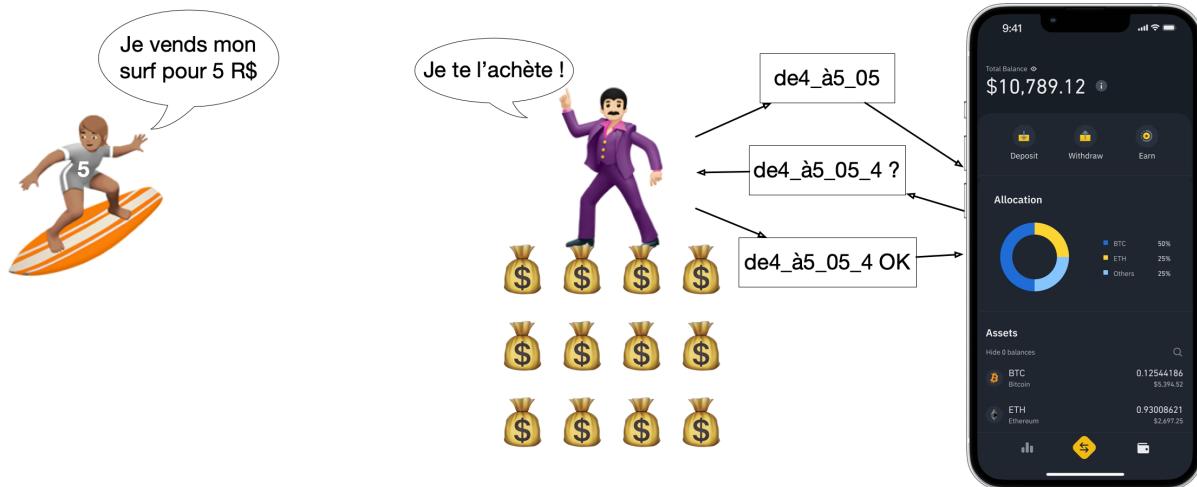


FIGURE 1.5. – La personne à l'adresse "5" vend son surf pour 5 R\$, et la personne à l'adresse "4" est intéressée à l'acheter. La personne "4" insère la transaction dans la plateforme d'échange de son choix, et la plateforme suggère d'ajouter 4 R\$ de frais de transaction. La personne "4" accepte.

### 1.2.2. Réseau

La chaîne de blocs est un système distribué, constitué de composants indépendants (les noeuds) connectés en réseau et communiquant en se passant des messages pour atteindre

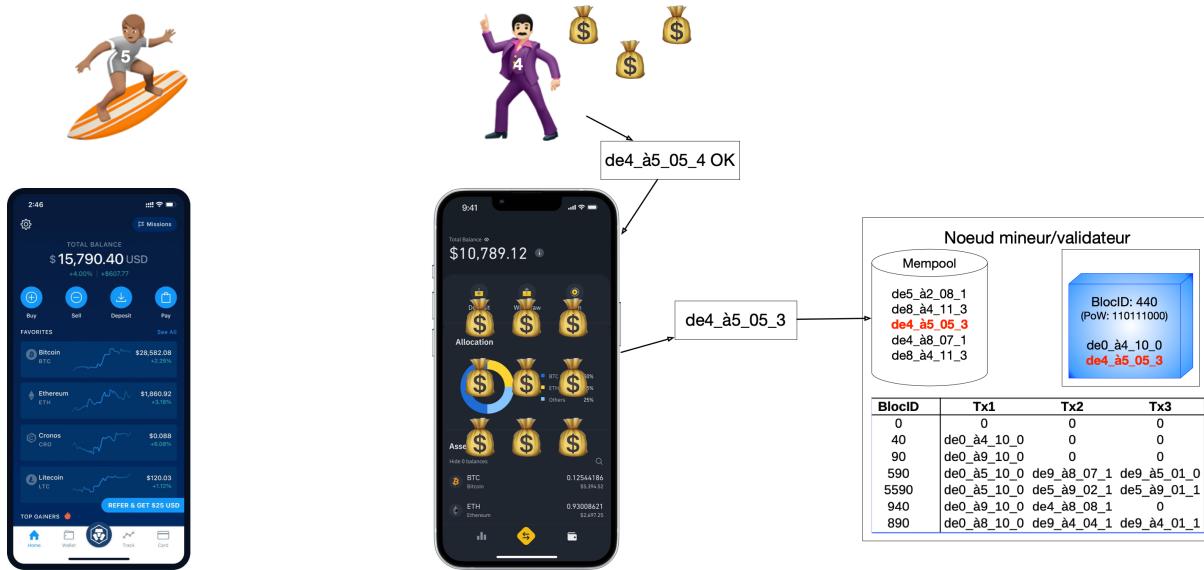


FIGURE 1.6. – La plateforme d'échange de la personne "4" gèle 9 R\$ de son compte, et soumet la transaction "de4\_à5\_05\_3" ("l'utilisateur "4" transfère 5 R\$ à l'utilisateur "5" et 3 R\$ de frais de transaction au noeud mineur ou validateur qui insère la transaction) au réseau de la chaîne de blocs. La plateforme d'échange encaisse 1 R\$ de commission au passage.

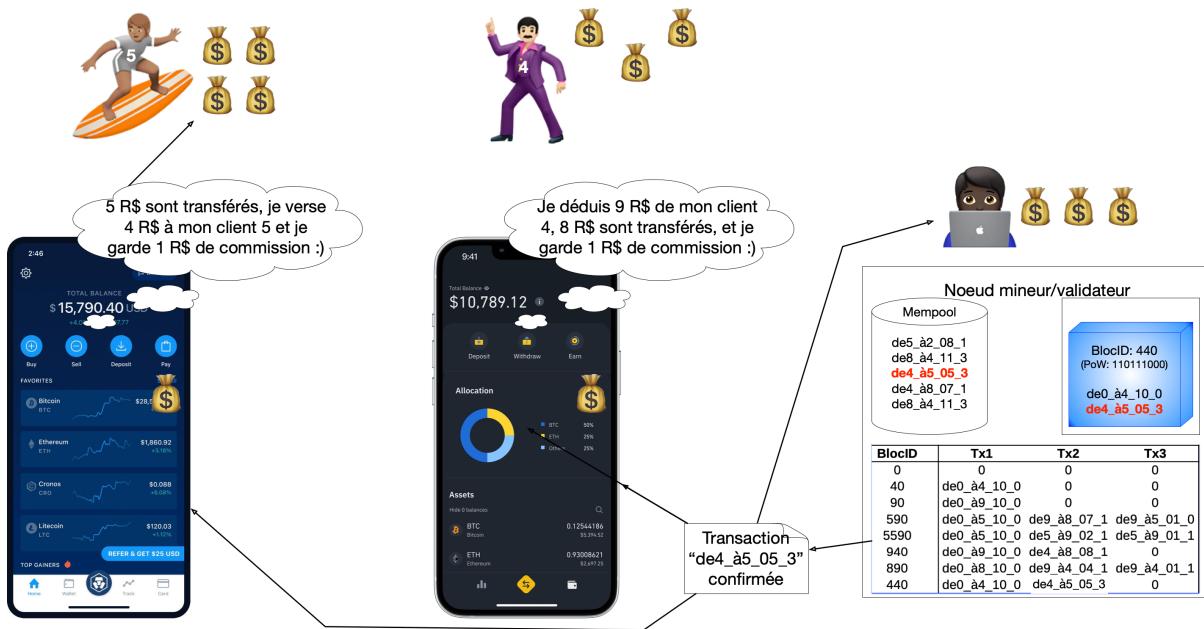


FIGURE 1.7. – Un mineur a inséré la transaction "de4\_à5\_05\_3" dans un bloc, qui a été finalisée : la transaction est donc exécutée. Le noeud-mineur empoche les frais de transaction de 3 R\$, la plateforme d'échange de l'acheteur garde 1 R \$ de commission, et la plateforme d'échange du vendeur reçoit 5 R\$. Cette dernière transfère 4 R\$ à la personne "5" et s'attribue une commission de 1 R\$.

un but commun. Il s'agit d'un système de communication pair-à-pair, comme internet, qui permet l'auto-organisation du réseau pour autant que les noeuds soient suffisamment actifs et connectés.

La structure de ce réseau est minimale, comme l'était celui de l'internet à ses débuts : le réseau est constitué de noeuds bénévoles, les messages sont diffusés de façon décentralisée et sans hiérarchie grâce à des protocoles standardisés, et il est résilient : les noeuds peuvent joindre ou quitter le réseau quand ils le veulent. Un nouveau noeud découvre ses pairs grâce à un ping-pong. La figure 1.8 illustre les réseaux Bitcoin et Ethereum.

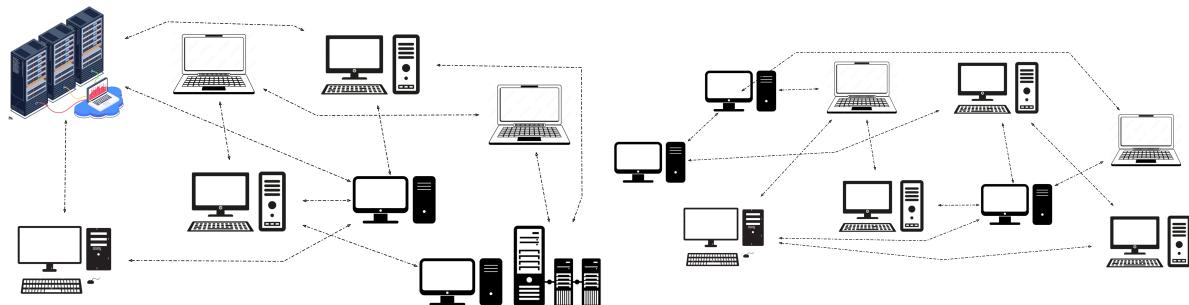


FIGURE 1.8. – Réseaux Bitcoin (à gauche) et Ethereum (à droite).

Le 11 avril 2023, Ethereum comptait 9846 noeuds [2], et plus de 10 000 noeuds complets (full nodes) opéraient dans le réseau de Bitcoin.

La diffusion (ou transmission simultanée, broadcast protocol) des transactions et des nouveaux blocs entre les noeuds du réseau se fait sur la base de deux protocoles :

- protocole de commérage (gossip protocol) : chaque noeud communique les nouveaux blocs et transactions avec ses pairs.
- protocole de meilleur effort (best-effort protocol) : comme sur internet, les mempools et les noeuds évaluent les transactions et nouveaux blocs qui circulent au mieux de leurs capacités techniques, en terme de vitesse, puissance de calcul, fiabilité du réseau, etc.

Il donc est possible qu'un noeud ne capte une nouvelle transaction ou un nouveau bloc, et donc ne l'intègre pas à son mempool ou à sa copie locale du registre donnant lieu à des versions différentes du registre.

### 1.2.3. Bloc

Un bloc contient tout simplement une liste de transactions. Pour assurer la sécurité de la chaîne de blocs, chaque bloc comporte aussi :

- le hachage (voir ci-dessous) du bloc-parent pour prouver son enchaînement à la chaîne de blocs,
- un arbre de Merkle<sup>4</sup> des transactions qu'il comporte,

<sup>4</sup>Selon Wikipedia, "en informatique et en cryptographie, un arbre de Merkle ou arbre de hachage est une structure de données contenant un résumé d'information d'un volume de données". Dans le cas des chaînes de blocs, les feuilles de l'arbre de Merkle contiennent le hachage des nouvelles transactions, et les noeuds non-feuilles contiennent le hachage de leurs enfants.

- la signature digitale du mineur qui l'a calculé.

Les blocs dans Ethereum comportent un arbre de Merkle où le hachage du bloc lui-même est une feuille, et les non-feuilles sont les hachages des blocs antécédants, jusqu'au bloc-genèse.

La figure 1.9 montre un bloc tel que lu directement dans le registre de Bitcoin, et un bloc interprété par un "explorateur de blocs", i.e. un site web reprenant les informations d'un registre de chaîne de blocs et les rendant plus lisible pour les utilisateurs

Block #170	
Summary	Hashes
Number Of Transactions	2
Output Total	100 BTC
Estimated Transaction Volume	10 BTC
Transaction Fees	0 BTC
Height	<a href="#">170 (Main Chain)</a>
Timestamp	2009-01-12 03:30:25
Received Time	2009-01-12 03:30:25
Relayed By	<a href="#">Unknown</a>
Difficulty	1
Bits	486604799
Size	0.49 kB
Weight	1.716 kWU
Version	1
Nonce	1889418792
Block Reward	50 BTC

FIGURE 1.9. – A gauche : un bloc Bitcoin brut, à droite : le bloc 170 de la chaîne Bitcoin tel que rendu par l'explorateur de blocs "Bitcoin Explorer".

La taille d'un bloc de Bitcoin est limitée à 1 MB et comporte environ 1000 transactions. La taille d'un bloc Ethereum est de 80 kB avec 70 transactions en moyenne, mais l'inclusion de contrats intelligents rend ces nombres très variables.

## Fonction de hachage

Une fonction de hachage est une fonction qui prend une chaîne de caractères (comme un fichier numérique, par exemple) comme paramètre d'entrée, et qui produit une nouvelle chaîne de caractères, de longueur prédéfinie, comme paramètre de sortie. Un exemple d'application commun de la fonction de hachage est la signature numérique d'un fichier, fournissant un moyen d'identification unique du fichier, et rendant la rétractation de son auteur impossible.

Selon De Filippi ([33]), les caractéristiques d'une fonction de hachage sont les suivantes :

- presque impossible de générer la même empreinte numérique pour deux paramètres d'entrée différents

- risque de collision réduit rapidement en augmentant la longueur de la chaîne de caractères de sortie
- toute modification apportée à un paramètre d'entrée génère une chaîne de caractères de sortie complètement différente
- impossible de dériver ou déduire les informations contenues dans un fichier à partir de sa seule empreinte numérique

Dans la chaîne de blocs, le hachage fournit l'horodatage (timestamp), sans besoin de la signature d'un tiers.

#### 1.2.4. Noeud

Un noeud est une entité (un ordinateur ou un groupe d'ordinateurs) contrôlée par une personne ou un groupe de personnes (une plateforme d'échange par exemple) où une copie de la chaîne de blocs est mise à jour et le code source de la chaîne de blocs est exécuté. Les noeuds sont à l'écoute en permanence de nouveaux blocs et transactions diffusés sur le réseau de la chaîne de bloc.

Un noeud peut se spécialiser dans l'enchaînement de blocs ; il s'appellera mineur dans une chaîne de blocs basée sur la PoW, ou validateur dans une chaîne de blocs basée sur la PoS. D'autres noeuds offriront aussi les services d'une plateforme d'échange, comme illustré dans la figure 1.10.

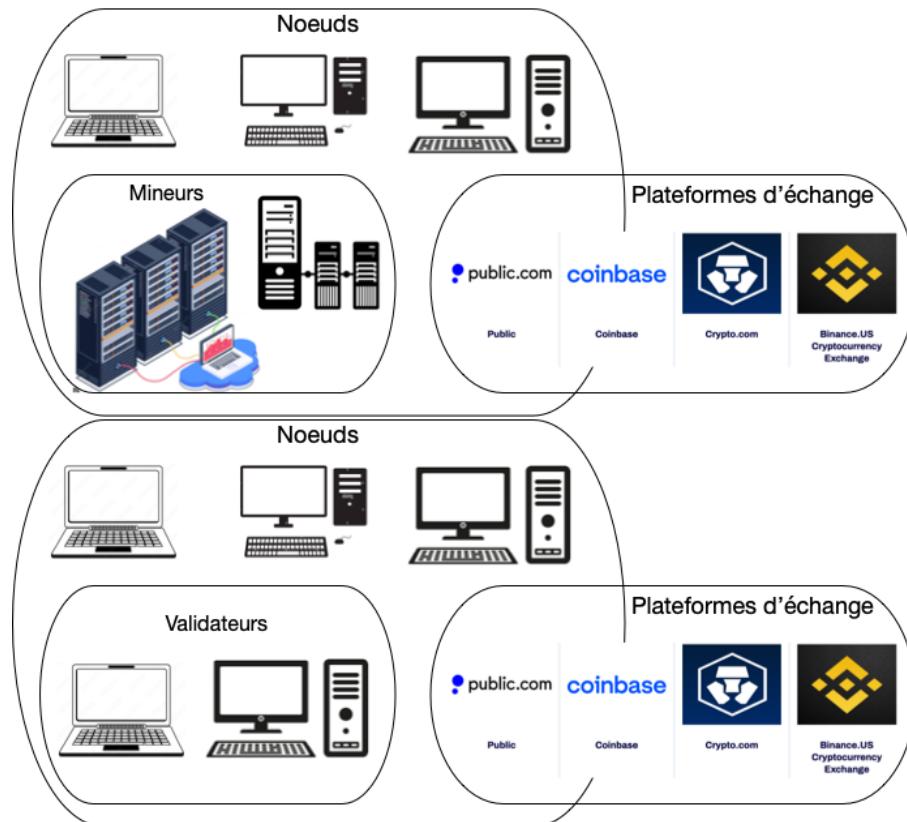


FIGURE 1.10. – Noeuds, mineurs ou validateurs et plateformes d'échange d'une chaîne de blocs basée sur la PoW (en haut) ou sur la PoS (en bas).

Au sein de la chaîne de blocs, le travail des noeuds se fait sur une base volontaire et ne rapporte aucune récompense. Cependant, à l'extérieur de la chaîne de blocs, un noeud peut monnayer ses services à des tiers en échange d'une commission.

Deux grandes catégories de noeuds participent au réseau de la chaîne de blocs :

- les noeuds spectateurs maintiennent et contrôlent la chaîne de blocs ;
- les noeuds mineurs ou validateurs insèrent des nouveaux blocs et gèrent les transactions.

La figure 1.11 illustre ces deux principaux types de noeuds.

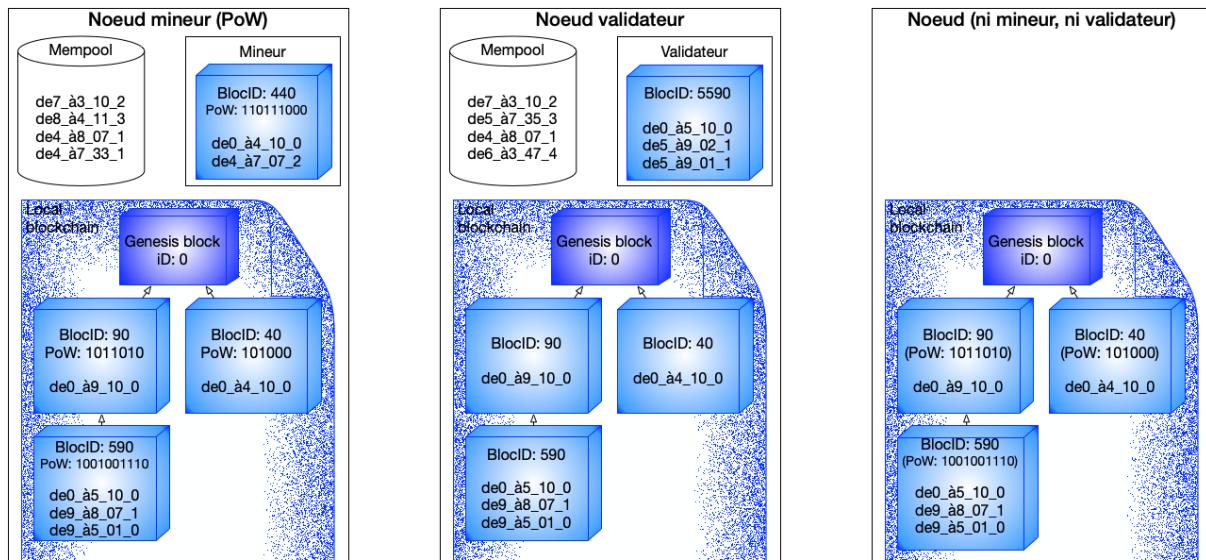


FIGURE 1.11. – Noeud-mineur (à gauche), noeud-validateur (au milieu) et noeud-spectateur (à droite).

Tous les types de noeuds (mineur, validateur ou spectateur) doivent obligatoirement maintenir et contrôler la chaîne de blocs, ce qui implique les tâches suivantes :

- stockage des logiciels et du registre ;
- écoute permanente et rediffusion de nouveaux blocs sur le réseau ;
- vérification de la validité des nouveaux blocs ;
- mise à jour de la copie locale de la chaîne de blocs ;
- services aux utilisateurs de la chaîne de blocs, comme la vérification du solde de cryptomonnaie associée à une adresse donnée.

Tous les types de noeuds peuvent injecter de nouvelles transactions sur le réseau à la demande des utilisateurs, mais ce travail n'est pas obligatoire, quoique le plus souvent rémunéré.

Les noeuds mineurs ou validateurs sont les véritables acteurs de la chaîne de blocs : en plus des tâches citées plus haut, ils créent des nouveaux blocs et gèrent les transactions, ce qui implique les tâches suivantes :

- calcul de la PoW (noeud-mineur) ou proposition et validation de blocs selon la PoS (noeud-validateur) ;

- mise à jour de la mempool (voir 1.4.1) et diffusion des transactions auprès des autres noeud ;
- choix des transactions à insérer dans un bloc, le plus souvent celles avec les frais les plus élevés ;
- validation des transactoins (vérification du solde du payeur), rejet des transactions invalides.

### Mineur

Dans une chaîne de blocs basée sur la PoW (voir section 1.3.1), le mineur est un type de noeud composé d'un ou plusieurs ordinateur(s) calculant la PoW nécessaire pour insérer un bloc dans une chaîne de blocs basée sur la PoW. Un mineur peut aussi être contrôlé et/ou hébergé par un noeud, ou offrir ses services à plusieurs noeuds. Dans Bitcoin, le minage des blocs se faisait au départ sur un simple PC, mais de nos jours, il est confié à des fermes de minage contre une rétribution, sous forme de frais de transaction, mais aussi sous forme de paiement simple, i.e. des frais sont payés même si le bloc n'est pas inséré dans la chaîne.

### Validateur

Dans une chaîne de blocs basée sur la PoS (voir section 1.3.2), une seule entité, le validateur, est responsable de l'insertion du prochain bloc de transactions, et donc du choix du bloc-parent et des transactions à insérer dans la chaîne. Les validateurs sont responsables de vérifier les nouveaux blocs propagés sur le réseau, et quand vient leur tour, ils créent et propagent des nouveaux blocs.

## 1.2.5. Jeton (token)

Un jeton est une unité de monnaie virtuelle, couramment appelée cryptomonnaie, qui n'existe que dans la chaîne de blocs, comme par exemple le bitcoin (BTC) dans Bitcoin, l'ether (ETH) dans Ethereum, ou le Binance Coin (BNB) dans Binance. Le jeton n'est pas une ressource ni un fichier, mais seulement une écriture qui n'existe que dans une transaction.

Le bitcoin (BTC) et le binance coin (BNB) peuvent se diviser jusqu'à 8 positions décimales : le BTC se divise en 10 000 000 de satoshis, et le BNB se divise en 10 000 000 jagers, le satoshi et le jager étant les unités les plus petites possibles dans leurs chaînes de blocs respectives. La plus petite unité de l'ether (ETH) est un wei, équivalent à  $10^{-18}$  ETH ; 1 ETH est égal à 1,000,000,000,000,000,000 weis.

Les jetons sont créés par la chaîne de blocs lorsqu'un bloc est finalisé sous forme de récompense au mineur (PoW) ou au validateur (PoS). La création de jetons s'appelle minage (mining) dans les chaînes de blocs à PoW, et "battre la monnaie" (minting) dans les chaînes de blocs à PoS. Pour ce faire, les mineurs et validateurs incluent une transaction de récompense dans chaque bloc soumis à la chaîne de blocs. La récompense est le seul moyen par lequel de nouveaux jetons sont créés dans une chaîne de blocs.

Lors d'une transaction entre deux utilisateurs, les jetons transférés sont dissociés de l'adresse du vendeur et associés à l'adresse du destinataire. "Dépenser" un jeton revient à

changer l'adresse à laquelle il est associé (voir section 1.2.6). Le jeton est donc constitué de toutes les signatures numériques de tous ses propriétaires depuis sa création.

*Note :* le terme "jeton" est aussi utilisé pour parler des NFTs (voir section 1.5.2) et autres actifs associés à des adresses enregistrées dans les chaînes de blocs.

## Récompense

Les noeuds mineurs ou validateurs reçoivent deux types de récompense pour leur travail :

- la récompense lorsqu'un bloc est finalisé,
- sous forme de frais de transaction lorsqu'une transaction est exécutée (voir section 1.4).

Il s'agit d'une reconnaissance de la contribution d'une entité à la chaîne de blocs, et constitue la base de sa sécurité (voir section 1.3.4).

La récompense pour les premiers blocs insérés dans Bitcoin était de 50 BTC, puis est divisée par deux tous les 210 000 blocs. Depuis mai 2020, la récompense est de 6,25 BTC. La quantité totale de BTC produite sera d'au maximum 20 999 999,9769. Au-delà, aucune récompense ne sera attribuée, et donc aucun nouveau BTC ne sera créé, mais les mineurs continueront de récupérer les frais de transaction. La fin de l'émission de nouveaux BTC est prédictive vers 2040, et cette rareté donne lieu à beaucoup de spéculations.

Dans Etherum, la récompense était de 3 ETH par bloc à ses débuts, elle est actuellement de 2 ETH par bloc, mais peut changer si la communauté le décide ou si un embranchement survient.

### 1.2.6. Adresse, clés privée et publique, et portefeuille électronique (wallet)

Chaque jeton circulant dans une chaîne de blocs est obligatoirement associé à une adresse ; plusieurs jetons peuvent être associés à une même adresse, et un utilisateur peut voir autant d'adresses qu'il le souhaite. Dans Bitcoin, l'adresse change pour chaque transaction. Les adresses fonctionnent sur la base du chiffrement asymétrique, chacune comportant une paire de clés uniques : une clé publique et une clé privée. N'importe qui peut transférer des jetons à n'importe quelle adresse grâce à sa clé publique, mais seul l'utilisateur qui détient la clé privée d'une adresse peut autoriser une transaction, i.e. le transfert de jetons depuis son adresse à celle du vendeur. Par conséquent, n'importe quelle entité qui connaît la clé privée d'une adresse peut dépenser les jetons associés à cette adresse, et si un utilisateur perd ou oublie sa clé privée, il n'a plus accès à ses jetons.

Ainsi, un utilisateur ne possède pas de jetons comme tels, mais une clé privée permettant de transférer les jetons qui sont associés à une adresse.

Le portefeuille électronique (wallet) ne fait pas partie intégrante de la chaîne de blocs, mais d'un service externe. Il ne contient aucun jeton, mais seulement la clé privée qui autorise une transaction de transfert d'un jeton, et la clé publique, générée avec la clé privée, dont le hachage constitue l'adresse de l'utilisateur sur la chaîne de blocs.

Le portefeuille peut être physique (écriture sur un disque dur ou une clé USB ou sur une feuille de papier), ou logiciel (par exemple, les portefeuilles gérés par les plateformes d'échange).

Les initiés peuvent gérer leur(s) portefeuille(s) électronique(s) de façon autonome, mais la plupart des utilisateurs font appel à des services tiers de portefeuille électroniques, ou encore des plateformes d'échange qui vendent des services de portefeuille électroniques. La plupart du temps, la clé privée demeure la propriété du service de portefeuille électronique, qui encaisse les éventuelles récompenses et ristournes si un bloc est inséré, en plus des commissions.

*Note :* Bitcoin [50] définit le portefeuille électronique comme contenant les paires de clés, mais se chargeant aussi de calculer le solde de BTC associés à une adresse données. Selon [34], le portefeuille ne contient que la clé privée associée à une clé publique.

Dans les chaînes de blocs supportant les contrats intelligents comme Ethereum, les comptes sont de deux types :

- les comptes externes, dont les jetons sont contrôlés par une clé privée détenue par un utilisateur externe, comme dans Bitcoin ;
- les comptes de SC : chaque SC est associé à une adresse unique sur le réseau de la chaîne de blocs, dont la clé privée est gérée par le code du SC lui-même. Un SC peut lui-même créer des transactions, et des jetons peuvent être associés à son adresse.

## 1.3. Consensus

La chaîne de blocs étant un système distribué et asynchrone (les noeuds n'ont pas à attendre un accusé de réception (acknowledgement<sup>5</sup>) avant de poursuivre le calcul d'un prochain bloc), les noeuds peuvent avoir une vision différente de l'état global de la chaîne de blocs. En effet, à cause du délai de propagation des blocs entre les noeuds du réseau (voir section 1.2.2), il est probable qu'un noeud produise un bloc avant de recevoir la notification d'un nouveau bloc produit par un autre noeud, et comme chaque noeud maintient sa propre copie locale du registre, des versions différentes de la chaîne de blocs peuvent coexister. Ceci induit des conflits qui se manifestent sous la forme de fourches, et un algorithme de consensus doit être mis en place pour résoudre ces conflits.

En sciences informatiques, le consensus est la tâche qui consiste à "amener tous les processus d'un groupe à s'accorder sur une valeur spécifique basée sur le vote<sup>6</sup> de chacun des processus" [12] [9]. Plus spécifiquement, dans le cas des chaînes de blocs, le consensus est "le procédé par lequel un réseau de noeuds garantit l'ordre des transactions et valide le bloc de transactions" ; il est donc "responsable de générer un accord sur l'ordre et confirmer la validité (correctness) des transactions qui constituent un bloc" [35].

Par conséquent, l'algorithme de consensus doit s'assurer que la double-dépense et la répudiation et l'altération d'une transaction soient impossibles (voir sections 1.3.3 et 1.3.4, respectivement).

Il existe plusieurs algorithmes de consensus, les plus répandus étant la PoW et la PoS, décrits dans les sections 1.3.1 et 1.3.2. Ces deux algorithmes, ainsi que la plupart des algorithmes de consensus de chaîne de blocs, reposent sur le même processus, décrit ci-dessous.

---

<sup>5</sup>an acknowledgment (ACK) is a signal that is passed between communicating processes, computers, or devices to signify acknowledgment, or receipt of message, as part of a communications protocol [9].

<sup>6</sup>Dans la PoW, les noeuds votent avec leur CPU en décidant à quel bloc ils enchaîneront le suivant [17].

## Processus

Le consensus consiste à ce que tous les participants reconnaissent la fourche la plus longue comme étant la bonne, car c'est celle à laquelle le plus grand nombre de noeuds a participé, et donc celle qui concentre le plus de travail et/ou de confiance, selon le protocole. Seules les transactions insérées dans les blocs en faisant partie sont exécutées, les autres deviennent caduques. Notons que les autres blocs ne disparaissent pas, et il est toujours possible de miner un nouveau bloc à n'importe quel bloc-parent.

Les figures 1.12, 1.13, 1.14 et 1.15 illustrent le principe de consensus dans une chaîne de blocs :

1. au point de départ, toutes les copies locales sont identiques (figure 1.12) ;
2. les noeuds mineurs ou validateurs insèrent des nouveaux blocs dans leurs copies locales ; il existe alors des versions différentes de la chaîne de blocs (figure 1.13).
3. les noeuds diffusent les nouveaux blocs et ajoutent les blocs produits par leurs pairs à leurs copies locales respectives, qui redeviennent identiques (figure 1.14) ;
4. la fourche la plus longue "gagne" car c'est à celle-ci qu'il est le plus avantageux d'enchaîner les blocs subséquents (figure 1.15).

*Note* : l'exemple montre le consensus basé sur la PoW, mais il est valable pour la PoS : il suffit de remplacer "mineur" par "validateur", et enlever les PoW.

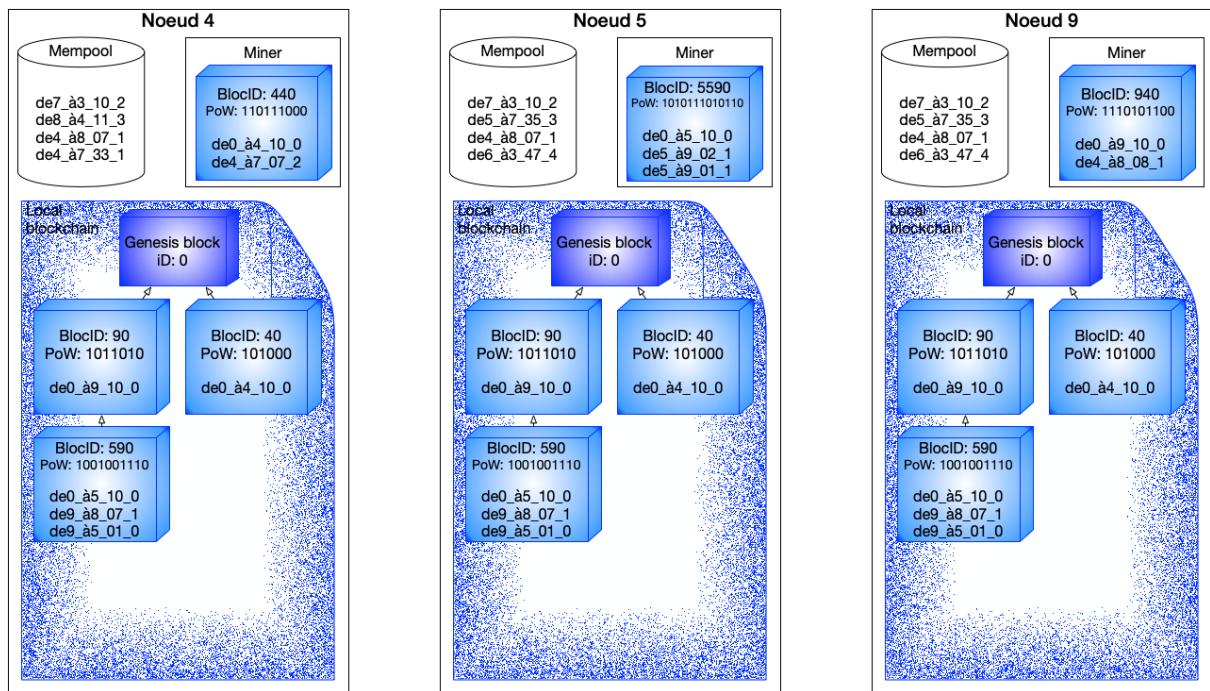


FIGURE 1.12. – Trois noeuds-mineurs participant à une chaîne de blocs possédant des copies locales identiques.

### 1.3.1. Preuve de travail (proof of work, PoW)

Pour insérer un bloc, les mineurs effectuent un travail selon différentes techniques propres à la chaîne de blocs, et c'est le bloc du mineur qui a effectué le meilleur travail qui est

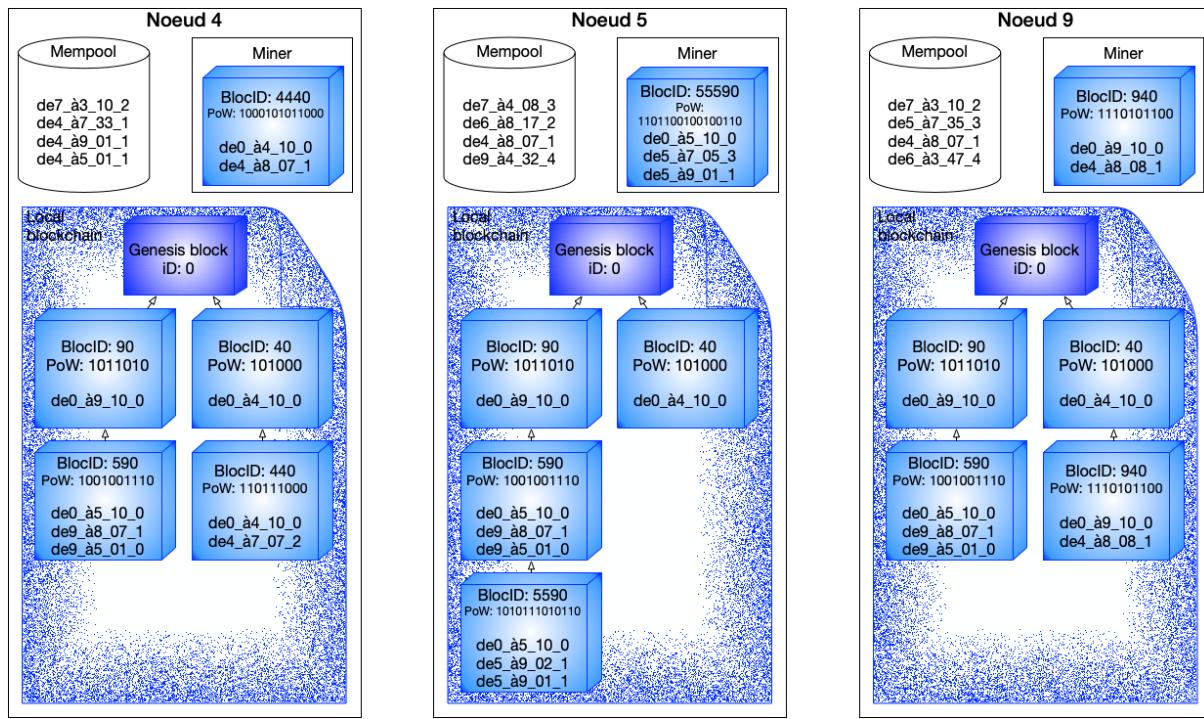


FIGURE 1.13. – Chaque mineur calcule un nouveau bloc et l'intègre à sa copie locale : les noeuds 4, 5 et 9 enchaînent les blocs 490, 5590 et 940, respectivement. Des versions différentes de la chaîne de blocs coexistent alors sur le réseau.

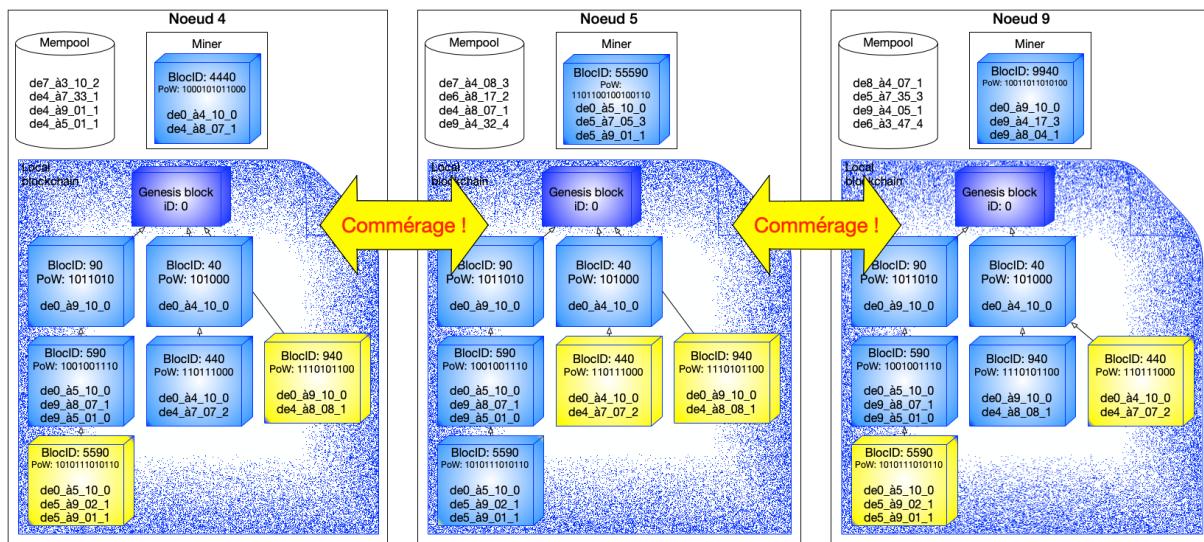


FIGURE 1.14. – Les noeuds échangent les nouveaux blocs et les intègrent à leurs copies locales.

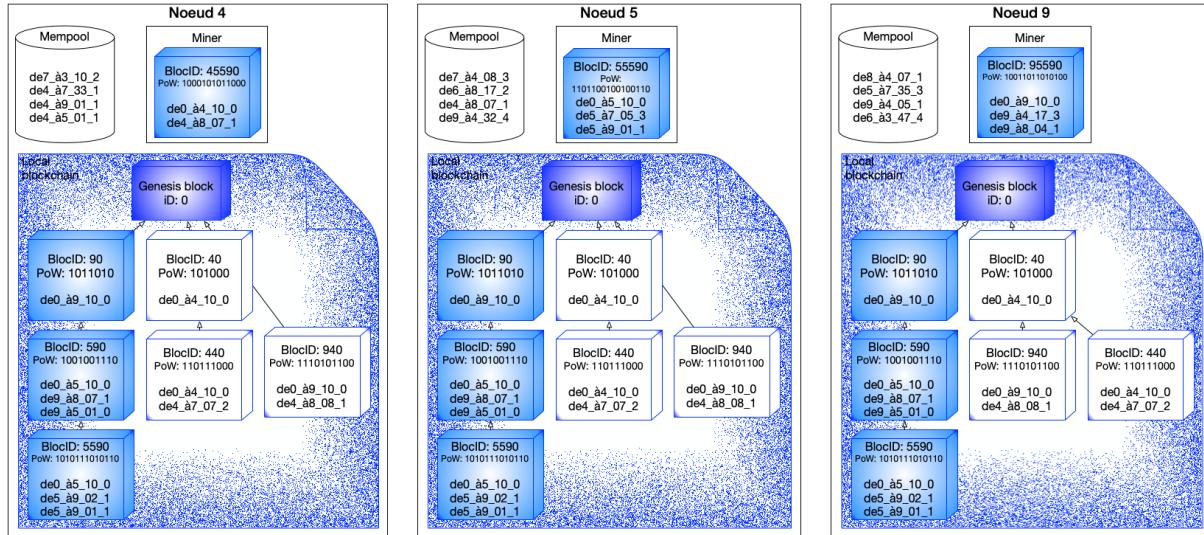


FIGURE 1.15. – La chaîne 0-90-590-5590 étant la plus longue, il est très probable que le prochain bloc miné sera attaché au bloc 5590.

inséré dans la chaîne. C'est avec ce mécanisme de consensus qu'à été créée la première blockchain connue, Bitcoin, en 2008. L'incitation à jouer selon les règles a fait son succès, car il est plus avantageux pour les mineurs de jouer selon les règles que de tricher (voir section 1.3.4) ; le système de récompense est donc suffisant pour garantir la sécurité de la chaîne de blocs.

La PoW de Bitcoin consiste à calculer la préimage d'un hachage, qui est un calcul difficile par définition, voire impossible. Le protocole de PoW demande donc aux mineurs de s'engager dans une course intense d'essais et d'erreurs pour trouver le nonce permettant d'insérer un bloc. Le rythme d'inclusion des blocs est régi par le niveau de difficulté du minage [22]. Le détail du minage du Bitcoin est expliqué dans de très nombreux sites web et livres, dont Bitcoin.fr [1], et de nombreux jeux pédagogiques y sont consacrés (voir section 2.3.1).

### 1.3.2. Preuve d'enjeu (proof of stake, PoS)

Dans une chaîne de blocs à PoS, le prochain bloc est inséré par un validateur choisi au hasard parmi tous les validateurs, et un comité de validateurs accepte (ou pas) le bloc proposé. Les validateurs reçoivent des récompenses, mais aussi des punitions en cas de mauvais comportement.

Dans le protocole de PoS, les utilisateurs qui souhaitent devenir validateurs mettent en gage un capital grâce à un SC, qui consiste à geler une partie des avoirs en jetons natifs de la chaîne de blocs à une adresse dédiée à la PoS pour une période définie. Dans Ethereum, la garantie (stake) se fait par incrément de 32 ETH, chacun faisant l'objet d'un SC (voir section 1.5) [22].

Dans Ethereum, le rythme de la PoS est divisé en époques, chacune composée de 32 créneaux ("slots") de 12 secondes. Les validateurs sont groupés en comité de 128 entités ; dans chaque créneau, un validateur est sélectionné et propose un bloc, et les 127 autres déterminent la validité du bloc proposé. Le validateur est sélectionné suite à un tirage

au sort entre tous les validateurs grâce au protocole de loterie ("lottery protocol"), dans lequel la probabilité d'être sélectionné est proportionnelle au montant de la garantie du validateur : plus les fonds mis en garantie par un validateur sont élevés, plus grandes sont les chances du validateur d'être sélectionné, et donc de recevoir une récompense, constituée de jetons fraîchement frappés (minted). Les validateurs valident les blocs de la chaîne en envoyant un vote (appelé attestation) en faveur d'un nouveau bloc sur le réseau, et les blocs qui obtiennent des votes représentant au moins deux tiers des ETH mis en garantie sont confirmés.

Au fur et à mesure que des validateurs joignent la chaîne de blocs, le taux d'acceptation de nouveaux validateurs est réduit, créant ainsi de la rareté, ce qui augmente la valeur des ethers mis en garantie, et donc la valeur de l'ether en général. [22] [18]

Selon Ethereum<sup>7</sup>, la PoS montre de nombreux avantages par comparaison à la PoW :

- pas d'utilisation excessive d'énergie par le calcul de la PoW ;
- plus facile de participer, car pas besoin d'équipement de pointe ;
- risque réduit de centralisation, car plus de noeuds sont incités à participer ;
- attaque à 51% exponentiellement plus "chère" pour l'attaquant que la PoW, car l'insertion d'un bloc dans une chaîne basée sur la PoS nécessite peu d'énergie, et à cause des punitions (voir détails ci-dessous) ;
- si une attaque à 51% survenait malgré les défenses cryptographiques et économiques, la communauté a la possibilité de ressusciter une fourche honnête.

### Récompenses et punitions

S'il est choisi, le validateur insère un bloc, puis reçoit la récompense et les frais de transaction si le bloc est finalisé, ainsi que des intérêts sur ses jetons mis en garantie. La récompense est proportionnelle au solde du validateur et inversement proportionnelle au nombre de validateurs présents sur le réseau. Une autre récompense est calculée selon la rapidité de l'attestation, encore une autre selon le nombre d'attestations reçues pour un bloc proposé, et enfin les preuves de mauvais comportement d'autres validateurs incluses dans un bloc proposé par un validateur contribuent aussi à sa récompense [2].

Le capital mis en garantie peut être détruit si le validateur agit de façon malhonnête ou s'il fait preuve de nonchalance. Si un validateur insère un bloc ou une transaction invalide, une amende sous forme d'ethers lui est infligée. Les validateurs perdent aussi un peu de leur garantie lorsqu'ils sont hors ligne.

Les amendes et punitions sont transférées à une adresse inutilisable (sans clé), ce qui décroît la quantité d'ethers en circulation, augmentant de ce fait sa valeur. Ceci constitue une récompense indirecte pour tous les participants de la chaîne de blocs, ce qui renforce sa sécurité.

#### 1.3.3. Inattaquable (?)

Comme tout système informatique, les chaînes de blocs sont vulnérables aux attaques par déni de service (DDoS), leurre d'identité (ou attaque Sybil) et ingénierie sociale. L'attaque à 51% et la double-dépenses sont quant à elles endémiques aux chaînes de blocs.

---

<sup>7</sup>(Traduction libre de l'autrice.)

Le caractère public, distribué, sans permission (permissionless) et anonyme des chaînes de blocs les rend particulièrement vulnérables aux DDoS et attaques Sybil, car il devient difficile de différencier les requêtes honnêtes des requêtes malveillantes. La solution consiste à imposer un coût économique à ce genre d'attaque :

- chaque requête entraîne un coût : les frais de transaction, ce qui rend la DDoS inefficace ;
- la mise en garantie de ressources (puissance de calcul (PoW) ou garantie en jetons (PoS)) élimine le risque d'attaque Sybil, avec laquelle un attaquant peut gagner une influence démesurée en créant un grand nombre d'identités.

Cependant, même avec un algorithme de consensus bien défini, un système décentralisé comme celui de la chaîne de blocs demeure un problème de généraux byzantins où un nombre trop grand de noeuds malveillants peut renverser la majorité de noeuds honnêtes. La menace d'une attaque à 51%, où une entité ou un ensemble d'entités contrôle plus de 50% de la puissance de calcul (PoW) ou des garanties (PoS) subsiste, rendant la double-dépense possible (voir ci-dessous). Avant de passer à la PoS, Ethereum combattait la centralisation des ressources en récompensant les blocs Ommer<sup>8</sup>, afin d'inciter les noeuds plus faibles (i.e. expérimentant plus de latence) à continuer à participer à la surveillance de la chaîne.

### **Double-dépense (double-spending)**

La double-dépense est une brèche endémique aux chaînes de blocs, qui découle du fait qu'elles sont décentralisées. Elle consiste à déclencher une action dans le monde réel grâce à une transaction, puis réutiliser les mêmes jetons dans une deuxième transaction, qui sera finalisée dans une autre fourche, et faire en sorte que cette deuxième fourche soit la plus longue. La première transaction est alors invalidée, mais l'action a déjà eu lieu dans le vrai monde.

Dans Bitcoin, tous les blocs sont conservés, et un revirement de finalisation d'un bloc peut toujours survenir. Dans Ethereum, les blocs non-finalisés sont définitivement éliminés, le risque est donc limité à la portion la plus récente de la chaîne de blocs (voir section 1.4.2).

### **1.3.4. Immutable, inaltérable**

La technologie des chaînes de blocs garantit son inaltérabilité et l'immuabilité des transactions grâce à la cryptographie et au fait qu'il est plus avantageux pour les noeuds de bien se comporter que de tenter de tricher.

En effet, étant donné que chaque bloc contient le hachage de son bloc-parent, qui lui-même contient le hachage de son bloc-parent, et ainsi de suite jusqu'au bloc-genèse, l'altération d'un bloc de la chaîne rend tous les blocs subséquents invalides. Les noeuds vigilants détectent les blocs invalides et abandonnent la fourche qui les contient.

La réPLICATION du registre sur tous les noeuds du réseau renforce aussi sa sécurité : non seulement un noeud malveillant doit modifier tous les blocs subséquents au bloc altéré, mais il doit aussi convaincre ses pairs d'adopter sa version du registre et enchaîner des blocs sur la fourche altérée.

---

<sup>8</sup>Un bloc ommer, ou bloc-oncle, est enchaîné au même bloc-parent qu'un bloc de la fourche la plus longue, mais ne fait pas partie de la fourche la plus longue.

Pour altérer un bloc, le noeud malveillant doit donc effectuer les actions suivantes :

- recalculer les hachages et les arbres de Merkel de tous les blocs subséquents à celui qu'il a altéré, ou calculer des nouveaux blocs pour faire en sorte que la fourche altérée soit la plus longue ;
- diffuser les blocs altérés pour que ses pairs les incluent dans leurs copies locales ; or cette copie étant largement minoritaire, sa diffusion reste limitée.

Pendant ce temps, les noeuds honnêtes continuent à calculer des nouveaux blocs, qui eux sont valides ; leurs fourches grandissent donc plus rapidement que celle qui est attaquée.

Ceci constitue la clé de la sécurité d'une chaîne de blocs : le noeud malveillant perd son énergie sur une chaîne qui sera très probablement écartée. Il est plus économique de simplement enchaîner un bloc valide à une fourche, et continuer à travailler afin de rendre cette fourche la plus longue.

### S'attribuer la récompense d'un copain

Le premier exemple montre le cas où un noeud malveillant tente de modifier la numérotation d'un bloc pour s'approprier la récompense du minage :

1. au départ, tous les blocs de la chaîne sont valides ;
2. le noeud malveillant change le numéro du bloc pour s'approprier la récompense du minage, mais il doit aussi modifier la PoW pour que le bloc soit valide (figure 2.7) ;
3. le noeud malveillant doit changer les numéros des blocs subséquents au bloc qu'il a altéré, car chaque bloc contient un hachage des données de son bloc-parent (figure 2.8) ;

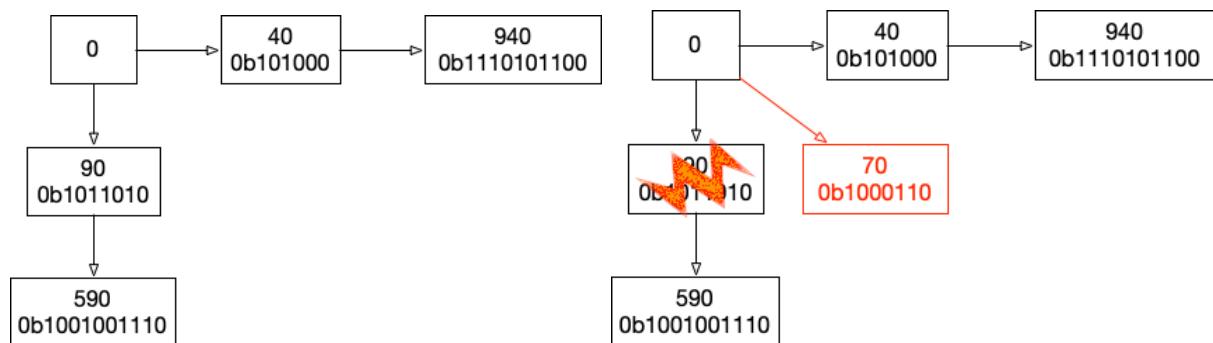


FIGURE 1.16. – A gauche : la chaîne de blocs au moment où le noeud 7 tente de l'altérer. A droite : le noeud 7 modifie la numérotation du bloc 90 et recalcule la PoW pour qu'elle corresponde au numéro de bloc 70.

Il est donc aussi coûteux de tenter d'altérer un bloc pour s'approprier la récompense que de miner un nouveau bloc, d'autant plus que les fourches dont ne fait pas partie le bloc altéré grandissent plus vite que la fourche où se trouve le bloc.

### S'attribuer les jetons d'une transaction dont on n'est pas le destinataire

Le deuxième exemple illustre le cas où le noeud malveillant altère une transaction afin de s'attribuer les jetons destinés à un autre noeud :

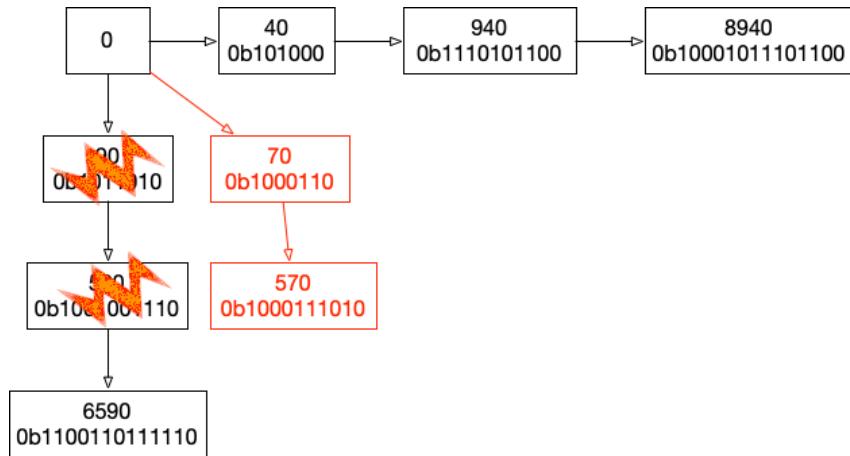


FIGURE 1.17. – Le noeud malveillant doit altérer les blocs subséquents. Les chaînes 0-90-590-6590 et 0-40-940-8940 grandissent plus rapidement que la chaîne modifiée 0-70-570.

1. au départ, tous les blocs de la chaîne sont valides ;
2. le noeud malveillant modifie une transaction pour s'attribuer les jetons ; dans l'exemple, le noeud 4 tente de s'attribuer les jetons que le noeud 8 souhaite transférer au noeud 7 dans le bloc 55483880 ; il modifie donc la transaction "de8\_à7\_12\_3" en "de8\_à4\_12\_3" (table 2.2 de droite).
3. Le noeud malveillant diffuse le bloc contenant la transaction modifiée auprès de ses pairs.
4. Suite à la modification effectuée par le noeud malveillant, la transaction "de7\_à4\_09\_1" inclue dans le bloc subséquent 355483880 devient invalide, car le solde du noeud 7 est maintenant négatif (table 2.3 de gauche).
5. Tous les blocs subséquents au bloc dans lequel le noeud malicieux a modifié une transaction sont invalidés (table 2.3 de droite).
6. Les noeuds vigilants détectent les transactions invalides, n'incluent pas ce bloc et abandonnent cette fourche.

Par conséquent, la transaction altérée par le noeud malicieux ne sera jamais finalisée, et donc jamais exécutée.

BlocID	Tx1	Tx2	3	4	5	7	8		BlocID	Tx1	Tx2	3	4	5	7	8
0									0							
.....									483880	de0_à4_10_0						
483880	de0_à4_10_0		10	10	0	0	30		5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30
5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30		55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15
55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15		355483880	de0_à3_10_0	de7_à4_09_1	21	13	24	7	15
355483880	de0_à3_10_0	de7_à4_09_1	21	13	24	7	15		7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2
7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2									

TABLE 1.2. – A gauche : la chaîne de blocs au moment où le noeud 4 tente de l'altérer. A droite : le noeud 4 tente de s'attribuer les jetons que le noeud 8 souhaite transférer au noeud 7.

BlocID	Tx1	Tx2	3	4	5	7	8	BlocID	Tx1	Tx2	3	4	5	7	8
0								0							
.....								.....							
483880	de0_à4_10_0		10	10	0	0	30	483880	de0_à4_10_0		10	10	0	0	30
5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30	5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30
55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15	55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15
		de8_à4_12_3	10	16	24	5	15			de8_à4_12_3	10	16	24	5	15
355483880	de0_à3_10_0	de7_à4_09_1	21	25	24	-5	15	355483880	de0_à3_10_0	de7_à4_09_1	21	25	24	-5	15
7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2	7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2

TABLE 1.3. – A gauche : la transaction "de7\_à4\_09\_1" inclue dans le bloc 355483880 devient invalide. A droite : tous les blocs subséquents sont invalidés.

## 1.4. Transactions

Une transaction consiste à changer l'adresse à laquelle un jeton est associé. Cette définition correspond à celle utilisée dans Bitcoin et pour les transactions ordinaires (entre utilisateurs) dans Ethereum, mais devra être étendue lorsqu'interviendront les contrats intelligents (section 1.5).

Une transaction finalisée sur une chaîne de blocs permet de pérenniser des informations comme des transactions financières et des biens numériques (comme les NFTs), et éventuellement d'autres objets comme des votes. Contrairement au système bancaire traditionnel, les transactions ne sont pas exécutées une à une, par ordre chronologique d'émission et/ou de priorité, mais par blocs.

La figure 1.18 montre une transaction sur la chaîne de blocs Bitcoin telle que visualisée grâce à un "explorateur de blocs". La transaction contient les champs suivants :

1. le numéro d'identification de la transaction sous forme de hachage ;
2. l'adresse de l'acheteur et le montant d'entrée ;
3. les frais de transaction ;
4. l'adresse du vendeur, le montant de la vente, l'adresse de retour et le montant du retour ;
5. le statut de la transaction (voir section 1.4.2).

Le montant d'entrée, de vente et de retour et les frais de transaction sont sous forme de UTXO (voir section 1.4.3). Chaque transaction comporte aussi la signature cryptographique de l'émetteur de la transaction pour autoriser le transfert des jetons [47] [70].



FIGURE 1.18. – Transaction Bitcoin telle que représentée par l'explorateur de blocs Bock Explorer [5].

Pour simplifier les exemples, les transactions seront indiquées de la façon suivante : "de4\_à5\_05\_3" correspond à la transaction où l'utilisateur "4" transfère 5 R\$ à l'uti-

lisateur "5" et 3 R\$ de frais de transaction au noeud mineur ou validateur qui insère la transaction dans un bloc. Notons que les frais de transaction ne sont payés que si la transaction est finalisée.

### 1.4.1. Mempool

Le mempool (ou memory pool) représente la zone d'attente des transactions d'une chaîne de blocs. Chaque noeud a son propre mempool. Une fois qu'une transaction est vérifiée par un noeud, celle-ci patiente à l'intérieur du mempool jusqu'à ce qu'elle soit captée par un noeud mineur ou validateur et insérée dans un bloc. Plus le nombre de transactions au sein du mempool est élevé, plus le réseau est congestionné, ce qui se traduit par un temps de confirmation moyen plus long (latence).

Les noeuds diffusent à leurs pairs les transactions contenues dans leurs mempools grâce au protocole de commérage (voir section 1.2.2 et figure 1.19).

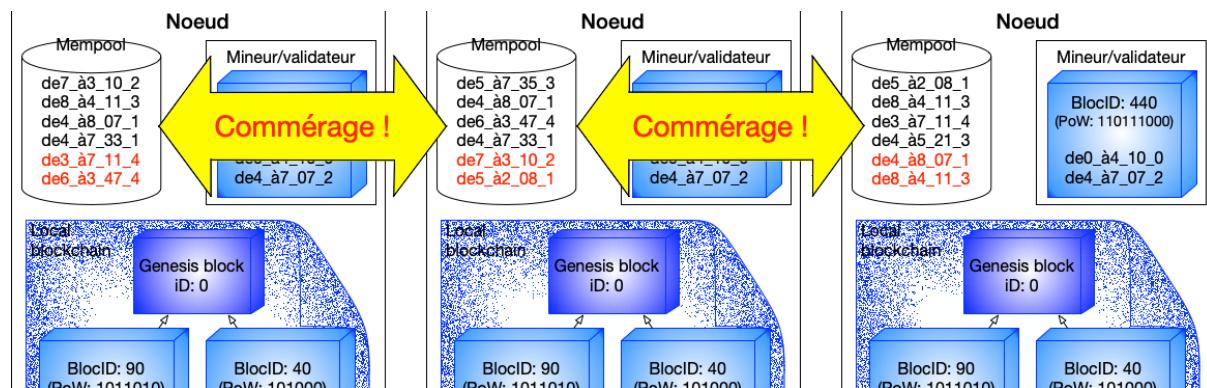


FIGURE 1.19. – Les noeuds diffusent les transactions sur le réseau.

Les mineurs et validateurs donnent la priorité aux transactions dont les frais de transaction sont les plus élevés [72]. Plusieurs mineurs ou validateurs peuvent choisir la même transaction, mais seulement un bloc par fourche ne peut la contenir. La vérification est sous la responsabilité des noeuds :

- si le solde est suffisant, la transaction est valide et éventuellement insérée (figure 1.21) ;
- sinon, la transaction est invalide et éliminée du mempool (figure 1.20).

### 1.4.2. Confirmation, finalisation et exécution

Lorsqu'un bloc est inséré dans la chaîne de blocs, les transactions qu'il contient sont inscrites dans le registre de la chaîne de blocs et celles-ci deviennent immuables, mais elles ne sont pas nécessairement exécutées. Les transactions contenues dans le blocs sont donc confirmées, mais ne sont pas encore finalisées. Pour que les transactions soient finalisées, il faut attendre que le consensus soit atteint, i.e. qu'une décision soit prise quant à déterminer quelle chaîne est conservée.

Les transactions peuvent avoir l'un des statuts suivants :

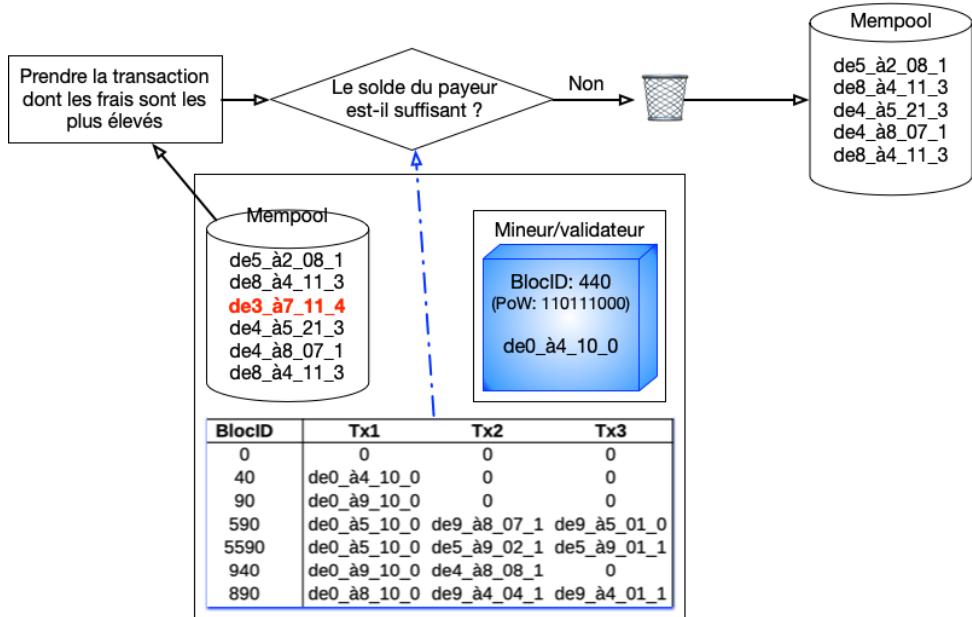


FIGURE 1.20. – Le noeud choisit la transaction dont les frais de transaction sont les plus élevés, puis vérifie si le solde du payeur est suffisant selon sa copie locale de la chaîne de blocs. Si le solde du payeur n'est pas suffisant, la transaction est rejetée et enlevée du mempool.

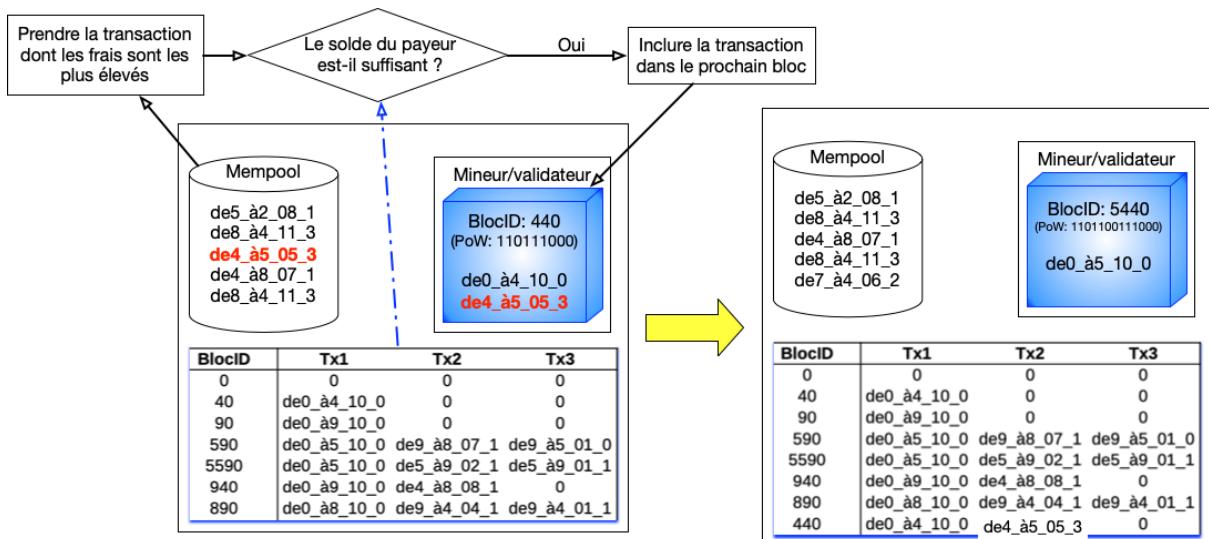


FIGURE 1.21. – Si le solde est suffisant, la transaction est inclue dans le nouveau bloc.

- une transaction est **non-confirmée** lorsqu'elle est en attente dans le mempool ;
- une transaction est **confirmée** lorsqu'elle est insérée dans un bloc par un mineur ou un validateur ;
- elle est **finalisée** si le bloc qui la contient fait partie de la chaîne la plus longue ;
- une fois finalisée, elle est **exécutée** dans le vrai monde lorsque les jetons sont transférés de l'adresse du vendeur à celle du destinataire.

Dans Bitcoin, les fourches pouvant se développer à l'infini, les transactions ne sont jamais finalisées. La décision d'exécuter les transactions repose donc sur la probabilité qu'une chaîne soit la plus longue ; une attente d'environ 60 minutes est considérée comme une garantie suffisante qu'un bloc fasse partie de la chaîne la plus longue.

Dans Ethereum, une époque correspond à une chaîne de 32 blocs, et lorsque le 32e bloc est enchaîné, ils sont tous confirmés. Si deux époques B et C sont enchaînées à la suite d'une époque A, les blocs de l'époque A sont finalisés, et toutes les transactions qu'ils contiennent sont exécutées. Les blocs des fourches concurrentes à l'époque A sont éliminées définitivement, ce qui a pour heureuse conséquence de réduire l'espace de stockage du registre, mais demande aux noeuds de faire acte de foi pour tout ce qui s'est passé entre le bloc-genèse et l'époque A.

En pratique, environ 98% les transactions soumises sont éventuellement finalisées, les autres s'étant perdues lors de leur diffusion ou rejetées car les frais de transaction ou le gaz étaient insuffisants.

### 1.4.3. Registre

Le registre d'une chaîne de blocs sont généralement de deux types : une liste d'utilisateurs avec leurs soldes, comme les comptes bancaires normaux, ou une liste de UTXO (unspent transaction output). La figure 1.22 illustre les deux types de registre :

- le registre de type "compte bancaire" correspond à la situation où l'utilisateur dépose tous les jetons qu'il reçoit dans un endroit donné, et quand il souhaite acheter un objet, il prend le montant dont il a besoin dans son coffre.
- Le registre de type UTXO correspond à la situation où à chaque fois qu'un utilisateur reçoit un certain nombre de jetons, il le met à un endroit différent dans sa maison. Lorsqu'il veut acheter un objet, il reprend les jetons qu'il a déposés à différents endroits dans sa maison, paye le montant et les frais demandés, puis dépose ce qui reste dans un autre endroit de sa maison.

La table 1.7 reprend la chaîne de blocs de la table 1.1 sous la forme d'un registre de type "compte bancaire".

Même s'il est plus simple et plus rapide de consulter un solde dans un registre de type "compte bancaire", la plupart des chaînes de blocs utilise un registre de type UTXO simplement pour économiser de l'espace de stockage. De plus, pour vérifier l'origine des jetons, il faut de toute façon remonter la chaîne et analyser toutes les transactions.

### UTXO (Unspent Transaction Output)

Dans la plupart des chaînes de blocs, le registre ne contient pas les soldes de jetons associés aux adresses des propriétaires, mais seulement les UTXO. Un UTXO correspond



FIGURE 1.22. – Représentation des registres de type "compte bancaire" et "UTXO".

BlocID	Tx1	Tx2	Tx3	Solde
0	0	0	0	{4: 0, 5: 0, 8: 0, 9: 0}
40	4100	0	0	{4: 10, 5: 0, 8: 0, 9: 0}
90	9100	0	0	{4: 0, 5: 0, 8: 0, 9: 10}
590	5100	98071	95010	{4: 0, 5: 12, 8: 7, 9: 1}
5590	5100	59021	59011	{4: 0, 5: 19, 8: 7, 9: 4}
940	9100	48081	0	{4: 1, 5: 0, 8: 8, 9: 11}
890	8100	94041	94011	{4: 5, 5: 0, 8: 12, 9: 3}
440	4100	0	0	{4: 20, 5: 0, 8: 0, 9: 0}
8940	8100	84030	94050	{4: 9, 5: 0, 8: 15, 9: 6}
88940	8100	48071	98031	{4: 1, 5: 0, 8: 37, 9: 2}
5440	5100	45040	0	{4: 16, 5: 14, 8: 0, 9: 0}
4440	4100	49011	45010	{4: 28, 5: 1, 8: 0, 9: 1}

TABLE 1.4. – Représentation d'un registre de type "compte bancaire".

à la sortie d'une transaction qui n'a pas encore été utilisée comme entrée dans une nouvelle transaction ; la somme des UTXO correspond donc au montant de cryptomonnaie autorisé à changer de propriétaire.

Les UTXO sont indivisibles ; si le montant à transférer est inférieur aux entrées en UTXO, alors le change est calculé et remboursé sous forme d'un nouvel UTXO. Le paiement se fait donc toujours sur la base d'une combinaison d'autres transactions entières [46]. Dans un registre de type UTXO, les frais de transaction n'apparaissent pas explicitement, mais sont calculés en soustrayant les UTXO sortants des UTXO entrants (fees = input UTXOs - output UTXOs).

Dans la transaction Bitcoin de la figure 1.18, le montant de la vente est de 0,01938783 BTC. L'acheteur prend un de ses UTXO dont le montant s'élève à 0,01956740 BTC pour payer la vente et les frais de transaction de 0,00017215 BTC. Il reste donc 0,00000742 BTC, qui lui sont retournés sous forme d'un UTXO. Notons que les frais de transaction ne font pas l'objet d'un UTXO, mais sont déduits de la différence entre les UTXO en entrée (0,01956740) et les UTXO en sortie ( $0,01938783 + 0,00000742 = 0,1939525$ ), c'est-à-dire  $0,01956740 - 0,1939525 = 0,00000742$ .

A titre d'exemple, la table 1.5 montre les entrées et sorties correspondant aux transactions de la chaîne de blocs de la table 1.1, et la table 1.6 celles de la fourche la plus longue.

Pour exécuter la transaction "48071"<sup>9</sup> insérée dans le bloc 88940, deux entrées (input) sont nécessaires : les 3 R\$ transférés par l'utilisateur "8" et les 5 R\$ transférés par l'utilisateur "9" dans le bloc précédent (8940), et deux sorties (output) en résultent : 7 R\$ sont transférés à l'utilisateur "8" comme paiement et 1 R\$ à l'utilisateur "8" pour les frais de transaction.

## Solde

Le solde de cryptomonnaie associé à une adresse est obtenu en remontant la chaîne jusqu'au bloc-genèse et en additionnant tous les UTXO associés à cette adresse. Ce calcul est fait à l'extérieur de la chaîne de blocs. La table 1.7 reprend l'exemple de la chaîne 0-40-940-8940-88940 de la table 1.6 : les 40 R\$ créés dans cette fourche ont été distribués parmi les utilisateurs "4", "5", "8" et "9", selon les transactions dont ils ont fait l'objet.

A tout moment, la somme de tous les UTXO est exactement égale à l'ensemble de tous les jetons émis dans la chaîne de blocs depuis le bloc-genèse. Dans la fourche 0-40-940-8940-88940, quatre blocs ont été insérés au total, ce qui correspond à la création de 40 R\$, et dix sorties (output) n'ont pas été utilisées lors des transactions, dont le total est aussi de 40 R\$.

---

<sup>9</sup>La transaction "48071" signifie que l'utilisateur "4" paye 7 R\$ à l'utilisateur "8" et paye 1 R\$ de frais de transaction au noeud-mineur ou validateur "8".

BlocID	PoW	Tx1	Tx2	Tx3	Tx1	Tx2	Tx3
0	0	0	0	0	output: 10à4		
40	0b101000	4100	0	0			
90	0b1011010	9100	0	0	output: 10à9		
590	0b1001001110	5100	98071	95010	output: 10à5	input: 10à9 output: 7à8 output: 1à9 output: 1à9	input: 1à9 output: 1à5
5590	0b1010111010110	5100	59021	59011	output: 10à5	input: 10à5 output: 2à9 output: 1à5 output: 7à5	input: 7à5 output: 1à9 output: 1à5 output: 5à5
940	0b1110101100	9100	48081	0	output: 10à9	input: 10à4 output: 8à8 output: 1à9 output: 1à4	
890	0b1101111010	8100	94041	94011	output: 10à8	input: 10à9 output: 4à4 output: 1à8 output: 5à9	input: 5à9 output: 1à4 output: 1à8 output: 3à9
8940	0b10001011101100	8100	84030	94050	output: 10à8	input: 10à8 output: 3à4	input: 10à9 output: 5à4
88940	0b10101101101101100	8100	48071	98031	output: 10à8	input: 3à4 input: 5à4 output: 7à8 output: 1à8	output: 5à9 output: 3à8 output: 1à8 output: 6à9

TABLE 1.5. – Entrées (input) et sorties (output) correspondant aux transactions de la chaîne de blocs de la table 1.1.

BlocID	PoW	Tx1	Tx2	Tx3	Tx1	Tx2	Tx3
0	0	0	0	0	output: 10à4		
40	0b101000	4100	0	0			
940	0b1110101100	9100	48081	0	output: 10à9	input: 10à4 output: 8à8 output: 1à9 output: 1à4	
8940	0b10001011101100	8100	84030	94050	output: 10à8	input: 10à8 output: 3à4	input: 10à9 output: 5à4
88940	0b10101101101101100	8100	48071	98031	output: 10à8	input: 3à4 input: 5à4 output: 7à8 output: 1à8	input: 5à9 output: 3à8 output: 1à8 output: 1à9

TABLE 1.6. – Fourche la plus longue de la chaîne de blocs de la table 1.1.

<b>UTXO</b>	<b>noeud:</b>	<b>solde:</b>
output: 8à8	4	1
output: 1à9	5	0
output: 1à4	8	$8+7+10+7+1+3+1 = 37$
output: 7à8	9	$1+1 = 2$
output: 10à8		
output: 7à8		
output: 1à8		
ouput: 3à8		
ouput: 1à8		
output: 1à9		

TABLE 1.7. – Détermination du solde associé à une adresse donnée.

## 1.5. Contrat intelligent (smart contract, SC)

La différence fondamentale entre la chaîne de blocs Ethereum et les chaînes de blocs classiques comme Bitcoin réside dans le fait qu'elle supporte les contrats intelligents. Cointelegraph [64] et AIMultiple [28] donne des exemples d'applications des SCs dans le vrai monde :

- finance décentralisée (Decentralized Finance, DeFi) : services de prêt, emprunt, hypothèque, courtage, ... , en marge de l'industrie bancaire. Ce secteur représentait une valeur de 94 milliards de dollars américains en 2021.
- jetons non-fongibles (non-fungible tokens, NFTs) : actifs digitaux uniques représentant des contenus de jeux vidéo en ligne, dont la rareté peut être prouvée grâce au registre de la chaîne de blocs
- contrats légaux : signature électronique, droits d'auteur
- immobilier : propriété fractionnée de biens, cadastre
- organisations autonomes décentralisées (Decentralized Autonomous Organizations, DAOs) : corporations dont la propriété et la rétribution est gérée par des SCs
- marché des données (data marketplace) : essais cliniques, assurances
- logistique : gestion de la chaîne d'approvisionnement, expédition
- internet des objets (Internet of Things, IoT) : par exemple, lorsque la quantité de savon à lessive est trop bas, la machine à laver passe automatiquement la commande.

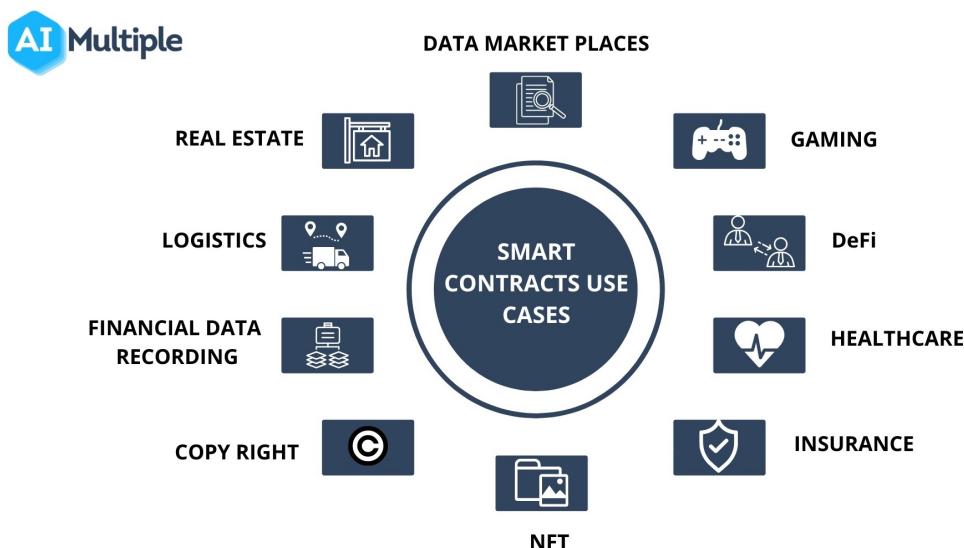


FIGURE 1.23. – Exemples d'utilisation de contrats intelligents en 2023 selon [28].

De Filippi [33] fait cependant remarquer que peu d'applications autres que financières peuvent être entièrement codifiées à partir d'informations stockées sur une chaîne de blocs, la plupart des autres applications nécessitant des informations externes et/ou une intervention humaine. Par exemple, dans l'exemple de la machine à laver, la livraison du savon implique un humain.

Dans le cadre de ce travail, les contrats intelligents seront utilisés pour la création, la vente et l'achat de NFTs et la mise en garantie (staking) des validateurs.

## Définition

La définition du contrat intelligent donnée dans le "Lexique de la blockchain" [52] a été retenue car elle est représentative de ce qui semble accepté dans le monde des chaînes de blocs en général (comme dans [14] par exemple) :

Programme autonome fonctionnant sur un protocole de type blockchain qui, une fois activé (par le paiement du gaz nécessaire), exécute automatiquement les opérations prévues dans la blockchain concernée, ce sans intervention humaine. Un smart contract fonctionne comme tout algorithme ou comme toute instruction conditionnelle de type « si – alors » (soit : si telle condition est vérifiée, alors telle conséquence doit obligatoirement se produire). Dès lors, un smart contract est fréquemment utilisé pour automatiser des transactions financières, ce en particulier dans le monde de la finance décentralisée (DeFi), ou par exemple pour fixer les conditions de création et de “vie” d'un jeton.

Par leur référence au "paiement du gaz", les auteurs lient les contrats intelligents à la chaîne Ethereum, mais il est important de mentionner qu'il existe d'autres chaînes de blocs gérant des contrats intelligents.

## Ni contrat, ni intelligent

La remarque suivante des auteurs du "Lexique de la blockchain" [52] est particulièrement pertinente, et devrait être discutée avec les élèves au moment d'aborder les contrats intelligents :

(...) un smart contract n'est ni « smart », ni un « contrat ». En droit suisse, il n'est pas impossible que l'adhésion à un smart contract puisse être considérée comme un acte concluant donnant naissance à un contrat valable. Cela étant, le code régissant le smart contract n'étant généralement pas compréhensible pour son adhérent, les termes du contrat ainsi conclu devraient être fixés, non pas par le smart contract lui-même, mais par les communications off-chain (ndlr : en dehors de la chaîne de blocs, le plus probablement sur une plateforme d'échange) faites par l'offrant. Ce sera notamment le cas lorsque l'offre d'adhésion au smart contract est véhiculée par un site web publicitaire, par un white paper (ndlr : livre blanc, soit l'écrit fondateur d'une technologie), voire par une vidéo présente sur des sites publics comme Youtube.

En effet, le contrat intelligent n'a rien d'intelligent, car il ne fait qu'exécuter des instructions bêtement, sans faire "preuve de discernement, de jugement, de bon sens" [43].

Un contrat intelligent est un automate exécuteur de clause, car il ne modère pas son contenu, et une fois lancé, il est difficile de l'amender ou de le terminer.

De Filippi [33] présente le SC comme un logiciel non-traditionnel car il est exécuté collectivement.

### 1.5.1. Fonctionnement

Chaque fois que son adresse apparaît dans une transaction, le SC est exécuté sur la machine virtuelle de tous les ordinateurs et serveurs du réseau de la chaîne de blocs sur

une machine virtuelle. Dans Ethereum, appelée EVM (Ethereum Virtual Machine) (voir figure 1.24).

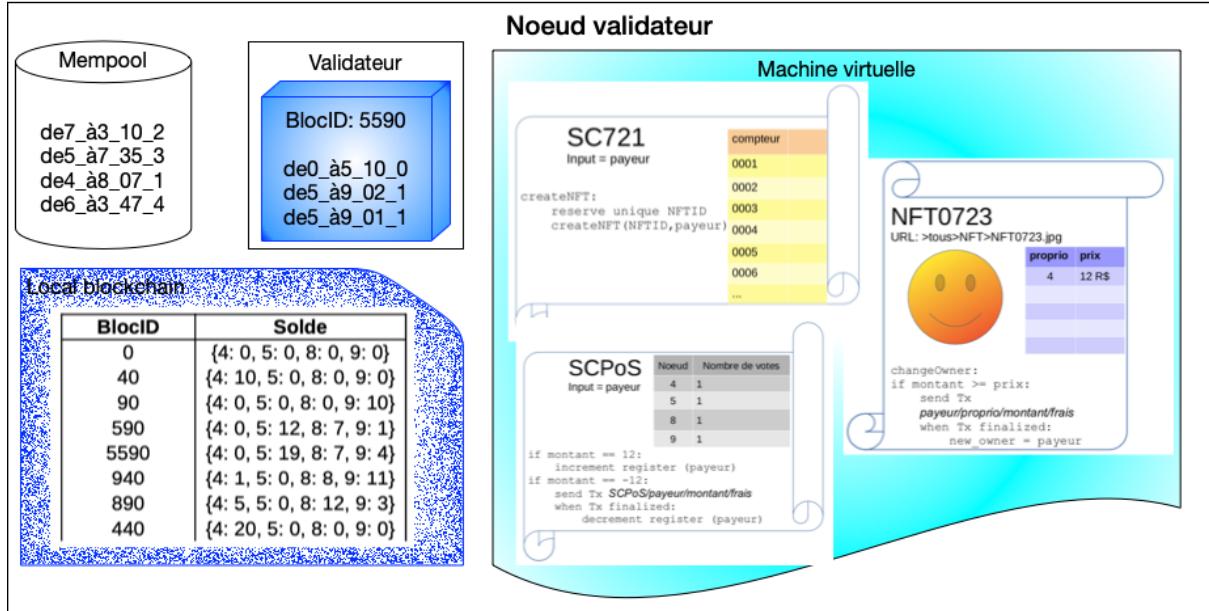


FIGURE 1.24. – Les noeuds de la chaîne de blocs Ethereum exécutent les SCs sur l’EVM (Ethereum Virtual Machine).

Le SC vérifie d’abord si les conditions prédéfinies de son exécution sont remplies ; si elles le sont, le reste du code s’exécute automatiquement, sans intervention externe. Un SC déterministe fonctionne de façon complètement autonome, sans besoin d’informations provenant de l’extérieur ; un SC non-déterministe requiert un état externe (comme la météo, une localisation, un prix ou un taux de change) pour l’exécution de son code.

## Gaz

Le gaz est similaire aux frais de transaction dans Bitcoin, mais le gaz est payé par l’émetteur de la transaction chaque fois qu’un contrat intelligent est exécuté, et non pas seulement au passage de la transaction. Le gaz est proportionnel à la complexité du SC, et est sensé représenter la puissance de calcul déployée pour exécuter un SC et/ou effectuer une transaction en lien avec le SC. Ceci permet notamment de limiter le nombre de SC en exécution, et ainsi limiter la consommation d’énergie et décourager le déploiement de SC dont le code tournerait en boucle à l’infini. Si le gaz vient à manquer pendant l’exécution d’un SC, la transaction est annulée, la chaîne de blocs rétrogradée, mais le gaz n’est pas remboursé à l’émetteur de la transaction.

Lorsqu’un SC est exécuté, le gaz qui lui est associé est "brûlé" : les jetons sont transférés à l’adresse nulle et deviennent inutilisables. Comme dans le cas des récompenses et punitions, le nombre d’ethers en circulation diminue, ce qui a pour effet d’augmenter la valeur de l’ether, et constitue une récompense indirecte pour les noeuds qui font tourner les SCs.

Pour donner un ordre de grandeur, le déploiement d’un SC de création de NFTs sur Ethereum (section 1.5.3) peut coûter entre 400 et 1000 chf [67]. Une fois mis en place

dans l'EVM, jusqu'à  $2^{256} \approx 10^{77}$  NFTs peuvent être conçus avec le même SC de création de NFT ; le gaz est de l'ordre de 60 chf en moyenne pour créer un NFT avec un SC de création de NFT existant.

### 1.5.2. Jeton non-fongible (non-fungible token, NFT)

La définition du NFT choisie dans le cadre de ce travail est celle de [14] :

Actif numérique personnalisé par son auteur, émis et échangeable sur une blockchain, et possédant les caractéristiques d'une cryptomonnaie : infalsifiabilité, unicité, enregistrement des échanges dans un registre immuable, sécurité des échanges, etc. En particulier, un token est transférable (et non duplicable) entre deux parties sur Internet, sans nécessiter l'accord d'un tiers.

Les jetons comme la cryptomonnaie sont interchangeables : l'adresse à laquelle ils sont associés change selon les transactions, alors que chaque NFT est unique et possède sa propre adresse, qui reste la même, peu importe les transactions dont il fait l'objet.

Les œuvres d'art digitales représentées par des NFTs ne sont pas directement insérées dans la chaîne de blocs pour des raisons pratiques d'espace de stockage mémoire. Les œuvres sont donc stockées à l'extérieur de la chaîne ("off-chain"), idéalement sur des supports fiables, pérennes et dont les coûts sont raisonnables, et c'est l'URL de ce support qui est inclus dans le NFT. Par conséquent, si l'URL devient caduque ("link rot"), par exemple si le support n'est plus disponible ou si l'URL change, le NFT perd toute sa valeur, mais la chaîne de blocs n'est pas altérée [23].

Dans ce travail, nous nous limiterons à l'application la plus répandue des NFTs, soit ceux représentant des œuvres digitales accessibles sur le web.

#### Droits d'auteur

Le NFT a été une révolution car il a introduit pour la première fois la notion de propriété d'un fichier numérique partagé [33], et des redevances sur l'utilisation d'œuvres numériques peuvent être mises en place dans un SC NFT. Cependant, les NFTs n'ont rien à voir avec les droits d'auteur : une entité qui achète un NFT ne détient pas ses droits d'auteur, et une entité qui crée puis vend un NFT ne perd pas ses droits d'auteur.

### 1.5.3. SC NFT

Deux types de SCs en lien avec les NFTs sont considérés dans le cadre de ce travail : le SC ERC-721, pour la création des NFTs, et le SC NFT, qui gère la vente et l'achat d'un NFT.

#### Création d'un NFT

Dans le vrai monde, n'importe qui peut créer un NFT, soit en créant lui-même le SC NFT, soit par l'intermédiaire d'une plateforme d'échange spécialisée, comme OpenSea, qui au passage prend une commission de 2,5% sur toutes les transactions.

Dans Ethereum, un SC standard de type ERC-721 est utilisé pour créer les SCs des NFTs [73], dont voici quelques fonctions [62], [73] :

```
function transferFrom(address _from, address _to, uint256 _tokenId) external payable;
```

Cette fonction correspond à la vente d'un NFT : elle demande comme argument l'adresse du propriétaire actuel, l'adresse de l'acheteur et l'identifiant du NFT, et son rôle et de modifier le numéro d'identification du propriétaire du NFT dans le contrat intelligent.

```
function approve(address _approved, uint256 _tokenId) external payable; Cette fonction vérifie si le propriétaire du NFT a bien autorisé la vente.
```

```
function ownerOf(uint256 _tokenId) external view returns (address); Cette fonction prend comme argument le numéro d'identification d'un NFT, et renvoie son propriétaire.
```

### Vente et achat d'un NFT

Les NFTs sont des SCs possédant une adresse sur le réseau de la chaîne de blocs et un numéro d'identification globalement uniques. Un SC NFT est un mécanisme pour implanter un accord de vente entre le propriétaire du NFT et un acheteur, qui comporte trois fonctions :

1. vérifier la propriété du NFT
2. coordonner son transfert
3. éventuellement prévoir les redevances.

Le déroulement du processus de vente/achat (changement de propriétaire du NFT et des jetons) se fait automatiquement, sans l'intervention d'un utilisateur de la chaîne de blocs ni interaction directe entre le vendeur et l'acheteur : les jetons correspondant au montant de la vente sont d'abord associés à l'adresse du SC NFT, et lorsque la condition est remplie (i.e. la transaction est finalisée), c'est le SC NFT lui-même qui soumet automatiquement une autre transaction pour que les jetons soient transférés à l'adresse du vendeur.

Dans le vrai monde, n'importe qui peut acheter un NFT en soumettant une transaction directement sur le réseau de la chaîne de blocs, mais la plupart du temps l'opération s'effectue par le biais d'une plateforme d'échange spécialisée comme OpenSea.

### 1.5.4. SC PoS

Un nouveau contrat intelligent est créé chaque fois qu'un utilisateur met une partie de ses avoirs en garantie. Ce SC PoS possède une adresse unique sur le réseau et est déployé sur la machine virtuelle. Les processus de tirage au sort des validateurs et la gestion des récompenses et des punitions sont basés sur les SC PoS.

Selon la chaîne de blocs, le SC PoS comprend les fonctions suivantes :

- gestion de la durée de vie de la mise en garantie ;
- remboursement de la garantie à la demande du propriétaire ;
- gestion des récompenses et punitions du propriétaire.

# 2

## Transposition didactique

---

<b>2.1. BlocNote . . . . .</b>	<b>40</b>
2.1.1. Valeurs . . . . .	40
2.1.2. Modèle . . . . .	42
<b>2.2. Mécanismes de la chaîne de blocs illustrés avec BlocNote . . . . .</b>	<b>48</b>
2.2.1. Exemple d'achat avec cryptomonnaie . . . . .	48
2.2.2. Double-dépense . . . . .	50
2.2.3. S'attribuer la récompense d'un copain . . . . .	50
2.2.4. S'attribuer les jetons d'une transaction dont on n'est pas le destinataire . . . . .	52
2.2.5. Création d'un NFT . . . . .	53
2.2.6. Achat et vente d'un NFT . . . . .	54
<b>2.3. Modalités . . . . .</b>	<b>58</b>
2.3.1. Jeu pédagogique (instructional games) . . . . .	60
2.3.2. Revue des jeux pédagogiques . . . . .	61
2.3.3. Activité déconnectée... ou pas . . . . .	67
<b>2.4. Découpage des activités . . . . .</b>	<b>68</b>

Ce chapitre est consacré à la transposition didactique, soit le passage des savoirs concernant les chaînes de blocs à l'apprentissage par les élèves. La première partie est consacrée à BlocNote, en deuxième partie les mécanismes des chaînes de blocs sont illustrés avec BlocNote, la troisième partie discute des modalités pédagogiques, et finalement, en quatrième partie, la mise en place d'activités pour les élèves est explicitée.

### 2.1. BlocNote

Dans cette section, les éléments exposés au chapitre 2 sont repris, simplifiés et/ou écartés afin d'élaborer un modèle de chaîne de blocs adapté aux élèves de la maturité gymnasiale.

#### 2.1.1. Valeurs

BlocNote est principalement basé sur deux valeurs :

- limitation de l'empreinte écologique
- système décentralisé... démocratique ? méritocratique ?

L'aspiration à la limitation de l'empreinte écologique est renforcée de la plusieurs façons :

- L'utilisation excessive d'énergie pour le calcul de la PoW de Bitcoin (discutée dans la section ??) est inacceptable à notre époque. La première activité de BlocNote illustre le consensus avec la PoW, mais celle-ci peut rapidement être ignorée dès la deuxième activité (voir section 2.4).
- La PoW se veut utile, comme plusieurs auteurs l'ont déjà proposé [71], [16], [45] [36] [13], mais à un niveau adapté à des élèves de la maturité gymnasiale.
- Par définition, la PoS ne demande aucun calcul, elle n'a donc pratiquement pas d'empreinte écologique, et peut être utilisée pour les activités 3, 4 et 5.
- La plupart des activités se déroulent en mode déconnecté, les ordinateurs n'étant utilisés que pour la vérification de la chaîne de blocs. De plus, les programmes et le registre se trouvant sur l'intranet du Collège, BlocNote ne contribue pas au trafic sur le grand internet.

L'aspect démocratique de la chaîne de blocs se manifeste de plusieurs façon :

- au niveau du groupe-classe, les règles peuvent être changées suite à un vote, dont les modalités (majorité, unanimité, deux tiers, proportionnellement au nombre de jetons détenus, ...) peuvent aussi être discutées ;
- au niveau intra-groupe, la prise de décision et la distribution des rôles est gérée par les élèves ;
- l'enseignant agit comme modérateur et source d'information, mais ne valide pas les blocs et transactions, cette responsabilité revenant aux élèves-noeuds ;
- la décentralisation : il n'y a pas de registre de référence, chaque noeud possède sa copie locale ;
- la transparence :
  - la chaîne de bloc est affichée aux yeux de tous et vérifiable par tous ; n'importe quel élève peut recalculer à tout moment les blocs et les soldes des noeuds pour les vérifier ;
  - les programmes de BlocNote sont sous licence de logiciel libre, et le registre est stocké dans un répertoire partagé par tous les élèves du Collège Rousseau (chapitre 4).

Notons que les chaînes de blocs relèvent plus de la méritocratie que de la démocratie, car la prise de décisions ne se fait pas sur la base de "un utilisateur, un vote", mais dépend de la contribution de l'utilisateur, calculée sur la base du nombre de jetons qu'il détient. Ceci peut mener à une réflexion sur l'aspect méritocratique de l'école en général, où le plus performant est le plus reconnu, ce qui peut sembler injuste d'un point de vue démocratique, car les plus faibles sont moins rétribués.

Dans la philosophie de Nakamoto, fondateur de Bitcoin, une chaîne de blocs devrait être publique et transparente par essence [50]. Le fait que quiconque puisse participer à la chaîne de blocs ou se retirer à tout moment est difficile à représenter dans le contexte scolaire, car la présence des élèves est obligatoire, et l'activité débouche sur une note sensée représenter les compétences et la participation des élèves. BlocNote est donc une chaîne de blocs avec permission, mais une fois les utilisateurs identifiés (de par leur simple présence en classe), leurs actions et leur influence sont les mêmes que celles d'une chaîne publique sans permission.

## 2.1.2. Modèle

BlocNote contient les principaux mécanismes et composants communs à la plupart des chaînes de blocs comme Bitcoin et Ethereum. Cette section présente les principes du modèle et la justification des simplifications qui ont été opérées. Notons que tous les éléments de BlocNote peuvent être modifiés par ses utilisateurs, comme dans une vraie chaîne de blocs.

### Grands principes

- toutes les chaînes de blocs commencent par le bloc-genèse (ne contenant aucune transaction ni récompense) ;
- chaque bloc permet de remonter toute la chaîne jusqu'au bloc-genèse ;
- n'importe quel noeud peut enchaîner un bloc à n'importe quel bloc-parent, laissant une liberté totale quant à l'évolution des fourches de la chaîne de blocs ;
- deux algorithmes sont explorés pour la finalisation des blocs :
  - la chaîne se développe librement, et à la fin de l'activité, les blocs formant la chaîne la plus longue sont finalisés. Cette modalité est fidèle au principe de consensus de Bitcoin.
  - si deux blocs B et C sont insérés à la suite d'un bloc A, le bloc A est finalisé, et tous ses cousins deviennent caduques. Cette modalité, proche de celle d'Ethereum, permet de perdre moins d'énergie sur des chaînes stériles et ainsi rendre la chaîne plus dynamique en produisant plus de blocs (et donc de transactions) valides.
- chaque bloc est unique, et son numéro est unique, moyennant la contrainte qu'un noeud donné ne peut enchaîner qu'un seul bloc à un bloc-parent donné (expliqué ci-dessous) ;
- la cryptomonnaie associée à BlocNote est le Rousseau dollars (R\$) ;
- chaque bloc finalisé donne lieu à une récompense de 10 R\$ pour son mineur ou validateur ;
- seuls les blocs faisant partie de la chaîne la plus longue comptent, et donc seulement les récompenses et les transactions contenues dans ces blocs sont effectives, toutes les autres ne sont pas considérées ;
- chaque bloc contient au minimum la transaction de récompense, et possiblement une ou deux transactions "ordinaires" ;
- le mineur ou validateur reçoit aussi en récompense les frais de transaction associés aux transactions incluses dans un bloc finalisé ;
- un bloc contenant une transaction invalide (par exemple si le solde du payeur est insuffisant pour couvrir le montant à payer et les frais de transaction) est rejeté en entier.

### Simplifications

- acteurs : les élèves jouent à la fois le rôle des utilisateurs, qui souhaitent faire passer des transactions, des noeuds, qui surveillent et mettent à jour la chaîne de blocs,

et des mineurs ou validateurs, qui calculent et enchaînent les blocs. Il n'y a pas de plateforme d'échange.

- les adresses des utilisateurs, noeuds et mineurs sont constitués d'un seul chiffre, entre 1 et 9. L'anonymat devrait en principe découler du fait que les élèves sont sensés garder leur numéro secret, mais il sera facile de le deviner de par leurs déplacements dans la salle et ajouts de transactions et de blocs sur le panneau d'affichage. Moyennant une discussion avec les élèves, cette pseudonymité donne l'occasion de discuter de ce qui se passe dans les vraies chaînes de blocs (voir section ??), et les informations disséminées sur les réseaux de façon plus générale.
- les jetons ne sont pas divisibles et aucune punition n'est prévue ;
- aucun élément cryptographique n'est repris (clés privées et publiques, hachage, ...) ;
- le registre n'est ni basé sur les UTXOs, ni sur la tenue des soldes de chaque utilisateur (détails ci-dessous) ;
- les noeuds diffusent les blocs et transactions en les affichant sur un panneau ou sur le mur de la classe, ou via les fichiers .csv sur l'intranet du collège dans le cas où les ordinateurs sont utilisés (voir chapitre 3), sans autre forme de protocole de commérage.

L'enchaînement de blocs, la PoW, le registre, les transactions, le registre, les SCs et NFTs sont détaillés ci-dessous.

## Blocs

Le numéro d'identification d'un nouveau bloc est composé du numéro du noeud mineur ou validateur qui l'a créé et du numéro du bloc-parent qui a été choisi, réunissant de ce fait les informations suivantes :

- le numéro d'identification du noeud mineur ou validateur ;
- le numéro du bloc-parent ;
- sa profondeur dans la chaîne de blocs.

Par exemple, si le noeud mineur ou validateur "4" enchaîne un bloc au bloc-parent 8530, alors le numéro du nouveau bloc sera 48530.

Il est ainsi possible de remonter la chaîne de blocs jusqu'au bloc-genèse simplement avec le numéro du bloc, et ainsi d'identifier rapidement les blocs-parents de n'importe quel bloc. Par exemple, les parents du bloc 48530 sont 8530, 530, 30 et 0, formant la fourche 0-30-530-8530-48530 (voir figure 1.2), à laquelle les mineurs "4", "8", "5" et "3" ont participé. La fourche la plus longue est aussi facilement détectée car elle correspond simplement à celle contenant le bloc avec le numéro le plus long.

Cependant, cette numérotation des blocs a pour conséquence qu'un noeud donné ne peut enchaîner qu'un seul bloc à un bloc-parent donné. Par exemple, les noeuds 3, 4 et 5 peuvent produire les blocs 340, 440 et 540 à la suite du bloc 40, mais le noeud 4 ne peut pas enchaîner deux blocs au bloc 40, car ils auraient tous les deux le même numéro d'identification 440.

## PoW

L'explication de la PoW est l'un des objectifs de BlocNote, mais pas le principal, comme expliqué précédemment. La PoW est donc illustrée de façon relativement simple : elle consiste à calculer la notation binaire du numéro du nouveau bloc. Ceci permet d'illustrer deux aspects de la chaîne de blocs : d'une part, la difficulté augmente au fur et à mesure que la chaîne grandit, d'autre part, ce calcul rend la tricherie moins payante que de suivre les règles, comme expliqué à la section 1.3.4.

Le calcul de la notation binaire se veut aussi un rappel des cours d'informatique de première année, mais pourrait être remplacé par n'importe quel calcul déterministe. L'avantage du calcul de la notation binaire est qu'il est relativement difficile sur papier, mais presque instantané avec une calculatrice, un site web ou un interpréteur Python, donc difficile à calculer pour les élèves-noeuds, mais facile à vérifier par l'enseignant.

Les noeuds-élèves pouvant choisir à quel bloc enchaîner un nouveau bloc, l'évolution de la chaîne de blocs reste imprévisible, il n'est donc en principe pas possible de calculer les cibles de la PoW d'avance. Cependant, connaissant les numéros des noeuds, des élèves futés pourraient dresser une table de tous les numéros de blocs possibles et leurs PoW correspondantes. De plus, le calcul est plus facile selon le numéro du noeud (par exemple,  $10_{10} = 1010_2$  et  $90_{10} = 101000_2$ ). Il s'agit de faiblesses du modèle, mais la PoW n'étant pas le but principal de BlocNote, ce choix permet de passer aux autres objectifs sans trop insister sur le mécanisme de la PoW.

## Transaction

Les transactions sont représentées par un nombre comportant cinq chiffres : le numéro du noeud payeur, le numéro du noeud destinataire, le montant de la transaction (entre 1 et 99 R\$ inclus) et les frais de transaction (entre 0 et 9 R\$ inclus). Par exemple, la transaction 45053, notée "de4\_à5\_05\_3" dans les exemples du chapitre 2, décrit une transaction où l'utilisateur "4" transfère 5 R\$ à l'utilisateur "5" et 3 R\$ de frais de transaction au noeud mineur ou validateur qui insère la transaction dans un bloc. Ceci résume l'essentiel des transactions ayant lieu dans les chaînes de blocs, mais aussi dans le secteur bancaire du monde réel.

Les transactions des SC NFT et des SC PoS sont expliquées plus loin.

## Registre

Le registre de BlocNote n'est basé ni sur les UTXOs, ni sur les soldes des utilisateurs (voir section 1.4.3). Le choix s'est porté sur un registre ne contenant que les transactions, pour forcer le calcul du solde en dehors de la chaîne de blocs, tout en évitant la complexité des UTXO. Il s'agit aussi d'une opportunité pour les élèves de découvrir et expérimenter un fonctionnement différent de celui avec lequel ils sont familiers : pour vérifier un bloc ou insérer une transaction dans un bloc, ils doivent d'abord analyser la fourche concernée afin de calculer le solde du payeur pour s'assurer que les fonds sont suffisants pour couvrir le montant de la transaction et les frais de transaction.

Le tableau 1.1 montre le registre de BlocNote tel que visible lors des activités-élèves, et le tableau 1.22 montre le même registre, mais cette fois avec les soldes des comptes des utilisateurs, qui sont calculés à l'extérieur de BlocNote, par les élèves.

## Ordre chronologique vs logique

Pour calculer le solde d'un utilisateur ou vérifier la validité d'une transaction, il n'est pas nécessaire de tenir le compte des jetons de la chaîne en entier, mais que des transactions contenues dans les blocs-parents du bloc contenant la transaction concernée. En effet, seule la fourche la plus longue sera retenue, les blocs contenant les jetons "dépensés" dans d'autres fourches ne seront pas tenus en compte.

Le registre montre les blocs en ordre chronologique d'insertion dans la chaîne, mais pas l'affiliation aux blocs-parents. Le tableau 2.1 montre les mêmes données que celles des tables 1.1 et 1.22 mentionnées plus haut, mais réorganisées par fourches ; ce travail doit être effectué par les noeuds à l'extérieur de la chaîne afin de calculer le solde d'un utilisateur dans une fourche donnée.

TABLE 2.1. – Registre réorganisé par fourche et avec le solde des noeuds après chaque transaction.

BlocID	PoW	Tx1	Tx2	Tx3	Solde
0	0	0	0	0	{4: 0, 5: 0, 8: 0, 9: 0}
40	0b101000	04100	0	0	{4: 10, 5: 0, 8: 0, 9: 0}
940	0b1110101100	09100	48081	0	{4: 1, 5: 0, 8: 8, 9: 11}
8940	0b10001011101100	08100	84030	94050	{4: 9, 5: 0, 8: 15, 9: 6}
88940	0b1010110110110100	08100	48071	98031	{4: 1, 5: 0, 8: 37, 9: 2}
0	0	0	0	0	{4: 0, 5: 0, 8: 0, 9: 0}
90	0b1011010	9100	0	0	{4: 0, 5: 0, 8: 0, 9: 10}
890	0b1101111010	8100	94041	94011	{4: 5, 5: 0, 8: 12, 9: 3}
0	0	0	0	0	{4: 0, 5: 0, 8: 0, 9: 0}
90	0b1011010	09100	0	0	{4: 0, 5: 0, 8: 0, 9: 10}
590	0b1001001110	05100	98071	95010	{4: 0, 5: 12, 8: 7, 9: 1}
5590	0b1010111010110	5100	59021	59011	{4: 0, 5: 19, 8: 7, 9: 4}
0	0	0	0	0	{4: 0, 5: 0, 8: 0, 9: 0}
40	0b101000	04100	0	0	{4: 10, 5: 0, 8: 0, 9: 0}
440	0b110111000	04100	0	0	{4: 20, 5: 0, 8: 0, 9: 0}
5440	0b1010101000000	05100	45040	0	{4: 16, 5: 14, 8: 0, 9: 0}
0	0	0	0	0	{4: 0, 5: 0, 8: 0, 9: 0}
40	0b101000	04100	0	0	{4: 10, 5: 0, 8: 0, 9: 0}
440	0b110111000	04100	0	0	{4: 20, 5: 0, 8: 0, 9: 0}
4440	0b1000101011000	04100	49011	45010	{4: 28, 5: 1, 8: 0, 9: 1}

## Attaques

BlocNote est résilient aux erreurs de crash (crash fault), involontaires (syntaxe, erreur de calcul, ...) et volontaires (altération d'un bloc ou d'une transaction), pour autant que les noeuds-élèves restent vigilants. Le système de récompense en R\$ devrait le prévenir des attaques malveillantes. Deux exemples sont illustrés aux sections 2.2.3 et 2.2.4.

L'acceptation des noeuds étant régulée par la présence en classe pour les activités déconnectées et par l'accès à l'intranet pour les programmes Python, les attaques DDoS et par leurre d'identité ne devraient pas avoir lieu, à moins que les élèves ne soient vraiment futés.

Sans cryptographie, il n'existe pas de moyen de vérifier si un noeud malveillant a introduit une transaction utilisant les jetons d'un autre noeud, car les transactions ne sont pas signées. Dans le vrai monde, les signatures numériques garantissent que le propriétaire d'un jeton a bien autorisé son transfert. BlocNote est donc particulièrement vulnérable au vol d'identité.

Des noeuds pourraient comploter et tenter une attaque à 51%, ce qui offrirait une occasion de discuter de la vulnérabilité des chaînes de blocs et les façons dont les noeuds honnêtes peuvent renverser les attaquants. Un exemple de double-dépense est discuté dans la section 2.2.2.

## SC NFT

Dans BlocNote, les contrats intelligents de création de NFTs et de PoS sont déjà mis en place dans la machine virtuelle (... qui consiste en un tableau d'affichage), et les utilisateurs font appel à eux pour créer des NFT et des PoS par le biais de transactions spéciales.

Les NFTs représentent des œuvres physiques ou digitales :

- les œuvres physiques sont exposées sur un mur, chacune avec une affiche représentant le SC NFT qui lui est associé.
- Les œuvres digitales sont stockées dans un répertoire protégé sur l'intranet du Collège Rousseau, et sont visibles grâce à un fichier HTML basique, par le biais d'un navigateur internet. Seuls les utilisateurs de l'intranet ont accès aux œuvres digitales.

Chaque SC NFT associé à une œuvre physique ou digitale est transcrit sur une feuille A4<sup>1</sup>, qui est ensuite affichée sur le tableau d'affichage appelé "machine virtuelle".

Le gaz pour la création d'un NFT est fixé à 5 R\$ ; le processus est illustré dans la section 2.2.5.

L'achat d'un NFT se fait en deux étapes : l'utilisateur paye le montant de la vente au SCNFT, qui ensuite émet une transaction pour transférer le montant au vendeur du NFT. Le mécanisme est illustré dans la section 2.2.6.

## SC PoS

Dans BlocNote, un seul SC tient le registre des mises en garantie, et la création d'une PoS ne comporte pas de gaz. Chaque jeton mis en garantie donne droit à une ballotte<sup>2</sup> pour le tirage au sort du prochain validateur.

---

<sup>1</sup>Dans l'esprit de réduction de l'empreinte écologique, les feuilles A4 sont issues des photocopies ratées de la benne de recyclage de la salle des maîtres.

<sup>2</sup>Le système des ballottes, boules de tirages au sort, employées à Venise pour l'élection de son doge, a donné en français le verbe ballotter et le substantif ballottage dont il est issu [9].

Par exemple, un utilisateur souhaite mettre une partie de ses avoirs en garantie. Il soumet la transaction "de4\_àSCPoS\_12\_3", qui signifie que l'utilisateur "4" met en garantie 12 R\$ et paye 3 R\$ de frais de transaction au mineur. Il obtient de cette façon 12 ballottes lors des tirages au sort.

Le remboursement de la PoS se fait en deux étapes : l'utilisateur soumet une transaction au SC PoS avec le montant négatif correspondant à sa PoS, et le SC PoS soumet automatiquement une transaction pour rembourser l'utilisateur :

1. l'utilisateur souhaitant retrouver sa garantie émet la transaction "de9\_àSCPoS\_-12\_2", qui signifie que l'utilisateur "9" demande le remboursement des 12 R\$ de sa PoS, et paye 2 R\$ de frais de transaction au mineur,
2. le SCPoS émet automatiquement la transaction "deSCPoS\_à9\_12\_0", qui signifie que le SC PoS rembourse les 12 R\$ à l'utilisateur "9" et paye 0 R\$ de frais de transaction au mineur. L'utilisateur perd alors 12 ballottes dans le tirage au sort.

Pour simplifier le mécanisme, aucun frais de transaction ne sont prévus pour le transfert des R\$ d'un SC à un utilisateur.

## 2.2. Mécanismes de la chaîne de blocs illustrés avec BlocNote

Dans cette section, les principaux mécanismes de la chaîne de blocs sont illustrés avec BlocNote.

### 2.2.1. Exemple d'achat avec cryptomonnaie

Afin d'aider le lecteur à former un modèle mental de l'utilisation de la cryptomonnaie dans le vrai monde, la situation où un utilisateur vend un objet, et un autre utilisateur souhaite l'acheter, est illustrée dans les figures 2.1, 2.2 et 2.6 :

- l'acheteur insère une transaction signifiant qu'il souhaite transférer une partie de ses jetons à l'adresse du vendeur, le plus souvent par le biais d'une plateforme d'échange (figure 2.1) ;

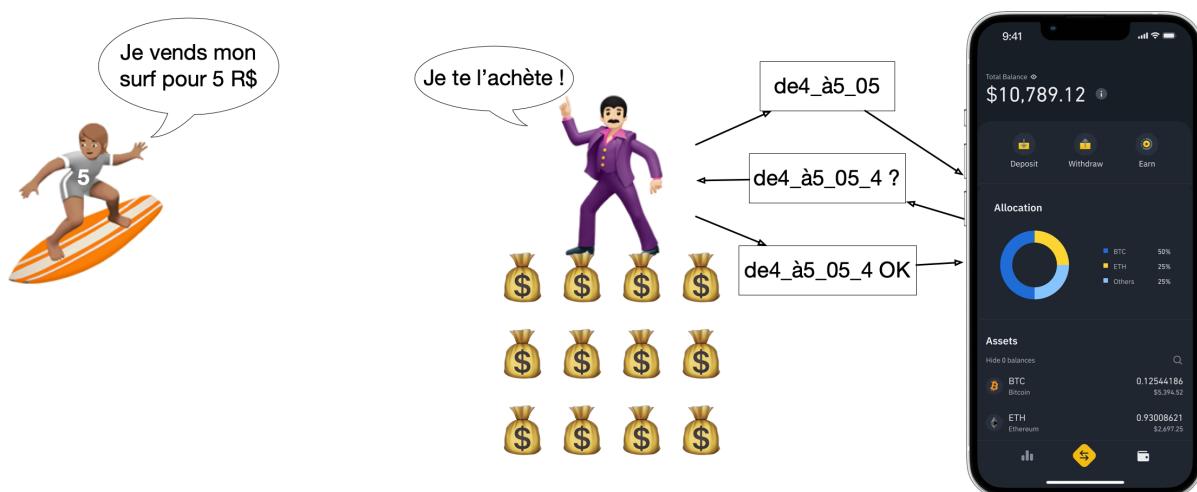


FIGURE 2.1. – La personne à l'adresse "5" vend son surf pour 5 R\$, et la personne à l'adresse "4" est intéressée à l'acheter. La personne "4" insère la transaction dans la plateforme d'échange de son choix, et la plateforme suggère d'ajouter 4 R\$ de frais de transaction. La personne "4" accepte.

- la plateforme d'échange soumet la transaction sur le réseau de la chaîne de blocs (figure 2.2) ;
- un noeud-mineur ou un noeud-validateur inclue la transaction dans un bloc, puis insère le bloc dans la chaîne de blocs (figure 2.6) ;
- si le bloc fait partie de la chaîne la plus longue, la transaction est finalisée (figure 2.6).

Si tout va bien, l'utilisateur-vendeur et l'utilisateur-acheteur respectent les conditions de la transaction, et l'acheteur récupère le bien mis en vente. En effet, la chaîne de bloc garantit le transfert des jetons de l'acheteur au vendeur, mais ne garantit pas l'exécution de la transaction dans le vrai monde.

Notons que si les utilisateurs étaient des initiés, ils auraient pu se passer des services des plateformes d'échange, du noeud et du mineur ou validateur, et ainsi insérer la transaction

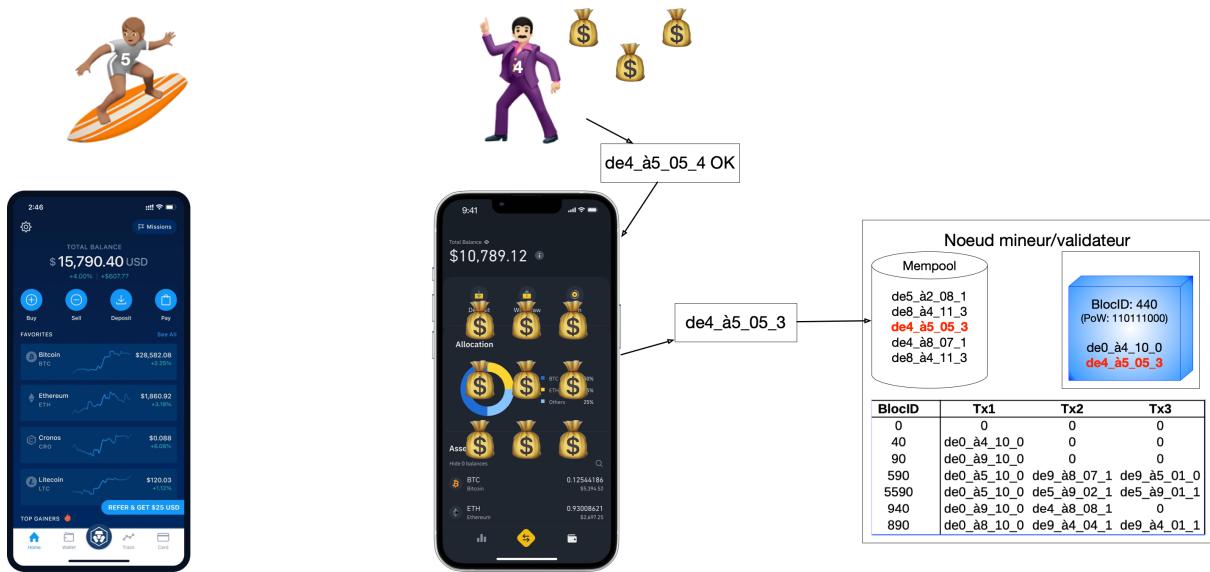


FIGURE 2.2. – La plateforme d'échange de la personne "4" gèle 9 R\$ de son compte, et soumet la transaction "de4\_à5\_05\_3" ("l'utilisateur "4" transfère 5 R\$ à l'utilisateur "5" et 3 R\$ de frais de transaction au noeud mineur ou validateur qui insère la transaction) au réseau de la chaîne de blocs. La plateforme d'échange encaisse 1 R\$ de commission au passage.



FIGURE 2.3. – Un mineur a inséré la transaction "de4\_à5\_05\_3" dans un bloc, qui a été finalisée : la transaction est donc exécutée. Le noeud-mineur empoche les frais de transaction de 3 R\$, la plateforme d'échange de l'acheteur garde 1 R \$ de commission, et la plateforme d'échange du vendeur reçoit 5 R\$. Cette dernière transfère 4 R\$ à la personne "5" et s'attribue une commission de 1 R\$.

directement dans la chaîne de blocs sans devoir payer les commissions impliquées dans l'utilisation des services.

### 2.2.2. Double-dépense

Pour reprendre l'exemple de la section 2.2.1, suite à la transaction "de4\_à5\_05\_3" insérée dans le bloc 440, l'utilisateur "5" donne son surf à l'utilisateur "4" dans le vrai monde.



FIGURE 2.4. – L'utilisateur 4 a reçu le surf, la transaction ayant été confirmée dans le bloc 440, puis finalisée grâce à l'ajout des blocs 4440 et 74440.

L'utilisateur "4" soumet ensuite une autre transaction "de4\_à6\_05\_3" qui utilise les mêmes jetons, cette fois pour acheter un skateboard à l'utilisateur "6". Il insère cette transaction dans le bloc 4890, qui est enchaîné dans une autre fourche que le bloc 440 où se trouve la première transaction. L'utilisateur "4" enchaîne des blocs à la suite du bloc 4890 afin que cette fourche devienne plus longue que la fourche (ou les fourches) contenant le bloc 440, les diffuse sur le réseau, et comme cette fourche est la plus longue, les autres noeuds l'adoptent.

La transaction "de4\_à5\_05\_3" reste confirmée, mais n'est plus finalisée. L'utilisateur "5" a déjà donné son surf à l'utilisateur "4", mais ne pourra pas utiliser les jetons qu'il avait reçu par la transaction "de4\_à5\_05\_3". Au final, l'utilisateur "4" aura dépensé deux fois les mêmes jetons.

### 2.2.3. S'attribuer la récompense d'un copain

Cet exemple montre le cas où un noeud malveillant tente de modifier la numérotation d'un bloc pour s'approprier la récompense du minage :

1. au départ, tous les blocs de la chaîne sont valides ;
2. le noeud malveillant change le numéro du bloc pour s'approprier la récompense du minage, mais il doit aussi modifier la PoW pour que le bloc soit valide (figure 2.7) ;



FIGURE 2.5. – L'utilisateur soumet la transaction "de4\_à6\_05\_3" pour acheter le ballon de l'utilisateur 6 en utilisant les mêmes jetons que pour la transaction précédente. La transaction est confirmée dans le bloc 4890, puis finalisée grâce à l'ajout des blocs 44890 et 444890.



FIGURE 2.6. – L'utilisateur 4 a reçu le surf et le ballon en dépensant deux fois les mêmes jetons.

3. le noeud malveillant doit changer les numéros des blocs subséquents au bloc qu'il a altéré, car chaque bloc contient un hachage des données de son bloc-parent (figure 2.8) ;

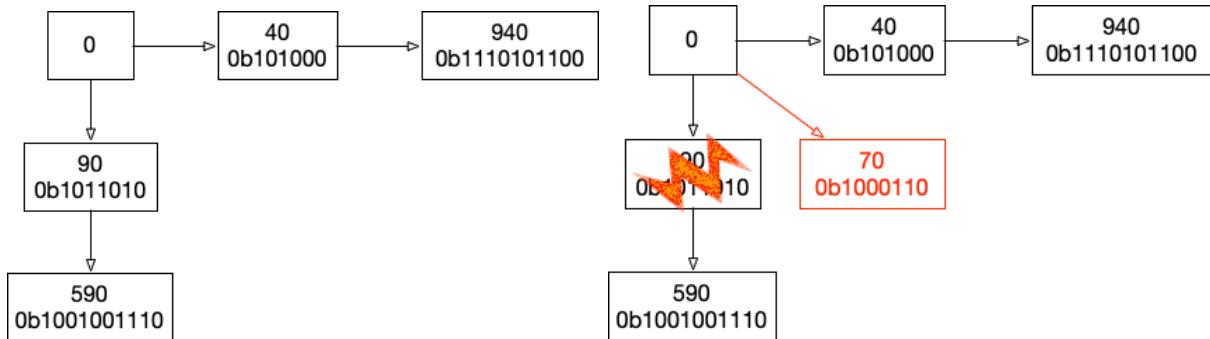


FIGURE 2.7. – A gauche : la chaîne de blocs au moment où le noeud 7 tente de l'altérer.  
A droite : le noeud 7 modifie la numérotation du bloc 90 et recalcule la PoW pour qu'elle corresponde au numéro de bloc 70.

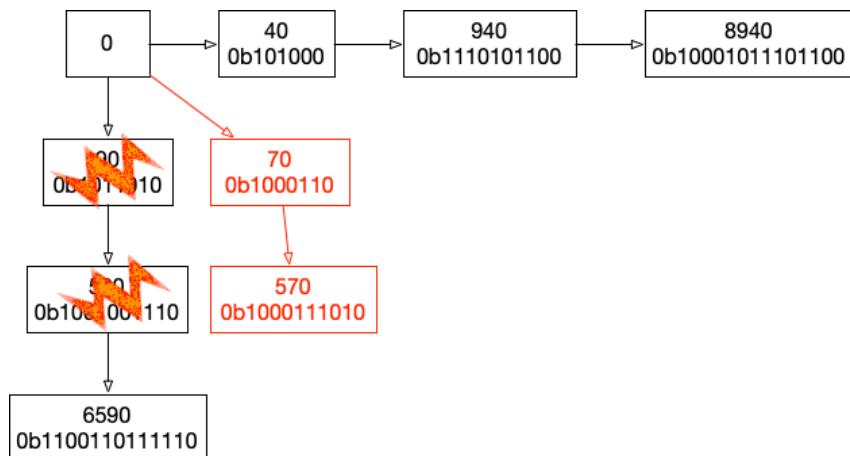


FIGURE 2.8. – Le noeud malveillant doit altérer les blocs subséquents. Les chaînes 0-90-590-6590 et 0-40-940-8940 grandissent plus rapidement que la chaîne modifiée 0-70-570.

Il est donc aussi coûteux de tenter d'altérer un bloc pour s'approprier la récompense que de miner un nouveau bloc, d'autant plus que les fourches dont ne fait pas partie le bloc altéré grandissent plus vite que la fourche où se trouve le bloc.

#### 2.2.4. S'attribuer les jetons d'une transaction dont on n'est pas le destinataire

Cet exemple illustre le cas où le noeud malveillant altère une transaction afin de s'attribuer les jetons destinés à un autre noeud :

- au départ, tous les blocs de la chaîne sont valides ;

2. le noeud malveillant modifie une transaction pour s'attribuer les jetons ; dans l'exemple, le noeud 4 tente de s'attribuer les jetons que le noeud 8 souhaite transférer au noeud 7 dans le bloc 55483880 ; il modifie donc la transaction "de8\_à7\_12\_3" en "de8\_à4\_12\_3" (table 2.2 de droite).
3. Le noeud malveillant diffuse le bloc contenant la transaction modifiée auprès de ses pairs.
4. Suite à la modification effectuée par le noeud malveillant, la transaction "de7\_à4\_09\_1" inclue dans le bloc subséquent 355483880 devient invalide, car le solde du noeud 7 est maintenant négatif (table 2.3 de gauche).
5. Tous les blocs subséquents au bloc dans lequel le noeud malicieux a modifié une transaction sont invalidés (table 2.3 de droite).
6. Les noeuds vigilants détectent les transactions invalides, n'incluent pas ce bloc et abandonnent cette fourche.

Par conséquent, la transaction altérée par le noeud malicieux ne sera jamais finalisée, et donc jamais exécutée.

BlocID	Tx1	Tx2	3	4	5	7	8		BlocID	Tx1	Tx2	3	4	5	7	8
0									0							
.....									.....							
483880	de0_à4_10_0		10	10	0	0	30		483880	de0_à4_10_0		10	10	0	0	30
5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30		5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30
55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15		55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15
355483880	de0_à3_10_0	de7_à4_09_1	21	13	24	7	15		355483880	de0_à3_10_0	de7_à4_09_1	21	13	24	5	15
7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2		7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2

TABLE 2.2. – A gauche : la chaîne de blocs au moment où le noeud 4 tente de l'altérer.  
A droite : le noeud 4 tente de s'attribuer les jetons que le noeud 8 souhaite transférer au noeud 7.

BlocID	Tx1	Tx2	3	4	5	7	8		BlocID	Tx1	Tx2	3	4	5	7	8
0									0							
.....									.....							
483880	de0_à4_10_0		10	10	0	0	30		483880	de0_à4_10_0		10	10	0	0	30
5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30		5483880	de0_à5_10_0	de4_à7_05_1	10	4	11	5	30
55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15		55483880	de0_à5_10_0	de8_à7_12_3	10	4	24	17	15
355483880	de0_à3_10_0	de8_à4_12_3	10	16	24	5	15		355483880	de0_à3_10_0	de8_à4_12_3	10	16	24	5	15
355483880	de0_à3_10_0	de7_à4_09_1	21	25	24	-5	15		355483880	de0_à3_10_0	de7_à4_09_1	21	25	24	-5	15
7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2		7355483880	de0_à7_10_0	de8_à5_11_2	21	13	35	19	2

TABLE 2.3. – A gauche : la transaction "de7\_à4\_09\_1" inclue dans le bloc 355483880 devient invalide. A droite : tous les blocs subséquents sont invalidés.

## 2.2.5. Création d'un NFT

Le processus de création d'un NFT est illustré avec l'exemple suivant : l'utilisateur associé à l'adresse "4" a réussi à insérer un bloc dans la chaîne de blocs, et a donc gagné 10 R\$. Il en profite pour créer un NFT contenant sa dernière création artistique pour la mettre en vente, qu'il estime valoir au minimum 12 R\$.

Pour ce faire,

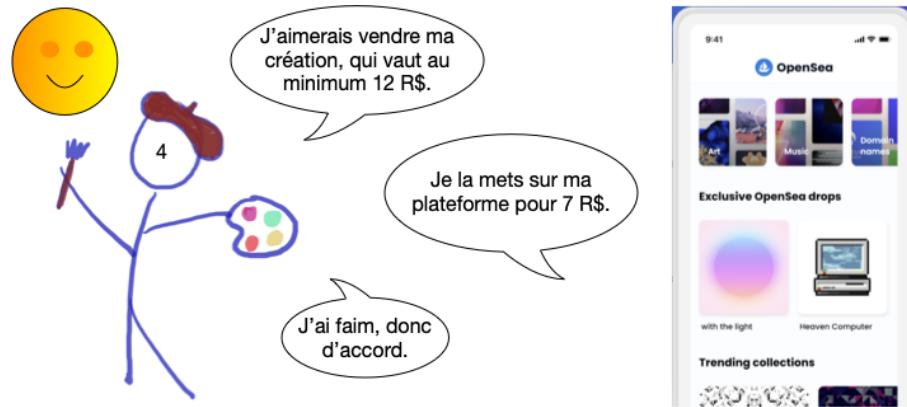


FIGURE 2.9. – L'utilisateur associé à l'adresse "4" souhaite créer un NFT.

1. l'utilisateur "4" doit payer un gaz de 5 R\$ pour faire tourner le SC ERC-721, et 1 R\$ pour que sa transaction soit choisie par un validateur. Il soumet donc la transaction "de4\_àSC0721\_05\_1" dont le destinataire est SC ERC-721 (figure 2.9). L'URL où se trouve l'oeuvre digitale et son prix de vente sont inclus dans les métadonnées de la transaction.
2. La transaction est diffusée dans le réseau de la chaîne de blocs, et éventuellement insérée dans un bloc (voir figure 2.10).
3. Si le bloc fait partie de la chaîne la plus longue, la transaction est finalisée et le SC ERC-271 exécuté. Un nouveau SC NFT contenant les métadonnées (l'URL et le prix de vente) est alors ajouté à l'EVM (voir figure 2.11).

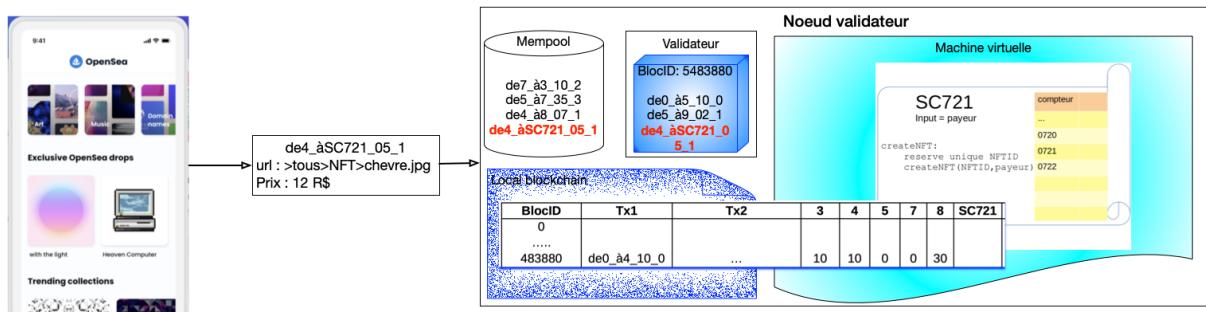


FIGURE 2.10. – La transaction "de4\_àSC0721\_05\_1" est diffusée sur le réseau. Le validateur associé à l'adresse "5" choisit cette transaction pour l'insérer dans le prochain bloc, dont le numéro est 5483880.

## 2.2.6. Achat et vente d'un NFT

Une fois le SC NFT créé, ce même SC est utilisé pour toutes les transactions de vente et d'achat du NFT. Pour simplifier le mécanisme, aucun frais de transaction ne sont prévus pour le transfert des R\$ du SC NFT au vendeur du NFT.

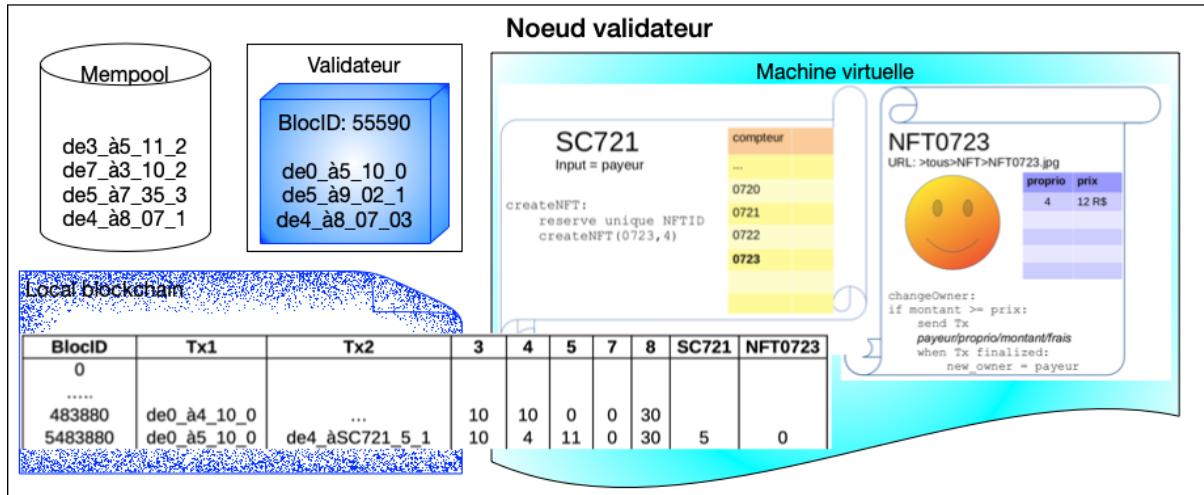


FIGURE 2.11. – Le bloc 5483880 a été finalisé, la transaction "de4\_àSC0721\_5\_1" est exécutée : le SC721 crée un nouveau SC "NFT0723" dont le propriétaire est l'utilisateur "4" et le prix de vente est de 12 R\$.

Par exemple, l'utilisateur associé à l'adresse "8" souhaite acheter le NFT0723 dont le prix a été fixé à 12 R\$ par son propriétaire actuel, l'utilisateur "4" :

1. L'utilisateur associé à l'adresse "8" consulte sa plateforme d'achat et vente de NFTs, qui lui annonce un prix de vente de 12 R\$ et 3 R\$ de frais pour faire passer la transaction.
2. L'acheteur émet la transaction "de8\_àNFT0723\_12\_3", qui signifie que l'utilisateur "8" paye 12 R\$ pour acheter le NFT, et 3 R\$ de frais de transaction au mineur ou validateur. Les métadonnées contiennent le prix de revente du NFT, et paye le prix du NFT, les frais de transaction et le gaz pour l'exécution du SC NFT (voir figure 2.12).

*Note :* pour simplifier, le gaz nécessaire à l'exécution du SC NFT est inclus dans les frais de transaction.

3. Une fois la transaction finalisée (figure 2.13), l'exécution du SC NFT est déclenchée (figure 2.14) :
  - a) les jetons de l'acheteur sont transférés à l'adresse du SC NFT ;
  - b) le SC NFT envoie automatiquement une nouvelle transaction pour que les jetons soient transférés au vendeur, soit la transaction "deNFT0723\_à4\_12\_0", qui signifie que le SC NFT transfère les 12 R\$ au vendeur du NFT et paye 0 R\$ de frais de transaction au mineur ou validateur ;
  - c) une fois cette deuxième transaction finalisée, l'adresse du propriétaire et le prix de vente du NFT sont incrémentés (ceux du propriétaire précédent ne sont pas effacés).



FIGURE 2.12. – L'utilisateur associé à l'adresse "8" souhaite acheter le NFT0723.

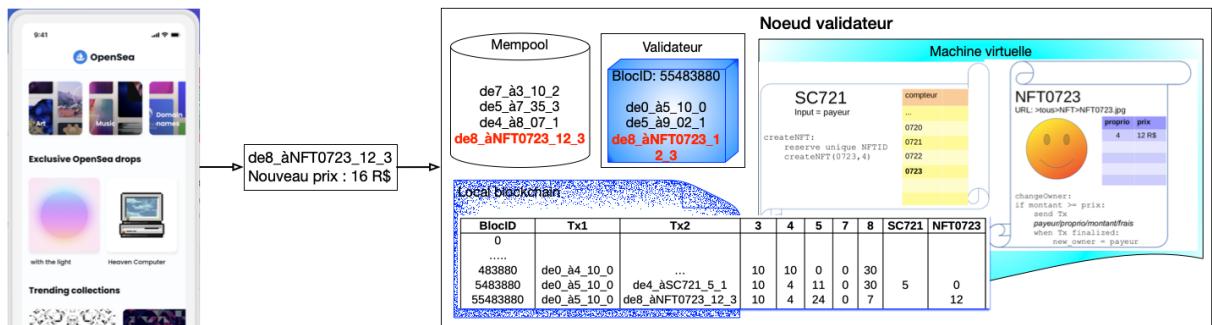


FIGURE 2.13. – La transaction "de8\_àNFT0723\_12\_3" est envoyée, puis choisie par le validateur "5" pour l'insérer dans le bloc 55483880.

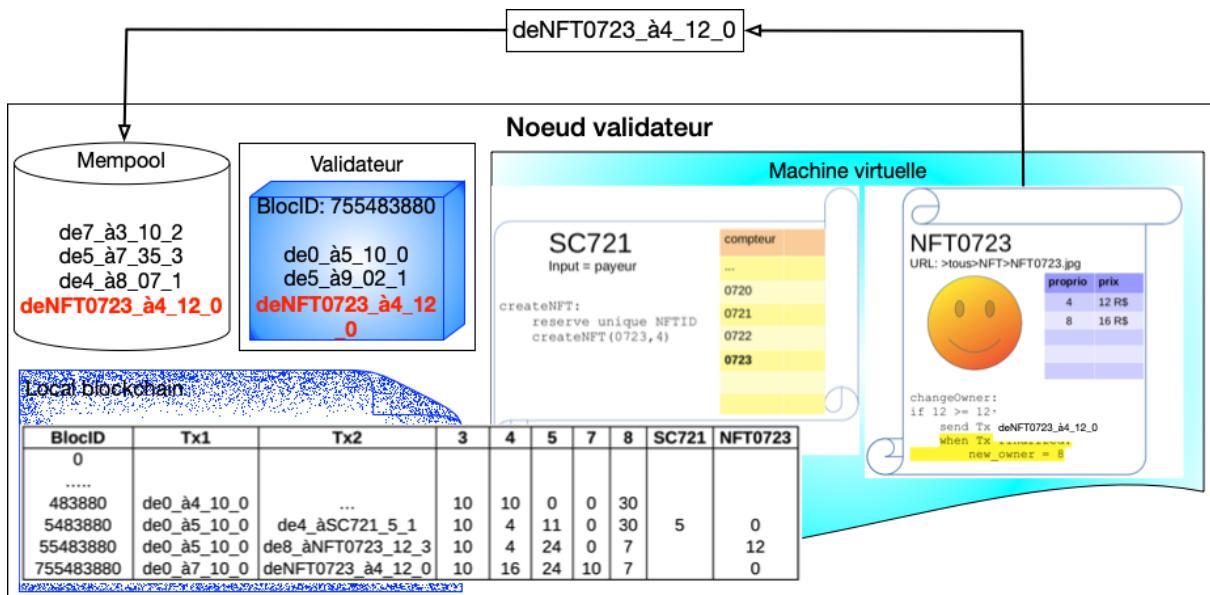


FIGURE 2.14. – Le bloc 55483880 a été finalisé, la transaction "de8\_àNFT0723\_12\_3" est exécutée : le NFT0723 affiche maintenant le nouveau propriétaire ("8") et le nouveau prix de vente (16 R\$), et envoie la transaction "deNFT0723\_à4\_12\_0" pour payer les 12 R\$ de la vente à l'utilisateur "4". Le validateur "7" choisit cette transaction et l'insère dans le bloc 755483880.

## 2.3. Modalités

Afin d'engager les élèves dans leurs apprentissages et encourager leur participation active, les activités sont présentées sous la forme d'un jeu pédagogique, en grande partie déconnecté. Ce choix se base sur l'hypothèse que le jeu est une façon naturelle de rendre les apprenants actifs, et une alternative plus engageante que les cours magistraux classiques. Comme nous l'avons vu en introduction, le CDIP et le programme d'étude du Collège de Genève, ainsi que le Plan d'études Romand (PER), vont dans ce sens. On le retrouve dans les capacités transversales que sont la collaboration, la communication, la démarche réflexive et le sens critique, la pensée créatrice, les stratégies et la réflexion métacognitive. BlocNote vise aussi à tenir compte des obstacles usuels d'ordre pédagogique : le niveau de difficulté est variable selon les activités, qui sont modulables selon les discussions en classe. Ce dernier aspect devrait aider à la motivation des élèves, qui contrôlent les règles du jeu. L'aspect intrinsèquement social des chaînes de blocs devrait aussi contribuer à leur motivation.

### Contexte

Dans l'esprit des objectifs pédagogiques discutés dans la section ??, le Collège de Genève a prévu le cours 3INDF (cours d'informatique de troisième année, discipline fondamentale) sous la forme d'une semaine décloisonnée transdisciplinaire : les élèves travaillent en groupes de quatre, pendant une semaine intensive (et non pas avec une dotation horaire hebdomadaire), sur un projet croissant l'informatique avec une deuxième discipline.

Selon le CDIP [69] : "certains contenus disciplinaires ne peuvent d'ailleurs pas être traités (ou du moins pas efficacement) en dehors du numérique, d'où la nécessité d'envisager de nouvelles possibilités d'enseignement". Ce cours fournit justement les conditions favorables pour aborder un système informatique vaste et tentaculaire comme les chaînes de blocs.

La figure 2.15 illustre la configuration imaginée pour le déroulement des activités BlocNote dans le cadre du cours 3INDF, soit des groupes de quatre élèves jouant le rôle des noeuds. Lors des activités BlocNote, les élèves sont appelés à afficher eux-mêmes les blocs et transactions sur le tableau d'affichage, encourageant de ce fait les déplacements physiques dans la salle. L'influence positive de l'aspect kinesthésique<sup>3</sup> des activités d'apprentissage a surtout été étudié et développé au niveau primaire, mais reste certainement valable aux niveaux supérieurs. Ces déplacements permettent aussi de mettre en évidence la latence entre le calcul et la diffusion de nouveaux blocs.

### Travail de groupe

La chaîne de blocs de BlocNote peut être travaillée individuellement, ce qui peut s'avérer utile pour un enseignant qui s'approprie l'activité ou un élève qui cherche à mieux comprendre ou approfondir sa compréhension. Cependant, pour vivre une expérience plus proche des chaînes de bloc du vrai monde et comprendre le rôle du protocole de consensus, le travail de groupe devient nécessaire. Cette modalité est d'ailleurs rendue obligatoires par les prescriptions du programme.

<sup>3</sup>Kinesthésie : perception des déplacements des différentes parties du corps.

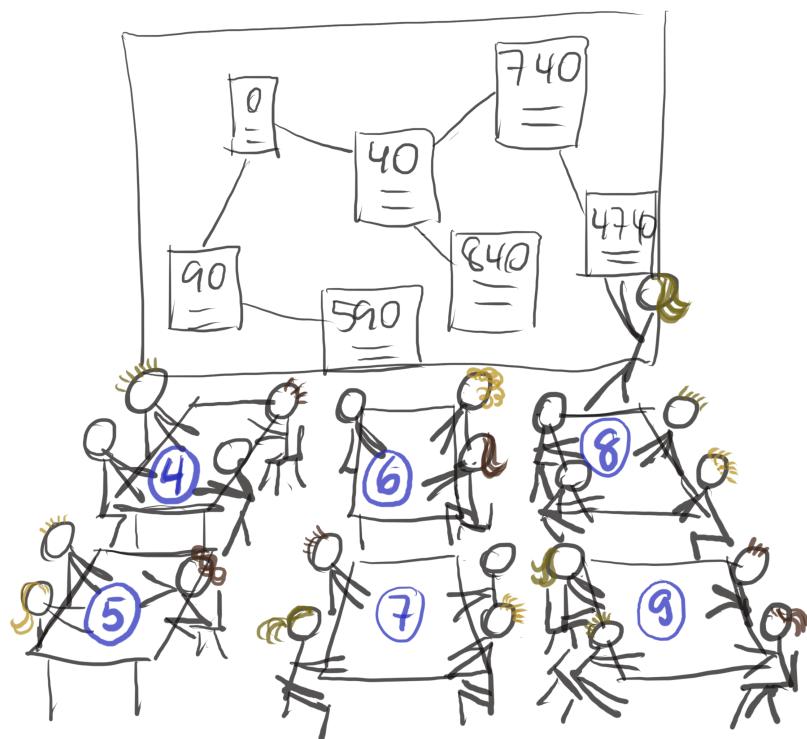


FIGURE 2.15. – Dans la mise en oeuvre des activités, des groupes de quatre élèves forment les noeuds, et les blocs et les transactions qu'ils contiennent sont affichés au mur.

Le travail de groupe n'est pas toujours facile ni bien vécu par les élèves, mais il constitue un apprentissage nécessaire pour les études supérieures, le monde du travail et la vie en société en général. Il est de nature sociale, ce qui revêt des avantages et des inconvénients dont il faut tenir compte pour s'assurer de l'efficacité des apprentissages. Dans son livre "Faire apprendre les sciences et la technologie à l'école", Potvin [58] consacre un chapitre à la thématique du travail de groupe ; deux aspects étroitement liés à ce travail sont résumés ci-dessous :

- les apports conceptuels, ingénieux, d'attitude et les compétences diverses de chaque membre du groupe profitent aux autres membres du groupe, et dans de bonnes conditions, chaque membre est valorisé et motivé de par sa contribution au travail de groupe ;
- les élèves peuvent se convaincre entre eux d'une idée fausse ("contamination mutuelle d'idées scientifiquement fausses"), et tomber dans un "délire" qu'il est difficile de rattraper.

Le retrait de l'enseignant est donc souhaitable pour laisser le plus de place possible à la créativité et la construction des connaissances par les élèves eux-mêmes, mais il doit rester suffisamment présent pour s'assurer que les élèves ne se perdent pas en chemin.

### 2.3.1. Jeu pédagogique (instructional games)

Dans son rapport technique, Hays [38] a analysé 274 documents concernant la conception, l'utilisation et l'évaluation de jeux. Parmi ces documents, 169 n'ont pas été utilisés pour l'analyse finale, la plupart ne reflétant que l'opinion de l'auteur ou comportant des failles méthodologiques majeures. Selon Hays, la recherche sur l'efficacité des jeux pédagogiques est fragmentée, remplie de termes mal définis et contaminée par des lacunes méthodologiques. L'auteure de ce travail a pu confirmer les observations de Hays lors de ses recherches : de nombreux articles portent sur les jeux pédagogiques, surtout dans le monde anglophone, mais la plupart raconte une activité ou l'opinion de l'auteur, sans analyse basée sur des expérimentations calibrées, et leurs références tournaient en rond pour éventuellement se rapporter au rapport de Hays [38], qui date de 2005...

BlocNote revêt les caractéristiques d'un jeu pédagogique telles que définies par Hays [38] :

- chaque activité présente un défi : enchaîner le plus de blocs possible, effectuer le plus de transactions possible, vendre/acheter le plus de NFTs possible, etc, ainsi que des défis intellectuels : comprendre les mécanismes de l'enchaînement des blocs, de la validation transactions, des contrats intelligents, etc.
- les élèves jouent les rôles des mineurs, noeuds, utilisateurs, créateurs de NFTs, validateurs (PoS), simultanément et/ou en parallèle, mais en tout cas en coordination ;
- les élèves collaborent au sein d'un noeud, les noeuds sont en compétition, mais doivent collaborer pour avancer.

Les conclusions et recommandations de Hays sont les suivantes :

- ne pas généraliser l'efficacité d'un jeu pédagogique dans un domaine d'apprentissage avec un groupe d'apprenants à tous les jeux dans tous les domaines pour tous les apprenants ;
- il n'existe aucune preuve qui indique que le jeu pédagogique est la méthode pédagogique à privilégier dans toutes les situations ;

- les jeux pédagogiques devraient s'inscrire dans un programme pédagogique complet qui inclut le résumé (debriefing) et un retour de façon à ce que les apprenants comprennent ce qui s'est passé pendant le jeu, et comment ces événements renforcent les objectifs pédagogiques ;
- le soutien pédagogique du jeu doit contribuer à l'efficacité pédagogique du jeu en permettant aux apprenants de se concentrer sur les données pédagogiques du jeu, et non pas sur les exigences du jeu.

Les activités développées dans ce travail s'articulent autour des deux dernières recommandations, en utilisant le jeu pédagogique comme une des nombreuses modalités possibles pour enseigner, sans le voir comme étant la panacée de la motivation, de l'engagement et des apprentissages des élèves.

### 2.3.2. Revue des jeux pédagogiques

Cette section donne un aperçu des jeux pédagogiques traitant les chaînes de blocs déjà existants, utilisés dans le vrai monde avec des vrais élèves, trouvés dans la littérature ou expérimentés dans le cadre de la formation GymInf.

#### Jeux déconnectés

Le jeu pédagogique le plus connu s'apparentant aux chaînes de blocs est le "MIT Beer Game", développé par Jay Wright Forrester de la MIT Sloan School of Management en 1960, dont le but est de représenter une chaîne d'approvisionnement (supply chain) en utilisant des petites feuilles de papier autoadhésives amovibles (postits) sur un panneau (voir figure 2.16). Il s'agit de la simulation de gestion la plus ancienne et la plus utilisée, maintenant disponible en ligne [53].

Les chaînes d'approvisionnement et les chaînes de blocs sont similaires dans leur dynamique : les différents acteurs agissent simultanément, dépendent les uns des autres, et les actions des uns ont des répercussions sur les autres.

Dans le "Blockchain Paper Game" [29] (voir figure 2.17), qui est une variation du ERPsim Lab développé aux HEC à Montréal (Canada), les étudiants jouent d'abord avec un jeu similaire au "MIT Beer Game" pour comprendre les processus de base de l'administration, puis les concepts de contrats intelligents et de registre distribué (distributed ledger) sont introduits. Même si ce jeu est orienté vers l'administration, son analyse du jeu pédagogique et la description de la méthode pour illustrer certains aspects de la chaîne de blocs ont été sources d'inspiration pour le développement de BlocNote.

#### Jeu sur ordinateur

Bloxxgame [27] (voir figure 2.18) est un outil informatique visant à "remplir une lacune dans la compréhension de la chaîne de blocs". Dans ce jeu, les élèves représentent les noeuds et l'enseignant gère la chaîne de blocs. Les interactions entre les différents éléments (noeuds, blocs, transactions, ...) sont automatisés ; par exemple, le solde de l'utilisateur est mis à jour automatiquement lorsque les blocs sont confirmés. L'aspect cryptographique est mis en oeuvre de façon fidèle à la réalité, et l'interface graphique très agréable .



FIGURE 2.16. – MIT Sloan School Beer Game

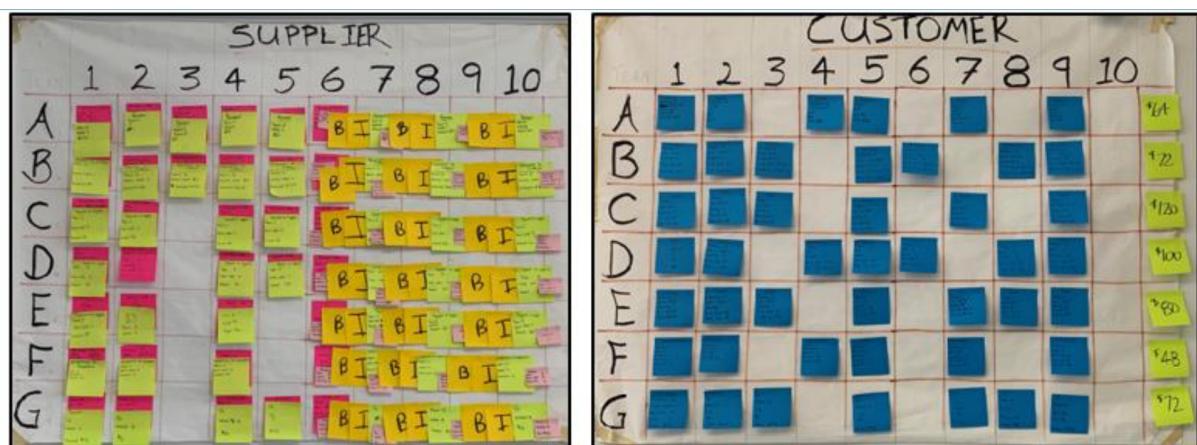


FIGURE 2.17. – Blockchain Paper Game

The screenshot displays the Bloxxgame blockchain interface with several panels:

- Nodes:** Shows 2 nodes: 0xBc2a83b36274b2 and 0xBc2a8ff690a6.
- Signature:** A form to sign data (e.g., 0xBc2a8ff690a6::1667649016) with a private key, featuring "Sign" and "Clear" buttons.
- Blockchain:** Displays Block 4 with Hash: ... and Status: unconfirmed. It shows the previous block's hash (78d0f81c0db4ecdb9efa6a905d21acb) and Merkle Root.
- Block:** A panel for creating a new block (Block# 4). It includes fields for Previous Blockhash (78d0f81c0db4ecdb9efa6a905d21acb), Merkle Root (empty), Difficulty (1), Timestamp (1667649233), and Nonce (0). Buttons include "Copy to Hasher", "Solve", and "Add Mining Reward (20)".
- Unspent Transactions:** Shows a transaction from address 0xBc2a8ff690a6 with amount 60.
- Hasher:** A panel showing the input for hashing: 3:1ca56603b10c9f1102caa92a8b3989606795b062::1667649152:0. It has "Hash" and "Clear" buttons.
- Merkle:** A panel showing Tx Hash 1 through Tx Hash 4, and Merkle Root Hash. It also has "Hash" and "Clear" buttons.
- Mempool:** A table for managing transactions with columns: Copy to Check, Input Address, Output Address, Amount, Public Key, SIG(Tx), and Tx Hash.
- Blockchain:** A summary of the blockchain chain:
  - Genesis:** Hash: 0489f...ad83b, Status: confirmed.
  - Block 1:** Hash: f5b4...b5dca, Status: confirmed.
  - Block 2:** Hash: 1ca56...5b062, Status: confirmed.
  - Block 3:** Hash: 78d0f...02a6d, Status: confirmed. It is connected by arrows to both Block 4 and Block 5.
  - Block 4:** Hash: ..., Status: unconfirmed.
  - Block 5:** Hash: 27a9f...69fc, Status: unconfirmed.
  - Block 6:** Hash: 31ca9...99398, Status: confirmed.

FIGURE 2.18. – Bloxxgame [https ://bloxxgame.io/play/](https://bloxxgame.io/play/)

Cependant, ce jeu très élaboré repose sur l'autorité de l'enseignant qui confirme, insère et finalise les blocs et les transactions que les noeuds-élèves produisent. Ceci contredit un des objectifs essentiels de la chaîne de blocs que ce travail cherche à faire comprendre aux élèves, soit l'absence de tiers de confiance. Bloxxgame est un jeu complet (bloc, PoW, Tx, clés publiques et privées, hachage, ...) et par conséquent complexe, il ne sera donc pas utilisé comme tel, mais pourrait servir lors de l'introduction de l'aspect cryptographique de la chaîne de blocs, pour des élèves plus avancés.

### Jeux hybrides

Le "Blockchain game" (voir figures 2.19 et 2.20) développé par Christianson [19] est un jeu clé-en-main où toutes les explications et les documents pour les élèves sont prêts à être imprimés. Le jeu est en partie déconnecté, où les élèves travaillent sur une PoW pour insérer des blocs, et en partie sur ordinateur, où un tableur est utilisé pour compiler les résultats .

Block	Course	Student	Grade	Nouce (1-3)	a	b	c	Value of Last 2 digits of prev Hash	Hash	Divid 3
1									212	
2										
3										
4										
5										
6										
7										
8										

**Hash = Nonce + a + b + c - Value of Last 2 digits of prev Hash**

Lookup Table

A	65	N	78
B	66	O	79
C	67	P	80
D	68	Q	81
E	69	R	82
F	70	S	83
G	71	T	84
H	72	U	85
I	73	V	86
J	74	W	87
K	75	X	88
L	76	Y	89
M	77	Z	90

Nonce = value between 1 and 3 that you will adjust to calculate a hash that can be equally divisible by 3

FIGURE 2.19. – Blockchain Game : tableau à imprimer pour aider les élèves à calculer la PoW.

Christianson conclue avec un résumé d'une dizaine de caractéristiques de la chaîne de blocs qui auraient été observés dans ce jeu, dont l'absence d'autorité centrale ; or le jeu en entier est centré sur le calcul de la PoW, qui est assez compliqué. Ce jeu ne semble pas permettre "aux apprenants de se concentrer sur les données pédagogiques du jeu, et non pas sur les exigences du jeu" [38], mais plutôt de relever un défi logique et mathématique. Dans le cadre de la formation Gyminf, dans le cours de "Sécurité et confidentialité", le Prof. Linus Gasser nous a fait jouer à un jeu pour illustrer l'enchaînement de blocs et la PoW avec un hash simplifié [37]. Les étudiants travaillent en groupes de trois pour calculer



Block	Course	Student	Grade	Nonce (1-3)	a	b	c	Value of Last 2 digits of Prev Hash	Hash
1	Parks 320	ad59da	F	1	80	65	70	12	212
2	Engineering 300	bd9ebc	B	1	69	66	66	4	198
3	Business 200	c67445	C	3	66	67	67	98	105
4	Parks 320	e2dd8a	B	3	80	69	66	5	213
5	Engineering 300	e2dd8a	D	2	69	69	68	13	195
6	Engineering 300	bde7af	B	2	69	66	66	95	108

FIGURE 2.20. – Blockchain Game : tableau pour compiler les résultats.

la PoW, soumettent leurs blocs par chat (le cours se faisait à distance), l'enseignant entre les données du bloc dans un programme javascript, qui vérifie la validité des blocs et affiche la chaîne de blocs, la chaîne la plus longue apparaissant en vert, les blocs orphelins en gris (voir figure 2.21). Le prof. Gasser a ensuite récompensé les étudiants du noeud qui avait placé le plus de blocs dans la chaîne la plus longue avec des chocolats.

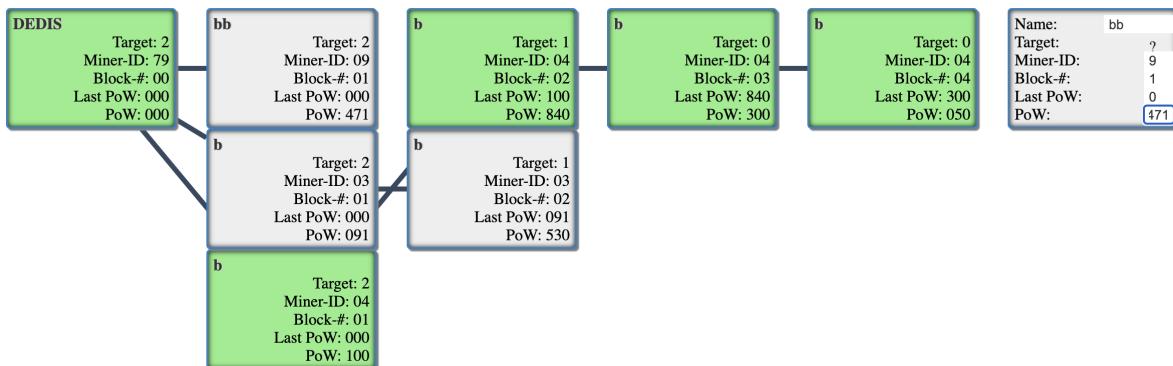


FIGURE 2.21. – Activité de chaînes de blocs de Linus Gasser [37].

Ici l'enseignant n'a pas cherché à démontrer toutes les caractéristiques de la chaîne de blocs avec son jeu, mais s'est concentré sur la PoW et le protocole de diffusion, et même si tous les blocs passaient par lui, il a agi en tant que relais, comme les relais présents dans internet, et non pas comme tiers de confiance. La chaîne de blocs CR est directement inspirée de ce jeu, mais vise plus d'objectifs pédagogiques.

Mes collègues Thierry et Olivier [55] ont repris le jeu du prof. Gasser : les élèves doivent calculer la PoW (sur une feuille en papier, donc de façon déconnectée) et entrent le résultat

tat dans un programme Python (voir figure 2.22). L’activité prévoit aussi une discussion sur les NFT, mais sans mise en activité.



FIGURE 2.22. – Activité de chaînes de blocs de Thierry et Olivier [55].

Mes collègues Daniel et Yves [31] ont développé le jeu du prof. Gasser pour y ajouter la pérennisation d’informations par la chaîne de blocs (des messages écrits par les élèves) et la signature des blocs avec la clé privée du mineur (voir figure 2.23). Le graphisme des blocs est particulièrement joli ! Les objectifs pédagogiques sont centrés sur la PoW, la sécurité et l’aspect décentralisé de la chaîne de blocs.

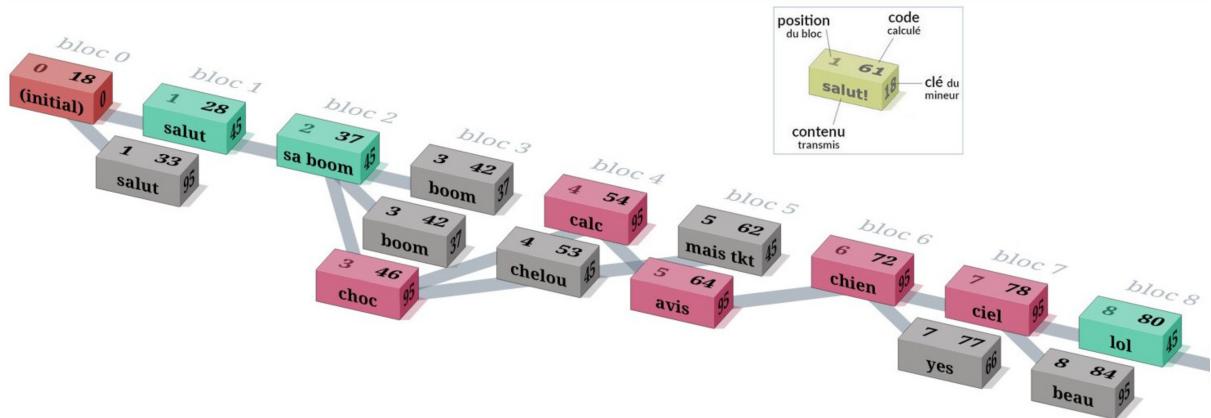


FIGURE 2.23. – Activité de chaînes de blocs de Daniel et Yves [31].

## Conclusion

Les jeux présentés dans cette section ont inspiré et servi à élaborer BlocNote, et l’autrice remercie leurs auteurs :) L’apport original de BlocNote consiste à complètement éliminer le tiers de confiance, réduire la part d’attention portée à la PoW, et concentrer les efforts sur les transactions, les SCs et les NFTs.

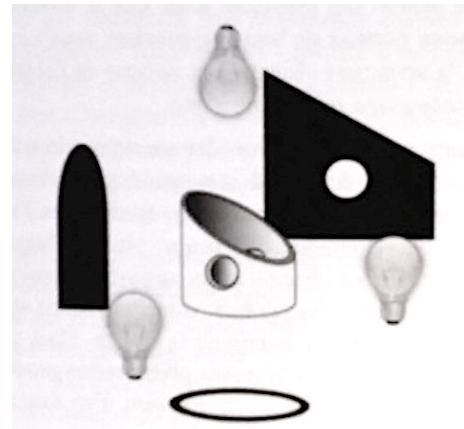
### 2.3.3. Activité déconnectée... ou pas

La majeure partie des activités se déroule de façon déconnectée, avec papier, marqueurs et grand tableau d'affichage, pour mettre en avant les interactions sociales, voire socio-constructivistes. Une discussion sur l'aspect numérique des chaînes de blocs demeure cependant incontournable.

En effet, la chaîne de blocs fournit un grand registre de transactions, financières ou autres, partagé presque instantanément. Ce caractère instantané est possible grâce aux ordinateurs et à l'internet, mais impossible dans un monde déconnecté. L'utilisation des ordinateurs demeure donc essentielle au fonctionnement d'une chaîne de blocs, permettant aussi d'accélérer les mécanismes (minage de blocs, diffusion des nouveaux blocs et transactions, validation et traitement d'une grande quantité de transactions par bloc), mais surtout de sécuriser la chaîne de blocs grâce à la cryptographie. Les fonctions de hachage et les algorithmes de signature numérique garantissent l'authentification de l'auteur d'une transaction et rendent la répudiation d'une transaction impossible, et ne sont pas envisageables sans l'utilisation d'un ordinateur. De plus,

L'activité déconnectée cherche à aider les élèves à s'approprier les concepts et mécanismes de la chaîne de blocs, mais demeure peu représentative de la réalité. Il est donc souhaitable de conduire une partie des activités avec l'aide d'un support numérique comme les programmes accompagnant BlocNote (chapitre 4), et éventuellement des logiciels plus élaborés comme Bloxxgame (section 2.3.2) .

Le fait d'approcher les chaînes de blocs d'abord par le jeu pédagogique déconnecté, puis avec des programmes informatiques, et les aller-retours entre l'activité du monde réel et celle sur ordinateur ainsi occasionnés, aident à ancrer les nouvelles connaissances. En effet, ces deux approches apportent des éclairages complètement différents du même objet complexe qu'est la chaîne de blocs, comme l'illustre la figure montrée à gauche : "la combinaison des perspectives nous permet de nous rapprocher de la structure objective de l'objet, mieux que chacune des projections prises une à une" [58].



Trois éclairages différents d'un même objet [58].

D'autre part, l'utilisation de programmes Python par les élèves offre une occasion de réactiver les connaissances et notions de programmation acquises lors des cours de première et deuxième année.

La proportion déconnectée vs connectée des activités est différentiable selon les buts recherchés et le contexte :

- Les activités peuvent se dérouler de façon entièrement déconnectées, sans ordinateur.
- La modalité la plus adéquate consiste à ce que l'activité se déroule de façon déconnectée, et que les programmes soient utilisés pour vérifier les blocs et transactions au fur et à mesure du développement des chaînes de blocs.

- Les élèves ont aussi la possibilité de travailler de façon autonome grâce aux programmes, pour s'entraîner à leur rythme, de façon autonome.
- Une activité avec Bloxxgame (voir section 2.3.2) compléterait le tableau brossé par BlocNote et ses programmes en ajoutant la cryptographie et la simultanéité des différents mécanismes.

## 2.4. Découpage des activités

Les activités sont découpées par thème, et sont modulaires : il n'est pas nécessaire de les faire dans l'ordre, et certains aspects peuvent être écartés ou repris dans une activité ou une autre. Notons que des nouvelles chaînes de blocs peuvent être lancées selon les phases des activités, siuté à la modification d'une règle, par exemple.

- Le seul prérequis pour l'activité 1 est celle de savoir transformer des nombres entiers en notation binaire, ce qui réactive des compétences supposées acquises en première année du Collège de Genève. Les autres activités n'ont aucun prérequis.
- la PoW (activité 1) ou la PoS (activité 5) sont interchangeables pour l'enchaînement des blocs dans les activités 3 et 4 ;
- l'activité 2 est indépendante des activités 3 et 4, mais l'activité 3 reste préalable à l'activité 4 ;
- la vente et l'achat de NFTs peuvent se faire avec de simples transactions (activité 3), suite à un accord oral entre le propriétaire et l'acheteur, ou avec les contrats intelligents (activité 4).

La table 2.4 résume les thèmes et contenus de chaque activité.

TABLE 2.4. – Découpage des activités

Activité 1	<p>enchaînement de blocs :</p> <ul style="list-style-type: none"> <li>— noeud-mineur, bloc, PoW, adresse, registre</li> <li>— notation binaire</li> <li>— immuabilité et sécurité : si un bloc est modifié, tous les suivants doivent l'être aussi</li> <li>— consensus : seuls les blocs de la plus grande chaîne sont considérés</li> </ul>
Activité 2	<p>décentralisation et consensus :</p> <ul style="list-style-type: none"> <li>— noeud-mineur ou noeud-validateur, copie locale de la chaîne de blocs</li> <li>— coexistence de versions différentes de la chaîne de blocs</li> <li>— protocoles de bavardage et de meilleur effort</li> <li>— personne et tout le monde possède la chaîne de blocs</li> <li>— absence de tiers de confiance ... démocratie ?</li> </ul>
Activité 3	<p>transactions :</p> <ul style="list-style-type: none"> <li>— noeud-mineur ou noeud-validateur, mempool</li> <li>— les mineurs choisissent les transactions du mempool qui sont les plus avantageuses (i.e. frais de transaction les plus élevés)</li> <li>— latence : délai entre la confirmation (inclusion d'une transaction dans un bloc) et sa finalisation</li> <li>— registre et calcul des soldes des utilisateurs</li> <li>— vérification de la validité des transactions</li> </ul>
Activité 4	<p>NFT :</p> <ul style="list-style-type: none"> <li>— noeud-validateur, transactions, SC</li> <li>— création de NFTs avec le SC ER-721</li> <li>— achat et vente de NFTs avec les SC NFT</li> </ul>
Activité 5	<p>PoS :</p> <ul style="list-style-type: none"> <li>— noeud-validateur, avec ou sans transactions, SC</li> <li>— mise en garantie avec le SC PoS</li> <li>— loterie et validation des blocs</li> <li>— 1R\$ = 1 ballotage : méritocratie ?</li> </ul>

# 3

## Programmes Python

---

<b>3.1. Présentation générale . . . . .</b>	<b>70</b>
3.1.1. Niveau . . . . .	71
3.1.2. Principes . . . . .	71
3.1.3. Architecture . . . . .	72
<b>3.2. Description des programmes . . . . .</b>	<b>72</b>
3.2.1. bchain_rousseau.py . . . . .	72
3.2.2. bchain_interac.py . . . . .	76
3.2.3. bchain_sim.py . . . . .	77
3.2.4. bchain_visualization.py . . . . .	77
<b>3.3. Activités in silico . . . . .</b>	<b>78</b>
3.3.1. Par où commencer ? . . . . .	78
3.3.2. Support aux activités . . . . .	79
<b>3.4. Mais bon, ma foi, c'est comme ça . . . . .</b>	<b>81</b>

Dans ce chapitre, les programmes de BlocNote sont présentés, leur utilisation par les élèves et enseignants discutée, et finalement, une auto-critique fait place à des pistes d'amélioration.

### 3.1. Présentation générale

BlocNote se base sur trois éléments fondateurs : les blocs, les transactions et les noeuds. Les jetons n'existent pas comme tels, ils sont implicites aux transactions. Seuls les blocs et les transactions sont traduits en classes<sup>1</sup>, décrites dans la section 3.2.1, car les noeuds ne sont jamais modifiés.

Les mécanismes suivants sont représentés :

- création des blocs et des transactions ;
- enchaînement des blocs, qui est basé sur la PoW, mais pourrait facilement être modifié pour représenter la PoS ;
- vérification des transactions → calcul des soldes des utilisateurs.

Les contrats intelligents ne sont pas implémentés.

---

<sup>1</sup>"Classes" comme dans "class Block :", et non pas comme dans "cours avec des élèves".

### 3.1.1. Niveau

Le groupe d'informatique du Collège Rousseau a fait le choix d'enseigner la programmation avec le langage Python dans le cadre des cours d'informatique de première et deuxième année, il est donc naturel que les programmes de BlocNote soient écrits en Python.

Les programmes de BlocNote sont inspirés de BlockSim [11], un outil de simulation de chaînes de blocs comme Bitcoin et Ethereum. Blocksim permet de tester les différents paramètres, comme la latence, affectant les flux, en particulier le nombre de transactions par seconde. Dans Blocksim, les blocs sont créés de façon aléatoire et autonome, sans interaction avec l'utilisateur (sauf pour fixer les paramètres de départ) ; ce logiciel n'est donc pas adapté aux activités en classe. De plus, il est trop complexe pour les élèves, et ne répond pas aux objectifs visés :

- un bloc simulé n'est conservé que si sa profondeur est supérieure au dernier bloc inséré, la chaîne ne comprend donc pas de fourches, ce qui ne permet pas de bien illustrer le processus de consensus ;
- les blocs sont produits automatiquement, sans interaction, selon les paramètres d'entrée ; il n'est donc pas utilisable en classe pour vérifier les blocs et les transactions soumis par les élèves.

Les programmes de BlocNote ont donc été écrit depuis zéro, avec un niveau sensé être accessible par des élèves du collège ayant fait les cours d'informatique de première et deuxième année du Collège de Genève, à l'exception de la notion de classes (programme bchain\_rousseau.py), et du programme bchain\_visualization.py, qui s'appuie sur le module Python NetworkX, qui va au-delà du niveau attendu des élèves.

### 3.1.2. Principes

Une bonne partie du fonctionnement des programmes se fait sur la base des numéros de blocs de type "chaîne de caractère", ce qui permet de remonter toute la chaîne jusqu'au bloc-genèse et d'identifier les blocs faisant partie de la plus longue chaîne et/ou miné/validé par un noeud donné. Les numéros de blocs pertinents peuvent être retenus dans une liste, et ensuite les transactions associées sont recherchées dans le registre de la chaîne de blocs.

Deux listes voyagent parmi les fonctions, et agissent un peu comme des variables globales :

- la liste liste\_noeuds, contenant les numéros des noeuds, est déterminée au départ puis jamais modifiée ;
- et la liste liste\_blocs, qui au départ est vide, puis incrémentée avec chaque nouveau bloc créé.

La chaîne de blocs produite par le programme bchain\_sim.py ou bchain\_interac.py est stockée dans un fichier .csv. Chaque ligne du fichier contient de cinq à six colonnes :

1. le numéro du bloc
2. la PoW
3. la transaction de récompense (obligatoire)
4. une deuxième transaction (pas obligatoire)

5. une troisième transaction (pas obligatoire)
6. et possiblement le solde de chaque noeud, selon le type de registre (voir section 1.4.3).

Des exemples sont montrés dans les tables 1.1 (sans solde) et 1.22 (avec les soldes).

### 3.1.3. Architecture

Quatre programmes accompagnent BlocNote : trois programmes servent à produire les chaînes de blocs, et un quatrième à les visualiser.

**bchain\_rousseau.py** : définition des classes et fonctions utilisées par les trois autres programmes

**bchain\_sim.py** : simule une chaîne de blocs, avec deux paramètres en entrée : le nombre de blocs et la liste des numéros de noeuds, et produit un fichier .csv contenant les blocs et les transactions (et éventuellement les soldes des noeuds). Ce programme peut être utilisé pour tester les différents paramètres (nombre de transactions par blocs, montants de la récompense, des transactions et des frais de transaction, ...), voir les effets de la modification des règles de BlocNote et produire des exemples de chaînes de blocs.

**bchain\_interac.py** est un programme qui vérifie les blocs entrés au fur et à mesure par les utilisateurs, et produit un nouveau fichier .csv à chaque fois qu'un bloc valide est inséré. Ce programme sert lors des activités en classe et aux élèves qui souhaiteraient s'entraîner, mais peut aussi être utilisé a posteriori pour vérifier une chaîne de blocs écrite sur papier.

**bchain\_visualization.py** bchain\_rousseau.py et bchain\_interac.py produisent un fichier .csv contenant tous les blocs de la chaîne de blocs, avec le même format. Ces fichiers .csv peuvent être ouverts avec un tableur (voir tables 1.7 ou 1.1) ou visualisés grâce à bchain\_visualization.py, qui produit une représentation graphique en format .png comme celles montrée à la figure 1.1.

La figure 3.1 représente l'architecture des programmes de BlocNote.

## 3.2. Description des programmes

Dans cette section, les programmes BlocNote sont brièvement expliqués. Les programmes complets sont donnés en annexe C.

### 3.2.1. bchain\_rousseau.py

Ce programme contient les définitions suivantes :

- les classes Block (bloc), Tx (transaction) et Reward (récompense)
- les fonctions communes à toutes les applications : mempool, calcul du solde, vérification des blocs, écriture des blocs dans un fichier .csv, etc.

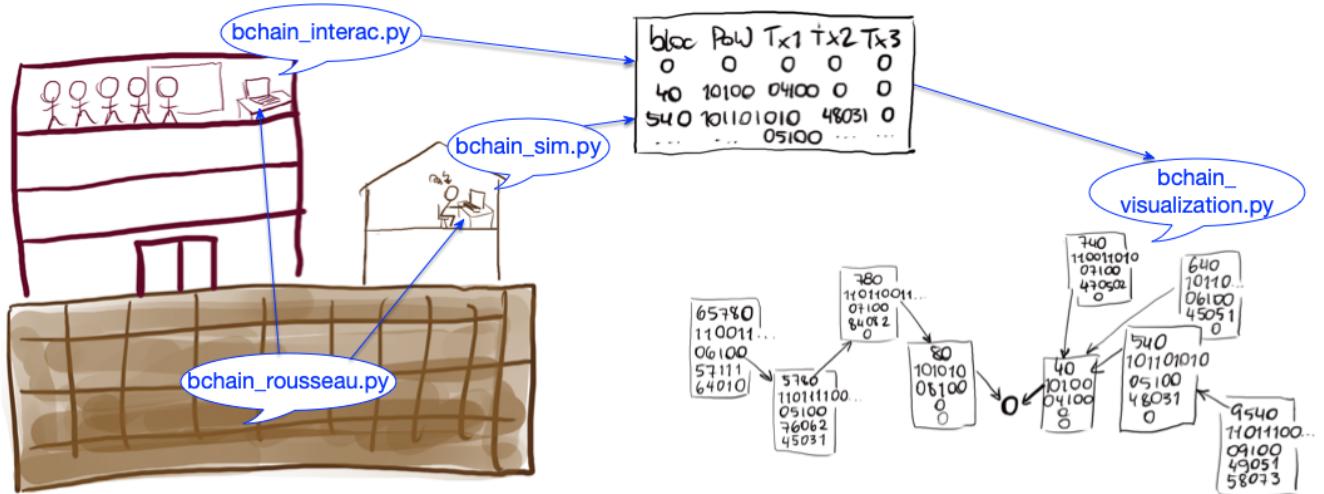


FIGURE 3.1. – Architecture des programmes de BlocNote.

### Définition des classes

La classe Block représentant les blocs de la chaîne de blocs contient trois attributs : le numéro du bloc, la PoW et la liste des transactions.

```
1 class Block:
2     def __init__(self, blockID, PoW, list_tx):
3         self.blockID, self.PoW, self.list_tx = blockID, PoW, list_tx
```

La classe Tx représentant les transactions contient quatre attribut : le payeur, le destinataire, le montant de la transaction et les frais de transaction.

```
1 class Tx():
2     def __init__(self,payeur,destinataire,amount,fee):
3         self.payeur, self.destinataire, self.amount, self.fee = payeur,destinataire,
            amount,fee
```

La classe Reward représente la transaction de récompense, et donc un cas spécial de transaction où le payeur est la chaîne elle-même, auquel le numéro '0' est attribué, le montant est de 10 R\$ et les frais de transaction nuls.

```
1 class Reward(Tx):
2     def __init__(self,destinataire):
3         self.payeur, self.destinataire, self.amount, self.fee = 0,destinataire,10,0
```

### Fonctions pour bchain\_sim.py

```
1 def createNewBlock(list_nodes,list_blocks)
```

Cette fonction crée un nouveau bloc :

1. déterminer le numéro du nouveau bloc -> createNewBlockID
2. revenir à l'étape 1 si le numéro du bloc fait déjà partie de la chaîne
3. calculer la PoW
4. insérer la transaction de récompense

5. créer une mempool -> createMempool(list\_nodes,10,10)
6. insérer 0, 1 ou 2 transactions, selon les soldes disponibles -> validateTx(mempool,dict\_balance)
7. retourner le nouveau bloc

```
1 def createNewBlockID
```

Cette fonction détermine le numéro du nouveau bloc :

1. choisir un mineur au hasard
2. choisir un bloc-parent au hasard
3. retourner un numéro de bloc

```
1 def createMempool
```

Cette fonction crée une liste de transactions. Elle prend en entrée la liste des numéros de noeuds, le nombre de transactions et le montant maximal d'une transaction. Ces deux derniers paramètres sont codés en dur lors de l'appel de cette fonction.

1. Les étapes ci-dessous sont répétées le nombre de fois correspondant au nombre de transactions :
  - a) choisir un payeur au hasard
  - b) déterminer le montant maximal des frais de transaction, qui correspond au cinquième du montant maximal de la transaction
  - c) déterminer les frais de transaction, au hasard entre 0 et le montant maximal des frais de transaction
  - d) déterminer le montant de la transaction, entre 1 R\$ et le montant maximal disponible (i.e. montant maximal de la transaction, après déduction des frais de transaction)
  - e) choisir un destinataire parmi les numéros de noeuds, mais sans celui du payeur
  - f) créer une transaction
2. retourner la liste des transactions

```
1 def validateTx
```

Cette fonction prend comme paramètre la liste de transactions créées par la fonction createMempool, puis retourne la liste des transactions qui sont valides (i.e. pour lesquelles le solde du vendeur est suffisant), en ordre décroissant de frais de transaction.

Pour chaque transaction de la liste de transactions,

1. calculer le solde du payeur -> check\_balance
2. si le montant est supérieur à la somme du montant de la transaction et des frais de transaction,
  - a) ajouter la transaction à la liste de transactions valides,
  - b) déduire le montant de la transaction et les frais de transaction du solde du vendeur. *Note* : ce solde n'est pas modifié globalement, mais que dans cette fonction.
3. retourner la liste de transactions valides. *Note* : il est possible que la liste soit vide, en particulier au début de la chaîne, quand peu de blocs ont été créés et le solde de la plupart des noeuds est encore nul.

## Fonctions générales

```
1 def checkBalance
```

Cette fonction prend comme paramètre la liste de noeuds, le numéro du bloc qui vient d'être inséré et la liste de blocs et retourne un dictionnaire contenant les soldes des utilisateurs. Le nouveau bloc et les blocs-ancêtres du nouveau bloc sont analysés pour déterminer les soldes des noeuds :

1. créer un dictionnaire des soldes, tous initialisés à 0 ;
2. déterminer les blocs précédents le nouveau bloc jusqu'au bloc-genèse -> findAncestors ;
3. pour chaque bloc-ancêtre,
  - a) déterminer le mineur et la liste de transactions ;
  - b) pour chaque transaction,
    - i. ajouter les frais de transaction au solde du mineur
    - ii. soustraire le montant de la transaction et les frais de transaction du solde du payeur (ne pas faire s'il s'agit d'une transaction de récompense)
    - iii. mettre à jour le dictionnaire des soldes
4. répéter l'étape 2 pour le nouveau bloc ;
5. retourner le dictionnaire des soldes.

```
1 def findAncestors
2 def findAncestorIDs
```

Ces fonctions prennent comme paramètre le numéro du bloc qui vient d'être inséré et la liste list\_blocks, et retournent une liste de blocs-ancêtres ou les numéros d'identification des blocs-ancêtres du nouveau bloc, respectivement.

1. déclarer une liste list\_ancestors ou list\_ancestorsID vide
2. déterminer les numéros des blocs-parents jusqu'au bloc-genèse
3. parcourir tous les blocs de la chaîne de blocs, et si le numéro d'un bloc correspond à celui d'un bloc parent, l'ajouter dans la liste de blocs-parents
4. retourner la liste de blocs-parents.

Les fonctions writeToFile(list\_rows), writeCSVRows(list\_blocks), et writeCSVRowsWithBalance(list\_nodes,list\_blocks) transcrivent la list\_blocks dans un fichier .csv.

## Fonctions pour bchain\_interac.py

```
1 def readBlocksFromCSVFile
```

Cette fonction remplit la list\_blocks à partir d'un fichier .csv :

1. créer une liste list\_blocks vide
2. donner le nom du fichier .csv à lire ou déterminer quel fichier .csv est le plus récent -> newest
3. créer un bloc de la classe Block avec chaque ligne du fichier et l'ajouter à la list\_blocks

4. retourner list\_blocks

```
1 strToRealTx()
```

Cette fonction lit les transactions du fichier .csv, qui sont de type str, et crée des objets de la classe Tx.

### 3.2.2. bchain\_interac.py

Le programme bchain\_interac.py revêt plusieurs aspects propres à l'interaction avec l'élève ou l'enseignant qui cherche à vérifier la validité de la chaîne de blocs créée dans le monde réel. Ce programme prend en entrée les blocs insérés par l'utilisateur de façon interactive, et produit un fichier .csv ne contenant que les blocs valides. A chaque bloc inséré, un nouveau fichier .csv est produit, et à chaque exécution, le programme reprend automatiquement le fichier .csv le plus récent.

1. un bloc est créé à partir des informations entrées au clavier par l'utilisateur -> createUserBlock ;
2. les blocs de la chaîne de blocs la plus récente sont insérés dans list\_blocks -> readBlocksFromCSVFile ;
3. les soldes des utilisateurs sont calculés avant l'insertion du nouveau bloc à partir de list\_blocks -> checkBalance ;
4. vérifier que le bloc entré par l'utilisateur a bien un bloc-parent dans list\_blocks :
  - si oui, le bloc est inséré dans list\_blocks et le programme se poursuit,
  - sinon, un message d'erreur est affiché, aucun nouveau fichier .csv n'est créé et le programme s'arrête.
5. le solde des utilisateurs est recalculé suite aux transactions contenues dans le nouveau bloc :
  - si un solde est négatif, un message d'erreur est affiché, aucun nouveau fichier .csv n'est créé et le programme s'arrête,
  - sinon, le programme se poursuit.
6. le nouveau bloc est ajouté à list\_blocks ;
7. un nouveau fichier .csv contenant la list\_blocks mise à jour est créé.

## Fonctions

```
1 def createUserBlock
```

Cette fonction crée un bloc à partir des informations entrées au clavier par l'utilisateur :

1. demander à l'utilisateur d'entrer au clavier le numéro du bloc, la PoW et les transactions autres que la récompense (s'il y en a)
2. émission de messages d'erreur si les informations sont invalides, par exemple :
  - si le numéro du bloc ne se termine pas par '0' ;
  - si la PoW ne correspond pas à la notation binaire du numéro du bloc ;
  - si le nombre de transactions autres que la récompense est supérieur à deux.

3. création de la transaction de récompense ;
4. validation des transactions entrées par l'utilisateur -> validateTxEntry ;
5. si le bloc est valide, un bloc de la classe Block est retourné; sinon, un bloc nul (0,0,0) est retourné.

```
1 def validateTxEntry
```

Cette fonction explique à l'utilisateur comment entrer une transaction au clavier avec le bon format, vérifie si le nombre entré par l'utilisateur comporte cinq chiffres, et si le premier correspond à un numéro de noeud de list\_nodes. Un message d'erreur est affiché si le format de la transaction est invalide, mais cette fonction ne vérifie pas sa validité en terme de solde du payeur.

### 3.2.3. bchain\_sim.py

Le programme bchain\_sim.py est une version simplifiée de BlockSim [11] produisant automatiquement une chaîne de blocs (mais avec des fourches) à partir des paramètres d'entrée : la liste des noeuds, le nombre de blocs à produire, le montant maximal d'une transaction et le nombre de transactions dans le mempool. Ce programme sert à simuler des chaînes BlocNote afin de tester les différents mécanismes développés pour les activités-élève et l'impact des différents paramètres sur le comportement de la chaîne de blocs. Il permet d'expérimenter de nouveaux algorithmes de consensus ou modifier les mécanismes selon les discussions avec les élèves et de produire des exemples de chaînes de blocs BlocNote.

### 3.2.4. bchain\_visualization.py

Le programme bchain\_visualization.py prend en entrée un fichier .csv produit par le programme bchain\_sim.py ou bchain\_interac.py et produit une représentation graphique de la chaîne de blocs en format .png, comme celle montrée aux figures 1.2 (sans les transactions) et 3.2 (avec les transactions).

Les principales étapes du programme sont les suivantes :

1. choisir la source des données : soit le plus récent des fichiers de simulation, soit le plus récent des fichiers créés interactivement, soit un fichier en particulier ; dans ce dernier cas, il faut entrer manuellement le nom du fichier.

```
1 #newest_csv_file = newest('bchains_sim/')
2 newest_csv_file = newest('interac_bchains/')
3 with open(newest_csv_file, newline='') as open_file:
4 #with open('bchains/bchain_sim20221026_193342.csv', newline='') as open_file:
```

2. remplir list\_blocs avec les informations contenues dans le fichier .csv. *Note* : les éléments de list\_blocs ne sont pas de la classe Block.
3. les noeuds du graphe correspondent aux numéros des blocs
4. les arêtes du graphe joignent un bloc à son bloc-parent
5. les (labels) sont créés à partir de la PoW et des transactions. Il est possible de choisir quelles informations sont affichées en modifiant les lignes suivantes :

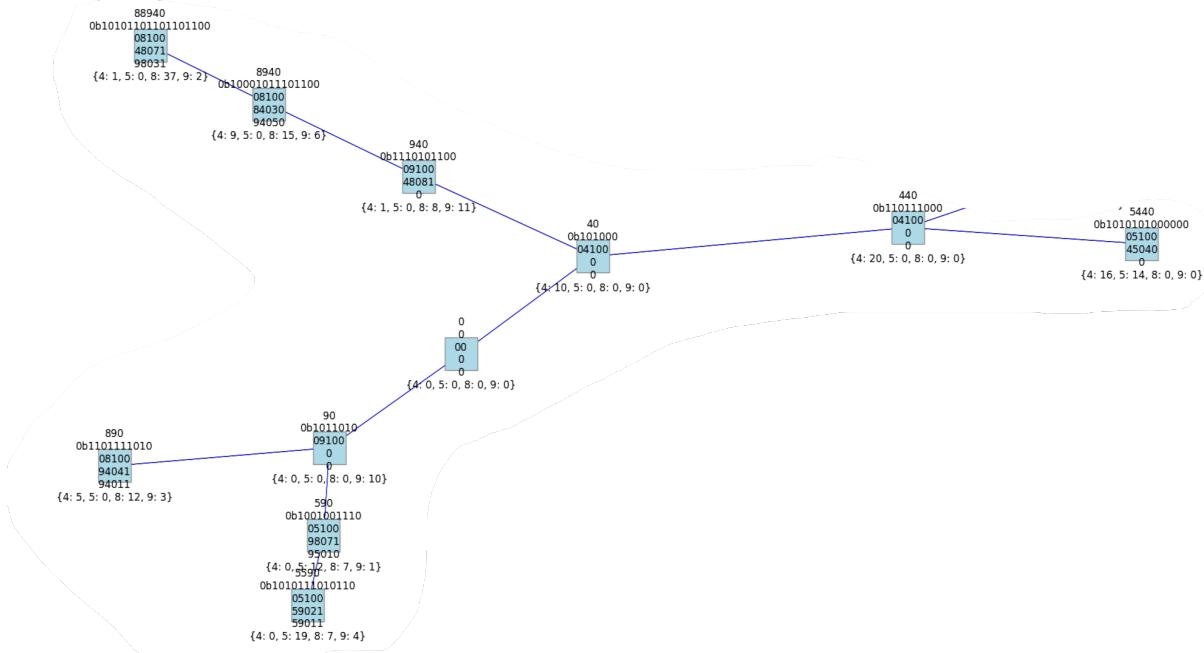


FIGURE 3.2. – Représentation graphique d'une chaîne de blocs BlocNote montrant les transactions.

```

1 dict_labels[node] = node+'\n'+PoW + '\n' + tx
2 #dict_labels[node] = node + '\n' + PoW

```

6. un graphe est créé.

Beaucoup de lignes de commande reposent sur le module Python NetworkX, dont voici la référence : <https://networkx.org>.

### 3.3. Activités *in silico*

Dans cette section, une marche à suivre pour les débutants est explicitée, et les activités-élèves (voir chapitre 3 et annexe B) sont revues en modalité connectée.

#### 3.3.1. Par où commencer ?

Les marches à suivre décrites dans cette section devraient (en principe) permettre à tout utilisateur de l'intranet du Collège Rousseau de produire une chaîne de blocs BlocNote.

##### Simulation

La marche à suivre suivante permet de créer une chaîne de blocs automatiquement :

1. copier les programmes bchain\_rousseau.py, bchain\_sim.py et bchain\_visualization.py dans un même répertoire ;
2. créer un sous-répertoire bchain\_sim, dans lequel les fichiers .csv seront déposés ;

3. s'assurer que le bon répertoire est sélectionné dans le programme bchain\_rousseau.py :

```
1 filename = 'bchain_sim/bchain_sim' + str(tstamp) + '.csv'
2 # filename = 'interac_bchains/bchain_interac' + str(tstamp) + '.csv'
```

4. ajuster la liste list\_nodes :

```
1 list_nodes = [4,5,8,9]
```

5. exécuter le programme bchain\_sim.py ;

6. exécuter le programme bchain\_visualization.py pour visualiser la chaîne de blocs.

### Modalité interactive

Pour produire une chaîne de blocs BlocNote de façon interactive,

1. copier les programmes bchain\_rousseau.py, bchain\_interac.py et bchain\_visualization.py dans un même répertoire ;

2. créer un sous-répertoire interac\_bchains, dans lequel les fichiers .csv seront déposés ;

3. s'assurer que le répertoire interac\_bchains contient au minimum un fichier .csv, contenant au minimum le bloc-genève (0 0 0 0 0) ;

4. s'assurer que le bon répertoire est sélectionné dans le programme bchain\_rousseau.py :

```
1 # filename = 'bchain_sim/bchain_sim' + str(tstamp) + '.csv'
2 filename = 'interac_bchains/bchain_interac' + str(tstamp) + '.csv'
```

5. s'assurer que list\_nodes correspond bien aux numéros des noeuds participants :

```
1 list_nodes = [4,5,8,9]
```

6. exécuter le programme bchain\_interac.py et entrer une à une les informations associées à un bloc, par exemple en copiant les blocs et les transactions de la table 1.1 ;

7. exécuter le programme bchain\_visualization.py à n'importe quel moment pour visualiser la chaîne de blocs.

### 3.3.2. Support aux activités

Comme expliqué dans la section 2.3.3, l'utilisation des programmes est modulable selon l'objectif pédagogique visé.

#### Activités 1 et 3

L'utilisation du programme bchain\_interac.py par l'enseignant ou un élève-arbitre est recommandée pour vérifier les transactions au fur et à mesure que les noeuds-élèves enchaînent des blocs, et pour vérifier le résultat final.

## Activité 2 : algorithme de diffusion et consensus

L'algorithme de diffusion des nouveaux blocs et le consensus reposent sur le fait que le programme bchain\_interac.py reprend automatiquement le fichier .csv le plus récent pour construire la chaîne de blocs : la version de la chaîne de blocs dépend du moment où le programme est exécuté, ce qui peut causer la coexistence de versions concurrentes du registre, comme dans les vraies chaînes de blocs.

Pour illustrer ces mécanismes, chaque noeud-élèves est composé d'un ordinateur contrôlé par un groupe d'élèves. La gestion des programmes et des fichiers se fait soit localement (sur le bureau de l'ordinateur par exemple), soit dans un répertoire commun sur l'intranet du Collège Rousseau. Cette activité renforce et met en pratique les notions vues le chapitre "stockage des données" du cours d'informatique de première année du Collège de Genève, dont apprendre à manipuler des fichiers et des répertoires, ce qui constitue une compétence transversale qui leur sera utile dans leur vie privée et professionnelle.

L'algorithme de bavardage peut être implémenté de trois façons :

- répertoire commun : tous les noeuds-élèves travaillent dans le même répertoire de l'intranet du Collège Rousseau. Chaque noeud-élève travaille sur sa propre copie de bchain\_interac.py, et les fichiers .csv sont directement produits dans ce répertoire commun.
  - répertoires privés : chaque noeud-élève travaille localement dans son propre répertoire, sur le bureau de l'ordinateur par exemple, les fichiers .csv sont produits dans ce répertoire privé :
    1. les élèves copient manuellement leur fichier .csv dans un répertoire partagé après avoir calculé un nouveau bloc,
    2. avant de calculer un nouveau blocs, ils copient le plus récent fichier .csv du répertoire commun dans leur répertoire, ce qui écrase leur copie locale. Si un bloc qu'ils avaient calculé n'en fait pas partie, ils peuvent tenter de le réinsérer.
  - fichier .csv commun : chaque noeud-élève travaille dans son propre répertoire, les fichiers .csv sont produits dans ce répertoire privé :
    1. les élèves ne copient que la ligne qui correspond à leur nouveau bloc dans un fichier .csv commun "nouveaux\_blocs.csv" se trouvant dans un répertoire partagé sur l'intranet du Collège Rousseau, à l'aide d'un tableur ou d'un éditeur de texte,
    2. puis copient les lignes du fichier .csv commun dans leur fichier .csv local à l'aide d'un tableur ou d'un éditeur de texte, les vérifient, les conservent si elles sont justes ou les éliminent si elles sont fausses.

Si un bloc calculé par un noeud-élèves ne fait pas partie de la version la plus récente, ce noeud-élèves peut tenter de le réinsérer.

## Activités 4 et 5

Les programmes de BlocNote n'offrent pas de support aux SCs, dont le niveau de complexité va bien au-delà du niveau attendu pour le cours 3INDF (voir section 2.3).

Le mécanisme de PoS pourrait cependant être implémenté par les élèves, dans le cadre d'un cours d'option complémentaire, par exemple.

### 3.4. Mais bon, ma foi, c'est comme ça.

Les programmes discutés dans ce chapitre ont été créés pour développer le modèle de BlocNote et servir de support aux activités en classe, mais le but de ce travail n'étant pas d'enseigner la programmation, beaucoup de place est laissée à l'amélioration.

Quelques pistes à cet effet sont avancées ci-dessous :

- Certains éléments sont codés en dur, et doivent être modifiés directement dans le programme. Par exemple, la liste des noeuds list\_nodes codée en dur au tout début de bchain\_interac.py et bchain\_sim.py, et doit être ajustée manuellement selon les numéros effectivement utilisés.
- Les transactions de récompense commencent par un zéro, or le fichier .csv ne rend pas le zéro, car il interprète le numéro de transaction comme un nombre. Par exemple, 04100 signifie que le noeud 4 reçoit une récompense de 10 R\$, mais le fichier csv l'affiche comme 4100. Ceci rend incohérents le registre et la représentation graphique, où la première transaction de chaque bloc contient 4 chiffres (plutôt que 5).
- Par défaut, BlocNote fonctionne avec le consensus de PoW, mais il serait très facile d'implémenter la PoS.
- Les messages d'erreur sont à compléter.
- Les allers-retours entre le type int et le type str dans les programmes bchain\_interac.py et bchain\_visualization.py sont à revoir.
- Les programmes sont un mélange peu savant d'anglais et de français.

# 4

## Conclusion

Les activités découlant de ce travail constituent une opportunité unique de faire vivre aux élèves un système alternatif méritocratique, où l’importance d’un acteur dans la chaîne de blocs est proportionnelle au nombre de jetons qu’il détient, coopétitif, où les participants coopèrent tout en y trouvant un intérêt personnel, et sécuritaire par définition, car il est plusavantageux de surveiller ses pairs et de jouer selon les règles du jeu plutôt que de tenter de tricher.

Les choix pédagogiques opérés pour modéliser la chaîne de blocs semblent atteindre leur objectif, soit de favoriser le questionnement et les discussions au sein des participants, comme relaté ci-dessous.

### Tests avec des vraies personnes

Une version simplifiée des activités 1 et 3 a été testée avec trois groupes de trois à quatre personnes (donc quatre à cinq noeuds car l'auteure de ce travail participait aussi), dont la plupart sont enseignants d'informatique, mathématiques et/ou physique, un préparateur, un enseignant d'économie, et un autre de chimie. Les participants étaient groupés autour d'une table. Ils s'attendaient tous à une activité sur les ordinateurs, mais nous avons écrit les blocs et les transactions sur un tableau magnétique ou simplement sur une feuille A3 posée sur la table.

Les conditions matérielles ont été discutées, et l'affichage de feuillets, sur lesquels le bloc et ses transactions sont écrits à la main par les noeuds affichés, sur un grand mur à l'aide de scotch semble être la solution la plus prometteuse pour un grand groupe. Le stylo-craie sur une vitre a été écarté car pas assez voyant, les postits risquent de se décoller, et un tableau magnétique n'est pas assez grand. Un enseignant a fait remarquer que la configuration où un grand mur sert à l'affichage des blocs et les élèves représentant un noeud sont assis à une table illustre les aspects public (les noeuds sont diffusés à tout le réseau) et privé (les calculs des blocs se font à l'intérieur des noeuds).

Un enseignant futé a enchaîné ses blocs à ses propres blocs, et ses blocs ne contenaient que la transaction de récompense, mais sa chaîne n'a pas été la plus longue à la fin de l'activité, il n'a donc reçu aucun R\$. Un autre enseignant soutient que le calcul de la notation binaire d'un nombre plus grand que 1000 est trop difficile pour les élèves ; d'une part, cette observation a été faite auprès d'élèves de première année, et l'activité est prévue pour des élèves de troisième année. D'autre part, cette expérience n'est pas partagée par tous les enseignants. Mais le plus important demeure que le but de BlocNote n'est pas

d'expliquer la PoW, mais plutôt de travailler sur les transactions, ce qui peut se faire sans PoW, ou éventuellement avec la PoS, pour laquelle aucun calcul n'est nécessaire.

Le grand succès a été le foisonnement de questions en lien avec les chaînes de blocs qui ont été soulevées par les participants ; BlocNote semble remplir ses promesses de représentation.

## Applications et extensions

Afin de répondre à l'objectif de la mise en place d'une chaîne de blocs utile avec une empreinte écologique la plus limitée possible, d'autres exemples d'applications que celui des NFTs pourraient être mis en place :

- une solution commerciale locale, citoyenne, durable et solidaire, comme le Léman [34] mentionné en introduction ;
- pérénisation de promesses, par exemple celles des politiciens ;
- une EchoChaîne de bons voeux écologiques, comme "je renonce à prendre l'avion pour mes vacances d'été" ou "je ne mangerai pas dans un restaurant avec de la vaisselle jetable pendant deux mois", avec une distribution d'Ecocoins lorsque la promesse est remplie ;
- d'autres types de PoW utiles pourraient être implémentés : chaque bloc contient une question ("donner un synonyme de embellir en allemand"), et pour attacher un nouveau bloc, celui-ci doit contenir la réponse juste.

Plusieurs variantes de l'activité ont été proposées lors du test avec des vraies personnes :

- introduire une phase d'activité intra-noeud, où les quatre élèves formant un noeud développent une chaîne de blocs entre eux, ce qui leur permet de s'assurer de comprendre les règles, s'entraîner et définir des stratégies pour les phases où les noeuds sont en coopération ;
- introduire une phase de coopération inter-noeud, où l'activité est notée proportionnellement au nombre de R\$ accumulés ;
- plusieurs ont pensé qu'un affichage avec le projecteur serait bénéfique, ce qui est possible avec les programmes rousseau\_interac.py et rousseau\_visualisation.py ;
- Un enseignant a beaucoup insisté pour que les élèves aient accès à un programme qui calcule automatiquement le solde des noeuds ; ceci est possible avec rousseau\_interac.py, mais l'activité devrait quand même démarrer en mode déconnecté avant de passer aux ordinateurs, pour les raisons discutées à la section 2.3.3.

D'autres variantes permettraient d'illustrer d'autres aspects de la chaîne de blocs :

- un système d'amendes dans le cas où un bloc invalide est diffusé ou si un noeud ne fait pas son travail, comme dans Ethereum ;
- une récompense pour les blocs-ommer (voir section 1.3.3) rétribuerait peut-être plus justement le travail accompli par les noeuds ;
- La variante "défi" consisterait à ce que chaque noeud tire au hasard un défi à garder secret, par exemple :
  - insérer un bloc avec une PoW invalide,
  - insérer des transactions invalides dans le mempool,

- insérer des transactions invalides dans un nouveau bloc ;
- Les noeuds reçoivent des points s'ils réalisent un défi et s'ils démasquent les actions malicieuses des autres noeuds.
- le côté spéculatif des chaînes de blocs pourrait se traduire en une activité où les élèves achètent leurs notes avec des transactions en R\$.

### **Conclusion de la conclusion**

Dans le cadre de l'éducation à la culture numérique, BlocNote vise à amener les futurs citoyens à comprendre les mécanismes d'un outil qui prend de plus en plus de place dans le fonctionnement de notre société et d'en questionner le bienfondé, les usages et la sécurité. L'expérimentation auprès de collègues enseignants montre que BlocNote donne lieu à des questions pertinentes sur le fonctionnement et le sens des chaînes de blocs. En plus de son application dans le monde des finances, nous espérons que les élèves retiendront la chaîne de blocs comme revêtant aussi un grand potentiel en tant qu'outil démocratique pour des applications communautaires.

# A

## Acronymes

**Mempool** zone d'attente des transactions d'une chaîne de blocs (memory pool), voir section 1.4.1

**NFT** jeton non-fongible (Non-Fungible Token), voir section 1.5.2

**PoW** preuve de travail (Proof of Work), voir section 1.3.1

**PoS** preuve d'enjeu (Proof of Stake), voir section 1.3.2

**SC** contrat intelligent (Smart Contract), voir section 1.5

**Tx** transaction, voir section 1.4

# B

## Activités élèves

---

B.1.	Activité 1 : enchaînement de blocs et preuve de travail . . . . .	86
B.2.	Activité 2 : consensus . . . . .	86
B.3.	Activité 3 : transactions . . . . .	86
B.4.	Activité 4 : contrats intelligents et NFTs . . . . .	86
B.5.	Activité 5 : preuve d'enjeu . . . . .	86

---

**B.1. Activité 1 : enchaînement de blocs et preuve de travail**

**B.2. Activité 2 : consensus**

**B.3. Activité 3 : transactions**

**B.4. Activité 4 : contrats intelligents et NFTs**

**B.5. Activité 5 : preuve d'enjeu**

# C

## License of the Documentation

Copyright (c) 2023 Marie Di Marco.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation ; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

The GNU Free Documentation Licence can be read from [4].

# Bibliographie

- [1] bitcoin.fr est un site d'informations et de nouvelles autour de bitcoin. 22
- [2] ethereum.org. 4, 13, 23
- [3] Geeks for geeks.
- [4] Gnu operating system.
- [5] <https://blockexplorer.one/>. 27
- [6] <https://coinmarketcap.com/rankings/exchanges/>. 10
- [7] <https://foldingathome.org/?lng=en>.
- [8] Nym.
- [9] wikipedia.com. 6, 7, 19, 46
- [10] Futura sciences, 2022.
- [11] Maher Alharby and Aad van Moorsel. Blocksim : An extensible simulation tool for blockchain systems. *frontiers in Blockchain*, 3(28), June 2020. i, 71, 77
- [12] Baeldung. Baeldung helps developers explore the java ecosystem and simply be better engineers. 19
- [13] Saez Y. Baldominos A. Coin.ai : A proof-of-useful-work scheme for blockchain-based distributed deep learning. *Entropy (Basel)*., doi : 10.3390/e21080723. PMID : 33267437 ; PMCID : PMC7515252., 2019. 41
- [14] Blockchain Partner by KPMG Blockchain France. Lexique. 5, 36, 38
- [15] Dead Coins 1700+ Cryptocurrencies Forgotten by This World (2023 Updated). 99bitcoins.
- [16] A. Ricottone C. G. Oliver and P. Philippopoulos. “proposal for a fully decentralized blockchain and proof-of-work algorithm for solving np-complete problems.”. August 30th 2017. 41
- [17] Christian Cachin. Blockchains and consensus protocols : Snake oil warning. IBM Research - Zurich, 2020. 5, 6, 19
- [18] Amy Castor. Why ethereum is switching to proof of stake and how it will work. Technology review, MIT, 4 mars 2022. 23
- [19] J Scott Christianson. How to teach blockchain with “the blockchain game !”, February 2022. 64

- [20] Inc. 30 Knightsbridge Road Suite 620 Piscataway NJ 08854 CITE Sunbird, Sunbird Software. Largest bitcoin mining farms.
- [21] Mitchell Clark. Blockchain, explained blocks ? chains ? how does this whole thing work ? 7
- [22] Ethereum community. Proof-of-work (pow). 22, 23
- [23] NFT School contributors like you. Nft school, octobre 2021. 38
- [24] TheWeek Correspondent. 5 of the largest bitcoin mining farms in the world.
- [25] crypto.com. A deep dive into blockchain scalability.
- [26] Raynor de Best. Countries that mine the most bitcoin (btc) 2019-2022, Aug 8, 2022.
- [27] Schneider B. Dettling, W. Bloxxgame – A Simulation Game for Teaching Blockchain. 9th International Conference, GALA 2020, Laval, France, December 9–10, 2020, Proceedings. Springer Cham, 2020. i, 61
- [28] Cem Dilmegani. Top 9 smart contract use cases & examples in 2023. 35
- [29] Natalie Aman East Carolina University Dr. Scott Abney, Dr. Mark Angolia. Using a paper-based supply chain game to introduce blockchain concepts. Number Paper ID 30605. American Society for Engineering Education, Asee's Virtual Conference, 22-26 juin 2020. 61
- [30] D. Adinda et al. La Transposition didactique. Master cft, Faculté des sciences de l'éducation, Université de Strasbourg, 2015.
- [31] Yves Dethurens et Daniel Kessler. Expérimenter un cours transdisciplinaire. GymInf, Didactique de l'informatique, Rendu 100, 11 mars 2022. 66
- [32] Figaro. Bourse et placements.
- [33] Primavera De Filippi. Blockchain et cryptomonnaies. Number 5e tirage. Que sais-je ? Humensis, 2021. 4, 6, 14, 35, 36, 38
- [34] solutions developper à Monnaie Léman Florian Dubath. <https://monnaie-leman.org>. 19, 83
- [35] Hyperledger Foundation. Hyperledger – open source blockchain technologies. 19
- [36] Yeung Fuk. Useful Computation on the Block Chain. PhD thesis, Harvard University, 2019. 41
- [37] Linus Gasser. Cours de sécurité et confidentialité du programme gyminf. In fichier pdf à usage interne, Juin 2022. 64, 65
- [38] Robert T. Hays. The effectiveness of instructional games : A literature review and discussion. Technical report, NAVAL AIR WARFARE CENTER TRAINING SYSTEMS DIV ORLANDO FL, 1er novembre 2005. 60, 64
- [39] <https://commons.wikimedia.org/w/index.php?curid=99450981> HocusPocus00 Own work, CC BY-SA 4.0. The number of daily confirmed bitcoin, ethereum and litecoin transactions from january 2011 through january 2021. source data : <https://bitinfocharts.com>, 31 January 2021.
- [40] Marco Iansiti and Karim R. Lakhani. Merriam-Webster dictionary. Merriam Webster, <https://www.merriam-webster.com/dictionary/blockchain>, 2022. 4
- [41] Copyright © 2023 Insider Inc. Cryptocurrencies overview.
- [42] Aaron Broverman Kat Tretina. 10 best cryptocurrencies in january 2023 10 best cryptocurrencies in january 2023, January 2023.

- [43] Larousse. Dictionnaire Larousse. Harrap's, https://www.larousse.fr/dictionnaires/francais/blockchain/188331, 2022. 4, 36
- [44] Stuart D. Levi, Arps-Slate Meagher Alex B. Lipton, Skadden, and Flom LLP. An introduction to smart contracts and their potential and inherent limitations. Technical report, Harvard Law School forum on corporate governance, 26 mai 2018.
- [45] Wei Li. Adapting blockchain technology for scientific computing. 41
- [46] Bitcoin.com © 2022 Saint Bits LLC. How do bitcoin transactions work ? 32
- [47] Andreas Lymouras. A shallow dive into bitcoin's blockchain part 2 - transactions. 27
- [48] modulo (ressources pour l'enseignement de l'informatique au gymnase). Vie privée et surveillance.
- [49] Kirsty Moreland. How to read a blockchain transaction history.
- [50] Satoshi Nakamoto. Bitcoin : A peer-to-peer electronic cash system. 4, 19, 41
- [51] Laurence Ndong. Didactique des sciences et formation des enseignants de sciences de la vie et de la terre. Recherches en didactique des sciences et des technologies, https://doi.org/10.4000/rdst.420 :179–208, 2011.
- [52] J.-H. Morin M. Reymond Centre de droit bancaire et financier O. Depierre, C. Lapinte. Lexique de la blockchain, février 2022. 4, 36
- [53] MIT Sloan School of Management. The beer game, très vieux. 61
- [54] The National Institute of Standards and Technology (NIST). Blockchain. 5
- [55] Thierry Gerez Olivier La Spada. Expérimenter un cours transdisciplinaire, portfolio pièce 110. Didactique de l'informatique, Gyminf, mars 2022. 65, 66
- [56] Joël Orizet. Cryptomonnaies : la demande de licence bancaire de bitcoin suisse échoue. ICTjournal, mars 2021.
- [57] des finances et de la souveraineté industrielle et numérique Par Bercy Infos, le 12/04/2022 Innovation et data. Ministère de l'économie. Qu'est-ce que la blockchain ?, 4 avril 2022. 4
- [58] Patrice Potvin. Faire apprendre les sciences et la technologie à l'école. Presses de l'université Laval, 2018. 60, 67
- [59] Farran Powell and Benjamin Curry. 10 best crypto apps and exchanges of 2023, January 2023. 9
- [60] TPG (Transport publics genevois). Tp publicité sa.
- [61] MORITZ PUTZHAMMER. Which cryptocurrencies will survive a market crash ?, 11 August 2022.
- [62] Rujia Li★1 3 Qi Wang1 Shiping Chen4 Qin Wang★2, 4. Non-fungible token (nft) : Overview, evaluation, opportunities and challenges (tech reportv2). Technical report, 1 Southern University of Science and Technology 2 Swinburne University of Technology 3 University of Birmingham 4 CSIRO Data61, arXiv :2105.07447v3 [cs.CR], 25 octobre 2021. 38
- [63] rameerez. No, your nft is not on the blockchain.
- [64] Ezra Reguerra. Cryptopedia : Learn the basics of smart contracts and how they work. 35

- [65] Gabriel Rodriguez. 4 best crypto exchanges of january 2023. 10
- [66] Eric Rosenberg. How much energy it takes to power bitcoin, September 15th 2022.
- [67] The WIRED staff. The wired guide to the blockchain the idea of creating tamper-proof databases has captured the attention of everyone from anarchist techies to staid bankers. 37
- [68] Isabelle Vuillemin David De Vito Gabrielle Stiassny. Esii education numérique informatique. Site pédagogique officiel de l'enseignement genevois, 2021.
- [69] Conférence suisse des directeurs cantonaux de l'instruction publique. Evolution de la maturité gymnasiale projet actualisation du plan d'études cadre : Chapitre ii - thématiques transversales. Confédération suisse, département fédéral de l'économie, de la formation et de la recherche, 20 décembre 2020. 58
- [70] PA 18950 THE COMPUTER LANGUAGE COMPANY INC. 5521 State Park Road, P.O. Box 265 Point Pleasant. Definition : Bitcoin transaction. 27
- [71] L. ; Ramljak D. ; Davidović T. ; Urošević D. ; Jakšić Krüger T. ; Jovanović Đ. Todorović, M. ; Matijević. Proof-of-useful-work : Blockchain mining by solving real-life optimization problems. Symmetry, [https://doi.org/10.3390/sym14091831\(14\)](https://doi.org/10.3390/sym14091831(14)), 2022. 41
- [72] Clément Wardzala. Mempool. 28
- [73] Dieter S. Jacob E. Nastassia S William, E. Eip-721 : Non-fungible token standard," ethereum improvement proposals, no. 721, january 2018. [online serial]. Available : <https://eips.ethereum.org/EIPS/eip-721>, 2018. 38
- [74] Changpeng Zhao. 9