

# 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 [26] 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 unplugged activity 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. Introduction</b>	<b>2</b>
1.1. Motivations et buts . . . . .	2
1.2. Enjeux . . . . .	6
1.2.1. Comment amener des étrangers méfiant à coopéter . . . . .	6
1.2.2. Limites . . . . .	7
1.2.3. Critiques . . . . .	8
1.2.4. Anonyme ? Surveillance ? . . . . .	9
1.3. Objectifs pédagogiques . . . . .	10
1.3.1. Obstacles . . . . .	11
1.3.2. Choix . . . . .	12
1.3.3. BlocNote . . . . .	12
1.4. Structure du travail . . . . .	13
<b>2. Chaînes de blocs</b>	<b>16</b>
2.1. Chaîne de blocs . . . . .	17
2.1.1. Définitions . . . . .	17
2.1.2. Définition pour les élèves . . . . .	19
2.1.3. Représentation . . . . .	19
2.2. Fonctionnement . . . . .	21
2.2.1. Exemple d'achat avec cryptomonnaie . . . . .	22
2.2.2. Réseau . . . . .	24
2.2.3. Bloc . . . . .	26
2.2.4. Jeton (token) . . . . .	28
2.2.5. Noeud . . . . .	29
2.2.6. Adresse, clés privée et publique, et portefeuille électronique (wallet)	31
2.3. Consensus . . . . .	31
2.3.1. Preuve de travail (proof of work, PoW) . . . . .	32
2.3.2. Preuve d'enjeu (proof of stake, PoS) . . . . .	35
2.3.3. Inattaquable (?) . . . . .	36

2.3.4. Immutable, inaltérable . . . . .	37
2.4. Transactions . . . . .	40
2.4.1. Mempool . . . . .	41
2.4.2. Confirmation, finalisation et exécution . . . . .	41
2.4.3. UTXO et solde . . . . .	43
2.5. Contrat intelligent (smart contract, SC) . . . . .	47
2.5.1. Fonctionnement . . . . .	49
2.5.2. Jeton non-fongible (non-fungible token, NFT) . . . . .	50
2.5.3. SC NFT . . . . .	50
2.5.4. SC PoS . . . . .	55
<b>3. Transposition didactique</b>	<b>57</b>
3.1. BlocNote . . . . .	57
3.1.1. Valeurs . . . . .	57
3.1.2. Modèle . . . . .	58
3.2. Modalités . . . . .	64
3.2.1. Jeu pédagogique (instructional games) . . . . .	66
3.2.2. Revue des jeux pédagogiques . . . . .	67
3.2.3. Activité déconnectée... ou pas . . . . .	73
3.3. Découpage des activités . . . . .	74
<b>4. Progammes Python</b>	<b>76</b>
4.1. Présentation générale . . . . .	76
4.1.1. Niveau . . . . .	77
4.1.2. Principes . . . . .	77
4.1.3. Architecture . . . . .	78
4.2. Description des programmes . . . . .	78
4.2.1. bchain_rousseau.py . . . . .	78
4.2.2. bchain_interac.py . . . . .	81
4.2.3. bchain_sim.py . . . . .	83
4.2.4. bchain_visualization.py . . . . .	83
4.3. Activités in silico . . . . .	84
4.3.1. Par où commencer ? . . . . .	84
4.3.2. Support aux activités . . . . .	85
4.3.3. Algorithme de diffusion et consensus . . . . .	85
4.4. Mais bon, ma foi, c'est comme ça. . . . .	86
<b>5. Conclusion</b>	<b>87</b>
<b>A. Acronymes</b>	<b>90</b>

<b>B. Activités élèves</b>	<b>91</b>
B.1. Activité 1 : enchaînement de blocs et preuve de travail . . . . .	91
B.2. Activité 2 : consensus . . . . .	91
B.3. Activité 3 : transactions . . . . .	91
B.4. Activité 4 : contrats intelligents et NFTs . . . . .	91
B.5. Activité 5 : preuve d'enjeu . . . . .	91
<b>C. Programmes</b>	<b>92</b>
C.1. bchain_sim.py . . . . .	92
C.2. bchain_interac.py . . . . .	92
C.3. bchain_visualization.py . . . . .	95
C.4. bchain_rousseau.py . . . . .	96
<b>D. License of the Documentation</b>	<b>102</b>

# 1

## Introduction

---

<b>1.1. Motivations et buts . . . . .</b>	<b>2</b>
<b>1.2. Enjeux . . . . .</b>	<b>6</b>
1.2.1. Comment amener des étrangers méfiants à coopéitionner . . . . .	6
1.2.2. Limites . . . . .	7
1.2.3. Critiques . . . . .	8
1.2.4. Anonyme ? Surveillance ? . . . . .	9
<b>1.3. Objectifs pédagogiques . . . . .</b>	<b>10</b>
1.3.1. Obstacles . . . . .	11
1.3.2. Choix . . . . .	12
1.3.3. BlocNote . . . . .	12
<b>1.4. Structure du travail . . . . .</b>	<b>13</b>

### 1.1. Motivations et buts

Les images des figures 1.1 et 1.2 illustrent le fait que les chaînes de blocs sont maintenant ancrées dans notre quotidien, surtout sous la forme de cryptomonnaies. Les plateformes d'échange cherchent à attirer une clientèle peu informée en la matière en leur faisant miroiter la perspective d'un enrichissement.



FIGURE 1.1. – Publicité de cryptomonnaie sur les trams à Genève [59] [55].



FIGURE 1.2. – Cartes-cadeau en vente à la Poste et dans les stations-service.

La table 1.1 montre le volume d'échange de cryptomonnaies de trois plateformes d'échange en date du 23 avril 2023. Le prix par unité donne la valeur d'un jeton de cryptomonnaie en dollars américains, la capitalisation boursière correspond au nombre de jetons en circulation multiplié par la valeur de chaque jeton (en dollars américains), "en circulation" donne une estimation du nombre de jetons en circulation, alors que le volume total correspond au nombre total de jetons émis par la chaîne de blocs depuis le bloc-genèse. A titre de comparaison, la capitalisation boursière de Nestlé était de 347 milliards de dollars américains le 28 avril 2023.

Cryptomonnaie	prix par unité	capitalisation	en circulation	volume total
Bitcoin	27 714	533 milliards	19,4 millions	13,4 milliards
Ethereum	1879	224 milliards	120 millions	7,17 milliards
Tether	1,0004	81,5 milliards	81,4 milliards	20,6 milliards
Binance Coin	330	51,5 milliards	156 millions	677 millions

TABLE 1.1. – Volume d'échange de cryptomonnaies selon [40].

Un autre exemple de la présence des chaînes de blocs dans notre quotidien est celui des NFTs, sous forme d'objets de collection ou d'actifs dans les jeux vidéo. Les NFTs sont une des applications possibles des chaînes de blocs, et ont généré des échanges à hauteur de 200 millions à 17 milliards de dollars en 2021, selon les sources. Quelques exemples sont montrés dans la figure 1.3 :

- CryptoKitties est un jeu où des chatons virtuels sont créés, reproduits et collectionnés. Le prix moyen d'un Cryptokitty est de 60\$, mais certains atteignent des montants vertigineux, comme le Cryptokitty Dragon qui a été vendu pour 600 ETH en 2018, ce qui valait 172 000 \$ à l'époque.
- En 2021, l'oeuvre "The Merge" a été vendue 91,8 M\$ à un collectif de 29 983 personnes.
- En 2019, le NFT "Hyperion" de Gods Unchained a été cédé pour 137 ETH, soit 229 542 chf avec le taux de change du 23 avril 2023.

- Depuis 2020, la NBA (National Basketball Association) vend des clips vidéo extraits de vrais matchs de basketball sous forme de NFTs ; le prix moyen d'un "Moment" est de 80\$.

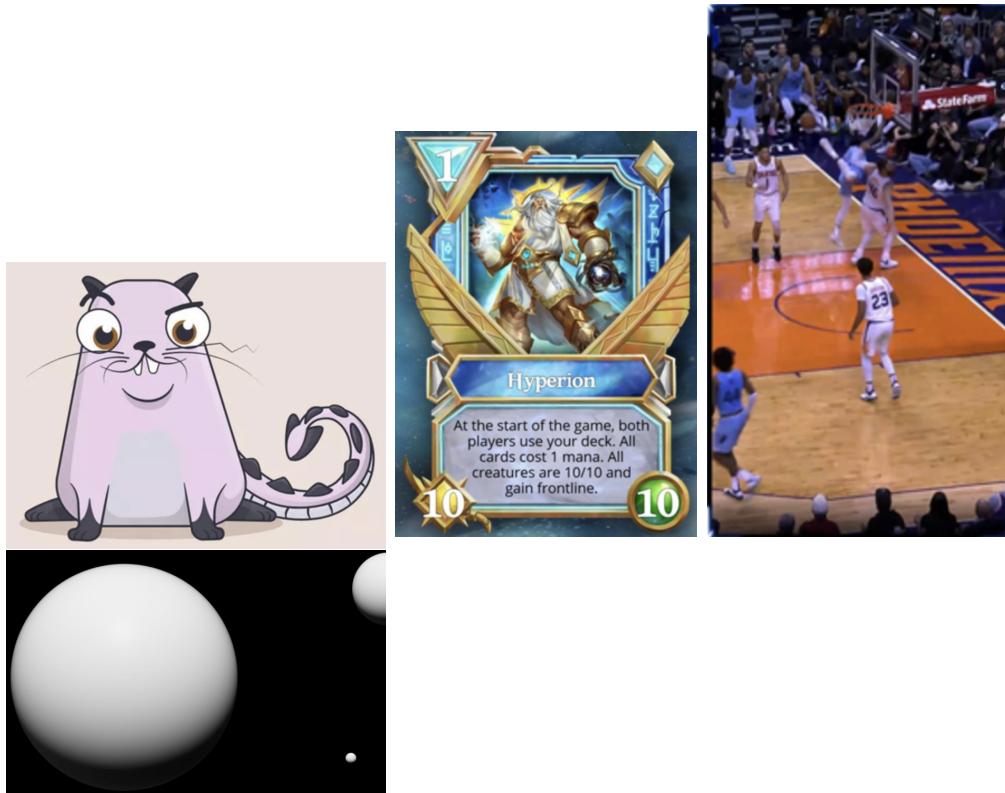


FIGURE 1.3. – Dragon (Crypokitties), The Merge, Hyperion (Gods Unchained) et NBA Top Shot.

D'autres applications de la chaîne de blocs poursuivent des buts communautaires, voire philanthropiques :

- NYM [8] utilise la chaîne de blocs pour sécuriser les transferts de données et récompenser les noeuds participant ;
- Folding@Home [7] demande aux citoyens de contribuer à la puissance de calcul nécessaire pour la simulation de protéines dans le but de soutenir le développement de médicaments et combattre les "menaces sanitaires globales" ;
- CALYPSO, MedChain, OmniLedger sont quelques uns des projets basés sur la technologie de la chaîne de blocs développés à l'EPFL [36] dont le but est d'établir un registre "réellement" décentralisé ou partager les données médicales.



Plus localement, "le Léman est la monnaie complémentaire, locale, du bassin lémanique transfrontalier. Par sa charte, la Communauté de paiement du Léman promeut une économie durable et solidaire. C'est une monnaie citoyenne, sa gouvernance est démocratique. On peut payer en billets ou en monnaie électronique, grâce à son application Biletujo" [33].

Il est essentiel d'amener les futurs citoyens à porter un regard critique sur les diverses incitations dont ils sont la cible, et qu'ils soient en mesure de faire des choix éclairés. A cet effet, le présent travail développe un modèle de chaîne de blocs et des activités pour les élèves de la maturité gymnasiale afin de leur fournir des connaissances et des outils suffisants pour mener une réflexion sur les chaînes de blocs et leurs applications.

La chaîne de blocs CR se veut une introduction à la pensée propre aux systèmes décentralisés, cherchant à développer des modèles mentaux utiles dans tous les domaines. Les participants expérimentent lors d'activités de coopétition (où les concurrents collaborent [32]) un nouveau modèle de gouvernance collectif et décentralisé, et se familiarisent avec la structure, les mécanismes et les dynamiques des différents acteurs et applications d'un système informatique complexe et intriqué que constitue la chaîne de blocs.

L'idée de ce travail est de développer un jeu éducatif déconnecté, basé sur un modèle simplifié de chaîne de bloc (CR), soutenu *in silico* par des programmes Python pour simuler, vérifier et visualiser la chaîne de blocs.

## 1.2. Enjeux

De Filippi [32] dit des chaînes de blocs qu'il s'agit d'une "technologie de rupture susceptible de changer le monde, au même titre que l'imprimerie, le fil barbelé et l'internet : ni bon, ni mauvais, mais pas neutre non plus". Il s'agit d'une évolution idéologique comme celle de l'internet : d'abord développée par des geeks et des punks, suivis par les pirates et les arnaqueurs, ensuite rejoints par les grandes institutions comme les gouvernements et les banques, et finalement adoptée par la masse.

La chaîne de blocs est déjà ancrée dans notre quotidien, comme discuté en introduction, et a de fortes chances de devenir incontournable, comme l'internet, sous la forme de système bancaire, de votation, de gestion associative, de coopétition entre entrepreneurs, etc.

### 1.2.1. Comment amener des étrangers méfiants à coopétitionner

Le défi que toute chaîne de blocs doit relever, et dont Nakamoto [49] a été le pionnier, est celui de convaincre des étrangers à investir dans un environnement où les participants peuvent apparaître et disparaître à tout moment (crash fault) et envoyer des transactions invalides, volontairement ou pas (problème des généraux byzantins). Les deux éléments-clés qui ont permis l'essor des cryptomonnaies sont les suivants :

- la cryptographie, qui empêche un utilisateur de dépenser les jetons associés à l'adresse d'un autre utilisateur,
- le consensus, qui garantit que les fonds d'une transaction n'ont pas été utilisés plusieurs fois (double-spending problem, voir section 2.3.3).

Les mécanismes suivants sont mis en place afin d'inciter des étrangers méfiants à coopé-titionner :

- chacun met des ressources à disposition (puissance de calcul, garantie en jetons natifs, ...),
- chacun reçoit une récompense pour sa contribution à la sécurité et à l'épanouissement de la chaîne de blocs,
- et chacun a la possibilité de vérifier les actions de leurs pairs (transparence).

### 1.2.2. Limites

Selon 99BITCOINS [15], en date du 3 novembre 2022, il y avait 1719 cryptomonnaies tombées en désuétude (dead coins) pour diverses raisons : leur développement s'est arrêté, manque d'utilisateurs et de traders, ou parce qu'elles ont été reconnues comme étant des escroqueries, etc.

De Filippi [32] met en lumière le fait que les chaînes de blocs sont coûteuses en terme des besoins de stockage, de puissance et de bande passante, ce qui les rend inefficaces quand elles grandissent. Elles ne pourront donc jamais rivaliser avec des systèmes de transactions comme celui de Visa, par exemple.

La figure 1.4 montre le nombre de transactions effectuées par jour par les chaînes de blocs Bitcoin, Ethereum et Litecoin, soit entre 256 000 et 1,1 million de transactions par jour. A titre de comparaison, Visa traite 150 millions de transactions par jour, et a la capacité de traiter 24 000 TPS (transactions par seconde), soit plus de 2 milliards par jour, contre 7 TPS avec Bitcoin [24], 15 TPS avec Ethereum 15 TPS, et 50 000 TPS avec Solana [60]. Selon [32], Bitcoin finaliserait une transaction chaque 7 minutes (soit 0,0024 TPS) et Ethereum une transaction chaque 10 à 20 secondes (0,05 à 0,1 TPS). Si on considère qu'un bloc est finalisé toutes les 10 minutes sur Bitcoin, et qu'un bloc comporte environ 1000 transactions, alors le taux serait de 0,028 TPS.

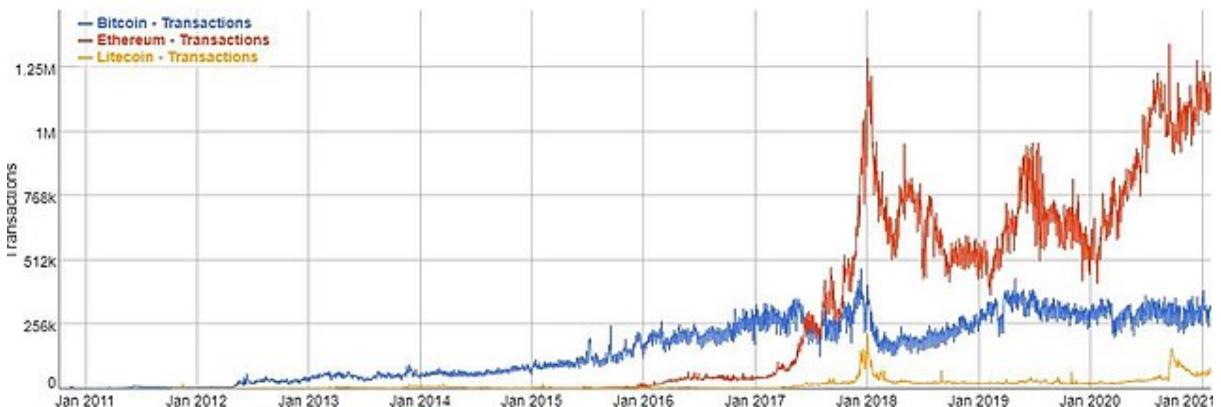


FIGURE 1.4. – Nombre de transactions effectuées par jour par les chaînes de blocs Bitcoin, Ethereum et Litecoin de janvier 2011 à janvier 2021 [38].

Le fait que la chaîne de bloc soit un système distribué, asynchrone et sans permission la rend vulnérable à l'attaque des 51%, où une entité rassemble 51% ou plus de la puissance de validation du réseau (voir section 2.3.3). L'entité prend alors le contrôle total de la chaîne de blocs, et peut dès lors altérer le registre, s'attribuer des jetons, bloquer des transactions, créer des transactions invalides, refuser des transactions légitimes, dépenser les mêmes jetons plusieurs fois, etc.

La mise à l'échelle ("scalability", ou potentiel évolutif) représente un obstacle majeur pour les cryptomonnaies. Semblable au triangle pédagogique de Jean Houssaye, selon lequel toute situation pédagogique privilégie la relation de deux éléments sur trois du triangle (savoir, enseignant, élève), une chaîne de blocs peut réaliser la décentralisation, la mise à l'échelle ou la sécurité, mais jamais les trois simultanément (voir figure 1.5). Des compromis sont dès lors inévitables.

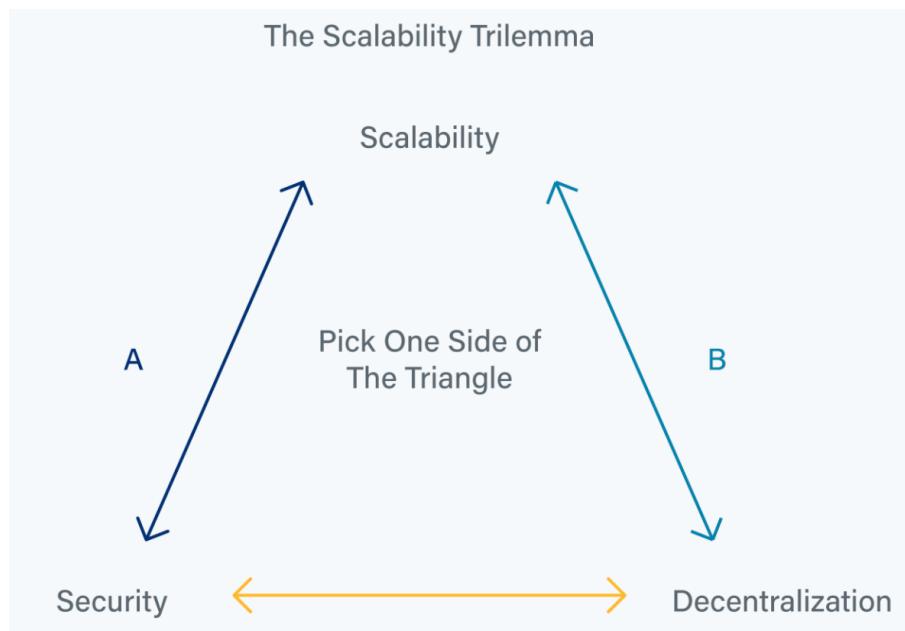


FIGURE 1.5. – Triangle de mise à l'échelle des chaînes de blocs (blockchain scalability trilemma) [24].

Une transaction finalisée sur une chaîne de blocs ne garantit en rien sa réalisation dans le vrai monde. Les contrats intelligents vont plus loin en déclenchant des actions autres que le simple transfert de jetons virtuels, ce qui pose la question à savoir qui ou quoi contrôlera si la qualité du rendu est celle qui était attendue, si chaque entité engagée a rempli son engagement, si une entité malveillante prétend qu'une autre n'a pas fait sa part, ou si une entité disparaît.

### 1.2.3. Critiques

Le calcul de la PoW des cryptomonnaies telles que Bitcoin est en tant que tel immatériel, mais sa production est très matérielle, et a de graves conséquences sur l'économie et le territoire [32]. Les opérations de minage du Bitcoin sont similaires à celles d'un centre de données (data center) traditionnel : de l'électricité doit être fournie, d'importants équipements de ventilation et de refroidissement doivent être mis en place, et l'emplacement doit être protégés des conditions extérieures et des cyberattaques. Les propriétaires, localisations géographiques et beaucoup de détails ne sont pas connus du public. Les spéculateurs mentionnent le plus souvent la Chine, l'Islande, la Russie, les Etats-Unis, la Suisse et les Pays-Bas comme étant les pays où les fermes de minage les plus importantes sont installées [19] [25] [23].

Selon les sources, la consommation attribuée au minage de Bitcoin se situerait entre 95 et 130 TWh par année ; à titre de comparaison, la consommation annuelle de la Suisse est de 56 TWh [65].

Le site Bitcoin.fr déclare que "Pratiqué sur le sol français (ou européen), le minage de cryptomonnaies n'est à conseiller qu'aux personnes qui disposent, à des fins domestiques, de leur propre générateur d'électricité verte (solaire, éolien, géothermique, hydraulique) sans avoir la possibilité de revendre les surplus de production sur le réseau public." [1]

La cryptomonnaie associée à une chaîne de blocs garantit la sécurité du réseau : plus le réseau est populaire, plus il est surveillé, plus il est sécuritaire. Cependant, elle donne lieu à des spéculations qui influencent l'économie mondiale, jusqu'à déstabiliser le marché.

Satoshi et les autres cyberpunk souhaitaient créer un système d'échange découplé du système bancaire, et aujourd'hui, les cours du Bitcoin et de l'Ether suivent celui de la bourse. La figure 1.6 montre le cours des cryptomonnaies selon [31]. Notons que seul BitcoinZ a connu une variation positive de +63 716 689 900,00 %, et que la seule autre cryptomonnaie à ne pas avoir perdu de valeur est le BunnyCoin, avec une variation de 0%.

Bitcoin	Ethereum	XRP	Litecoin
1 BTC =	1 ETH =	1 XRP =	1 LTC =
<b>22 988,618 \$</b>	<b>1 581,721 \$</b>	<b>0,406 \$</b>	<b>87,431 \$</b>
-0,17 % ↓	-1,39 % ↓	-1,05 % ↓	-0,17 % ↓

### Comparatif

Libellé	Cours	Var.%	Cap. boursière	Volume (24h)
BitcoinZ	63 716,690 \$	+63 716 689 900,00 % ↑	100 000 000 000 000 \$	20 367 \$
BunnyCoin	173,555 \$	0,00 % →	17 882 509 011 380 \$	190 \$
Rimbit	7 703,690 \$	-0,42 % ↓	1 522 103 168 633 \$	95 833 \$

FIGURE 1.6. – Capitalisation boursière des cryptomonnaies le 27 janvier 2023 [31].

### 1.2.4. Anonyme ? Surveillance ?

Les jetons produits et échangés par une chaîne de blocs sont associés à des adresses cryptographiques, et comme il n'existe pas de dictionnaire associant ces adresses à des personnes physiques ou morales, le propriétaire de jetons reste anonyme. Tout le monde peut voir les transactions, mais personne ne sait qui se trouve derrière l'adresse qui a créé une transaction ou possède des jetons. Ceci laisse la place au blanchiment d'argent, au marché noir, à l'évasion fiscale et autres manœuvres criminelles.

Même si la chaîne de blocs est en principe anonyme, les transactions étant associées à des adresses, il est possible de corrélérer les transactions à des actions dans le vrai monde, par exemple l'achat ou la vente d'un bien physique, voire numérique, et ainsi est tracer toutes les transactions associées à une adresse par triangulation, et identifier le propriétaire de l'adresse par inférence statistique. De Filippi [32] parle donc de "pseudonymité" plutôt que d'anonymité.

Ceci a aussi pour conséquence que des jetons fraîchement produits auraient plus de valeur que des "vieux" jetons, car ils ont moins d'historique. En effet, il est toujours possible de retracer les adresses à travers lesquelles un jeton a transité, et ainsi l'associer à une adresse ayant une mauvaise réputation, ou un utilisateur connu, par exemple [32].

D'un point de vue éthique, les caractéristiques de transparence et d'irréversibilité de la chaîne de blocs pourrait dériver vers une surveillance permanente et généralisée [32] sous forme de sousveillance<sup>1</sup>

D'autre part, Florian DUBATH, développeur du léman électronique [33], fait remarquer qu'une chaîne de blocs utilisée à des fins commerciales ne peut pas être anonyme de par la loi, car les comptes sont vérifiés par les autorités financières du pays où est émise la monnaie (principe "know your customer"). Il est à noter que le niveau d'anonymité influe aussi sur la sécurité du système : plus il est facile d'identifier un utilisateur, plus il est facile de le punir ou le rejeter en cas de mauvais comportement.

### 1.3. Objectifs pédagogiques

Nous verrons que l'enseignement des chaînes de blocs pourrait contribuer à atteindre certains objectifs du cours d'informatique du Collège de Genève, soit *de faire découvrir et d'acquérir les notions de base concernant la représentation (information et données), l'organisation (réseaux, bases de données) et le traitement automatique des données (algorithmique et programmation). Il encourage l'élève à faire un usage responsable des technologies numériques (bonnes pratiques, sécurité)*. [67]

Les modalités d'enseignement proposées dans ce travail sont principalement centrées sur le jeu pédagogique, répondant à un autre des objectifs du cours d'informatique du Collège de Genève [67], soit que *l'enseignement se déroule essentiellement au travers de travaux pratiques individuels ou en groupe, ou sous forme d'ateliers, pour inciter les élèves à évaluer de façon autonome et immédiate la qualité de leur analyse et de leurs modélisations*.

Le couplage de la CR avec les NFTs permet une symbiose entre un concept de sciences informatiques et l'art, dans laquelle une réelle réflexion sur un projet d'oeuvres numériques peut être menée. Ceci rejoint les objectifs du cours d'informatique du Collège de Genève, ainsi que de la CDIP [67] :

Dans les deux filières de l'enseignement général (Collège de Genève, Ecole de culture générale), l'enseignement de la science informatique est articulé à celui de la culture numérique, dispensée sous forme de compétences transversales approchées dans l'ensemble des autres disciplines.

Un travail sur un projet commun, entre l'informatique et une autre discipline du Collège de Genève, sans tenir compte des frontières entre les savoirs disciplinaires, en complétant ces derniers par des savoirs pratiques et professionnels et en réorganisant les savoirs disciplinaires en un système total. Il s'agit de mener une réflexion critique sur les paradigmes disciplinaires et les cultures scientifiques. [68].

Outre les applications de cryptomonnaies et NFTs, les chaînes de blocs constituent un système innovant, qui a pu émerger grâce aux progrès technologiques en informatique et électronique, et qui a le potentiel de fournir des outils de gestion des données appelés à remplacer les systèmes utilisés présentement, qui sont en général opaques et font appel à un tiers de confiance.

<sup>1</sup>La notion de «sousveillance» décrit un dispositif inverse à celui de la surveillance, soit un système dans lequel les personnes surveillées observent à leur tour les surveillants. Il s'agit donc de regarder «par le bas» ce qu'il se passe en haut en s'appropriant les mêmes outils [47].

### 1.3.1. Obstacles

Le concept de chaîne de blocs posera des obstacles de compréhension aux élèves, non seulement en ce qui concerne les mécanismes, mais surtout parce que les élèves n'ont probablement jamais été exposés à un système décentralisé, sans tiers de confiance. Les banques constituent un exemple simple et évident d'un système centralisé et opaque, et il est très probable que les élèves n'imaginent même pas qu'un système à l'opposé (décentralisé et transparent) puisse exister et fonctionner. Ce choc cognitif constitue à la fois une source de difficulté et une opportunité d'apprentissage.

De par sa complexité, la chaîne de blocs amène avec elle un vocabulaire très élargi, et certainement nouveau pour les élèves. Etant un sujet d'actualité, ce vocabulaire comporte beaucoup d'ambiguités, et étant maintenant adopté par le grand public, des contradictions et des expressions à la mode portent à confusion.

BlocNote vise aussi à déboulonner certaines idées fausses et rétablir certains faits :

- les chaînes de blocs ne sont pas que des cryptomonnaies ;
- la cryptomonnaie est purement virtuelle et n'a aucune valeur en dehors de la confiance que les utilisateurs placent dans la chaîne de blocs où elle a été générée, elle n'a aucun lien avec le vrai monde ;
- le jeton de cryptomonnaie n'est pas constitué d'une ressource ou d'un fichier, mais n'existe que dans une transaction. Il n'y a pas de transfert réel entre le payeur et le destinataire. [32]
- le portefeuille électronique (wallet) ne comporte pas de monnaie ni de cryptomonnaie ;
- le contrat intelligent (smart contract) n'est ni un contrat, ni intelligent.

Les bases des mécanismes de la chaîne de blocs reposent sur les mathématiques, elle serait donc en principe déterministe, pourtant les humains opèrent des choix qui influencent directement son développement, comme l'illustre la figure 1.7. Cette influence humaine sur le système informatique que constitue la chaîne de blocs rend son traitement complexe et intriqué.

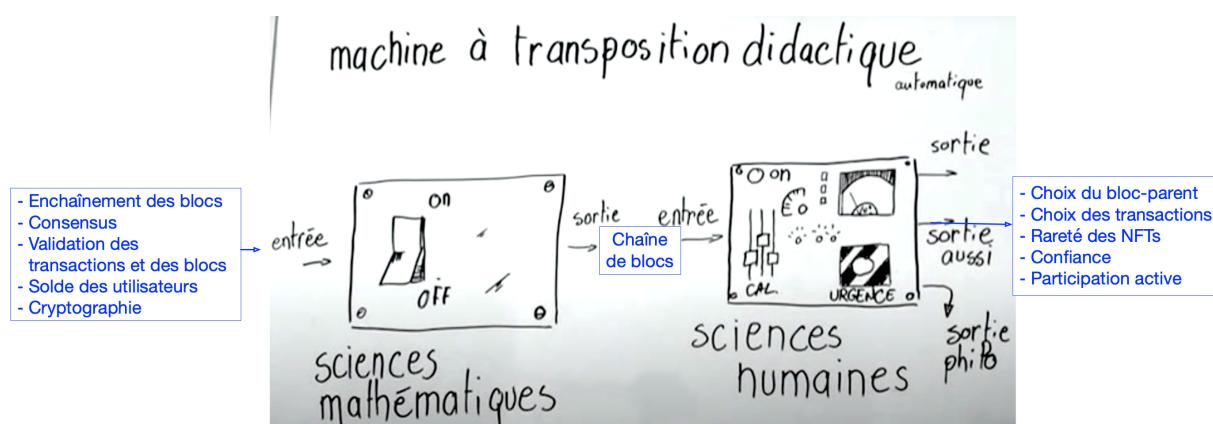


FIGURE 1.7. – Transposition didactique [50] à la sauce CR (en bleu).

### 1.3.2. Choix

La vastitude et la complexité des chaînes de blocs rivalisent avec celles de l'internet et du web ; le stratagème décrit par Potvin [57] a donc été adopté pour l'approcher :

(...) procéder à de multiples réductions divergentes du grand phénomène qui nous intéresse, mais qui demeure pourtant inaccessible dans son intégralité, puis à une étude minutieuse de chacune des projections que ces réductions nous fournissent. Ces réductions (...) ne visent pas à trouver la solution unique ultime (... mais) entretiennent l'ambition d'étudier sous différents angles divers aspects du phénomène à l'étude, (...) utiles et testables de la réalité complexe qui nous intéresse.

Il a donc été nécessaire de procéder à des choix dans l'élaboration de BlocNote sur la base du meilleur compromis possible entre la fidélité de la représentation, qui pourrait causer une surcharge mentale, et la simplicité de la vulgarisation, qui pourrait faire obstacle au transfert des apprentissages acquis dans le vrai monde.

Afin de motiver ces choix, une revue de littérature a d'abord été effectuée afin de déterminer quels aspects de la chaîne de blocs méritaient d'être traités (chapitre 2), et quels ont été les choix faits par d'autres enseignants (section 3.2.2).

Les aspects étudiés dans le présent travail seront principalement le registre inaltérable (enchaînement des blocs) et décentralisé (processus de consensus par PoW ou PoS), les transactions, les contrats intelligents et les jetons non-fongibles (NFTs). Outre l'aspect



La sécurité de la chaîne de blocs repose entièrement sur la cryptographie, mais ce sujet est si vaste et si riche qu'il constitue en lui-même des années d'enseignement. Il ne sera donc pas traité dans la CR.

cryptographique, les aspects commerciaux (efficacité, de spéculation, throughput, latence, scalability, processus industriels, chaîne d'approvisionnement, ...) sont écartés du modèle de la CR.

Nous verrons dans la revue bibliographique du chapitre 3 que la plupart des jeux pédagogiques se concentrent sur le calcul du nonce pour la PoW. L'imaginaire populaire est fasciné par la PoW et les fermes de minage de bitcoins, au point d'occultez le but des chaînes de blocs, qui consiste à pérenniser des transactions. De plus, la PoS prenant maintenant autant de place dans le monde réel que la PoW avec la récente conversion d'Ethereum, il est souhaitable d'initier les élèves à cet autre processus de consensus.

### 1.3.3. BlocNote

Le modèle de BlocNote est basé sur les chaînes de blocs Bitcoin et Ethereum pour les raisons suivantes :

- Bitcoin et Ethereum sont de loin les chaînes de blocs les plus répandues ;
- Bitcoin n'est pas la première chaîne de blocs utilisée comme système de transactions sans tiers de confiance, mais sans aucun doute la première à avoir mis ensemble les différents ingrédients permettant de gagner la confiance des utilisateurs et à devenir effective ;

- Ethereum a repris les grands principes déjà mis en oeuvre avec Bitcoin, et leur a ajouté les contrats intelligents. C'est la chaîne la plus avancée et la plus utilisée pour les contrats intelligents, car les autres souffrent du manque de standard [32].
- Selon [60], Bitcoin et Ethereum font partie des 20 cryptomonnaies dont la valeur, la durabilité et la demande indiquent qu'elles pourraient survivre un krash boursier et survivre à long-terme, et sont les deux meilleures cryptomonnaies en janvier 2023 selon Forbes [41].

Quoique très instructif, l'exemple du Bitcoin ne doit pas être le seul discuté avec les élèves, car il ne revêt qu'une partie des aspects des chaînes de blocs, et est le plus souvent associé à la spéculation financière. De plus, contrairement à ce qui est ancré dans l'imaginaire collectif, le Bitcoin n'est pas la première cryptomonnaie ; voir [32] et le livre blanc d'Ethereum [?] pour d'excellents historiques des chaînes de blocs et cryptomonnaies.

BlocNote est une chaîne de blocs développée pour les élèves de la maturité gymnasiale visant à combler l'immense gap conceptuel séparant la PoW et le processus de consensus avec la mise en place d'un système distribué maintenu par les élèves-noeuds, et non pas par l'enseignant. La comparaison entre les processus de PoW et de PoS permettra d'approfondir les deux concepts, et de nuancer la domination du bitcoin dans l'imaginaire populaire.

Les activités prennent la forme d'un jeu pédagogique centré sur le consensus et les transactions. La cryptomonnaie associée à BlocNote est le Rousseau dollars (R\$) ; le modèle de BlocNote sera développé au chapitre 3.

Et un autre objectif poursuivi par les activités développées sur la base de ce travail consiste à ce que les élèves aient du plaisir pendant les activités :)

## 1.4. Structure du travail

L'introduction (chapitre 1) cherche à mettre l'objet de ce travail dans le contexte du "vrai monde" et des pratiques sociales de référence que sont les programmes définis par la Confédération suisse et le Département de l'instruction publique de Genève.

Dans le chapitre 2, le principe de fonctionnement, les mécanismes et les composants principaux des chaînes de blocs sont exposés, de façon à identifier les "savoirs savants" pour poser les bases de BlocNote et justifier les choix qui ont été faits dans son élaboration et le développement des activités pour les élèves. Ce chapitre se veut aussi un support pour les enseignants qui devront répondre aux questions des élèves.

Le chapitre 3 détaille la transposition didactique des "savoirs savants" :

1. le modèle de BlocNote est d'abord exposé et les choix justifiés ;
2. le choix et le développement des modalités sont discutés sur la base d'une revue des activités et sites existants ;
3. et finalement, les concepts sont découpés en activités pour les élèves de la maturité gymnasiale.

Le chapitre 4 présente et discute brièvement des programmes Python de simulation, vérification et visualisation de BlocNote.

Les activités pour les élèves sont fournies en annexe B, qui se veulent "prêtes à photocopier" pour tous les enseignants qui souhaiteraient aborder les chaînes de blocs avec leurs élèves.

Le processus de transposition didactique prend ses sources dans le "vrai monde" et se termine par les apprentissages de l'élève ; la figure 1.7 met en lien les sections de ce travail avec ces différentes étapes.

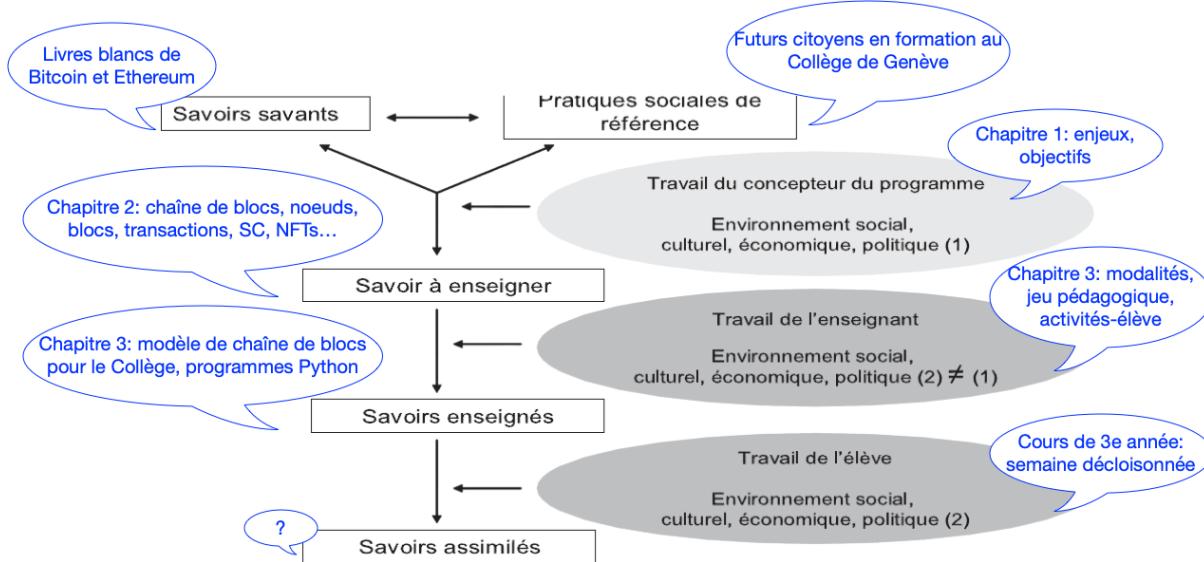


FIGURE 1.8. – Transposition didactique [29] à la sauce BlocNote (ajouts en bleu).

### Ordre de lecture recommandé

Nous avons vu que la chaîne de blocs est un système informatique complexe et intriqué. Par conséquent, le lecteur débutant aurait avantage à lire les activités-élève (annexe B) pour commencer. Dans le chapitre 2, les définitions font référence les unes aux autres, comme dans un dictionnaire, car on ne saurait dire qui de la poule ou de l'oeuf est apparu le premier. De plus, les mécanismes de la chaîne de blocs sont illustrés dans le chapitre 2 sur la base du modèle de la CR, expliqué au chapitre 3. Il faut donc tout lire simultanément, ou du moins faire des aller-retour entre les chapitres.



L'aimable lecteur est prié de se référer à l'annexe A pour les principaux acronymes, et de s'armer de patience et de résilience. Nous lui souhaitons néanmoins une bonne lecture :)

## Sources

Le chapitre 2 et le modèle de BlocNote ont été élaborés principalement sur la base des lectures suivantes :

- le papier blanc de Bitcoin [49], qui énonce les grands principes de la chaîne de blocs ;
- l'article BlockSim [11], qui donne une vue d'ensemble détaillée de l'architecture des chaînes de blocs ;
- le papier blanc de Ethereum [2], contenant un excellent historique des chaînes de blocs ;
- le livre de De Filippi [32], dressant un tableau complet des aspects techniques et sociétaux des chaînes de blocs.

# 2

## Chaînes de blocs

---

<b>2.1. Chaîne de blocs</b>	<b>17</b>
2.1.1. Définitions	17
2.1.2. Définition pour les élèves	19
2.1.3. Représentation	19
<b>2.2. Fonctionnement</b>	<b>21</b>
2.2.1. Exemple d'achat avec cryptomonnaie	22
2.2.2. Réseau	24
2.2.3. Bloc	26
2.2.4. Jeton (token)	28
2.2.5. Noeud	29
2.2.6. Adresse, clés privée et publique, et portefeuille électronique (wallet)	31
<b>2.3. Consensus</b>	<b>31</b>
2.3.1. Preuve de travail (proof of work, PoW)	32
2.3.2. Preuve d'enjeu (proof of stake, PoS)	35
2.3.3. Inattaquable (?)	36
2.3.4. Immuable, inaltérable	37
<b>2.4. Transactions</b>	<b>40</b>
2.4.1. Mempool	41
2.4.2. Confirmation, finalisation et exécution	41
2.4.3. UTXO et solde	43
<b>2.5. Contrat intelligent (smart contract, SC)</b>	<b>47</b>
2.5.1. Fonctionnement	49
2.5.2. Jeton non-fongible (non-fungible token, NFT)	50
2.5.3. SC NFT	50
2.5.4. SC PoS	55

---

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.

## 2.1. 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.

### 2.1.1. Définitions

Les définitions du dictionnaire Larousse [42] et du Merriem-Webster [39] 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 [32] 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" [49].

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 [56] 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" [51] 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, est répliqué (c'est-à-dire recopié) de manière décentralisée et de pair-à-pair, sur base volontaire, sur un grand nombre de noeuds 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.

Quelques remarques sur cette dernière définition :

- Le fait que la chaîne de blocs soit "sécurisée par des méthodes de cryptographie asymétrique" ne sera pas repris dans le modèle CR.
- Dans Bitcoin et Ethereum, les blocs ne sont pas de taille fixe.
- La CR tournera sur l'intranet du Collège Rousseau, mais peut être adaptée à tout type d'intranet.

Dans la philosophie de Nakamoto, fondateur de Bitcoin, une chaîne de blocs devrait être publique et transparente par essence [49]. A partir du moment où l'entrée des utilisateurs est contrôlée par des permissions, et/ou que les transactions sont approuvées par un tiers de confiance, alors il s'agit d'une base de donnée privée, au mieux infalsifiable et/ou distribuée, mais en tout cas pas décentralisée [14]. En ce sens, la chaîne de blocs BlocNote est privée, car seuls les utilisateurs de l'intranet du Collège Rousseau y ont accès.

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 attache dans le monde réel, et que sa valeur repose entièrement sur la confiance de ses utilisateurs.

Selon le NIST [53]<sup>1</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.

---

<sup>1</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. De plus, selon [32], une chaîne de blocs n'est pas une base de données car elle est administrée de manière collective.

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 [32], 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.

### 2.1.2. 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 utilisateurs de l'intranet du collège, dans lequel sont stockées toutes les transactions effectuées entre ses utilisateurs depuis sa création, sous forme de blocs non modifiables 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.

### 2.1.3. 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 2.3. 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 2.1 montre un exemple simplifié de registre sans les hachages.

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

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	0b101011010110	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	0b100010101100	04100	49011	45010

TABLE 2.1. – Exemple simplifié de registre d'une chaîne de blocs.

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 2.1 et dans les jeux pédagogiques discutés à la section 3.2.2. 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.

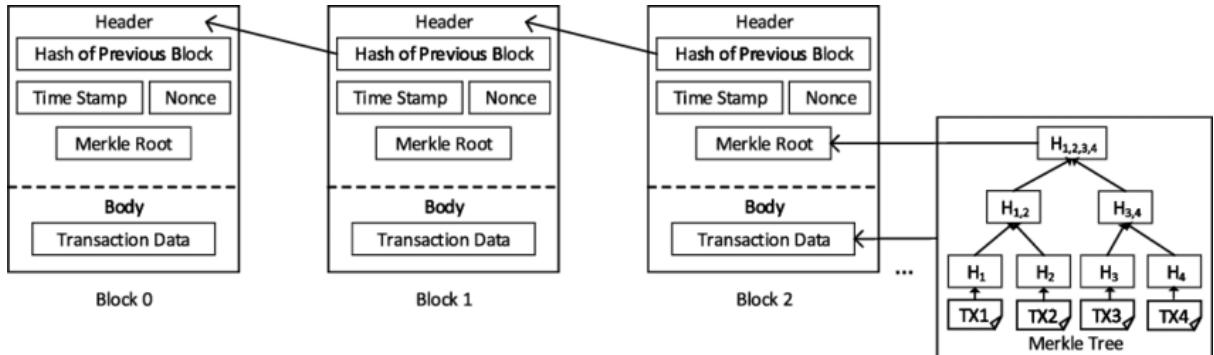


FIGURE 2.1. – Représentation graphique classique d'une chaîne de blocs [20].

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 2.2), montrant qu'un bloc peut être enchaîné à n'importe quel bloc-parent, n'importe où dans la chaîne, et qu'aucune chaîne ni bloc n'a de priorité sur une autre ; des fourches peuvent donc partir dans toutes les directions. Cette représentation permet aussi de mettre en évidence l'ordre logique des blocs.

Par exemple, la chaîne de blocs montrée à la figure 2.2 représente la même chaîne de blocs que le registre de la table 2.1. Elle comporte cinq fourches : 0-40-440-4440, 0-40-440-5440, 0-40-940-8940-88940, 0-90-590-5590 et 0-90-890. 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.

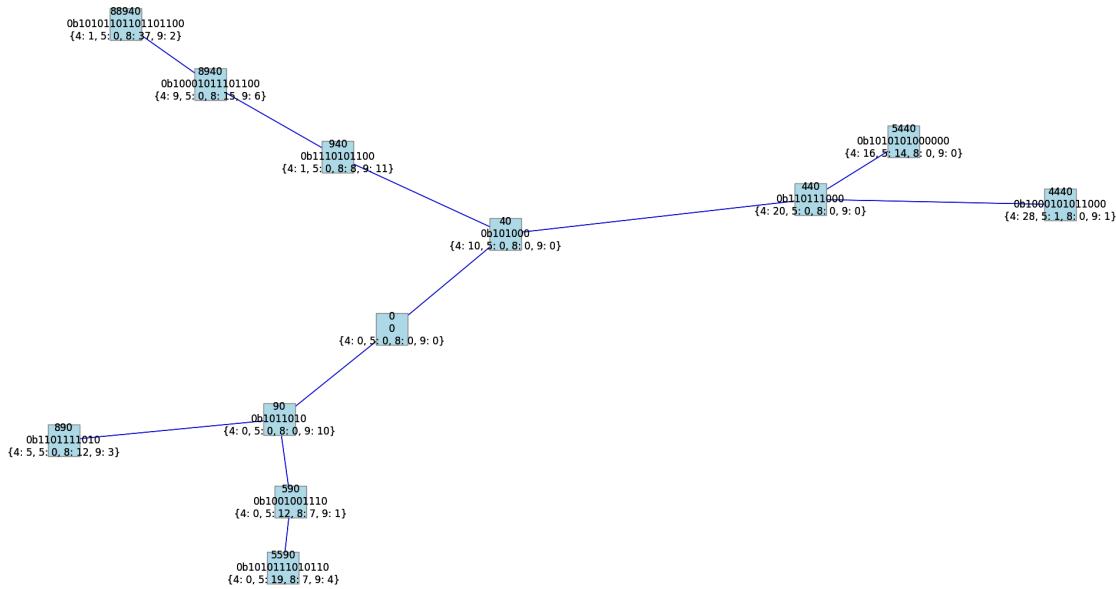


FIGURE 2.2. – Représentation graphique d'une chaîne de blocs BlocNote.

Nous verrons à la section 2.3 que seuls les blocs contenus dans la chaîne la plus longue sont conservés, tous les autres ne sont pas tenus en compte. Par exemple, les seules transactions qui seront exécutées dans la chaîne de blocs de la figure 2.2 sont celles contenues dans les blocs 0-40-940-8940-88940.

Notons qu'il existe deux types de fourches :

**fourche** résultant de la situation où plusieurs blocs ont la même profondeur ; ces fourches sont accidentnelles et temporaires, et sont résolues grâce au protocole de consensus (voir section 2.3).

**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.

## 2.2. 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 [?] 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 [58], les plateformes d’échange les plus performantes sont Kraken, Gemini et Crypto.com ; la figure 2.3 montre les plateformes d’échange selon leurs volumes, et la figure 2.4 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).

### 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 2.5, 2.6 et 2.7 :

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 2.5) ;
2. la plateforme d’échange soumet la transaction sur le réseau de la chaîne de blocs (figure 2.6) ;

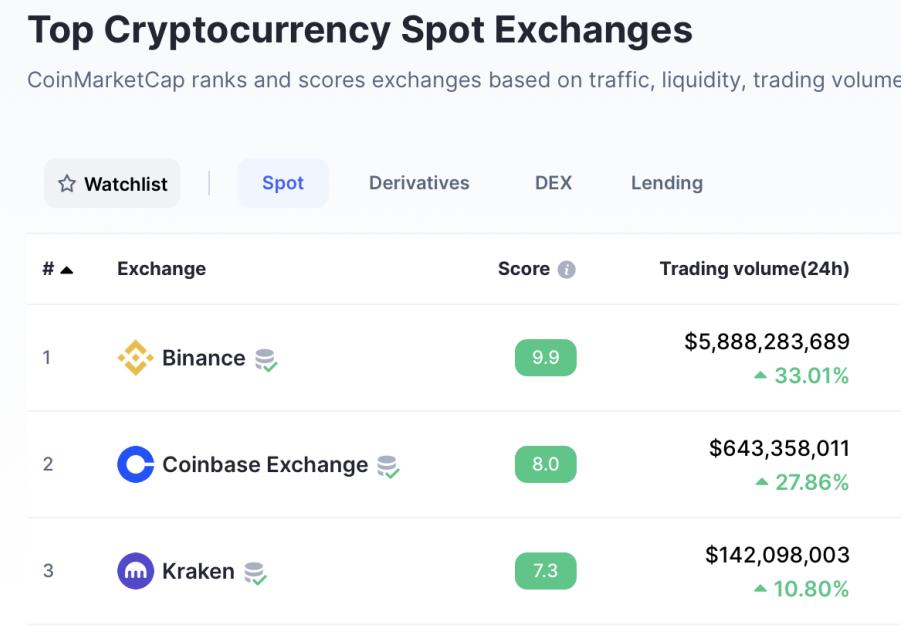


FIGURE 2.3. – Plateformes d'échange selon leurs volumes [6].



FIGURE 2.4. – Meilleures plateformes d'échange selon [64].

3. un noeud-mineur ou un noeud-validateur inclue la transaction dans un bloc, puis insère le bloc dans la chaîne de blocs (figure 2.7) ;
4. si le bloc fait partie de la chaîne la plus longue, la transaction est finalisée (figure 2.7) ;

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.

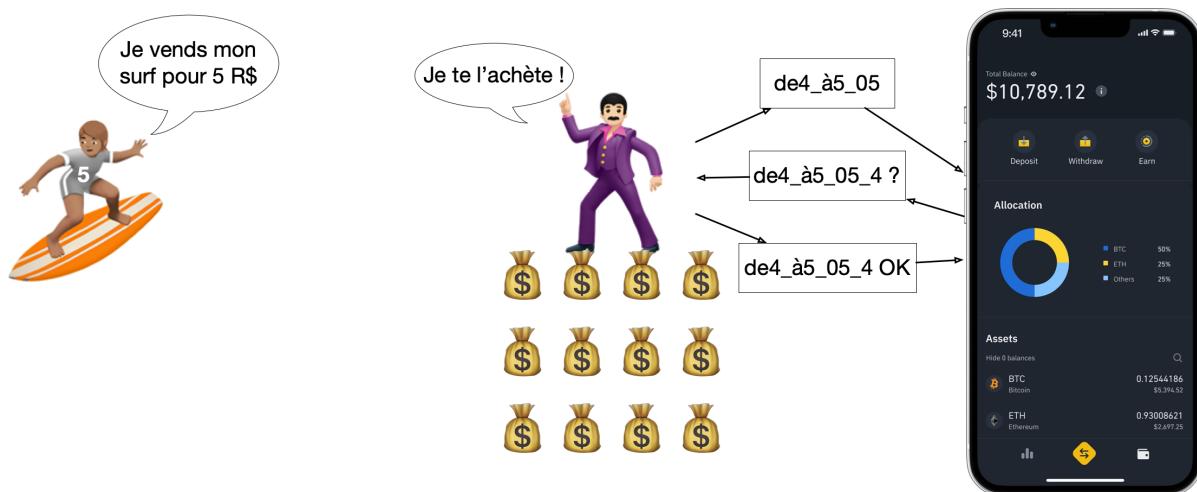


FIGURE 2.5. – La personne à l'adresse "5" vend son surf pour 5 R\$, et la personne à l'adresse "4" est intéressée à l'acheter. La personne "4" insère la transaction dans la plateforme d'échange de son choix, et la plateforme suggère d'ajouter 4 R\$ de frais de transaction. La personne "4" accepte.

## 2.2.2. 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 un but commun. La structure du réseau constituant une chaîne de blocs est minimale, comme l'était celui de l'internet à ses débuts : le réseau est constitué de noeuds bénévoles, les messages sont diffusés sur la base du meilleur effort (best-effort) de pair à pair (donc décentralisé et sans hiérarchie) grâce à des protocoles standardisés, et il est résilient : les noeuds peuvent joindre ou quitter le réseau quand ils veulent. Un nouveau noeud découvre ses pairs grâce à un ping-pong. La figure 2.8 illustre les réseaux Bitcoin et Ethereum.

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

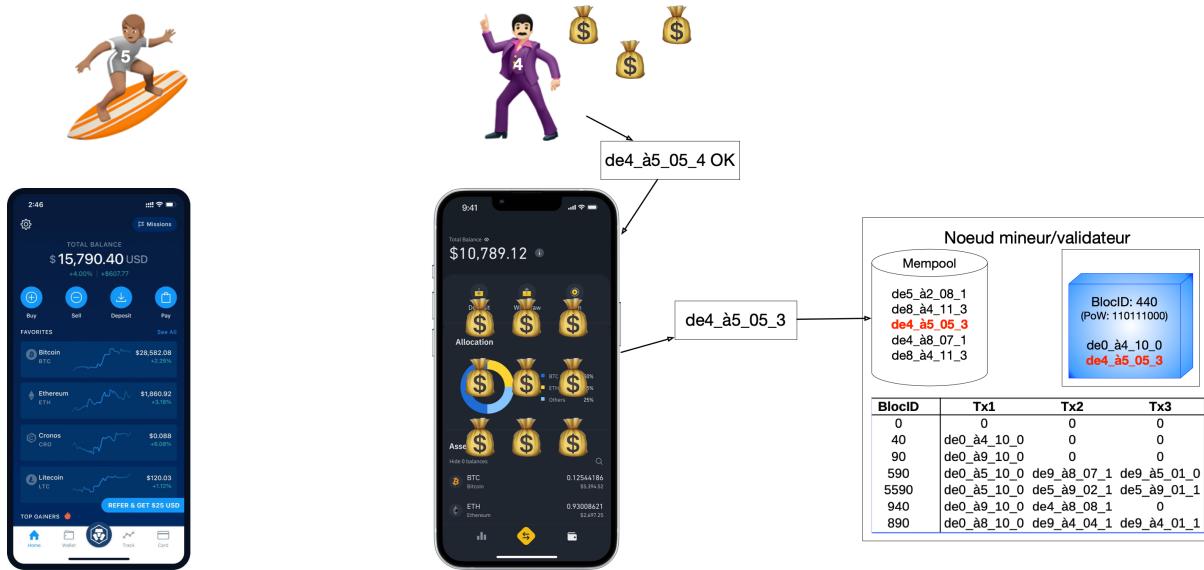


FIGURE 2.6. – La plateforme d'échange de la personne "4" gèle 9 R\$ de son compte, et soumet la transaction "de4\_à5\_05\_3" ("l'utilisateur "4" transfère 5 R\$ à l'utilisateur "5" et 3 R\$ de frais de transaction au noeud mineur ou validateur qui insère la transaction) au réseau de la chaîne de blocs. La plateforme d'échange encaisse 1 R\$ de commission au passage.

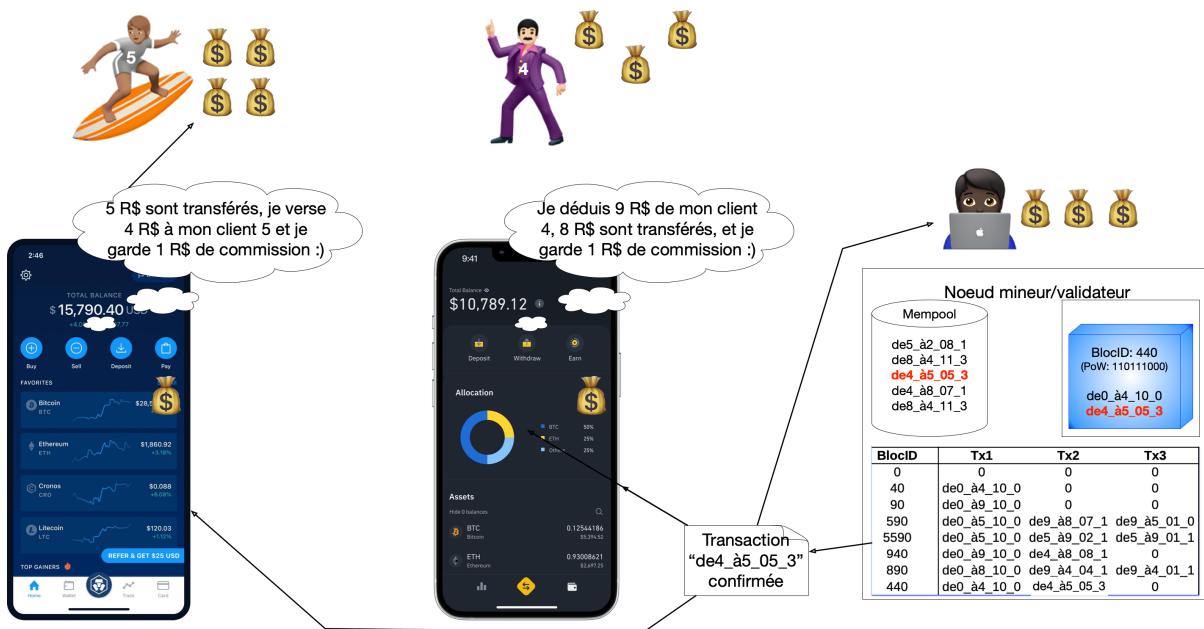


FIGURE 2.7. – Un mineur a inséré la transaction "de4\_à5\_05\_3" dans un bloc, qui a été finalisée : la transaction est donc exécutée. Le noeud-mineur empoché les frais de transaction de 3 R\$, la plateforme d'échange de l'acheteur garde 1 R \$ de commission, et la plateforme d'échange du vendeur reçoit 5 R\$. Cette dernière transfère 4 R\$ à la personne "5" et s'attribue une commission de 1 R\$.

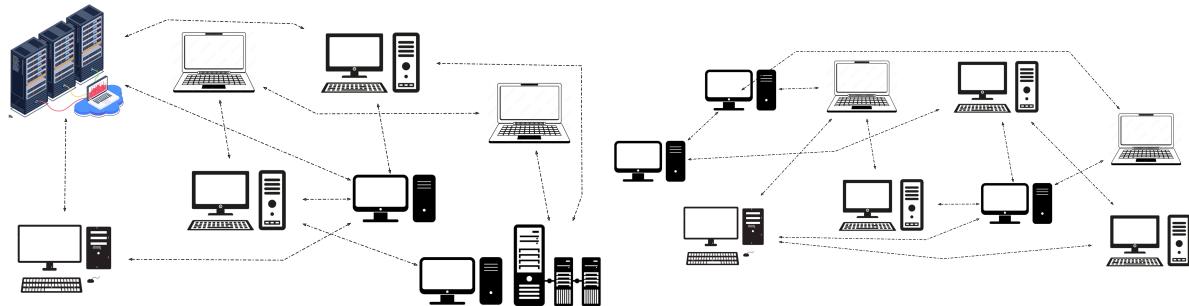


FIGURE 2.8. – Réseaux Bitcoin (à gauche) et Ethereum (à droite).

### Protocole de diffusion (broadcast protocol)

La transmission simultanée 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. Il n'y a pas de registre central, 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.
- 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. Nous verrons que les transactions et blocs finalisés par la chaîne de blocs sont ceux qui se trouvent sur le plus grand nombre de copies locales, et qu'un noeud doit parfois abandonner sa copie locale pour se mettre en rang avec ses pairs.

### 2.2.3. 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>2</sup> des transactions qu'il comporte,
- 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.

<sup>2</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 figure 2.9 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	
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
Hashes	
Hash	<a href="#">00000000d1145790a8694403d4063f323d499e655c83426834d4ce2f8dd4a2ee</a>
Previous Block	<a href="#">00000002a22cfbee1f2c846adbd12b3e183d4f97683f85dad08a79780a84bd55</a>
Next Block(s)	<a href="#">00000000c9ec538cab7f38ef9c67a95742f56ab07ba37c5be6b02808dbfb4e0</a>
Merkle Root	
Merkle Root	<a href="#">7dac2c5666815c17a3b36427de37bb9d2e2c5cc3f8633eb91a4205cb4c10ff</a>
Transactions	
F4184fc596403b9d638783cf57adfe4c75c605f6356fb91338530e9831e9e16	<a href="#">1Q2TWE3GMdB6BZKafqwxtnSzktFkuoG3eH</a>
2009-01-12 03:30:25	<a href="#">WAvgFt5Jvm3 10 BTC</a>
<a href="#">12cbQLTFMXRnSzktFkuoG3eH</a>	<a href="#">12cbQLTFMXRnSzktFkuoG3eH</a>
<a href="#">oMeFtpTu3S</a>	<a href="#">oMeFtpTu3S 40 BTC</a>

FIGURE 2.9. – 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 ([32]), 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.

### 2.2.4. 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 2.2.6). 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 2.5.2) 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 2.4).

Il s'agit d'une reconnaissance de la contribution d'une entité à la chaîne de blocs, et constitue la base de la sécurité (voir section 2.3.4).

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.

### 2.2.5. 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 2.10.

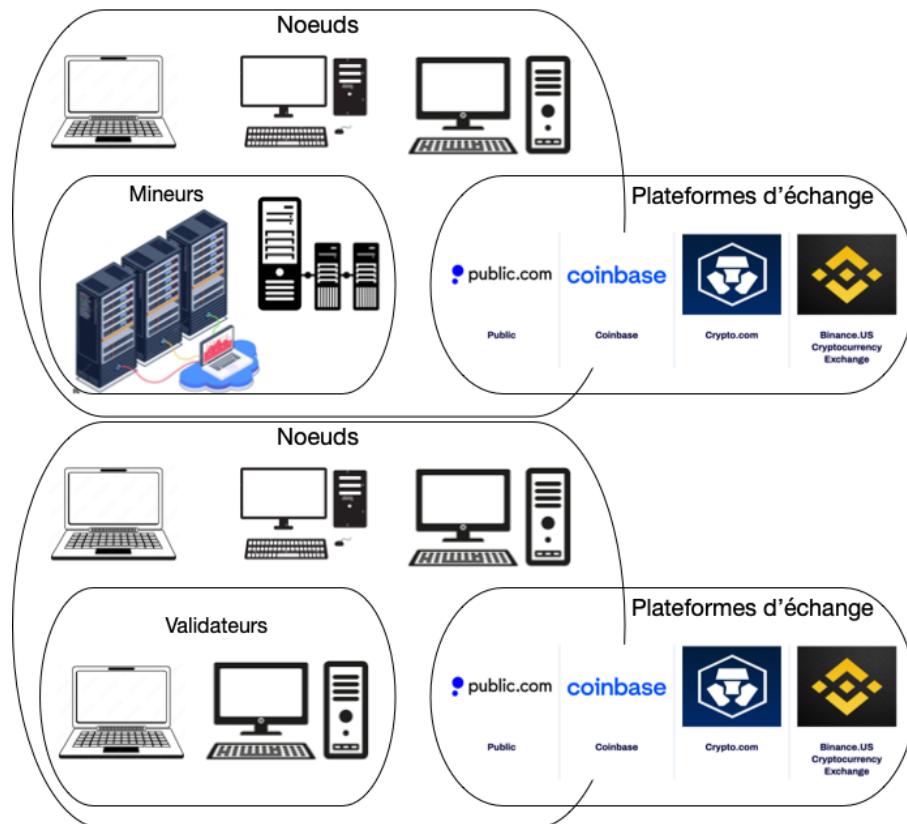


FIGURE 2.10. – Noeuds, mineurs ou validateurs et plateformes d'échange d'une chaîne de blocs basée sur la PoW (en haut) ou sur la PoS (en bas).

Au sein de la chaîne de blocs, le travail des noeuds se fait sur une base volontaire et ne rapporte aucune récompense, sauf lors de la finalisation d'un bloc. 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 2.11 illustre ces deux principaux types de noeuds.

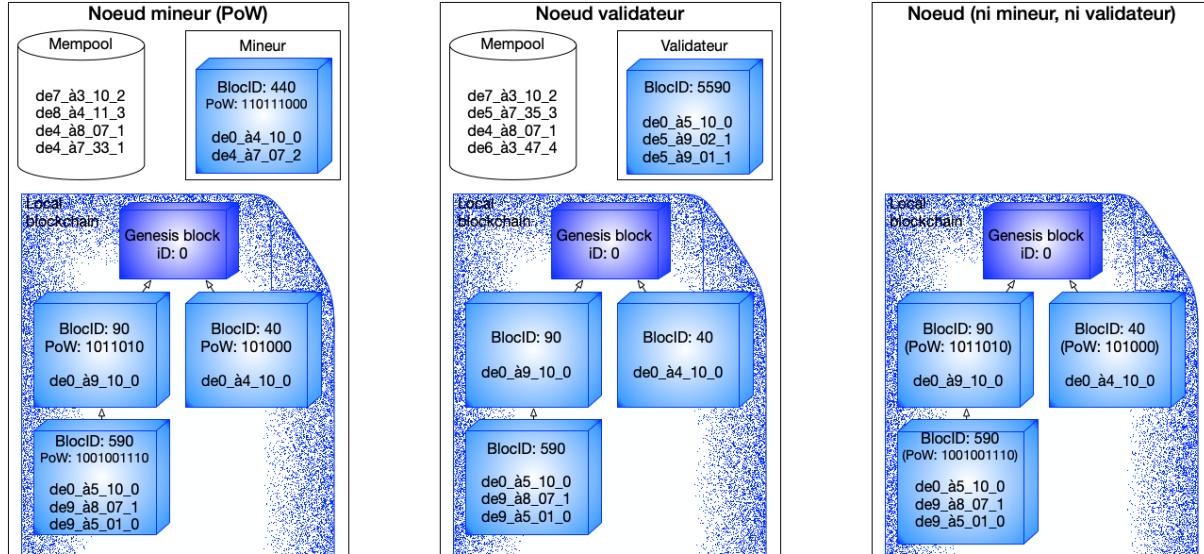


FIGURE 2.11. – Noeud-mineur (à gauche), noeud-validateur (au milieu) et noeud-spectateur (à droite).

Tous les noeuds (qu'ils soient mineur, ou validateur, ou pas) 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 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 doivent créer des nouveaux blocs et gérer 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 2.4.1) 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 transactions (vérification du solde du payeur), rejet des transactions invalides.

## 2.2.6. 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 l'adresse du vendeur à celle du destinataire. 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 son 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.

Notons que Bitcoin [49] 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 [33], le portefeuille ne contient que la clé privée associée à une clé publique.

Dans les chaînes de blocs supportant les contrats intelligents, 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.

## 2.3. Consensus

La chaîne de blocs étant un système distribué et asynchrone (les noeuds n'ont pas à attendre un accusé de réception (acknowledgement<sup>3</sup>) avant de poursuivre le calcul d'un

---

<sup>3</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].

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 2.2.2), 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 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" [34].

Il existe plusieurs algorithmes de consensus, les plus répandus étant la PoW et la PoS, décrits dans les sections 2.3.1 et 2.3.2. Ces deux algorithmes, ainsi que la plupart des algorithmes de consensus de chaîne de blocs, reposent sur le même processus, décrit ci-dessous.

## 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 2.12, 2.13, 2.14 et 2.15 illustrent le principe de consensus dans une chaîne de blocs :

1. au point de départ, toutes les copies locales sont identiques (figure 2.12) ;
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 2.13).
3. les noeuds diffusent les nouveaux blocs et ajoutent les blocs produits par leurs pairs à leurs copies locales respectives, qui redeviennent identiques (figure 2.14) ;
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 2.15).

*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.

### 2.3.1. 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 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

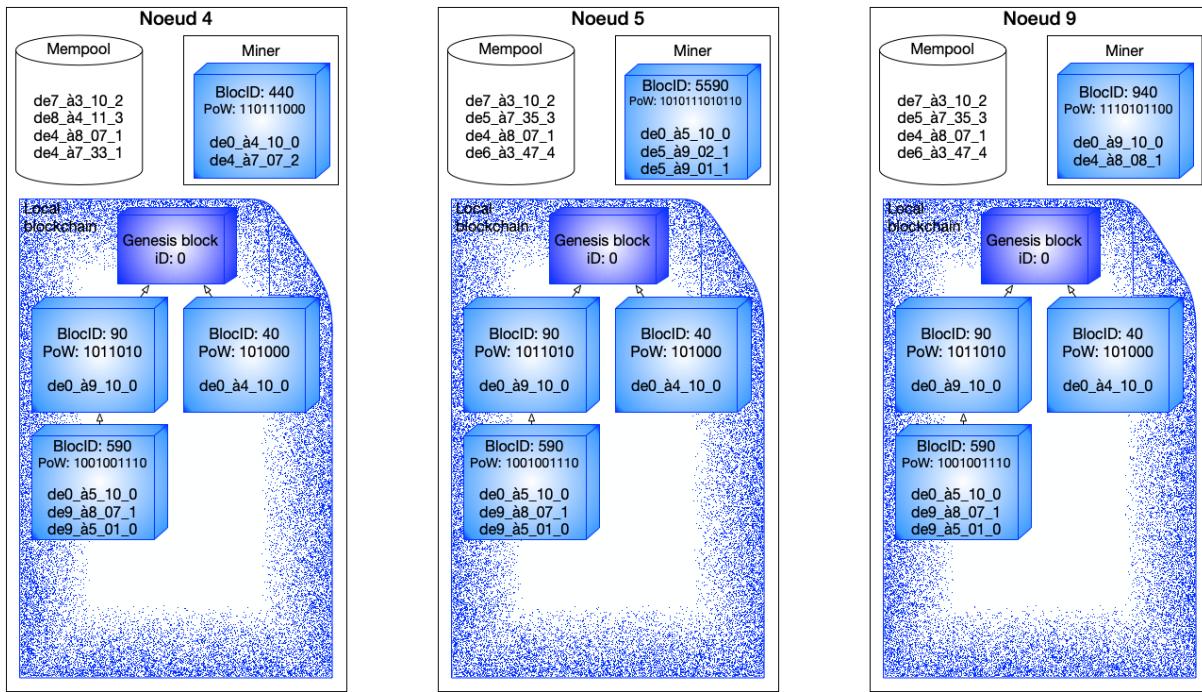


FIGURE 2.12. – Trois noeuds-mineurs participant à une chaîne de blocs possédant des copies locales identiques.

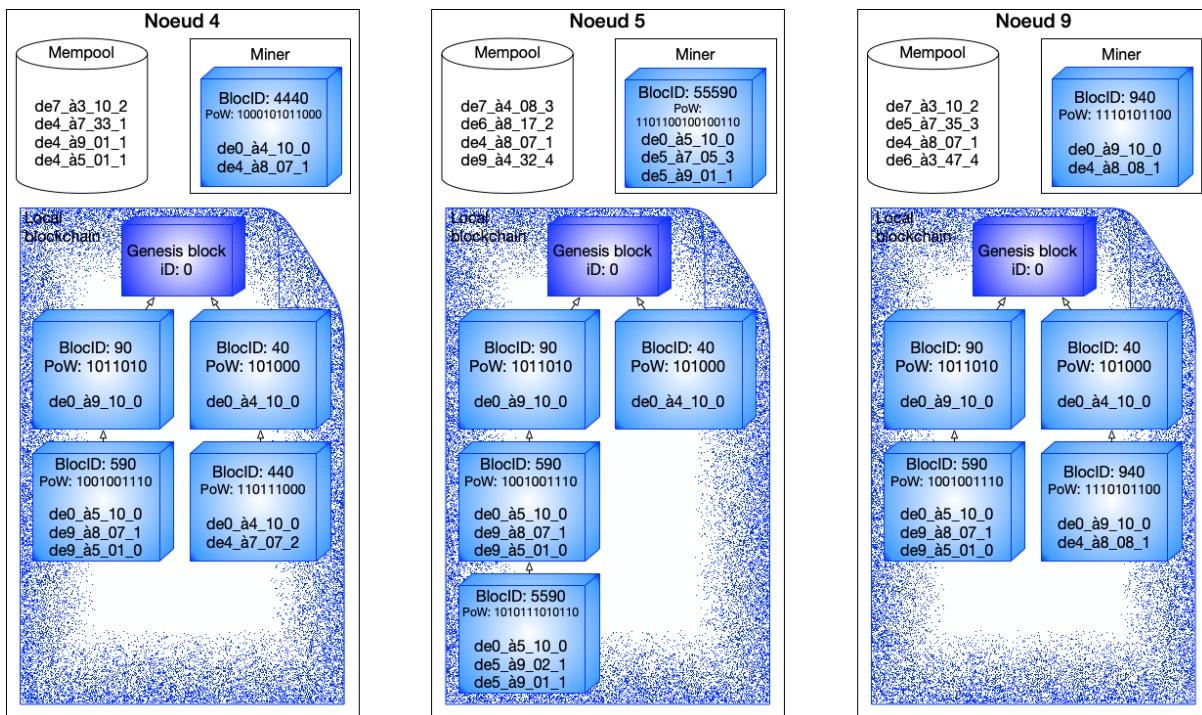


FIGURE 2.13. – Chaque mineur calcule un nouveau bloc et l'intègre à sa copie locale ; des versions différentes de la chaîne de blocs coexistent alors sur le réseau.

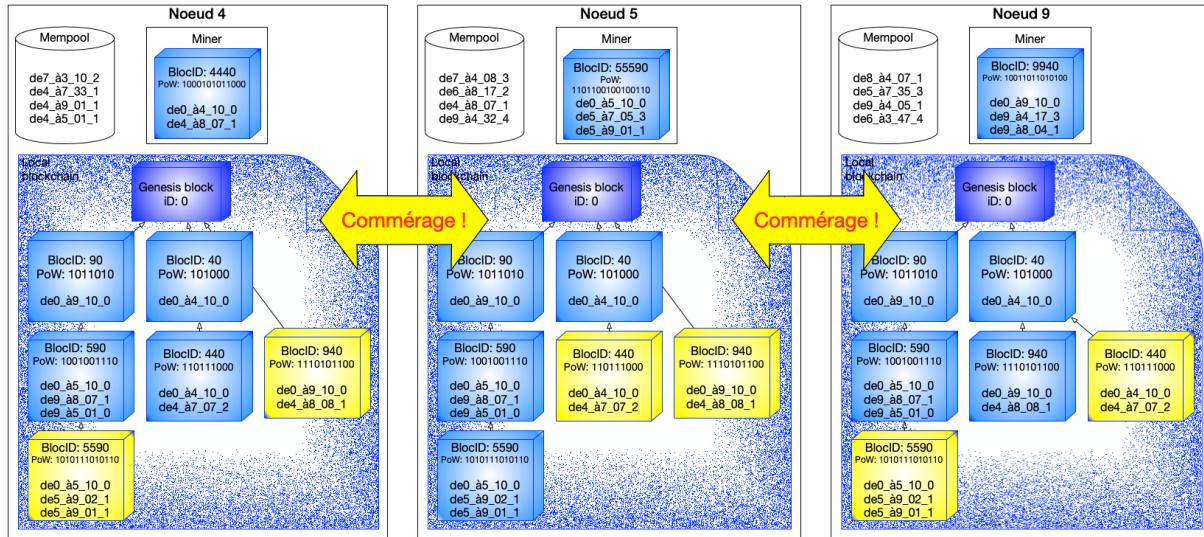


FIGURE 2.14. – Les noeuds échangent les nouveaux blocs et les intègrent à leurs copies locales.

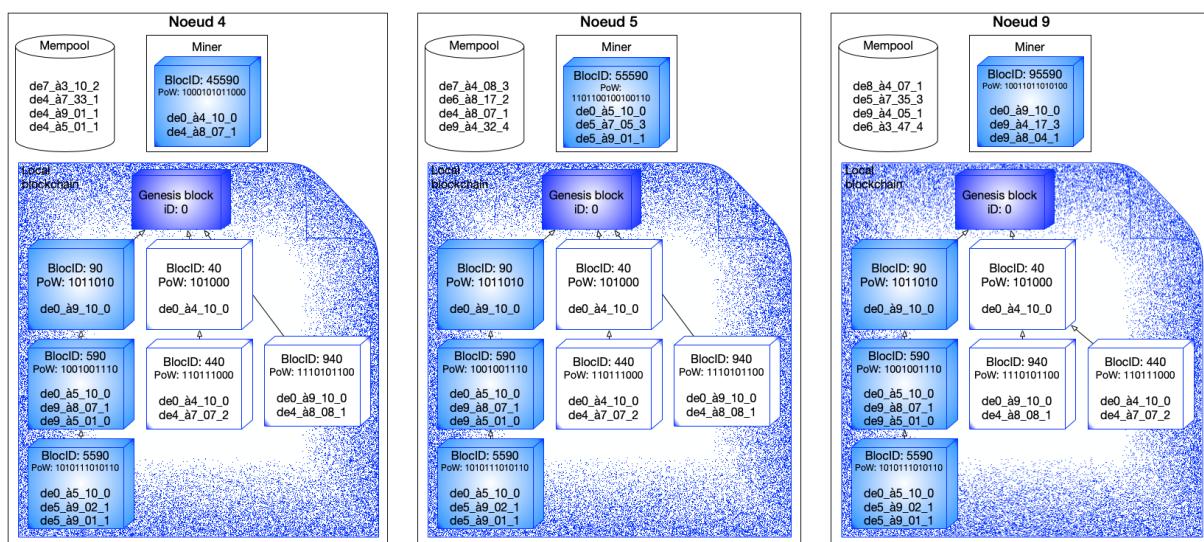


FIGURE 2.15. – La chaîne 0-90-590-5590 étant la plus longue, il est très probable que le prochain bloc miné sera attaché au bloc 5590.

section 2.3.4) ; 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 [21]. 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 3.2.1).

### Mineur

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.

### 2.3.2. 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 2.5) [21].

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. 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).

Selon Ethereum<sup>4</sup>, la PoS montre de nombreux avantages par comparaison à la PoW [21] :

- 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 ;

---

<sup>4</sup>(Traduction libre de l'autrice.)

- 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.

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. Dans Ethereum, le rythme de la PoS est divisé en époques, chacune composée de 32 créneaux ("slots") de 12 secondes. Dans chaque créneau, un validateur est sélectionné et propose un bloc, et un comité de validateurs est choisi pour déterminer la validité du bloc proposé. Les validateurs sont groupés en comité de 128 entités, dont une propose le nouveau bloc et 127 le vérifient. Les blocs qui obtiennent des votes représentant au moins deux tiers des ETH mis en garantie sont confirmés. Les validateurs valident les blocs de la chaîne en envoyant un vote (appelé attestation) en faveur d'un nouveau bloc sur le réseau. 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. [21] [17]

### 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.

### 2.3.3. Inattaquable (?)

Comme tout système informatique, les chaînes de blocs sont vulnérables aux attaques par déni de service,urre d'identité<sup>5</sup> (ou attaque Sybil) et ingénierie sociale. L'attaque à 51% et la double-dépenses sont quant à elles endémiques aux chaînes de blocs.

A cause de l'anonymat des chaînes de blocs, la difficulté pour contrer les attaques par déni de service et parurre d'identité réside dans la différenciation des sources de requêtes légitimes vs malicieuses. La solution consiste à imposer un coût économique à ce genre

---

<sup>5</sup>Dans un système sans permission, un attaquant peut gagner une influence démesurée en créant un grand nombre d'identités.

d'attaque : la mise en garantie de ressources (puissance de calcul (PoW) ou garantie en jetons (PoS)) et des frais de transaction.

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. Ethereum combat la centralisation des ressources en récompensant les blocs Ommers, afin d'inciter les noeuds plus faibles à continuer à participer à la surveillance de la chaîne. L'attaque à 51% rend la double-dépense possible et défait le caractère immuable de la chaîne de blocs.

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

La double-dépense est une brèche endémique aux chaînes de blocs, qui découle du fait qu'elles soient 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, puis 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.

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". L'utilisateur "4" soumet une autre transaction "de4\_à6\_05\_3" qui utilise les mêmes jetons, cette fois pour acheter un skateboard à l'utilisateur "8". Il insère cette transaction dans le bloc 590, 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 590 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.

#### **2.3.4. Immutable, inaltérable**

La technologie des chaînes de blocs garantit son immuabilité et son inaltérabilité 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.

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.

Deux exemples d'altération sont donnés ci-dessous.

### 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 2.16) ;
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 2.17) ;

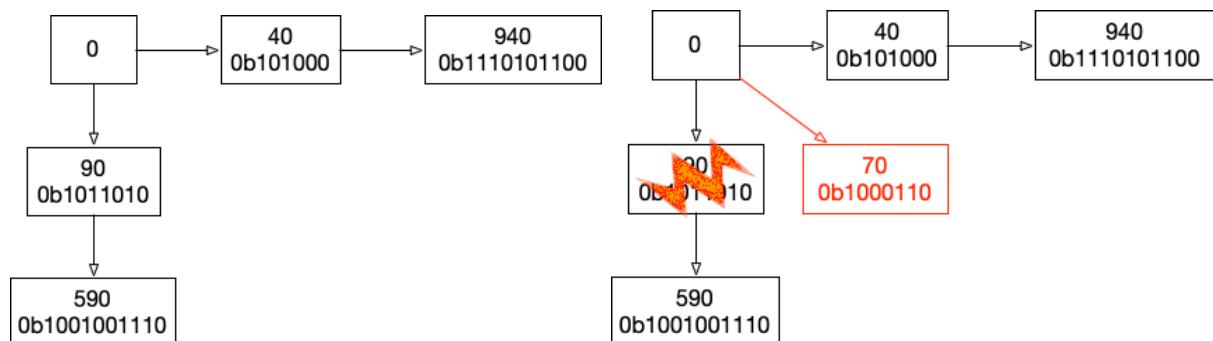


FIGURE 2.16. – A gauche : la chaîne de blocs au moment où le noeud 7 tente de l'altérer. A droite : le noeud 7 modifie la numérotation du bloc 90 et recalcule la PoW pour qu'elle corresponde au numéro de bloc 70.

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 :

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

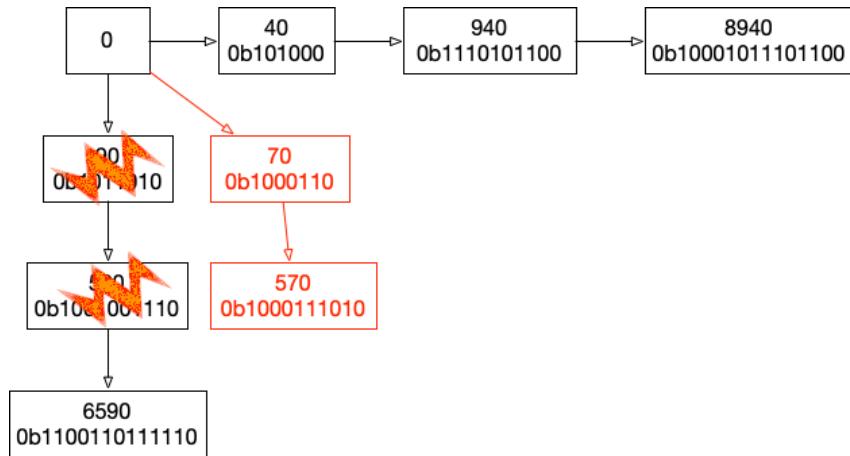


FIGURE 2.17. – Le noeud malveillant doit altérer les blocs subséquents. Les chaînes 0-90-590-6590 et 0-40-940-8940 grandissent plus rapidement que la chaîne modifiée 0-70-570.

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	10	0	0	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	10	16	24	5
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	24	7	15
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
		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 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.

## 2.4. 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 2.5).

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 2.18 montre une transaction sur la chaîne de blocs Bitcoin telle que visualisée grâce à un "explorateur de blocs". La transaction contient les champs suivants : le numéro d'identification de la transaction sous forme de hachage (1), l'adresse de l'expéditeur (2), les frais de transaction (3), l'adresse du destinataire (4) et le statut de la transaction (5) (détaillé dans la section 2.4.2). Une transaction comporte aussi la signature cryptographique de l'émetteur de la transaction pour autoriser le transfert des jetons [46] [69].



FIGURE 2.18. – Transaction telle que représentée par l'explorateur de blocs Bock Explorér [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'utilisateur "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.

### 2.4.1. 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 2.2.2 et figure 2.19).

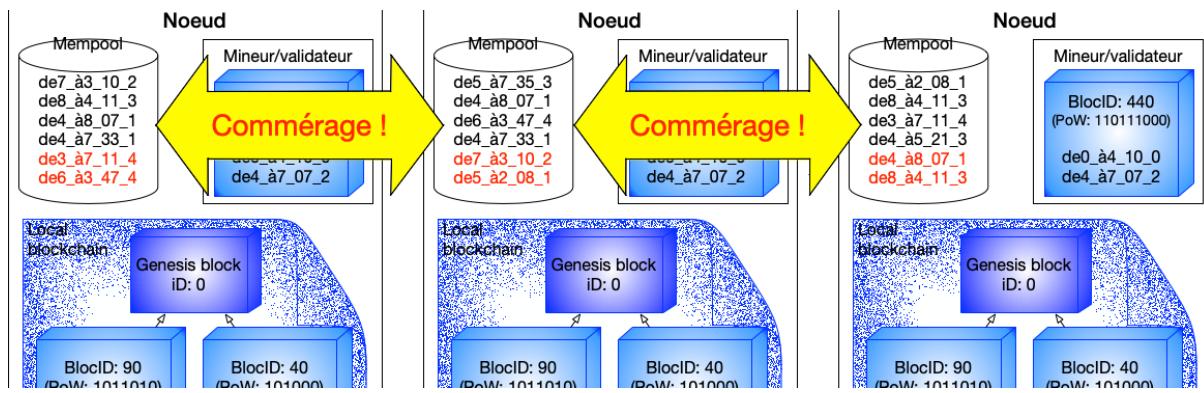


FIGURE 2.19. – Les noeuds diffusent les transactions sur le réseau.

Les mineurs et validateurs donnent la priorité aux transactions dont les frais de transaction sont les plus élevés [71]. 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 2.21) ;
- sinon, la transaction est invalide et éliminée du mempool (figure 2.20),

### 2.4.2. Confirmation, finalisation et exécution

Les transactions peuvent avoir l'un des statuts suivants :

- 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** lorsque les jetons sont transférés de l'adresse du vendeur à celle du destinataire.

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

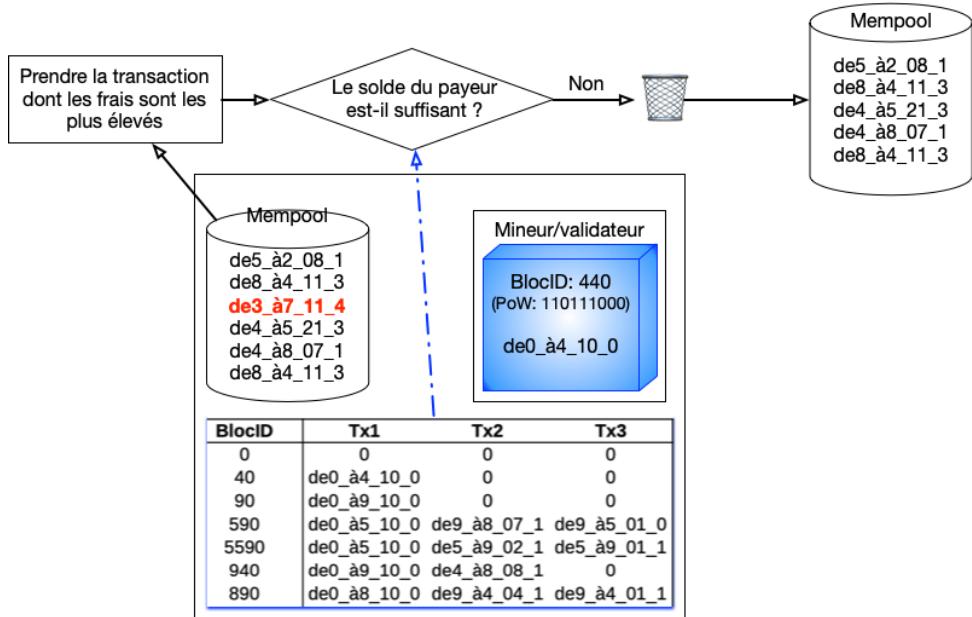


FIGURE 2.20. – Le noeud choisit la transaction dont les frais de transaction sont les plus élevés, puis vérifie si le solde du payeur est suffisant selon sa copie locale de la chaîne de blocs. Si le solde du payeur n'est pas suffisant, la transaction est rejetée et enlevée du mempool.

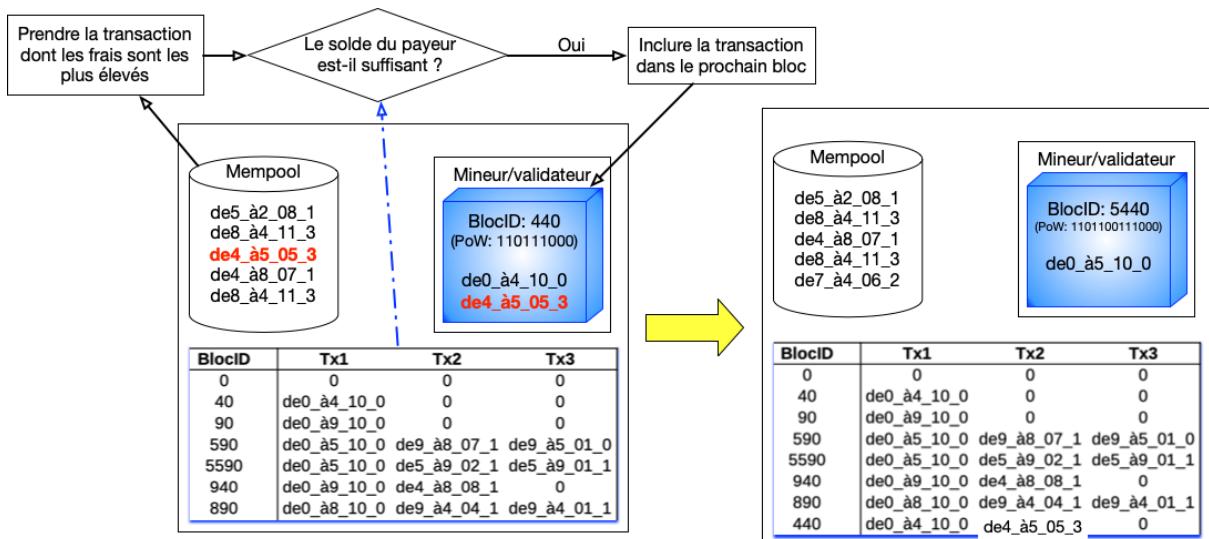


FIGURE 2.21. – Si le solde est suffisant, la transaction est inclue dans le nouveau bloc.

finalisées, il faut attendre que le consensus soit atteint, i.e. qu'une décision soit prise quant à déterminer quelle chaîne est conservée.

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.

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.

### 2.4.3. UTXO et solde

La figure 2.22 illustre les deux types de registre d'une chaîne de blocs :

- 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 est semblable à celui montré dans la table 2.7, qui reprend les blocs de la table 2.1, auxquels sont ajoutés le solde de chaque utilisateur après chaque transaction.
- 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.



FIGURE 2.22. – Représentation des registres de type "compte bancaire" et "UTXO" du point de vue de l'utilisateur.

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

La raison pour laquelle les chaînes de blocs utilisent un registre de UTXO plutôt que de tenir un registre des soldes de chacun des utilisateurs est simplement pour économiser de l'espace de stockage. La tenue d'un registre de type "compte bancaire" est plus rapide pour consulter un solde, mais utilise davantage d'espace de stockage, et pour vérifier l'origine des jetons, il faut de toute façon remonter la chaîne.

### 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 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.

A titre d'exemple, la table 2.5 montre les entrées et sorties correspondant aux transactions des tables 2.1 et 2.7.

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. La table 2.6 montre une des fourches extraite de la chaîne de blocs des tables 2.5, 2.1 et 2.7. Dans cette 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\$.

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 [45]. Par exemple, dans la fourche 0-40-940-8940-88940 montrée à la table 2.6, pour exécuter la transaction "48071" (signifiant que l'utilisateur "4" paye 7 R\$ à l'utilisateur "8" et paye 1 R\$ de frais de transaction au noeud-mineur ou validateur "7") 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.

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 output: 7à8	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 2.5. – Entrées (input) et sorties (output) correspondant aux transactions de la chaîne des tables 2.1 et 2.7.

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 output: 7à8	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 2.6. – Chaîne extraite de la table 2.5.

Note : dans un vrai 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).

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 2.7 reprend l'exemple de la chaîne 0-40-940-8940-88940 de la table 2.6 : 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.

<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 2.7. – Détermination du solde associé à une adresse donnée.

## 2.5. 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 [63] et AIMultiple [27] 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 of Things, IoT) : par exemple, lorsque la quantité de savon à lessive est trop bas, la machine à laver passe automatiquement la commande.

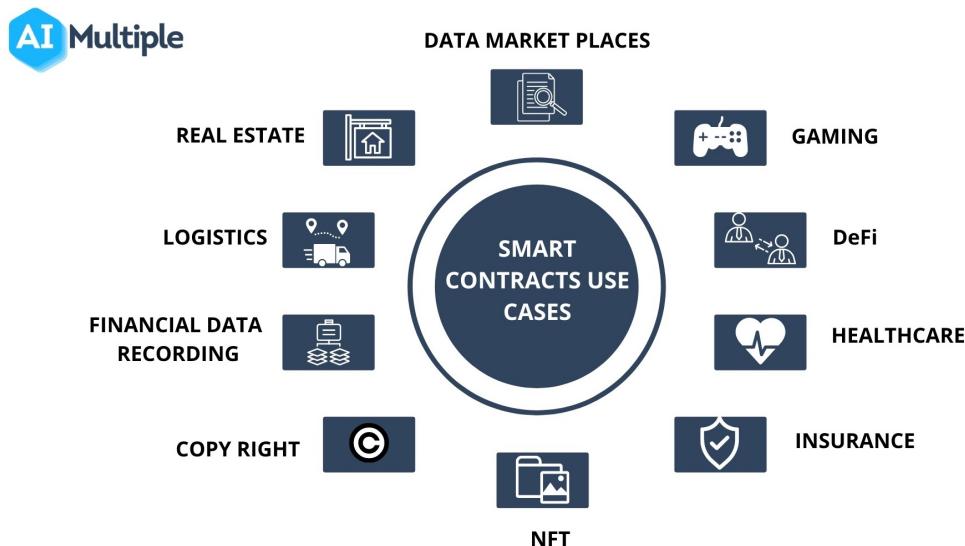


FIGURE 2.23. – Meilleurs exemples d'utilisation de contrats intelligents en 2023 selon [27].

De Filippi [32] 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" [51] 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" [51] 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" [42].

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

## 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 [32], et des redevances sur l'utilisation d'oeuvres 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.

## 2.5.1. 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, la machine virtuelle est appelée EVM (Ethereum Virtual Machine) (voir figure 2.24).

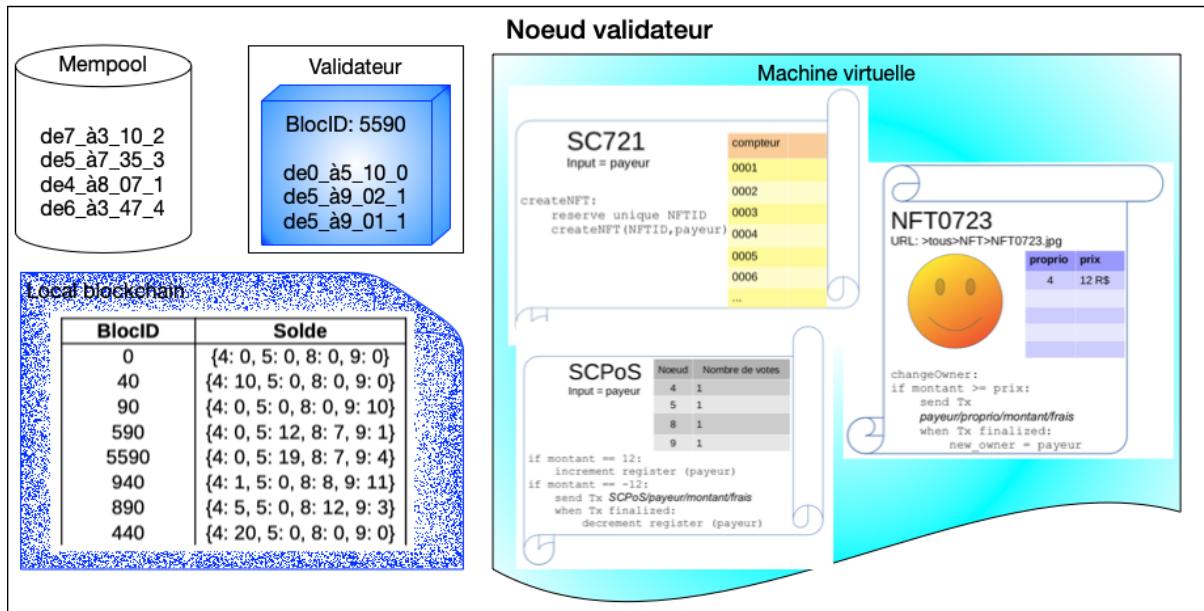


FIGURE 2.24. – Les noeuds de la chaîne de blocs Ethereum exécutent les SCs sur l’EVM (Ethereum Virtual Machine).

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'une 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 (voir section 2.5.3) peut coûter entre 400 et 1000 chf [66]. 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. Sur Ethereum, le gaz est de l'ordre de 60 chf en moyenne pour créer un NFT avec un SC de création de NFT existant.

## 2.5.2. 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 [22].

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

## 2.5.3. SC NFT

Deux SCs de gestion de 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 [72], dont voici quelques fonctions [61], [72] :

```
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.
```

Le processus est le suivant :

1. le créateur du NFT soumet une transaction dont le destinataire est SC ERC-721, et dont le montant correspond au gaz nécessaire pour l'exécution du SC ERC-721. L'URL où se trouve l'oeuvre digitale et son prix de vente sont inclus dans les métadonnées de la transaction (figure 2.25).
2. La transaction est diffusée dans le réseau de la chaîne de blocs, et éventuellement insérée dans un bloc (voir figure 2.26).
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 2.27).

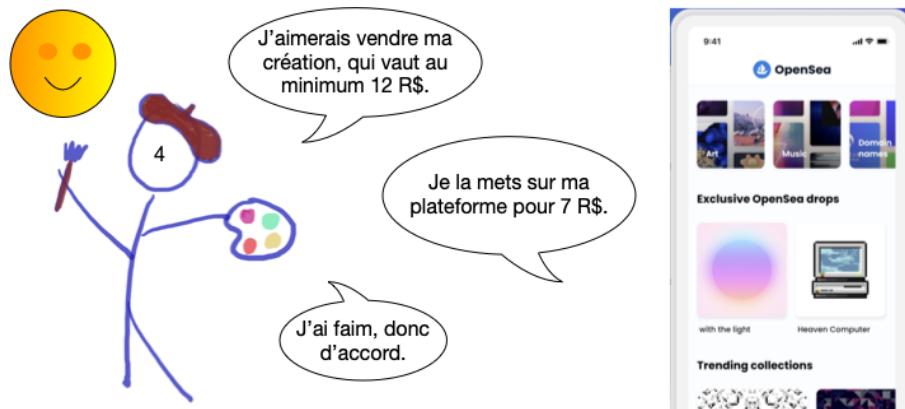


FIGURE 2.25. – L'utilisateur associé à l'adresse "4" a réussi à insérer un bloc dans la chaîne de blocs, et a donc gagné 10 R\$. Il en profite pour créer un NFT contenant sa dernière création artistique pour la mettre en vente. Pour ce faire, il doit payer un gaz de 5 R\$ pour faire tourner le SC ERC-721, et 1 R\$ pour que sa transaction soit choisie par un validateur.

## 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 l'acheteur, qui comporte trois fonctions :

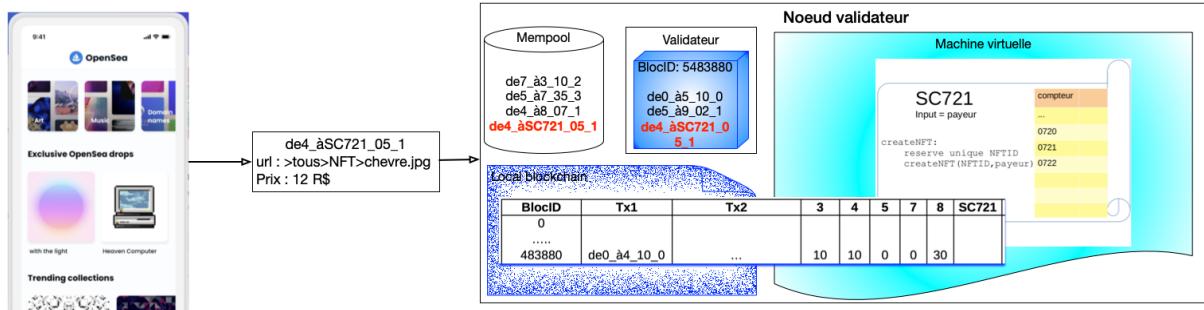


FIGURE 2.26. – La transaction "de4\_àSC0721\_05\_1" est diffusée sur le réseau. Le validateur associé à l'adresse "5" choisit cette transaction pour l'insérer dans le prochain bloc, dont le numéro est 5483880.

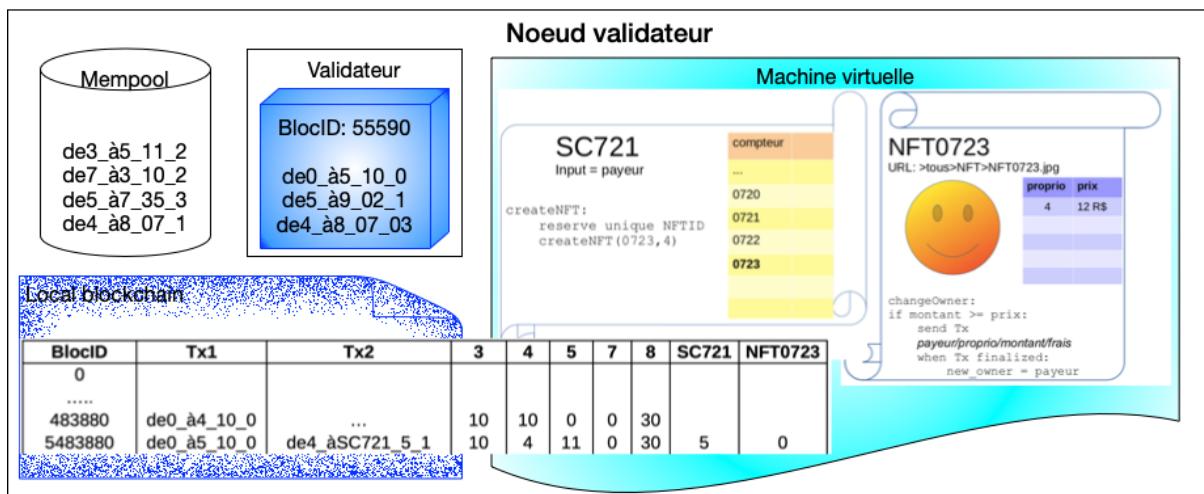


FIGURE 2.27. – Le bloc 5483880 a été finalisé, la transaction "de4\_àSC0721\_05\_1" est exécutée : le SC721 crée un nouveau SC "NFT0723" dont le propriétaire est l'utilisateur "4" et le prix de vente est de 12 R\$.

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. L'acheteur soumet une transaction dont 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 2.28). Note : pour simplifier, le gaz nécessaire à l'exécution du SC NFT est inclus dans les frais de transaction.
2. Une fois la transaction finalisée (figure 2.29), l'exécution du SC NFT est déclenchée (figure 2.30) :
  - a) les jetons de l'acheteur sont transférés à l'adresse du SC NFT,
  - b) le SC NFT envoie une nouvelle transaction pour que les jetons soient transférés au vendeur,
  - 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 2.28. – L'utilisateur associé à l'adresse "8" souhaite acheter le NFT0723. Il 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.

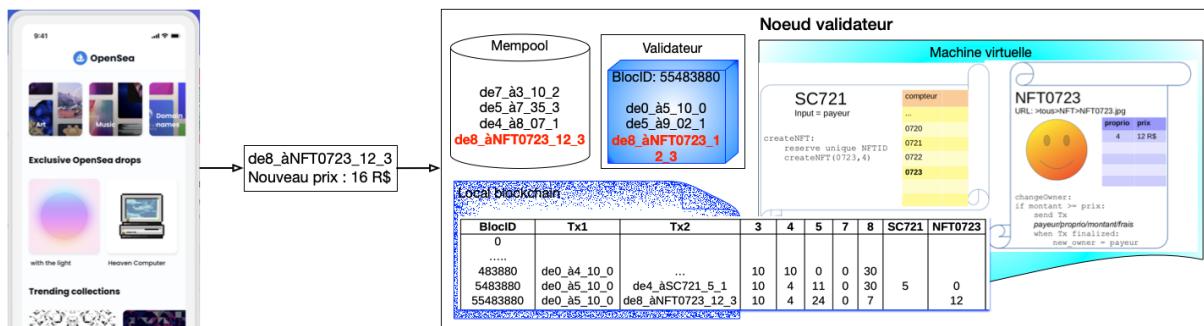


FIGURE 2.29. – La transaction "de8\_àNFT0723\_12\_3" est envoyée, puis choisie par le validateur "5" pour l'insérer dans le bloc 55483880.

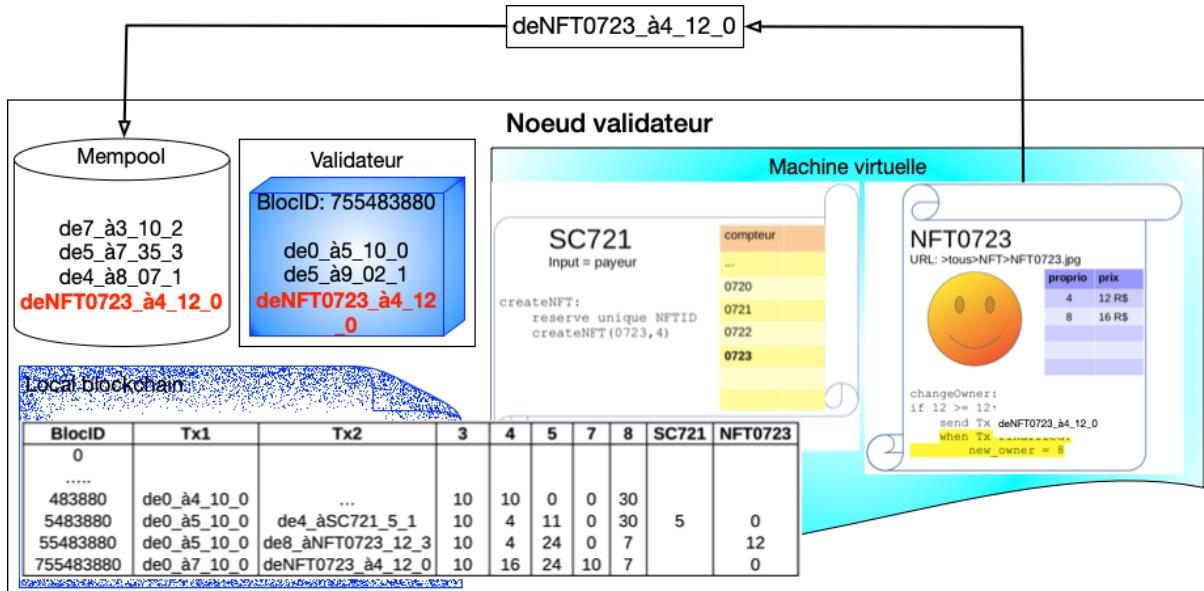


FIGURE 2.30. – Le bloc 55483880 a été finalisé, la transaction "de8\_àNFT0723\_12\_3" est exécutée : le NFT0723 affiche maintenant le nouveau propriétaire ("8") et le nouveau prix de vente (16 R\$), et envoie la transaction "deNFT0723\_à4\_12\_0" pour payer les 12 R\$ de la vente à l'utilisateur "4". Le validateur "7" choisit cette transaction et l'insère dans le bloc 755483880.

## 2.5.4. 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 :

- gérer la durée de vie de la mise en garantie ;
- rembourser la garantie à la demande du propriétaire ;
- gérer les récompenses et punitions du propriétaire.

W

# 3

## Transposition didactique

---

<b>3.1. BlocNote . . . . .</b>	<b>57</b>
3.1.1. Valeurs . . . . .	57
3.1.2. Modèle . . . . .	58
<b>3.2. Modalités . . . . .</b>	<b>64</b>
3.2.1. Jeu pédagogique (instructional games) . . . . .	66
3.2.2. Revue des jeux pédagogiques . . . . .	67
3.2.3. Activité déconnectée... ou pas . . . . .	73
<b>3.3. Découpage des activités . . . . .</b>	<b>74</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, qui reprend les savoirs discutés au chapitre 2 en un modèle adapté aux élèves de maturité gymnasiale, et en deuxième partie sont discutées les modalités et la mise en place d'activités pour les élèves.

### 3.1. 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.

#### 3.1.1. 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 façon suivante :

- L'utilisation excessive d'énergie pour le calcul de la PoW de Bitcoin (discutée dans la section 1.2.3) est inacceptable à notre époque. La première activité de la CR illustre le consensus avec la PoW, mais celle-ci peut rapidement être ignorée dès la deuxième activité.

- La PoW se veut utile, comme plusieurs auteurs l'ont déjà proposé [70], [16], [44] [35] [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 discutée parmi 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 chaîne de blocs n'élimine pas le problème de confiance, mais le distribue sur plusieurs acteurs. Il devient nécessaire de faire confiance à l'ensemble des participants. [32] (p108)
- la transparence :
  - la chaîne de bloc est affichée aux yeux de tous, modifiable 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 la CR 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 le 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, mais peut sembler injuste d'un point de vue démocratique, car les plus faibles sont moins rétribués.

Le fait que quiconque puisse participer à la chaîne de blocs ou se retirer à tout moment est plus 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 à permission, ce qui la rend moins vulnérable aux attaques par leurre d'identité.

### 3.1.2. 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 assez 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 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 1.2.4), 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.
- Dans la CR, il n'existe pas de moyen de vérifier si un noeud malveillant a introduit une transaction payable par 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.

L'enchaînement de blocs, la PoW, le registre, les transactions, le registre de la CR, 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 2.2), 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 la CR, 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 2.3.4.

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 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.

La cible du calcul reste relativement imprévisible car il n'est pas possible de prévoir quelle fourche sera la plus longue, et donc quel sera le numéro du prochain bloc. Cependant, des élèves futés pourraient calculer les PoWs d'avance. 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 de la CR

Le registre de BlocNote n'est basé ni sur les UTXOs, ni sur les soldes des utilisateurs (voir section 2.4.3). 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 2.1 montre le registre de BlocNote, et le tableau 2.22 montre le même registre, mais cette fois avec les soldes des comptes des utilisateurs.

## Ordre chronologique vs parentalité

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 n'étant 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 3.1 montre les mêmes données que celles des

tables 2.1 et 2.22 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 3.1. – 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	0b10101101101101100	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}

## 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.

Les SC NFT associés aux œuvres physiques et digitales sont affichés sur le tableau d'affichage appelé "machine virtuelle".

Le gaz pour la création d'un NFT est fixé à 5 R\$. Par exemple, l'utilisateur "4" soumet la transaction "de4\_àSC0721\_05\_1", qui signifie que l'utilisateur "4" paye 5 R\$ de gaz pour créer un NFT, et 1 R\$ de frais de transaction au mineur.

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. Par exemple, le prix du NFT0723 a été fixé à 12 R\$ par son propriétaire actuel, l'utilisateur "4".

1. 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,
2. le SC NFT émet automatiquement 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.

Pour simplifier le mécanisme, aucun frais de transaction ne sont prévus pour le transfert des R\$ du SC NFT au vendeur du NFT.

## 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 donnant une ballotte<sup>1</sup> pour le tirage au sort du prochain validateur.

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.

---

<sup>1</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].

## 3.2. 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 1.3, 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 [68] : "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 3.1 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>2</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 d'expérimenter la latence entre le calcul et la diffusion de nouveaux blocs.

### Travail de groupe

La chaîne de blocs de la CR 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 expérimenter le protocole de consensus, le travail de groupe devient nécessaire. Cette modalité est d'ailleurs rendue obligatoire par les prescriptions du programme.

<sup>2</sup>Kinesthésie : perception des déplacements des différentes parties du corps.

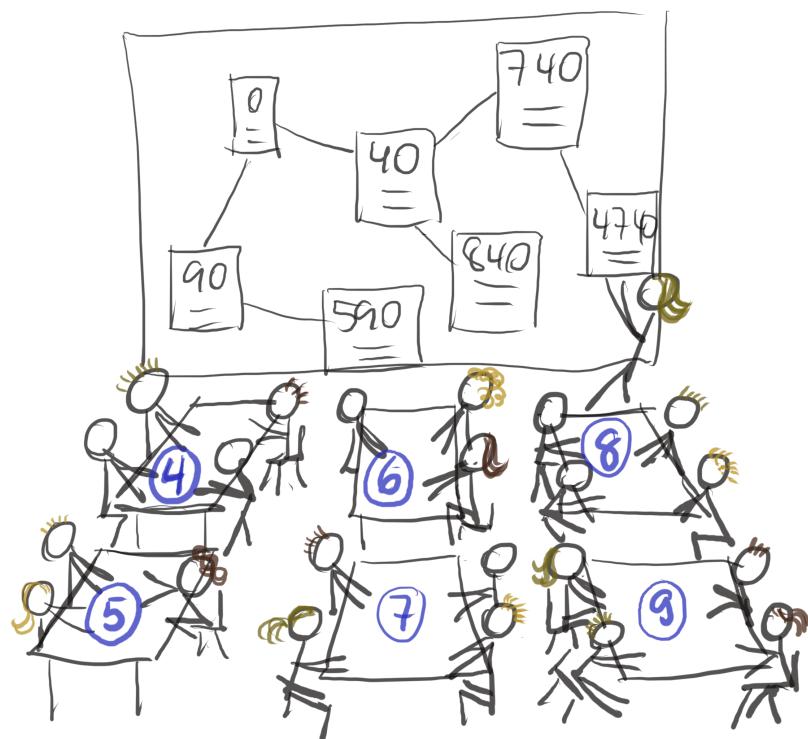


FIGURE 3.1. – Dans la mise en oeuvre des activités, des groupes de quatre élèves forment les noeuds, et les blocs et les transactions qu'ils contiennent sont affichés au mur.

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 [57] 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.

### 3.2.1. Jeu pédagogique (instructional games)

Dans son rapport technique, Hays [37] 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 [37], qui date de 2005...

La chaîne de blocs CR revêt les caractéristiques d'un jeu pédagogique telles que définies par Hays [37] :

- 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.

### 3.2.2. 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 3.2). Il s'agit de la simulation de gestion la plus ancienne et la plus utilisée, maintenant disponible en ligne [52].

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" [28] (voir figure 3.3), 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 la CR.

#### Jeu sur ordinateur

Bloxxgame [26] (voir figure 3.4) 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



FIGURE 3.2. – MIT Sloan School Beer Game

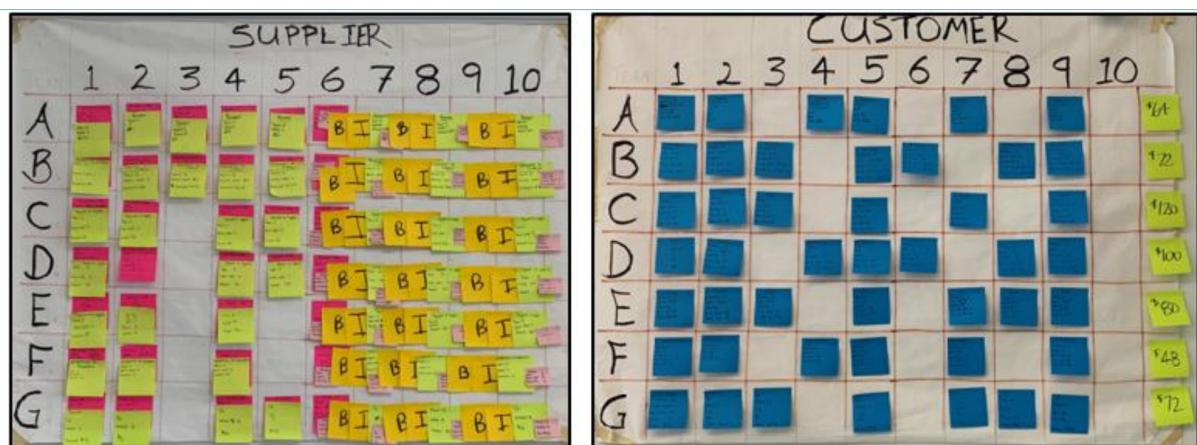


FIGURE 3.3. – Blockchain Paper Game

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 .

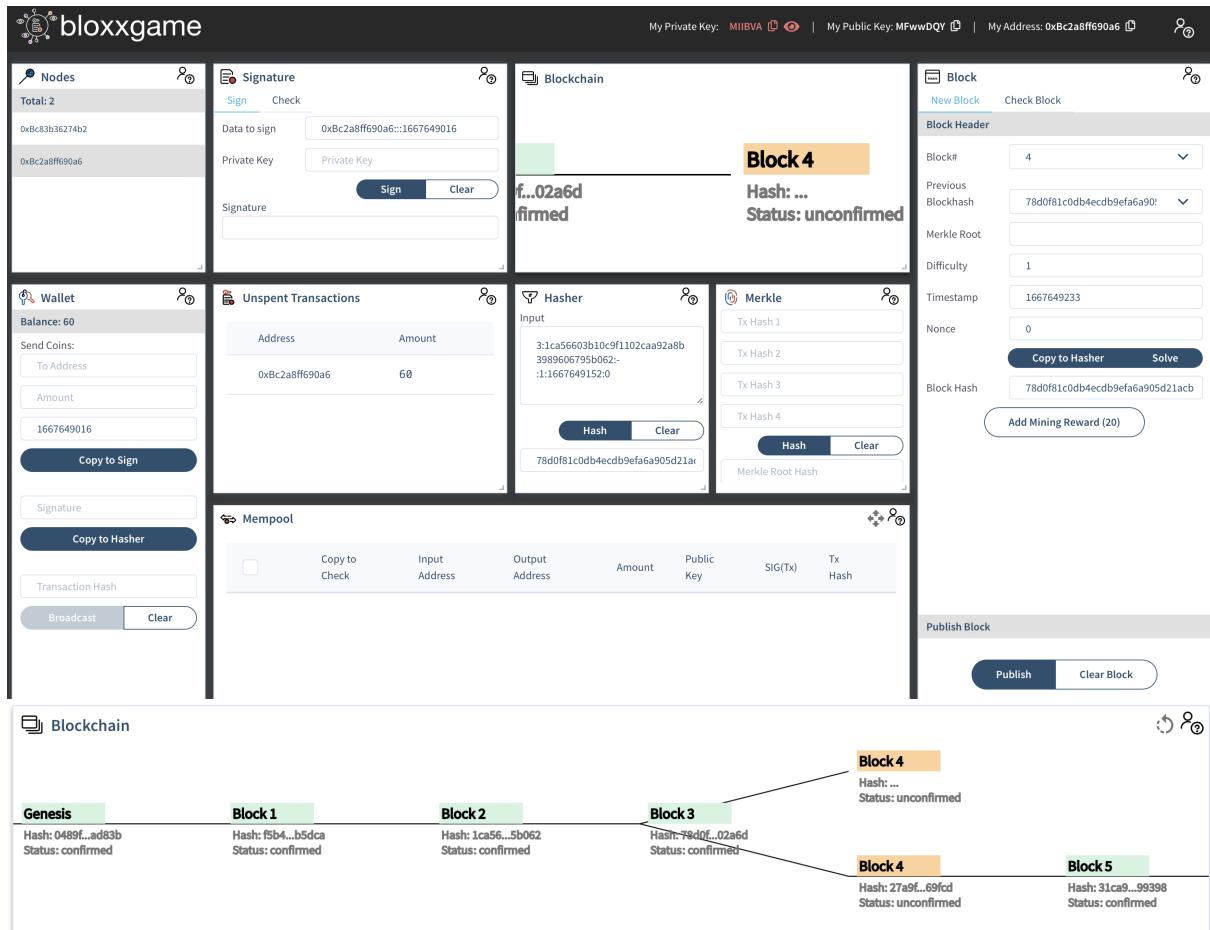


FIGURE 3.4. – Bloxxgame [https ://bloxxgame.io/play/](https://bloxxgame.io/play/)

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 3.5 et 3.6) développé par Christianson [18] 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 .

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

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 = Value of the first letter of the course

b = Value of the first letter of the student Public

Key

c = Value of the Grade

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

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

FIGURE 3.5. – Blockchain Game : tableau à imprimer pour aider les élèves à calculer la PoW.



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	204
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 3.6. – Blockchain Game : tableur pour compiler les résultats.

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" [37], 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 [36]. Les étudiants travaillent en groupes de trois pour calculer 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 3.7). 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.

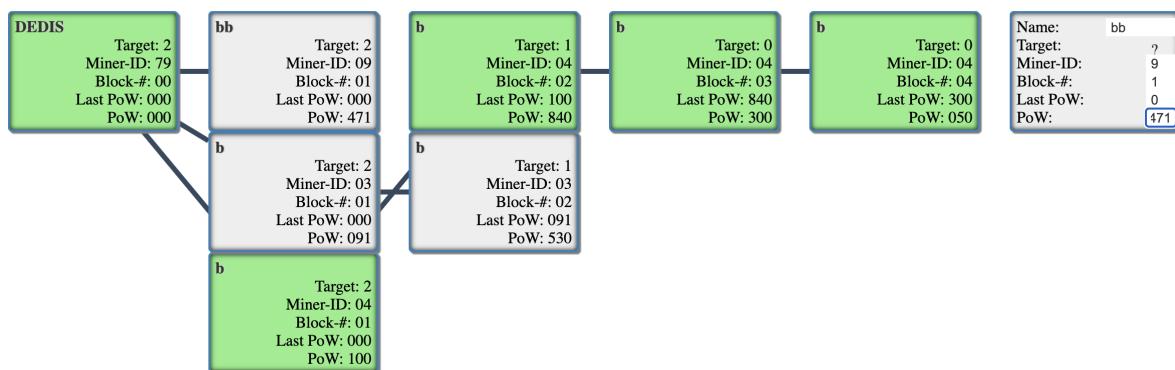


FIGURE 3.7. – Activité de chaînes de blocs de Linus Gasser [36].

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 [54] 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 dans un programme Python (voir figure 3.8). L'activité prévoit aussi une discussion sur les NFT, mais sans mise en activité.

Mes collègues Daniel et Yves [30] 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 3.9).

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.

## 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.

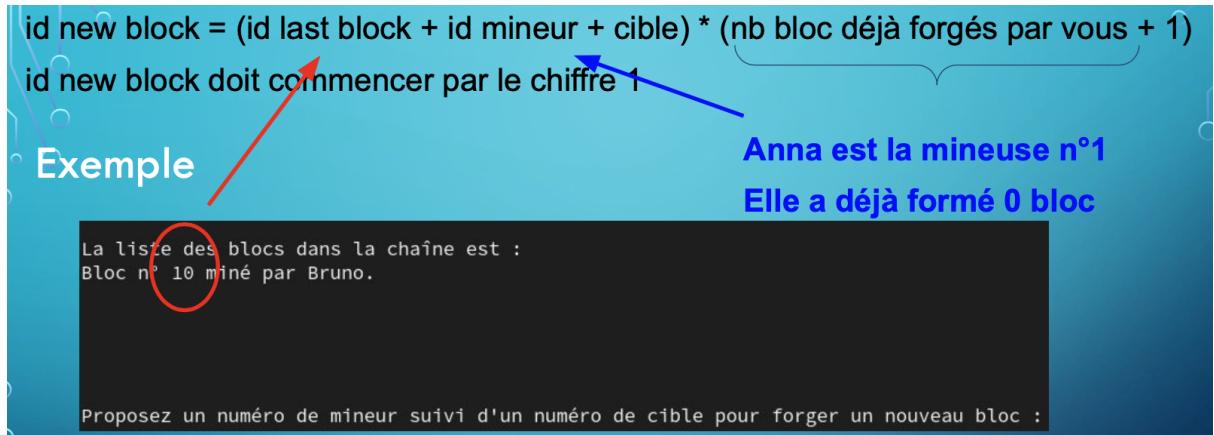


FIGURE 3.8. – Activité de chaînes de blocs de Thierry et Olivier [54].

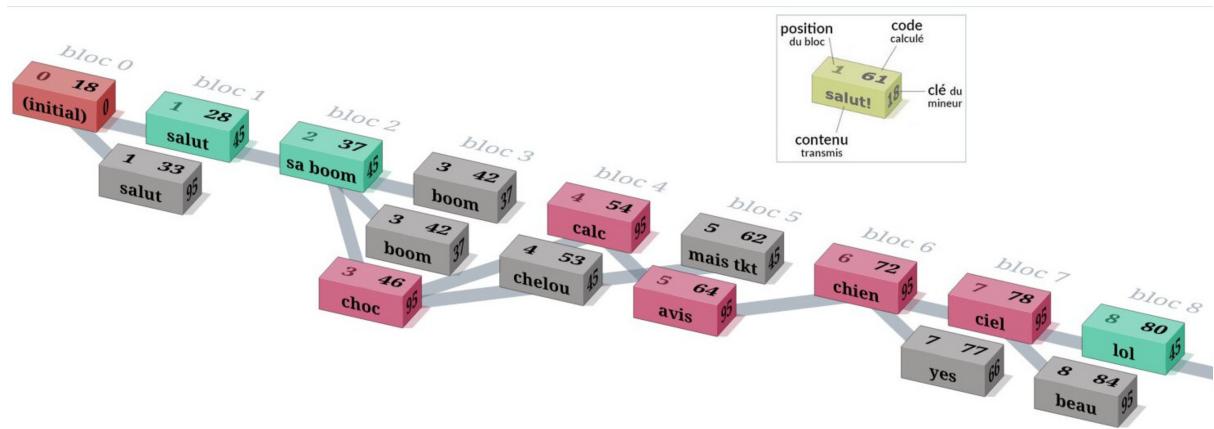


FIGURE 3.9. – Activité de chaînes de blocs de Daniel et Yves [30].

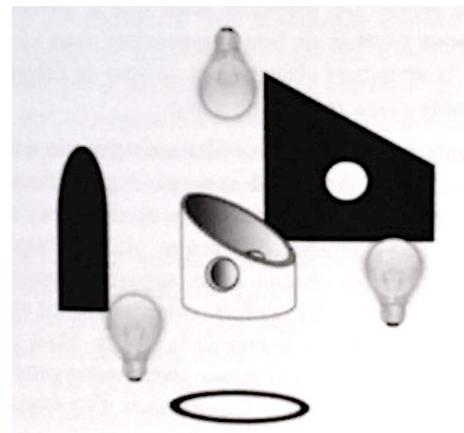
### 3.2.3. Activité déconnectée... ou pas

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 car elle permet 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 constitue donc une façon de 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 3.2.2) .

Le fait d'approcher les chaînes de blocs d'abord par le jeu, 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" [57].



Trois éclairages différents d'un même objet [57].

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.
- Il est aussi possible de transformer l'activité 2 de façon à ce que chaque noeud possède sa copie locale de la chaîne de blocs, et que les noeuds partagent les nouveaux blocs (sans affichage de la chaîne de blocs en entier), afin d'illustrer ce qui se passe dans le réseau d'une chaîne de blocs, où chaque noeud (voir chapitre 4).
- Une activité avec Bloxxgame (voir section 3.2.2) compléterait le tableau brossé par la CR et ses programmes en ajoutant la cryptographie et la simultanéité des différents mécanismes.

### 3.3. 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, après 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 3.2 résume les thèmes et contenus de chaque activité.

TABLE 3.2. – 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> </ul>
Activité 2	<p>consensus : seuls les blocs de la plus grande chaîne sont considérés</p> <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> </ul>
Activité 3	<p>absence de tiers de confiance ... démocratie ?</p> <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> </ul>
Activité 4	<p>vérification de la validité des transactions</p> <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> </ul>
Activité 5	<p>achat et vente de NFTs avec les SC NFT</p> <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>

# 4

## Programmes Python

---

<b>4.1. Présentation générale . . . . .</b>	<b>76</b>
4.1.1. Niveau . . . . .	77
4.1.2. Principes . . . . .	77
4.1.3. Architecture . . . . .	78
<b>4.2. Description des programmes . . . . .</b>	<b>78</b>
4.2.1. bchain_rousseau.py . . . . .	78
4.2.2. bchain_interac.py . . . . .	81
4.2.3. bchain_sim.py . . . . .	83
4.2.4. bchain_visualization.py . . . . .	83
<b>4.3. Activités in silico . . . . .</b>	<b>84</b>
4.3.1. Par où commencer ? . . . . .	84
4.3.2. Support aux activités . . . . .	85
4.3.3. Algorithme de diffusion et consensus . . . . .	85
<b>4.4. Mais bon, ma foi, c'est comme ça. . . . .</b>	<b>86</b>

Dans ce chapitre, les programmes de BlocNote sont présentés de façon à aider un futur utilisateur à les utiliser.

### 4.1. 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, car les noeuds ne sont jamais modifiés. Les mécanismes suivants sont représentés :

- création des blocs et des transactions ;
- l'enchaînement des blocs est basé sur la PoW, mais pourraient facilement être modifiés pour représenter la PoS ;
- vérification des transactions → calcul des soldes des utilisateurs.

Les contrats intelligents ne sont pas implémentés.

### 4.1.1. 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. Ce programme écrit en Python représente fidèlement la structure des blocs, mais ne peut pas être utilisé dans BlocNote car 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.

Des programmes 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.

### 4.1.2. 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 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 :

- liste\_noeuds, qui est déterminée au départ,
- et liste\_blocs, qui au départ est vide, puis est incrémentée avec chaque nouveau bloc créé.

Les programmes complets sont donnés en annexe B.

La chaîne de blocs produite par le programme bchain\_sim.py ou bchain\_interac.py est stockée dans un fichier .csv. Chaque ligne contient de 5 à 6 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 ...).

Des exemples sont montrés dans les tables 2.1 et 2.22.

### 4.1.3. Architecture

Quatre programmes accompagnent la CR : 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 la CR 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 contenant tous les blocs de la chaîne de blocs, avec le même format, de type .csv. Ces fichiers .csv peuvent être ouverts avec un tableur (voir tables ...) ou visualisés grâce à bchain\_visualization.py, qui produit une représentation graphique en format .png comme celles montrée à la figure ....

La figure 4.1 représente l'architecture des programmes de la CR.

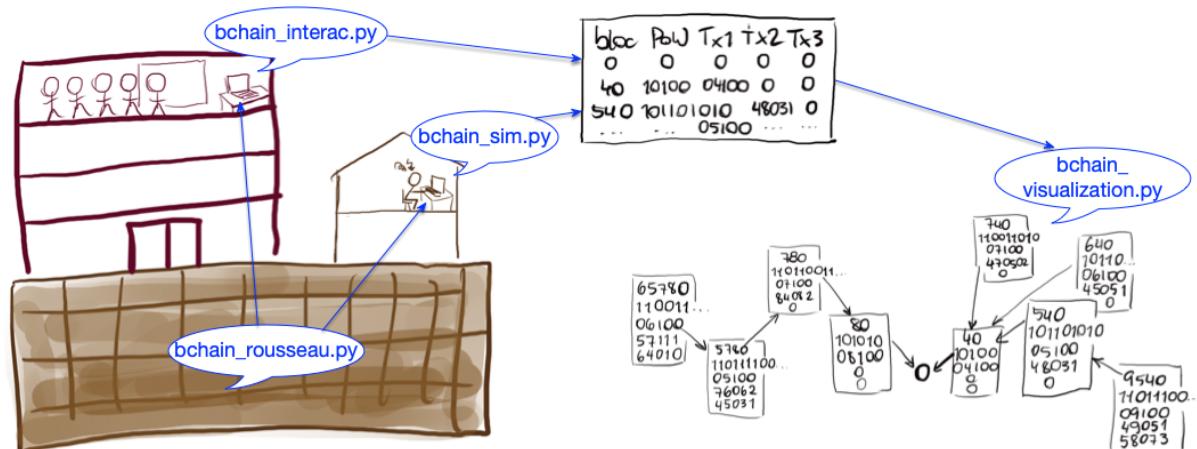


FIGURE 4.1. – Architecture des programmes de la CR.

## 4.2. Description des programmes

### 4.2.1. bchain\_rousseau.py

Ce programme remplit les fonctions suivantes :

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

## 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 certains 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. Les soldes des noeuds sont tenus dans un dictionnaire, qui au départ est vide, et toute la chaîne à laquelle le nouveau bloc appartient est analysée pour calculer les soldes.

1. déterminer les blocs précédents le nouveau bloc jusqu'au bloc-genèse -> findAncestors
2. pour chaque bloc-ancêtre,

- a) déterminer le mineur et la liste de transactions
- b) pour chaque transaction,
  - i. ajouter les traîns 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. ajouter les frais de la transaction au destinataire
- 3. 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 de blocs et la list\_blocks :

1. déterminer les numéros des blocs-parents jusqu'au bloc-genèse
2. 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
3. retourner la liste de blocs-parents.

```

writeToFile(list_rows)
writeCSVRows(list_blocks)
writeCSVRowsWithBalance(list_nodes,list_blocks)
readBlocksFromCSVFile()
```

### Fonctions pour bchain\_interac.py

```

strToRealTx()
updateBchain
1 def readBlocksFromCSVFile
```

Cette fonction permet d'ajouter les nouveaux blocs à une chaîne de blocs existante :

1. déterminer quel fichier .csv est le plus récent, grâce au module Python "os"
2. créer un bloc de la classe Block avec chaque ligne du fichier et l'ajouter à la list\_blocks (qui est vide au départ)
3. retourner list\_blocks

#### 4.2.2. 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 le plus récent.

1. la liste des noeuds list\_nodes codée en dur au tout début du script ;

2. un bloc est créé à partir des informations entrées au clavier par l'utilisateur -> createUserBlock ;
3. les blocs de la chaîne de blocs la plus récente sont insérés dans list\_blocks -> readBlocksFromCSVFile ;
4. les soldes des utilisateurs sont calculés avant l'insertion du nouveau bloc à partir de list\_blocks -> checkBalance ;
5. 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.
6. 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.
7. le nouveau bloc est ajouté à list\_blocks ;
8. 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 2
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.

### 4.2.3. 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 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.

- définir les classes Bloc et Transaction
- tester les mécanismes développés pour les activités-élève
- tester les paramètres du jeu
- produire des exemples de CR

#### BlockSim

Le programme de simulation de chaînes de blocs Blocksim est un excellent outil pour développer une chaîne de blocs en testant les différents paramètres. Dans son article, l'auteur fait excellent résumé du concept de chaînes de blocs, et sa formulation du modèle en couches et du flux des opérations (workflow) est très claire et éclairante. Le but de Blocksim est d'étudier le fonctionnement global de la chaîne de blocs, dont les blocs sont créés de façon aléatoire, il n'y a donc aucune interaction avec l'utilisateur, qui ne fait qu'ajuster les conditions de départ. Cet outil est trop complexe pour les élèves, et n'est pas adapté aux activités déconnectées, mais reste une excellent référence.

### 4.2.4. 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 2.2 (sans les transactions) et 4.2 (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

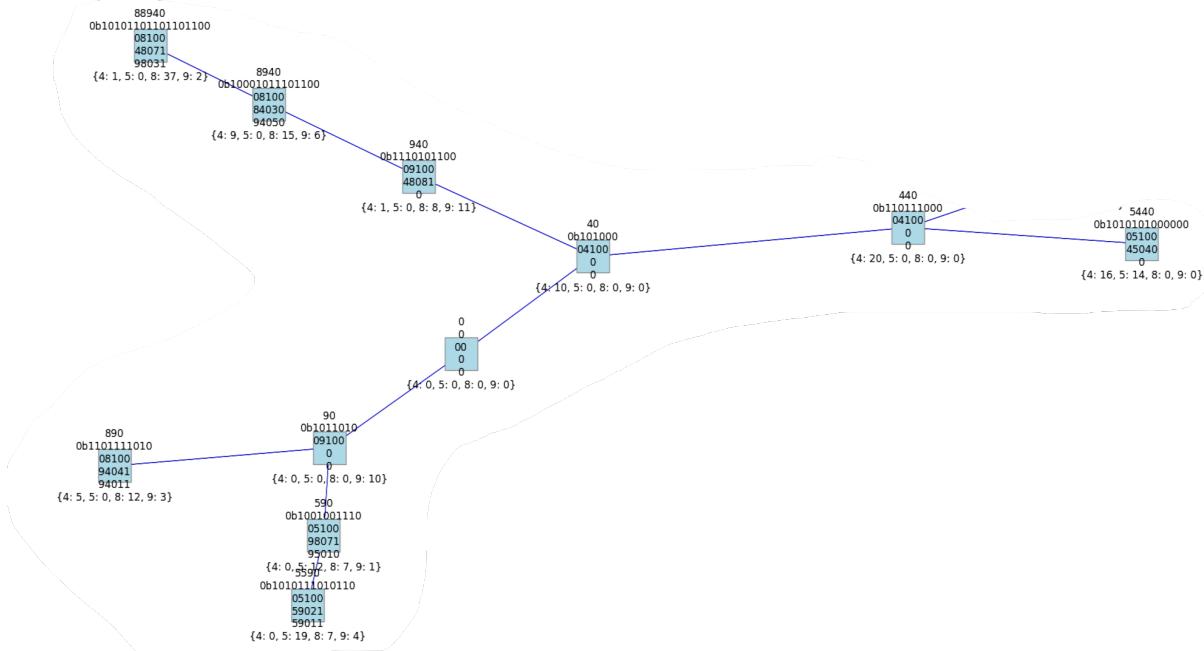


FIGURE 4.2. – Représentation graphique d'une chaîne de blocs BlocNote montrant les transactions.

- 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 :

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

- un graphe est créé.

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

## 4.3. Activités *in silico*

Dans cette section, les différents usages des programmes de BlocNote sont détaillés :

- une marche à suivre pour les débutants
- support pour les activités-élèves (voir annexe B)
- activités-élèves (voir annexe B) revues en modalité connectée

### 4.3.1. Par où commencer ?

La marche à suivre décrite dans cette section devrait (en principe) permettre à n'importe qui de produire une chaîne de blocs BlocNote.

- copier les programmes bchain\_rousseau.py, bchain\_interac.py et bchain\_visualization.py dans un même répertoire ;
- créer un sous-répertoire interac\_bchains, 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. s'assurer que list\_nodes correspond bien aux numéros des noeuds participants :

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

5. 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 2.1 ;

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

### 4.3.2. Support aux activités

La démarche décrite à la section 4.3.1 peut être utilisée par l'enseignant pendant les activités menées en classe afin de s'assurer que les blocs insérés sont valides, et pour vérifier les résultats, comme le calcul des soldes par exemple. Dans ce cas, un seul ordinateur suffit. Il peut aussi servir aux élèves pour s'entraîner à produire une chaîne de blocs et en comprendre les mécanismes. Dans ce cas, chaque élève ou noeud-élèves utilise sur son propre ordinateur, de façon indépendante.

### 4.3.3. 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 :

- les élèves copient manuellement leur fichier .csv dans un répertoire partagé après avoir calculé un nouveau bloc,

- 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 :

- 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,
- ils 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.

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

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

Quelques pistes sont avancées ci-dessous :

- Certains éléments sont codés en dur, et doivent être modifiés directement dans le programme.
- 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.

# 5

## 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 avantageux 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 la CR

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 ; la CR 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 et avec le moins d'impact possible sur l'environnement, d'autres applications que celle des NFTs pourraient être mises en place :

- une solution commerciale locale, citoyenne, durable et solidaire, comme le Léman [33] 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 :

- phase où une chaîne de blocs est construite parmi les quatre élèves formant un noeud, 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 coopétition
- phase où les noeuds sont en coopétition ; l'activité pourrait être notée proportionnellement au nombre de R\$ accumulés, qui est représentatif de la contribution de chaque noeud
- 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 3.2.3.

D'autres variantes permettraient d'illustrer d'autres concepts 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-oncles<sup>1</sup> 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,

---

<sup>1</sup>Les blocs-oncles ("uncle blocks") sont ceux qui sont attachés au même bloc-parent qu'un bloc finalisé.

- 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\$.

## Finalités

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 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

<b>NFT</b>	jeton non-fongible (Non-Fungible Token), voir section 2.5.2
<b>PoW</b>	preuve de travail (Proof of Work), voir section 2.3.1
<b>PoS</b>	preuve d'enjeu (Proof of Stake), voir section 2.3.2
<b>SC</b>	contrat intelligent (Smart Contract), voir section 2.5
<b>Tx</b>	transaction, voir section 2.4

# B

## Activités élèves

---

B.1.	Activité 1 : enchaînement de blocs et preuve de travail . . . . .	91
B.2.	Activité 2 : consensus . . . . .	91
B.3.	Activité 3 : transactions . . . . .	91
B.4.	Activité 4 : contrats intelligents et NFTs . . . . .	91
B.5.	Activité 5 : preuve d'enjeu . . . . .	91

---

**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

## Programmes

---

C.1. bchain_sim.py . . . . .	92
C.2. bchain_interac.py . . . . .	92
C.3. bchain_visualization.py . . . . .	95
C.4. bchain_rousseau.py . . . . .	96

---

### C.1. bchain\_sim.py

```
1 from bchain_rousseau import *
2
3 if __name__ == "__main__":
4
5     list_nodes = [4,5,8,9]
6     nb_blocks = 10
7     list_blocks = [Block(0,0,[])]
8
9     while len(list_blocks) < nb_blocks:
10         list_blocks.append(createNewBlock(list_nodes,list_blocks))
11
12     for block in list_blocks:
13         print(block)
14
15     list_rows = writeCSVRows(list_blocks)
16 # list_rows = writeCSVRowsWithBalance(list_nodes,list_blocks)
17     writeToFile(list_rows)
```

### C.2. bchain\_interac.py

```
1 from bchain_rousseau import *
2
3 def createUserBlock():
4     flag_erreur = False
5
6     print("Quel est le blocID ?")
```

```

7     blockID = input()
8     if blockID[-1] != "0":
9         print("Le blocID n'est pas valide.")
10        flag_erreur = True
11
12    print("Quelle est la PoW ?")
13    PoW = input()
14    if PoW != "0":
15        PoW = "0b" + PoW
16        target = blockID
17        PoW_expected = bin(int(target))
18        if PoW != str(PoW_expected):
19            print("La PoW n'est pas valide.")
20            flag_erreur = True
21
22    print("La transaction de r[U+FFFFD] compense est automatiquement ajout [U+FFFFD]e .")
23    nodeID = int(blockID[0])
24    reward = '0'+str(nodeID)+'10'+'0'
25    reward = strToRealTx(reward)
26    newblock_list_tx = [reward]
27
28    print("Combien de transactions autres que la r[U+FFFFD] compense ?")
29    nb_tx = input()
30    nb_tx = int(nb_tx)
31    if nb_tx == 0:
32        print("Pas de transaction autre que la r[U+FFFFD] compense .")
33    elif nb_tx > 2:
34        print("Il n'est pas possible d'ajouter plus de deux transactions .")
35    elif nb_tx >= 1:
36        print("Quelle est la premi[U+FFF]re transaction ?")
37        tx = input()
38        validateTxEntry(tx,list_nodes)
39        tx = strToRealTx(tx)
40        newblock_list_tx.append(tx)
41        if nb_tx == 2:
42            print("Quelle est la deuxi[U+FFF]me transaction ?")
43            tx = input()
44            validateTxEntry(tx,list_nodes)
45            tx = strToRealTx(tx)
46            newblock_list_tx.append(tx)
47
48    if flag_erreur:
49        print("Le bloc n'est pas valide .")
50        user_block = Block(0,0,0)
51    else:
52        user_block = Block(blockID,PoW,newblock_list_tx)
53
54    print('create userblock',user_block)
55    return user_block
56
57 def validateTxEntry(tx,list_nodes):
58     if len(tx) != 5:
59         print("La transaction n'est pas valide .")
60         print("""
61             Le format d'une transaction est compos[U+FFF] de 5 caract[U+FFF]res:
62             1 = l'adresse du payeur,
63             2 = l'adresse du destinataire,
64             3 et 4 = le montant de la transaction (entre 01 et 99),

```

```
65      5 = les frais de transaction (entre 0 et 9).
66
67      Par exemple, si le noeud 5 ach[U+FFFD]te un v[U+FFFD]lo pour 35 R$ au noeud
68          7
69      et accepte de payer des frais de transaction de 4 R$,
70      la transaction est 57354.
71      """)

72      strtx = str(tx)
73      if (strtx[0] or strtx[1]) not in list_nodes:
74          print("Transaction invalide")

75
76
77 def updateBchain(list_nodes,user_block,list_blocks):
78     user_block_parentID = str(user_block.blockID)[1:]

79     miner = user_block.blockID

80
81
82
83
84 if __name__ == "__main__":
85
86     list_nodes = [4,5,8,9]

87
88     user_block = createUserBlock()
89     print(user_block)
90     user_blockID = str(user_block.blockID)
91     user_block_parentID = str(user_block.blockID)[1:]

92
93     list_blocks = readBlocksFromCSVFile()
94     dict_balance = checkBalance(list_nodes, user_block_parentID, list_blocks)
95     print("Solde avant l'ajout du nouveau bloc:",dict_balance)

96
97     list_blockIDs = findBlockIDs(list_blocks)

98
99     if int(user_block_parentID) not in list_blockIDs:
100         print("Le nouveau bloc n'a pas de bloc-parent.")
101     else:
102         list_blocks.append(user_block)
103         new_dict_balance = checkBalance(list_nodes, user_blockID, list_blocks)
104         print(new_dict_balance)

105
106     flag_newblockisok = True
107     for balance in new_dict_balance:
108         if new_dict_balance[balance] < 0:
109             print("Block not valid !")
110             flag_newblockisok = False

111
112     if (flag_newblockisok):
113         list_rows = writeCSVRows(list_blocks)
114         # list_rows = writeCSVRowsWithBalance(list_nodes,list_blocks)
115         writeToFile(list_rows)
116     else:
117         print("No new cvs file.")
```

### C.3. bchain\_visualization.py

```

1 import csv
2 import os
3 import matplotlib.pyplot as plt
4 import networkx as nx
5 G = nx.Graph()
6
7 def newest(path):
8     files = os.listdir(path)
9     paths = [os.path.join(path, basename) for basename in files]
10    return max(paths, key=os.path.getctime)
11
12 list_blocs = []
13 list_nodes = []
14 newest_csv_file = newest('bchains_sim/')
15 newest_csv_file = newest('interac_bchains/')
16 with open(newest_csv_file, newline='') as open_file:
17 #with open('bchains/bchain_sim20221026_193342.csv', newline='') as open_file:
18     csv_reader = csv.reader(open_file, delimiter=',')
19     for row in csv_reader:
20         # row = [bloc.blocID,bloc.PoW,bloc.Tx]
21         bloc = (row[0],row[1],row[2],row[3],row[4])
22         list_blocs.append(bloc)
23         list_nodes.append(row[0])
24 print(list_blocs)
25
26 G.add_nodes_from(list_nodes)
27 print(list(G.nodes))
28
29 list_edges = []
30 for bloc in list_blocs:
31     if bloc[0] != '0':
32         parent=bloc[0][1:]
33         list_edges.append((bloc[0],parent))
34
35 G.add_edges_from(list_edges)
36 print(list(G.edges))
37
38 dict_labels = {}
39 nb_blocs = len(list_blocs)
40 #keys = range(nb_blocs)
41 for node in G.nodes:
42     for bloc in list_blocs:
43         if node == bloc[0]:
44             PoW = bloc[1]
45             tx = '0'+bloc[2]+"\n"+bloc[3]+"\n"+bloc[4]
46             dict_labels[node] = node+'\n'+PoW + '\n'+tx
47             #dict_labels[node] = node + '\n' + PoW
48 # dict_labels[node] = node+'\n'+PoW+'\n'+tx+'\n'+bloc[5]
49 # dict_labels[node] = node + '\n' + PoW + '\n' + bloc[5]
50 print("dict",dict_labels)
51
52 pos = nx.spring_layout(G)
53 # nx.draw(G, with_labels=True, font_weight='bold')
54 nx.draw(G,pos=pos,with_labels=False)
55 #nx.draw_networkx(G,pos=pos,labels=dict_labels)

```

```

56
57 nodenames = {}
58 for n in G.nodes():
59     nodenames[n] = 'firstline \n secondline \n thirdline'
60 #nodenames = {n:'firstline \n secondline \n thirdline' for n in G.nodes()}
61 #nx.draw_networkx_labels(G, pos=pos, labels=nodenames)
62
63 nx.draw_networkx_labels(G, pos=pos, labels=dict_labels)
64
65 nx.draw_networkx(G, pos = pos, labels = dict_labels,
66                   node_shape = "s", node_size = 1500,
67                   node_color = "lightblue",
68                   edge_color = "blue", #color of the edges
69                   edgecolors = "gray") #edges of the box of node
70
71
72 plt.show()

```

## C.4. bchain\_rousseau.py

```

1 import random
2 from random import randint
3 import os
4 import csv
5 import math
6 from datetime import datetime
7
8 infinity = 1000
9
10 class Block:
11     def __init__(self, blockID, PoW, list_tx):
12         self.blockID, self.PoW, self.list_tx = blockID, PoW, list_tx
13
14     def __str__(self):
15         plain_list_tx = []
16         for tx in self.list_tx:
17             strtx = getTxStr(tx)
18             plain_list_tx.append(strtx)
19         return f"block {self.blockID}, PoW {self.PoW}, Tx:{plain_list_tx}"
20
21
22 class Tx():
23     def __init__(self,payeur,destinataire,amount,fee):
24         self.payeur, self.destinataire, self.amount, self.fee = payeur,destinataire,
25         amount,fee
26
27     def __str__(self):
28         return f"Tx: from {self.payeur}, to {self.destinataire}, amount {self.amount},
29         fee {self.fee}"
30
31     def tx_str(self):
32         strtx = ""
33         strtx += str(self.payeur)
34         strtx += str(self.destinataire)
35         if self.amount < 10:

```

```
34         strtx += "0" + str(self.amount)
35     else:
36         strtx += str(self.amount)
37     strtx += str(self.fee)
38
39     return strtx
40
41 def strToRealTx(tx):
42     payeur = int(tx[0])
43     destinataire = int(tx[1])
44     montant = int(tx[2:4])
45     frais = int(tx[4])
46     realTx = Tx(payeur,destinataire,montant,frais)
47
48     return realTx
49
50
51 def getTxStr(tx):
52     tx_str = ""
53     tx_str += str(tx.payeur)
54     tx_str += str(tx.destinataire)
55     if tx.amount < 10:
56         tx_str += "0" + str(tx.amount)
57     else:
58         tx_str += str(tx.amount)
59     tx_str += str(tx.fee)
60
61     return tx_str
62
63
64
65 class Reward(Tx):
66     def __init__(self,destinataire):
67         self.payeur, self.destinataire, self.amount, self.fee = 0,destinataire,10,0
68
69 def findBlockIDs(list_blocks):
70     list_blockIDs = []
71
72     for block in list_blocks:
73         blockID = block.blockID
74         list_blockIDs.append(blockID)
75
76     return(list_blockIDs)
77
78 def findAncestorIDs(newblockID):
79     list_ancestorsIDs = []
80
81     for n in range(len(str(newblockID))):
82         list_ancestorsIDs.append(int(newblockID[n:]))
83
84     print('in fct findAncestorIDs, ancestors of',newblockID,'are',list_ancestorsIDs)
85     return(list_ancestorsIDs)
86
87 def findAncestors(list_ancestorIDs,list_blocks):
88     list_ancestors = []
89
90     for block in list_blocks:
91         for id in list_ancestorIDs:
```

```
92         if int(block.blockID) == id:
93             list_ancestors.append(block)
94             print('in fct findAncestors',block)
95
96     return(list_ancestors)
97
98 def createMempool(list_nodes,nb_tx,max_amount):
99     mempool = []
100
101    for i in range(nb_tx):
102        payeur = random.choice(list_nodes)
103        max_fee = max_amount // 5
104        fee = randint(0,max_fee)
105        max_amount -= fee
106        amount = randint(1,max_amount)
107
108        # choisir un destinataire au hasard parmi tous les noeuds, sauf le payeur
109        list_destinataires = list_nodes.copy()
110        list_destinataires.remove(payer)
111        destinataire = random.choice(list_destinataires)
112
113        tx = Tx(payer, destinataire, amount, fee)
114        mempool.append(tx)
115
116 # for tx in mempool:
117 # print(getTxStr(tx))
118     return mempool
119
120
121 def checkBalance(list_nodes,new_blockID,list_blocks):
122     print('$$ checkBalance $$')
123     dict_balance = {}
124     for node in list_nodes:
125         dict_balance[node] = 0
126
127     list_ancestorIDs = findAncestorIDs(new_blockID)
128     list_ancestors = findAncestors(list_ancestorIDs,list_blocks)
129
130     for block in list_ancestors:
131         miner = int(str(block.blockID)[0])
132         list_tx = block.list_tx
133         for tx in list_tx:
134             dict_balance[miner] += tx.fee
135             if tx.payeur != 0:
136                 dict_balance[tx.payeur] -= (tx.amount + tx.fee)
137                 dict_balance[tx.destinataire] += tx.amount
138             print("$$",block.blockID,dict_balance)
139
140 # print('in fct checkBalance, end balance:',dict_balance)
141     return dict_balance
142
143 def newest(path):
144     files = os.listdir(path)
145     paths = [os.path.join(path, basename) for basename in files]
146     return max(paths, key=os.path.getctime)
147
148 def readBlocksFromCSVFile():
149     list_blocks = []
```

```

150     newest_csv_file = newest('interac_bchains/')
151 # newest_csv_file = 'interac_bchains/bchain_intterac_genese.csv'
152 # newest_csv_file = 'interac_bchains/bchain_sim20221218_051429.csv'
153     print("Reading bchain from",newest_csv_file)
154     with open(newest_csv_file, newline='') as open_file:
155         csv_reader = csv.reader(open_file, delimiter=',')
156         for row in csv_reader:
157             blockID = int(row[0])
158             blockPoW = row[1]
159             block_listTx = []
160             for i in range(2,5):
161                 if len(row[i]) == 4:
162                     row[i] = '0'+row[i]
163                 if len(row[i]) == 5:
164                     tx = strToRealTx(row[i])
165                     block_listTx.append(tx)
166             block = Block(blockID,blockPoW,block_listTx)
167
168             list_blocks.append(block)
169
170     return list_blocks
171
172 def writeCSVRows(list_blocks):
173     list_rows = []
174
175     for block in list_blocks:
176         list_tx = []
177         for tx in block.list_tx:
178             list_tx.append(int(getTxStr(tx)))
179             for i in range(3):
180                 if len(list_tx) == 0:
181                     tx1 = 0
182                 else:
183                     tx1 = list_tx[0]
184                 if len(list_tx) > 1:
185                     tx2 = list_tx[1]
186                 else:
187                     tx2 = 0
188                 if len(list_tx) == 3:
189                     tx3 = list_tx[2]
190                 else:
191                     tx3 = 0
192             row = [block.blockID, block.PoW, tx1, tx2, tx3]
193             list_rows.append(row)
194
195     return list_rows
196
197 def writeCSVRowsWithBalance(list_nodes,list_blocks):
198     list_rows = []
199
200     for block in list_blocks:
201         list_tx = []
202         for tx in block.list_tx:
203             tx = getTxStr(tx)
204             if len(tx) == 4:
205                 tx = '0' + tx
206             list_tx.append(int(tx))
207             for i in range(3):

```

```
208         if len(list_tx) == 0:
209             tx1 = 0
210         else:
211             tx1 = list_tx[0]
212             if len(list_tx) > 1:
213                 tx2 = list_tx[1]
214             else:
215                 tx2 = 0
216             if len(list_tx) == 3:
217                 tx3 = list_tx[2]
218             else:
219                 tx3 = 0
220             dict_balance = checkBalance(list_nodes,str(block.blockID),list_blocks)
221             row = [block.blockID, block.PoW, tx1, tx2, tx3, dict_balance]
222             list_rows.append(row)
223
224     return list_rows
225
226
227 def writeToFile(list_rows):
228     ##### ----- ecriture de la bchain dans un fichier csv
229     tstamp = str(datetime.utcnow())
230     print('tstamp',tstamp)
231     tstamp = tstamp[:-7]
232     tstamp = tstamp.replace("-", "")
233     tstamp = tstamp.replace(" ", "_")
234     tstamp = tstamp.replace(":", "")
235     tstamp = tstamp.replace(".", "_")
236     print('tstamp',tstamp)
237     filename = 'bchain_sim/bchain_sim' + str(tstamp) + '.csv'
238 # filename = 'interac_bchains/bchain_interac' + str(tstamp) + '.csv'
239
240     # open the file in the write mode
241     with open(filename, 'w') as f:
242         writer = csv.writer(f)
243         for row in list_rows:
244             writer.writerow(row)
245
246 def createListBChainBlockIDs(list_blocks):
247     list_bchain_blockIDs = []
248     for block in list_blocks:
249         list_bchain_blockIDs.append(block.blockID)
250
251     return list_bchain_blockIDs
252
253 def getFee(tx):
254     return tx.fee
255
256 def validateTx(mempool,dict_balance):
257     list_validTx = []
258     print('validateTx',dict_balance)
259     for tx in mempool:
260         payeur_balance = dict_balance[tx.payeur]
261         if payeur_balance > tx.amount + tx.fee:
262             list_validTx.append(tx)
263             dict_balance[tx.payeur] -= tx.amount + tx.fee
264
265 # for tx in list_validTx:
```

```
266 # print(tx,tx.fee)
267
268     list_validTx.sort(key=getFee, reverse=True)
269     return list_validTx
270
271 def createNewBlockID(list_nodes,list_existing_blockIDs):
272     minerID = random.choice(list_nodes)
273     parentblockID = random.choice(list_existing_blockIDs)
274
275     new_blockID = str(minerID) + str(parentblockID)
276
277     return new_blockID
278
279 def createNewBlock(list_nodes,list_blocks):
280     list_existing_blockIDs = createListBChainBlockIDs(list_blocks)
281     new_blockID = createNewBlockID(list_nodes, list_existing_blockIDs)
282
283     while new_blockID in list_existing_blockIDs:
284         new_blockID = createNewBlockID(list_nodes, list_existing_blockIDs)
285
286     minerID = int(new_blockID[0])
287     reward = Reward(minerID)
288     new_block_list_tx = [reward]
289     mempool = createMempool(list_nodes,10,10)
290
291     dict_balance = checkBalance(list_nodes, new_blockID, list_blocks)
292     list_validTx = validateTx(mempool,dict_balance)
293
294     if (len(list_validTx)) > 0:
295         new_block_list_tx.append(list_validTx[0])
296         if (len(list_validTx)) > 1:
297             new_block_list_tx.append(list_validTx[1])
298
299     PoW = bin(int(new_blockID))
300     new_block = Block(new_blockID,PoW,new_block_list_tx)
301     return(new_block)
```

# D

## 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. 8, 35
- [2] ethereum.org. 15, 17, 24, 36
- [3] Geeks for geeks.
- [4] Gnu operating system.
- [5] <https://blockexplorer.one/>. 40
- [6] <https://coinmarketcap.com/rankings/exchanges/>. 23
- [7] <https://foldingathome.org/?lng=en>. 5
- [8] Nym. 5
- [9] wikipedia.com. 19, 31, 32, 63
- [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, 15, 77, 83
- [12] Baeldung. Baeldung helps developers explore the java ecosystem and simply be better engineers. 32
- [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. 58
- [14] Blockchain Partner by KPMG Blockchain France. Lexique. 18, 48, 50
- [15] Dead Coins 1700+ Cryptocurrencies Forgotten by This World (2023 Updated). 99bitcoins. 7
- [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. 58
- [17] Amy Castor. Why ethereum is switching to proof of stake and how it will work. Technology review, MIT, 4 mars 2022. 36
- [18] J Scott Christianson. How to teach blockchain with “the blockchain game !”, February 2022. 69
- [19] Inc. 30 Knightsbridge Road Suite 620 Piscataway NJ 08854 CITE Sunbird, Sunbird Software. Largest bitcoin mining farms. 8

- [20] Mitchell Clark. Blockchain, explained blocks ? chains ? how does this whole thing work ? 20
- [21] Ethereum community. Proof-of-work (pow). 35, 36
- [22] NFT School contributors like you. Nft school, octobre 2021. 50
- [23] TheWeek Correspondent. 5 of the largest bitcoin mining farms in the world. 8
- [24] crypto.com. A deep dive into blockchain scalability. 7, 8
- [25] Raynor de Best. Countries that mine the most bitcoin (btc) 2019-2022, Aug 8, 2022. 8
- [26] 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, 67
- [27] Cem Dilmegani. Top 9 smart contract use cases & examples in 2023. 47
- [28] 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. 67
- [29] D. Adinda et al. La Transposition didactique. Master cft, Faculté des sciences de l'éducation, Université de Strasbourg, 2015. 14
- [30] Yves Dethurens et Daniel Kessler. Expérimenter un cours transdisciplinaire. GymInf, Didactique de l'informatique, Rendu 100, 11 mars 2022. 71, 72
- [31] Figaro. Bourse et placements. 9
- [32] Primavera De Filippi. Blockchain et cryptomonnaies. Number 5e tirage. Que sais-je ? Humensis, 2021. 6, 7, 8, 9, 10, 11, 13, 15, 17, 19, 27, 47, 48, 58
- [33] solutions developper à Monnaie Léman Florian Dubath. <https://monnaie-leman.org>. 5, 10, 31, 88
- [34] Hyperledger Foundation. Hyperledger – open source blockchain technologies. 32
- [35] Yeung Fuk. Useful Computation on the Block Chain. PhD thesis, Harvard University, 2019. 58
- [36] Linus Gasser. Cours de sécurité et confidentialité du programme gyminf. In fichier pdf à usage interne, Juin 2022. 5, 71
- [37] 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. 66, 71
- [38] <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. 7
- [39] Marco Iansiti and Karim R. Lakhani. Merriam-Webster dictionary. Merriam Webster, <https://www.merriam-webster.com/dictionary/blockchain>, 2022. 17
- [40] Copyright © 2023 Insider Inc. Cryptocurrencies overview. 4
- [41] Aaron Broverman Kat Tretina. 10 best cryptocurrencies in january 2023 10 best cryptocurrencies in january 2023, January 2023. 13

- [42] Larousse. Dictionnaire Larousse. Harrap's, https://www.larousse.fr/dictionnaires/francais/blockchain/188331, 2022. 17, 48
- [43] 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.
- [44] Wei Li. Adapting blockchain technology for scientific computing. 58
- [45] Bitcoin.com © 2022 Saint Bits LLC. How do bitcoin transactions work ? 44
- [46] Andreas Lymouras. A shallow dive into bitcoin's blockchain part 2 - transactions. 40
- [47] modulo (ressources pour l'enseignement de l'informatique au gymnase). Vie privée et surveillance. 10
- [48] Kirsty Moreland. How to read a blockchain transaction history.
- [49] Satoshi Nakamoto. Bitcoin : A peer-to-peer electronic cash system. 6, 15, 17, 18, 31
- [50] 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. 11
- [51] J.-H. Morin M. Reymond Centre de droit bancaire et financier O. Depierre, C. Lapinte. Lexique de la blockchain, février 2022. 17, 48
- [52] MIT Sloan School of Management. The beer game, très vieux. 67
- [53] The National Institute of Standards and Technology (NIST). Blockchain. 18
- [54] Thierry Gerez Olivier La Spada. Expérimenter un cours transdisciplinaire, portfolio pièce 110. Didactique de l'informatique, Gyminf, mars 2022. 71, 72
- [55] Joël Orizet. Cryptomonnaies : la demande de licence bancaire de bitcoin suisse échoue. ICTjournal, mars 2021. 3
- [56] 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. 17
- [57] Patrice Potvin. Faire apprendre les sciences et la technologie à l'école. Presses de l'université Laval, 2018. 12, 66, 73
- [58] Farran Powell and Benjamin Curry. 10 best crypto apps and exchanges of 2023, January 2023. 22
- [59] TPG (Transport publics genevois). Tp publicité sa. 3
- [60] MORITZ PUTZHAMMER. Which cryptocurrencies will survive a market crash ?, 11 August 2022. 7, 13
- [61] 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. 50
- [62] rameerez. No, your nft is not on the blockchain.
- [63] Ezra Reguerra. Cryptopedia : Learn the basics of smart contracts and how they work. 47

- [64] Gabriel Rodriguez. 4 best crypto exchanges of january 2023. 23
- [65] Eric Rosenberg. How much energy it takes to power bitcoin, September 15th 2022. 8
- [66] 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. 50
- [67] Isabelle Vuillemin David De Vito Gabrielle Stiassny. Esii education numérique informatique. Site pédagogique officiel de l'enseignement genevois, 2021. 10
- [68] 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. 10, 64
- [69] PA 18950 THE COMPUTER LANGUAGE COMPANY INC. 5521 State Park Road, P.O. Box 265 Point Pleasant. Definition : Bitcoin transaction. 40
- [70] 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. 58
- [71] Clément Wardzala. Mempool. 41
- [72] 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. 50