

Présentation de l'état de l'art de la blockchain

...

Aller au-delà du Bitcoin

Généralités



C'est quoi la blockchain ?

Des données sont enregistrées
dans des blocs chaînés, distribués et
sécurisés par de la cryptographie
après validation par consensus

C'est quoi la blockchain ?

- Les blockchains sont un type spécifique de registre distribué (distributed ledger) et un moyen de vérifier et d'ordonner en blocs un nombre de transactions toujours croissant, avec diverses protections contre la falsification et la modification
 - Un grand livre distribué signifie qu'aucune autorité centralisée ne vérifie et n'exécute les transactions.
- Un réseau d'ordinateurs, sans organe de contrôle, conserve et valide un enregistrement du consensus de ces transactions avec un journal d'audit cryptographique
 - Les participants de la blockchain mettent à disposition les ordinateurs qui servent constituent les nœuds du réseau

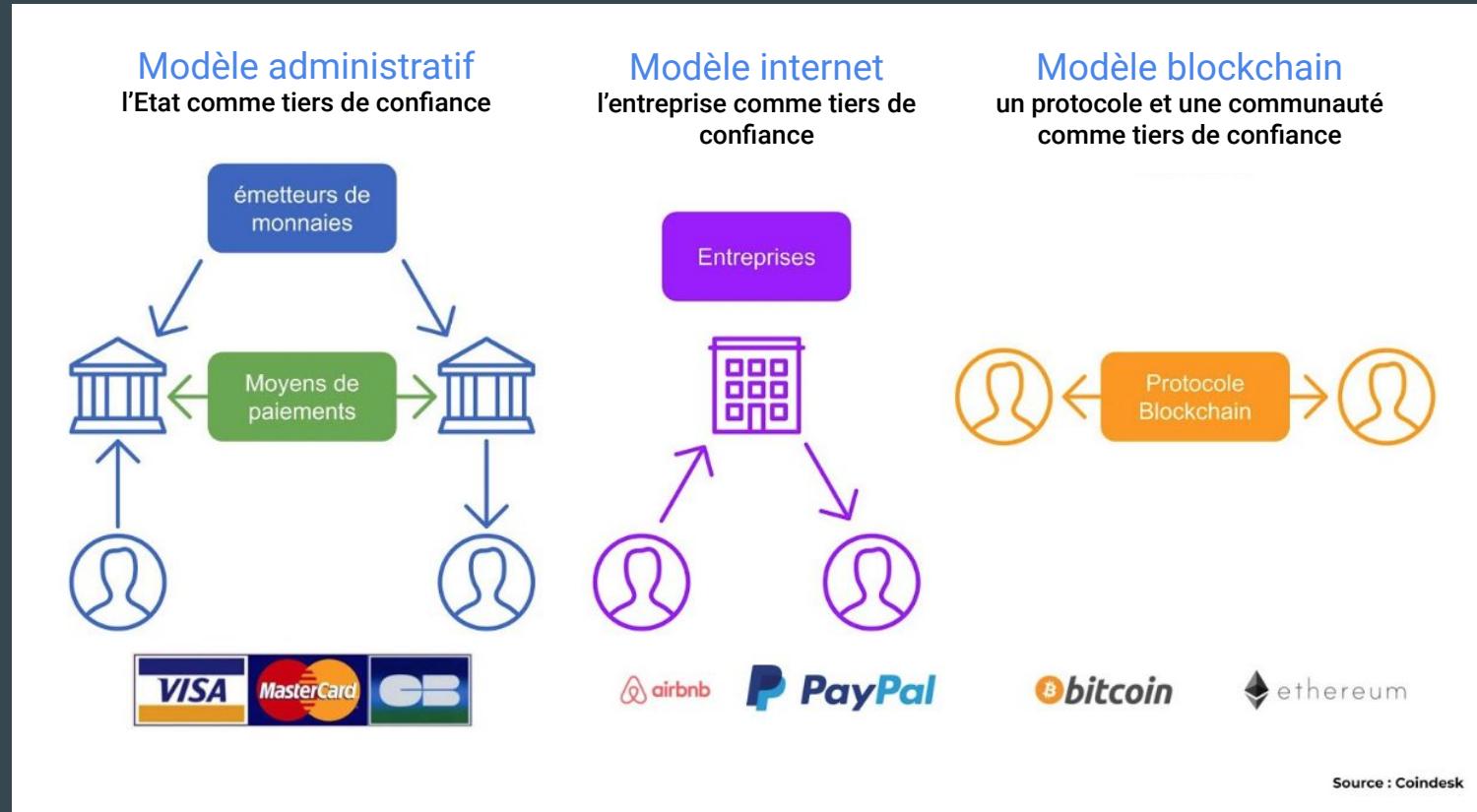
C'est quoi la blockchain ?

- **Cet ensemble de concepts permet de construire des applications décentralisées**
 - Elles sont à la fois distribuées sur le réseau dans la mesure où les registres qui stockent l'information sont hébergés sur plusieurs ordinateurs, et décentralisées car en cas de défaillance d'un nœud, le réseau peut toujours fonctionner

Caractéristiques principales

- La blockchain possède trois grandes caractéristiques :
 1. La transparence, puisque tout participant peut consulter l'ensemble des échanges inscrits dans le registre depuis sa création
 2. La sécurité, grâce à l'utilisation des techniques de cryptographie dont le hachage, qui rendent la falsification des données quasi impossible
 3. L'absence d'organe de contrôle bâtie sur une architecture peer-to-peer

Simplification des modèles



Réglementation

“Je considère la technologie de la blockchain comme un changement fondamental et je veux que l’Europe soit à la pointe de son développement”

(Commissaire bulgare Mariya Gabriel)

La technologie du blockchain est un changement fondamental selon la Commission

Modifié le 02/02/2018 à 11:45 | Publié le 02/02/2018 à 11:45

Écouter



Lire le journal numérique

Ouest France avec AFP

Bruxelles ne veut pas « rater » le virage numérique que représente le système du « blockchain », utilisé notamment pour la monnaie virtuelle, le bitcoin. La commissaire européenne Mariya Gabriel a annoncé jeudi le lancement d'un observatoire-forum sur cette question.

L'UE « ne peut pas se permettre de rater » la révolution numérique liée à la technologie du « blockchain », qui renforce la traçabilité et la sécurité des données et des transactions sur internet, a affirmé jeudi la commissaire européenne à l'Économie numérique.

Transaction

- Une partie ou l'ensemble des nœuds du réseau vérifie et le cas échéant exécute les transactions proposées selon un algorithme convenu entre les participants appelé mécanisme de consensus
 - Il n'y a pas donc pas besoin d'un intermédiaire entre les parties pour garantir la validité d'une transaction
 - Une fois la transaction validée et ajoutée à un bloc, celui-ci est propagé au réseau. Lorsque que le consensus sur sa validité est atteint par tous les nœuds du réseau, le bloc est alors chaîné

Transaction

- Ces transactions sont encodées et stockées sous forme de blocs chaînés sur les nœuds du réseau, constituant ainsi un journal d'audit
 - Toutes les technologies de blockchain utilisent les techniques de cryptographie asymétrique; une transaction est ainsi signée avec la clés privée du participant qui en est à l'initiative
 - Il n'est pas nécessaire de se faire confiance entre participants au réseau, car la technologie, qui s'exécute en toute transparence sur les nœuds des participants, fournit toute la confiance nécessaire
- Il en résulte des avantages de rapidité, de confidentialité et de baisse de coût

Exemple de transaction entre Alice et Bob

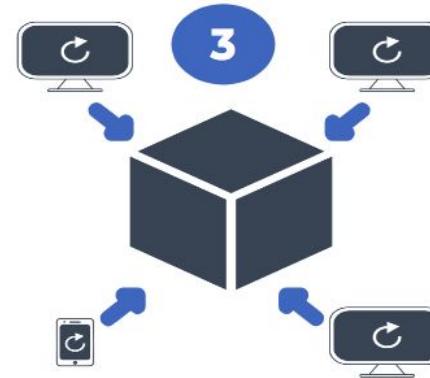
1

Bob négocie une transaction avec Alice



La transaction intègre
un bloc au sein du
réseau

2

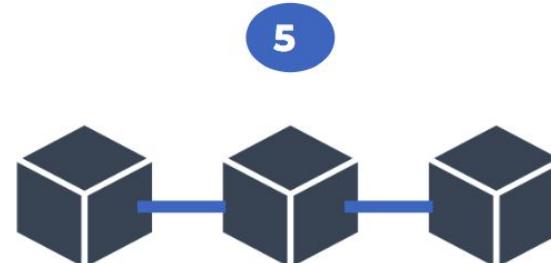


Le bloc est transmis à
chaque noeud du
réseau

Exemple de transaction entre Alice et Bob



Les membres décident que la transaction est validée en utilisant des techniques cryptographiques



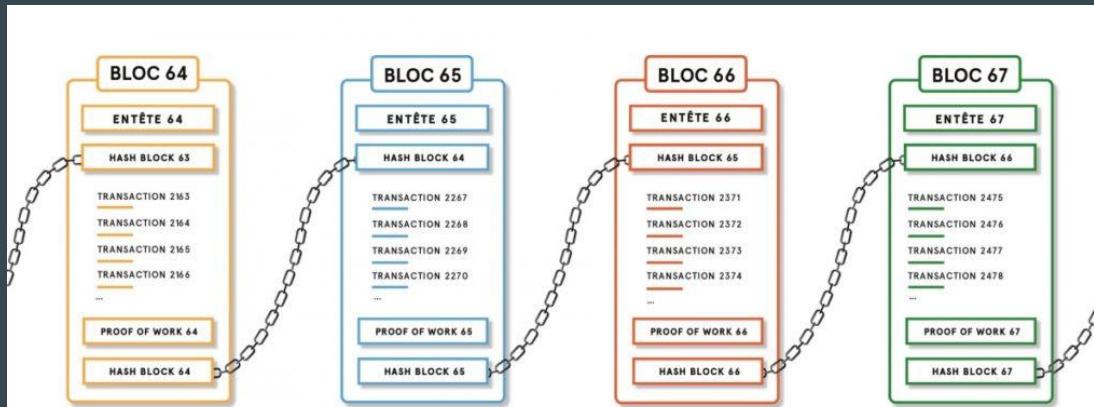
Le bloc s'ajoute à la chaîne de blocs de manière irrémédiable et irréfutable

6

La transaction est finalisée!

Registre distribué

- Une “simple” structure de données
 - Chaque bloc est lié au précédent pour former une chaîne
 - Un bloc = un ensemble de transactions ou de smart contracts
- Pour garantir la validité de la chaîne, un nœud est chaîné au nœud précédent par son pointeur de hachage
- Un pointeur de hachage est calculé sur l'adresse du bloc précédent et le hachage des données à l'intérieur du bloc



Registre distribué

- L'utilisation de la cryptographie assure l'authentification d'identité pour chaque transaction
- L'application des principes de non-répudiation / immuabilité préserve l'intégrité des données et crée un journal d'audit
- Les smart contracts sont chargés de l'exécution automatique de la logique métier lorsque certains critères sont remplis
- Le registre est partagé pour que chaque participant ait la même vue des mêmes données, mise à jour en temps réel, sous réserve d'autorisation
- Le consensus est distribué pour garantir que l'état du registre représente la vérité convenue de toutes les parties prenantes

Principes du consensus

- Problème du général byzantin
- Conditions / postulats
 - Chaque division de l'armée byzantine est dirigée par un général
 - Un général peut prendre la décision d'attaquer ou de se replier
 - Parmi les généraux, certains sont des traîtres
 - Un petit nombre de traîtres ne peuvent pas amener les généraux fidèles à adopter un mauvais plan
 - Parmi les généraux fidèles
 - Tous les généraux fidèles décident du même plan d'action
 - Tous les généraux fidèles reçoivent la même information à partir de laquelle ils vont prendre la même décision quoiqu'il arrive
 - Les informations envoyées par un général fidèle devraient être utilisées par tous les autres généraux fidèles
 - Les généraux communiquent entre eux par des messagers
 - Les messages peuvent être perdus ou corrompus

Principes du consensus

- **Action**
 - Les divisions ont entouré un château et se préparent à l'assaut
 - Pour gagner, elles doivent attaquer simultanément
- **Résultats**
 - Il n'y a pas de consensus si moins de 2/3 des généraux sont fidèles

Contraintes de fonctionnement

- Pour que la blockchain puisse être industrialisable, elle doit répondre à plusieurs critères
 - Faible latence : traitement des transactions dans un temps acceptable
 - Haut débit : nombre de transactions réalisées simultanément sans congestionner la blockchain
 - Taille : la place mémoire de la blockchain reste sous contrôle avec l'augmentation de son activité
 - Consommation : le fonctionnement de la blockchain n'entraîne pas une consommation de ressources excessive
 - Résilience : la blockchain doit posséder une tolérance aux pannes et résister aux nœuds malveillants
 - Irrévocabilité : une transaction validée par le système ne peut être remise en cause

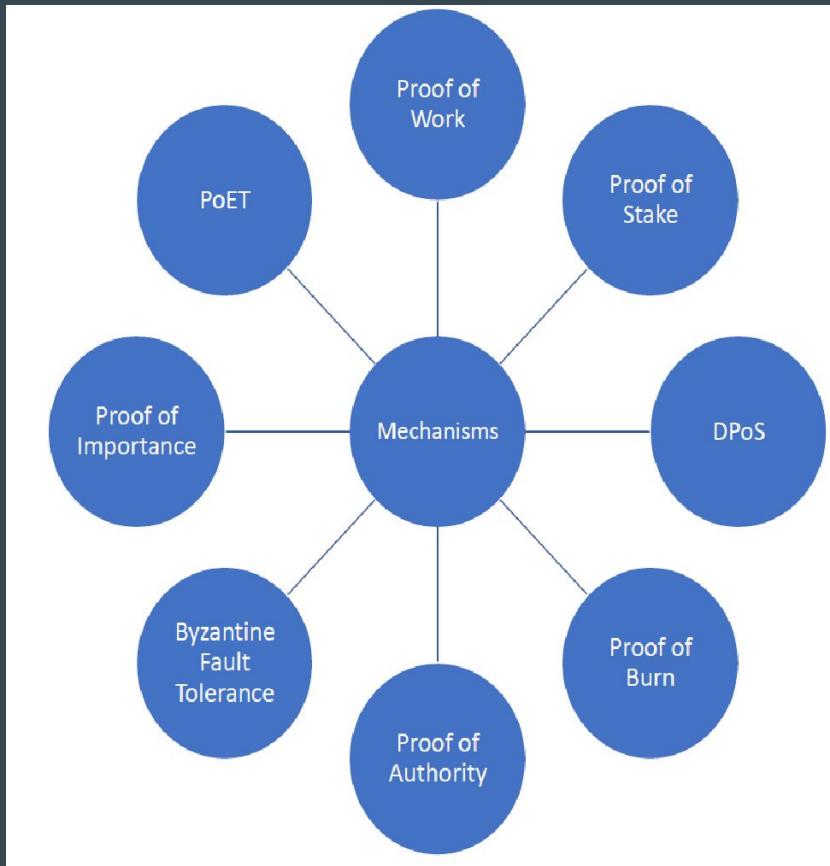
Contraintes de fonctionnement

- Le mécanisme de consensus tient un rôle crucial dans l'industrialisation de la blockchain
 - Un mauvais choix de mécanisme de consensus dans une blockchain peut entraîner sa fin rapidement
 - Ainsi la consommation énergétique démesurée requise par le mécanisme de consensus Proof of Work (PoW) oblige Ethereum à le remplacer
 - C'est un véritable challenge pour cette communauté que de changer de mécanisme de consensus tout en assurant sa continuité de service

Principes de bases des mécanismes de consensus

- **Gouvernance décentralisée**
 - Une autorité centrale seule ne peut assurer la finalité de la transaction
- **Structure du quorum**
 - Les nœuds échangent des messages de manières prédéfinies
- **Authentification**
 - Vérification de l'identité des participants
- **Intégrité**
 - Validation de l'intégrité de la transaction, le plus souvent en utilisant la cryptographie
- **Non-répudiation**
 - Vérification que l'expéditeur supposé a vraiment envoyé le message
- **Confidentialité**
 - Garantie que seul le destinataire prévu peut lire le message
- **Tolérance de panne**
 - Le réseau fonctionne efficacement et rapidement, même si certains nœuds sont lents ou hors de service
- **Performances**
 - Prise en compte du débit, de la latence, de la réactivité et de l'évolutivité du mécanisme

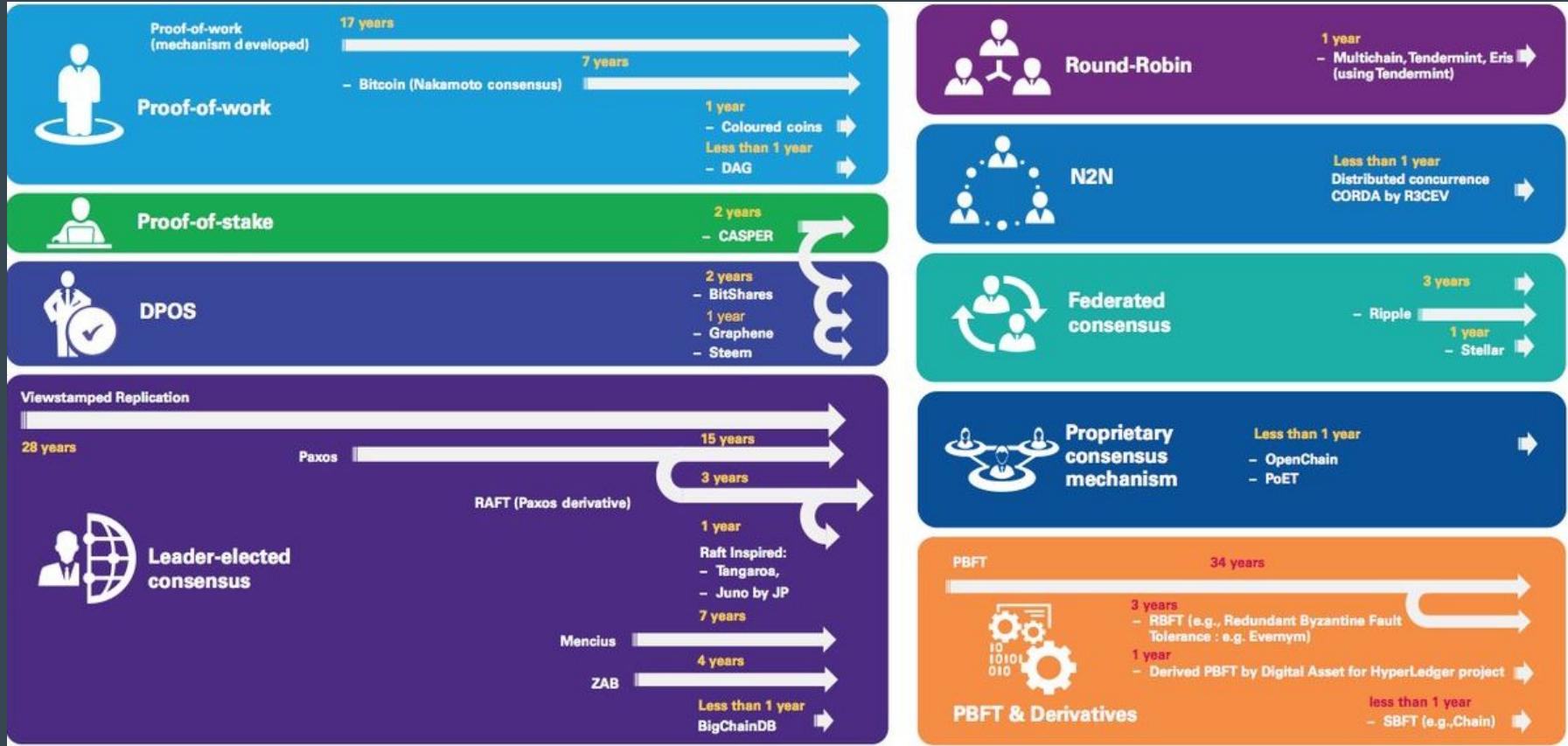
Vue d'ensemble des mécanismes de consensus



Différentes situations, différents consensus

Type de consensus utilisé	Type de blockchain	Description	Exemple de solutions
Proof of Work Proof of Stake Proof of Elapsed Time	Sans permission ou public	Tout le monde peut participer, on ne sait donc rien des participants et on n'a aucun contrôle sur leurs actions	Bitcoin Ethereum Sawtooth
Byzantine Fault Tolerance (BFT)	Avec permission	Les participants sont connus, mais on ne peut pas influencer leurs actions	R3 Corda EWF (Energy Web Foundation)
Federated Byzantine Tolerance		On a une bonne compréhension de ce qu'est un participant fautif	Ripple ou Stellar

Historique des consensus



Consensus d'Ethereum

Proof of Work ou preuve de travail ou minage :

Il s'agit de la résolution d'une **énigme mathématique** par les membres du réseau qui mettent à contribution la **puissance de calcul** de leur ordinateur. Les premiers à résoudre l'énigme sont **récompensés avec une cryptomonnaie**. L'opération est fortement **consommatrice en énergie**.

Proof of Stake ou preuve de participation :

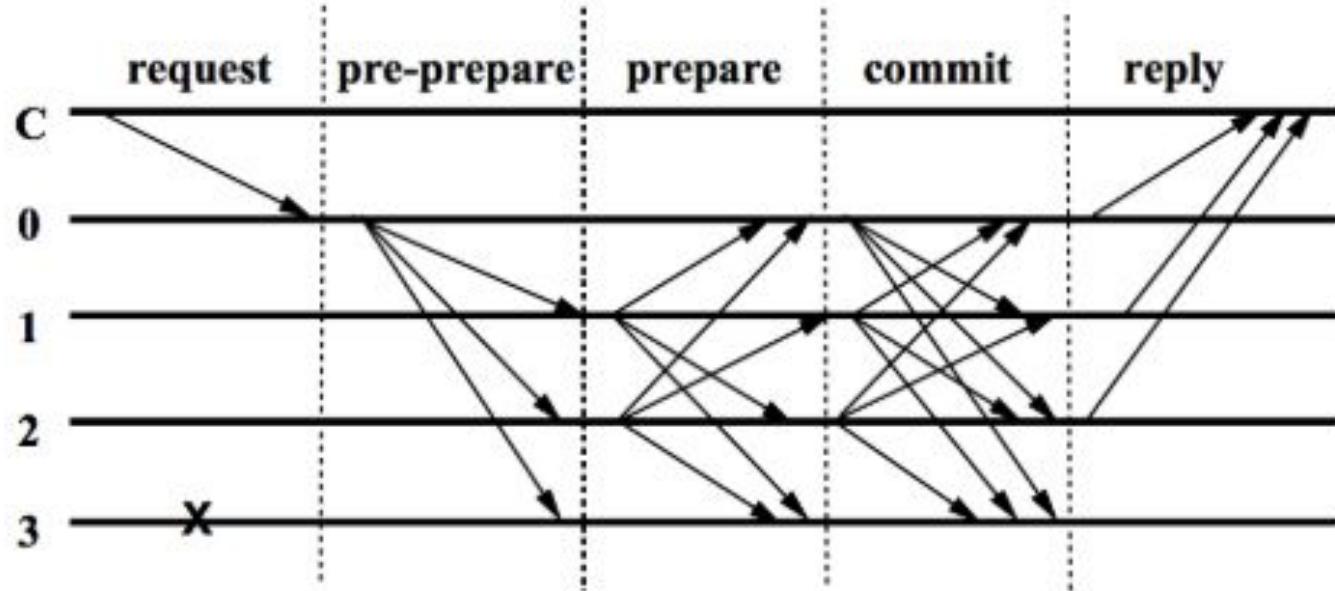
L'utilisateur doit prouver la **possession d'une certaine quantité de crypto-monnaie** (leur « participation » dans la crypto-monnaie) et **la mettre en gage** pour prétendre à pouvoir **valider** des blocs supplémentaires dans la chaîne de bloc et de pouvoir toucher la récompense, s'il y en a une, à l'addition de ces blocs.

Consensus d'Hyperledger

Byzantine Fault Tolerance

- Le consensus s'appuie sur un nœud primaire (général) et plusieurs nœuds secondaires (lieutenants)
- Les nœuds secondaires vérifient constamment la validité des décisions prises par le nœud primaire et peuvent collectivement nommer un nouveau nœud primaire si ce dernier se retrouve compromis
 - Ce mécanisme s'appuie sur des informations de contrôle qui augmentent exponentiellement avec le nombre de nœuds
 - Il est donc réservé exclusivement aux blockchains privées dont le nombre de nœuds est restreint (< 20)
- Un nouveau général est élu à chaque validation d'un nouveau bloc
 - Le général commence par choisir et ordonner les transactions du prochain bloc
 - Chaque lieutenant exécute les transactions et calcule un hash à partir des transactions
 - Si les 2/3 des lieutenants ont le même hash, le bloc est ajouté dans la blockchain

Consensus d'Hyperledger



Pros/Cons du Byzantine Fault Tolerance

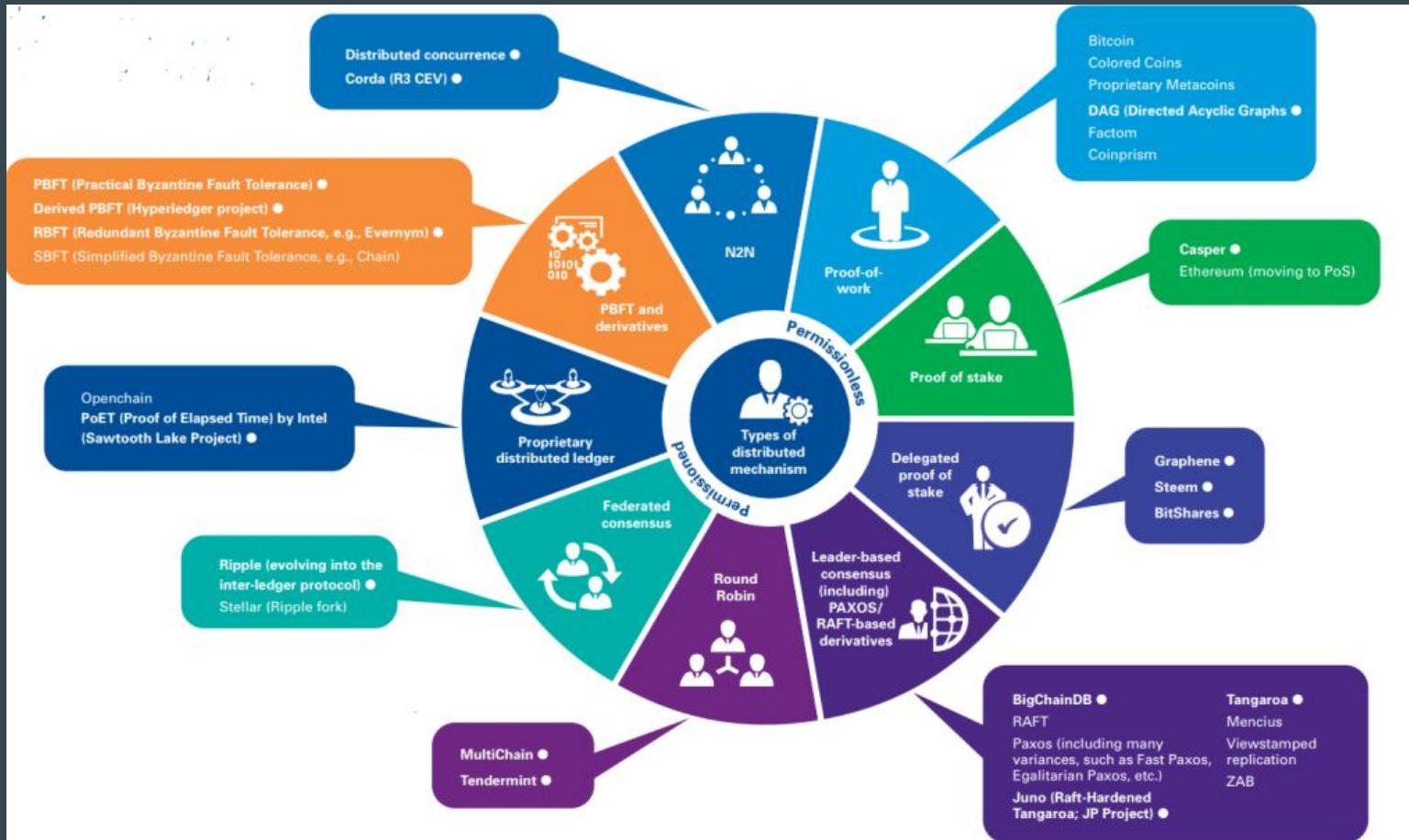
Avantages

- Pas énergivore
- Pas de forks de la blockchain
- Temps de transaction instantané

Inconvénients

- Pas d'anonymat
 - Cependant, le mécanisme de consensus Federated Byzantine Agreement implémenté par Ripple résout les problèmes d'anonymat et de scalabilité
- Mise à l'échelle difficile avec l'augmentation du nombre de nœuds en raison de l'overhead

Quel consensus pour quel blockchain ?



Base de données distribuée

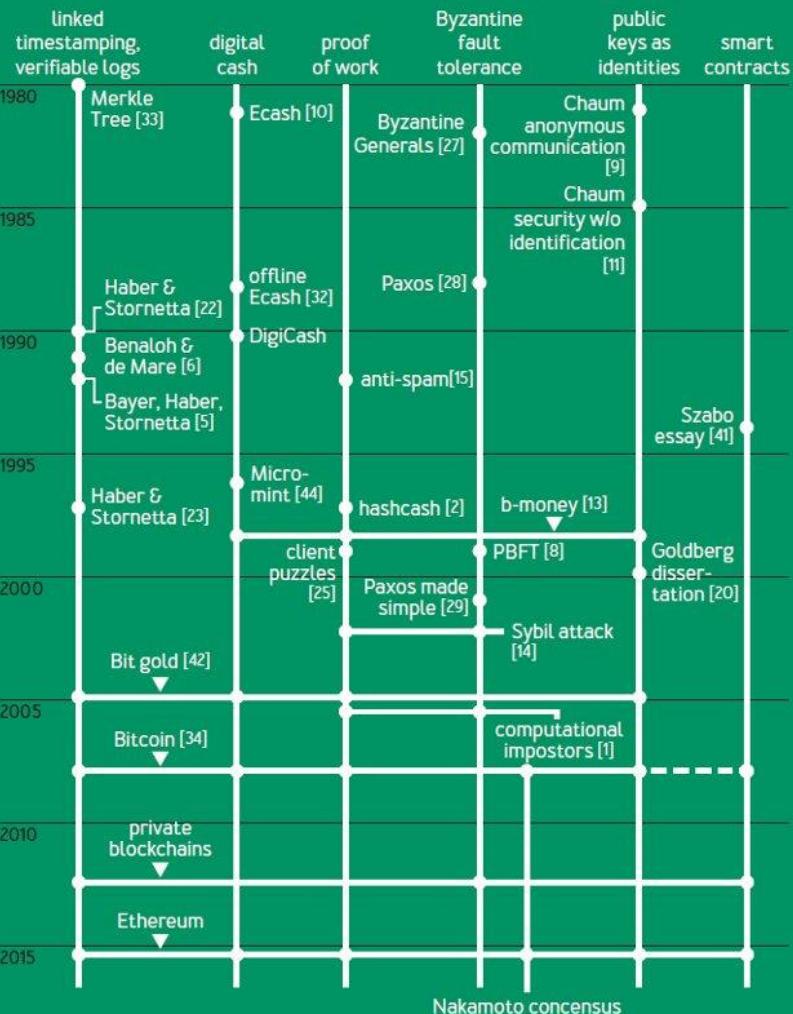
- Partagée par ses différents utilisateurs
- Chaque noeud du réseau possède une copie complète du registre
- Tout est public
- Transferts en P2P



Historique

La Blockchain n'est pas
apparue de nulle part

FIGURE 1: CHRONOLOGY OF KEY IDEAS FOUND IN BITCOIN



Première implémentation : Bitcoin

- Internet de la monnaie
- Cash numérique
- Fin 2008
 - white paper Satoshi Nakamoto : a peer to peer electronic cash system
- Février 2009
 - lancement du protocole
- 22 mai 2010
 - première transaction bitcoin 10 000 btc pour 2 pizzas (Pizza Day)



Industrialisation : Ethereum

- Un ordinateur mondial
- Une blockchain programmable
- Smart Contracts + Dapps
- Fin 2013
 - white paper publié par Vitalik Buterin
- Juillet/Août 2015
 - Crowdsale Ether => + de 15 millions de \$
- Juillet 2015
 - lancement du réseau public

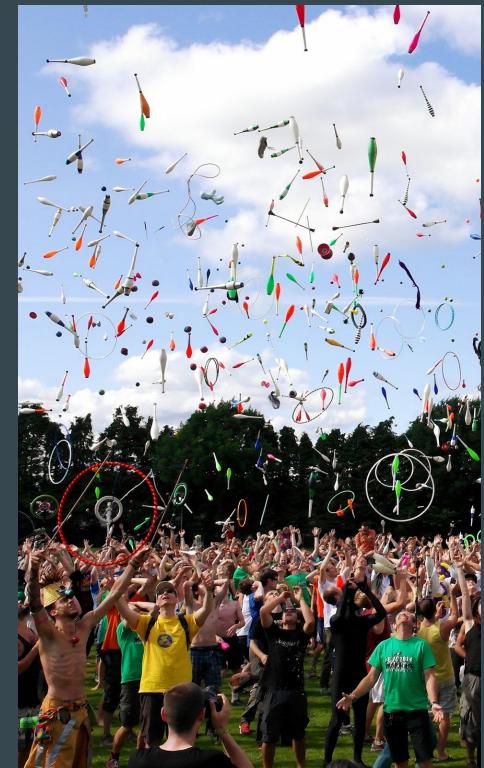


Blockchain publique

- Dans le cadre d'une blockchain publique, n'importe quel utilisateur peut
 - Lire les données
 - Créer son nœud distribué et participer ainsi au mécanisme de consensus
- Une crypto-monnaie est indispensable pour les blockchains publiques pour motiver les acteurs à valider des transactions
- Par défaut tous les nœuds d'une chaîne publique sont considérés comme non fiables. Il est donc nécessaire de mettre en place un mécanisme de consensus robuste à plusieurs types d'attaques dont :
 - Attaque Sybil (ou leurre d'identité) : Cette attaque est basée sur la création d'un grand nombre de comptes pour renverser un système communautaire
 - Attaque distributed denial-of-service (DDoS) : l'objectif de l'attaque DDoS est de rendre indisponible un service en utilisant un ensemble de terminaux accédant massivement et en simultané sur un système ciblé
- Le premier mécanisme de consensus à avoir été mis en place par les blockchains publiques Bitcoin et Ethereum est le Proof of Work (PoW)

Avantages d'une blockchain publique

- **Autonomie**
 - Règles établies par ses membres
- **Sécurité**
 - Blocs identifiés par de la cryptographie
- **“Eternelle”**
 - Pérennité des données par distribution
- **Transparence**
 - Audit et consultation de toute la BC par tous les membres
- **Automatisation**
 - Smart contract
- **Mondiale**
 - Accessible partout dans le monde



Blockchain privée

- Dans le cadre d'une blockchain privée, tous les nœuds participant au mécanisme de consensus sont enregistrés, connus et authentifiés par une entité communautaire de type consortium
- Une blockchain privée n'a besoin ni de crypto-monnaie, ni de preuve de travail puisque la motivation des acteurs ne repose pas sur les mêmes ressorts qu'une blockchain publique
- Le nombre de nœuds d'une blockchain privée est bien plus faible que celui d'une blockchain publique et ses nœuds sont considérés comme sûrs
- Un des premiers mécanismes de consensus à avoir été mis en place par les blockchains privées comme Hyperledger est le Practical Byzantine Fault Tolerance (PBFT)

Avantages d'une blockchain privée

- **Permissions**
 - Tous les participants sont connus
- **Performance**
 - Peu de noeuds
- **Séparation des données**
 - Certaines données peuvent être privées à quelques noeuds via les channels
- **Requête sur les données**
 - Utilisation de CouchDB pour faire des requêtes sophistiquées
- **Modularité**
 - L'architecture modulaire accepte les plugins



Smart contract

- Un smart contract est l'implémentation numérique d'un contrat dans une blockchain
 - Il peut être exécuté par les utilisateurs de la blockchain ou un autre smart contract
- Le principe est simple : disposer d'un ensemble d'instructions à exécution automatique entre deux parties, qui ne nécessitent pas de surveillance ou de mise en application par une tierce partie
- Le comportement d'un smart contract est déterministe afin que tous les nœuds de la blockchain puissent rejouer la même exécution dans un contexte d'appel donné
 - Pour cela les smart contracts utilisent les données endogènes de la blockchain
 - Les données exogènes doivent être introduites dans la blockchain par un système d'oracles
 - Le rôle d'un oracle est de fournir les valeurs externes à la blockchain attendues par les smart contracts de manière fiable et sécurisée

Exemple de smart contract d'une Dapp

- 1** Le smart contract du pari PSG - Rennes est mis dans la blockchain avec l'adresse du wallet contenant les gains à distribuer, la date d'échéance et l'adresse où trouver le résultat du match
- 2** Les paris sont ajoutés dans le smart contract jusqu'à la date de clôture
- 3** Un oracle ajoute le score du match dans la blockchain à l'adresse attendue par le smart contract
- 4** Le smart contract récupère le résultat du match à l'adresse attendue
- 5** Le smart contract distribue les gains dans les wallets des parieurs qui ont trouvé le bon résultat

C'est quoi une Dapp ?

- Application
 - Frontend classique (Javascript...)
 - Backend avec des smart contracts sur infrastructure décentralisée (blockchain, tangle...)
- Crypto-monnaies et tokens pour payer l'infrastructure, les services et les produits
- Le degré de décentralisation augmente avec le nombre d'utilisateurs



Pour quels types de projets ?

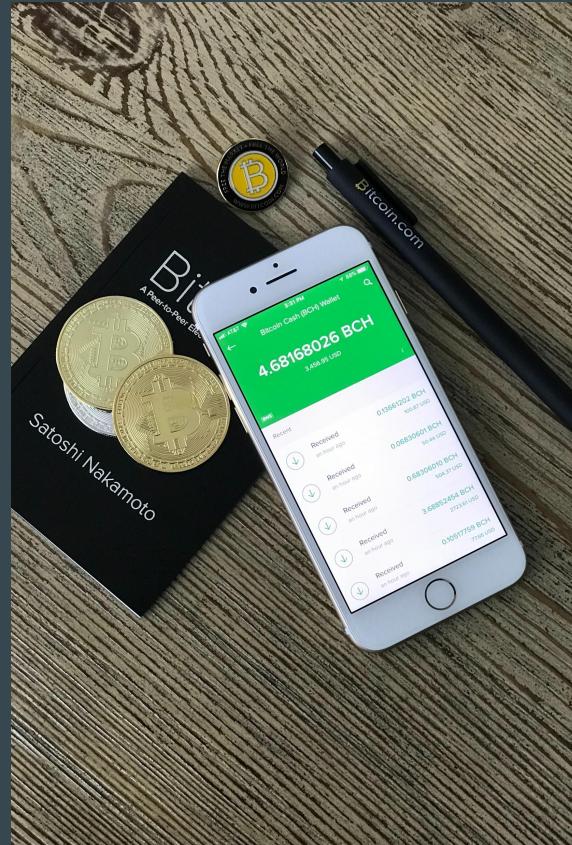
- Communications 29.7% (\$1,942,924,607)
- Finance 13.5% (\$884,907,884)
- Trading & Investing 11.7% (\$767,724,116)
- Gaming & VR 6.9% (\$451,867,213)
- Payments 6.1% (\$398,676,368)
- Commerce & Advertising 5.4% (\$354,344,000)
- Infrastructure 3.7% (\$241,747,400)
- Machine Learning & AI 2.8% (\$180,569,201)
- Drugs & Healthcare 2.5% (\$161,528,011)
- Supply & Logistics 2.5% (\$161,040,968)
- Data Storage 1.7% (\$113,952,711)
- Social Network 1.7% (\$112,810,800)
- Mining 1.6% (\$105,131,400)
- Energy & Utilities 1.5% (\$101,500,000)
- Marketplace 1.2% (\$81,452,015)
- Privacy & Security 1.0% (\$67,193,221)
- Data Analytics 1.0% (\$63,800,378)
- Identity & Reputation 0.8% (\$53,286,200)

- Compliance & Security 0.8% (\$52,828,101)
- Travel & Tourism 0.8% (\$52,731,006)
- Transport 0.7% (\$44,216,324)
- Gambling & Betting 0.5% (\$35,111,027)
- Art & Music 0.5% (\$32,100,000)
- Governance 0.5% (\$30,000,000)
- Education 0.3% (\$21,164,376)
- Recruitment 0.2% (\$12,653,158)
- Real Estate 0.2% (\$11,050,000)
- Content Management 0.2% (\$10,977,534)
- Commodities 0.0% (\$1,169,000)
- Events & Entertainment 0.0% (\$1,003,600)

ICOs réalisées en 2018

Cas d'usage : finances

- **Transfert de valeur de pair à pair**
 - Échange de monnaies de manière décentralisée (Bitcoin)
- **Échanges transfrontières**
 - Contourner les lourdeurs du protocole SWIFT (Ripple)
- **Financement participatif**
 - Financer son projet blockchain et trouver sa communauté (Polymath)
- **Cartes de fidélité universelle**
 - Regrouper toutes les cartes de fidélité des commerçants d'un centre commercial (Loyyal)
- **Assurance**
 - Automatiser la prise en charge de l'assurance selon les termes du smart contract (Etherisc)

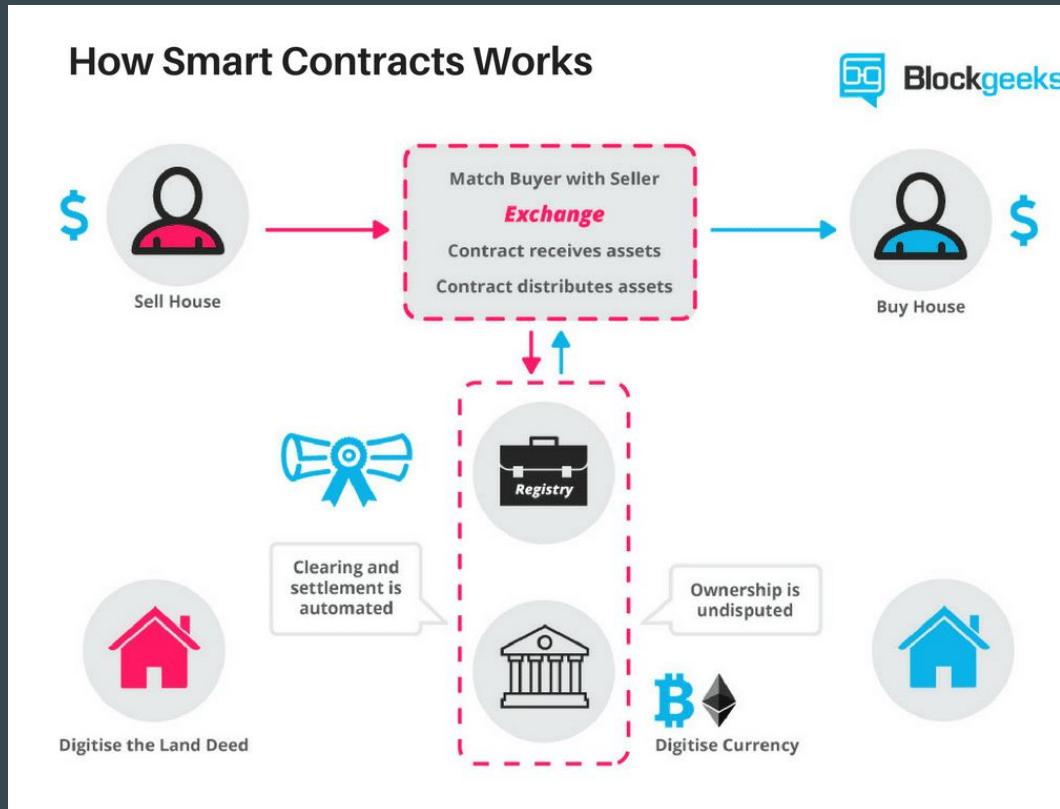


Cas d'usage : accords commerciaux

- **Du contrat commercial au smart contract**
 - Règles des transactions commerciales mis dans la blockchain (Ethereum)
- **Location**
 - Un smart contract envoie le loyer au propriétaire à l'échéance (Rentberry)
- **Transfert de propriété**
 - Un smart contract spécifie le propriétaire en fonction des règles dans le contrat (OpenBazaar)



Cas d'usage : immobilier



Cas d'usage : ressources humaines

- CV
 - Les écoles et les entreprises actualisent et vérifient le CV de la personne représentée par son smart contract (blockcerts)
- Promotions
 - Les objectifs donnés à un employé sont mis dans un smart contract et évalués tout au long de l'année (pas trouvé...)



Cas d'usage : héritage

- Testament
 - Les dernières volontés du testamentaire sont consignées dans un smart contract (WillChain)



Cas d'usage : loisirs

- **Billets de spectacle**
 - Acheter des billets de spectacle certifiés et les revendre au même tarif (Lava)
- **Plateforme de jeux**
 - Développer des jeux dans un écosystème blockchain (chimaera)

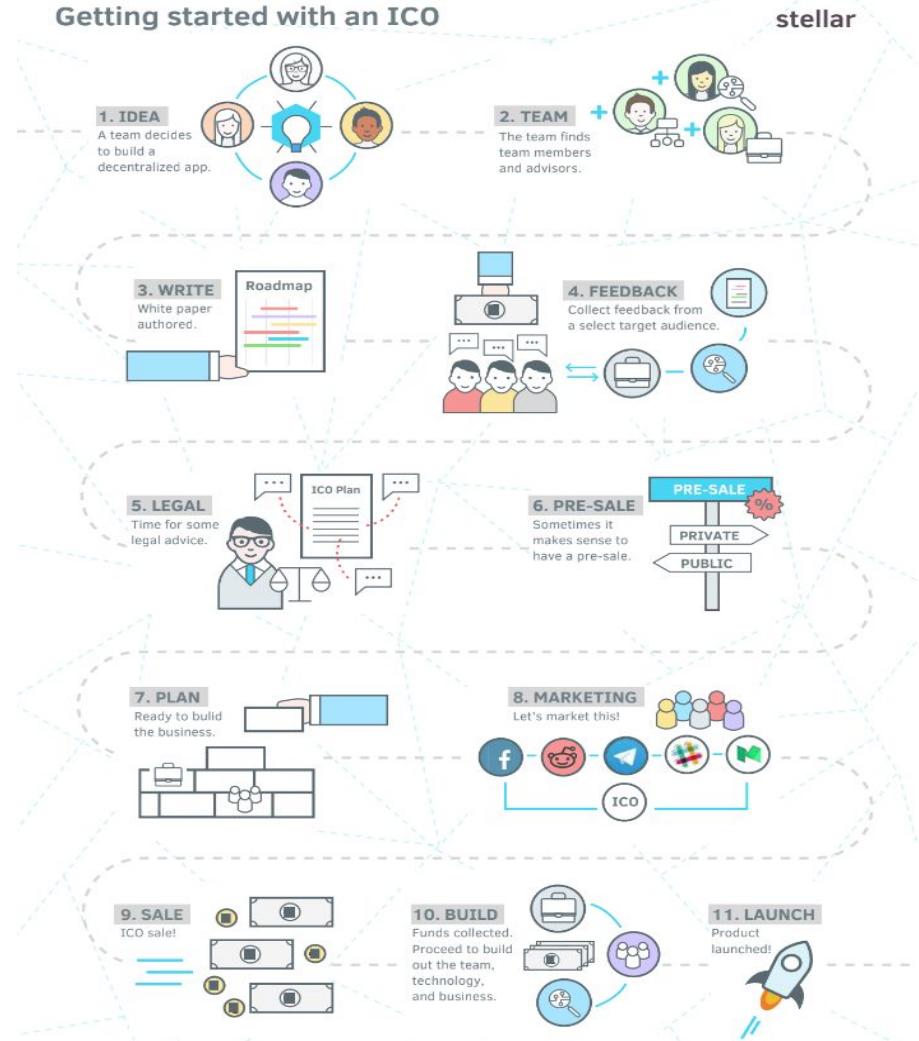


C'est quoi une ICO ou une STO ?

Un porteur de projet
organise un événement
pour lequel il met en vente des tokens
pour financer son projet
auprès de crypto-investisseurs

Méthodologie

Getting started with an ICO

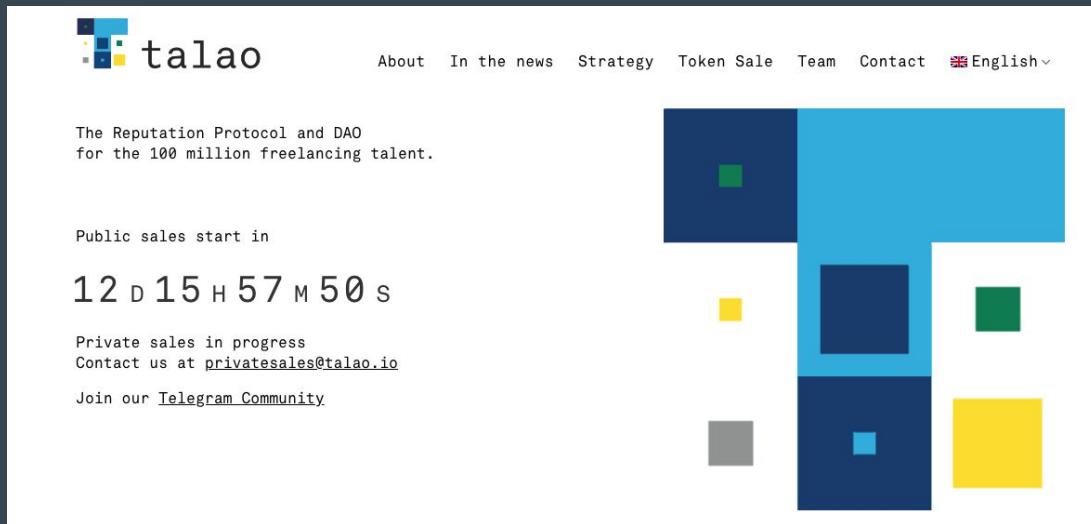


Exemple d'ICO



Sites web

- Site de l'ICO
 - <https://ico.talao.io/>
 - résumé du White Paper
 - accès au démonstrateur (site web)
- Site du produit
 - <https://talao.io/>



White Paper

- En anglais
 - http://ico.talao.io/wp-content/uploads/dlm_uploads/2018/03/Talao.io-White-Paper-V-1.0.pdf
- Plan
 - Marché
 - Plateforme Talao
 - Fonctionnalités
 - Tokenomics
 - Équipe et gouvernance
 - Roadmap

Table of Contents

EXECUTIVE SUMMARY.....	4
THE MARKET.....	6
Market Definition and Segmentation.....	6
Market Size.....	6
Existing Market Problems and Needs.....	7
Centralized Platforms own Talent data.....	7
Falsification of Resumes is common practice.....	7
Freelancing Talent lose 10 to 15% commission to platforms	8
Lack of power of Freelancing Talent versus their customers	8
THE TALAO PLATFORM.....	9
Talao Platform Value Proposition.....	9
Talao Business Ecosystem.....	10
Talent	10
Communities	10
Clients	11
Third Parties	11
Competitors	11
Key Benefits of the Talao Platform for the Ecosystem.....	15
Key Components of the Platform.....	16
The Professional Reputation Vault.....	16
Decentralized Autonomous Organization.....	18
Talent Marketplace Processes.....	20
The TALAO Token.....	24
TALAO Tokens creation	29
Talao Revenues Model.....	30
Talao Use cases	31
GOVERNANCE OF THE TALAO PROJECT.....	35
Management Board.....	35
Management Team Profiles	37
Advisors.....	38
Roles & Responsibilities	38
Composition of the Advisory Board	38
Statutory auditors	38
PROJECT ROADMAP.....	39
General Roadmap	39
Go-to market Roadmap	40
Technical Roadmap.....	43
Budget Allocation (post ICO)	45
Round A: Budget Allocation	46
Round B: Budget allocation	48
Group comparisons	49
Governing Law - Jurisdiction	50

Démonstrateur

- Disponible sur le site web
 - <https://talao.io/>
- Être opérationnel avant ICO
 - Convaincre les investisseurs
 - Nécessite du budget
- MVP
 - Pas toutes les fonctionnalités
 - Pas de bugs, bon design

Talao Requests Answers Circles My experts Add request

Welcome Gaël Durand

Created a new node corporate: Novolinko (entityreference_autocreate)
The changes have been saved.

Join circles Create a request Invite experts

Last requests

Search by keywords

Organisation

Circle

Type

Status

Domain(s)

Tags

Apply Reset

Senior stress engineer
Aviation
Open 3 Anonymous

Team leader Maintenance and operations support Railway project
Transportation systems
Expired 2 EXPERING

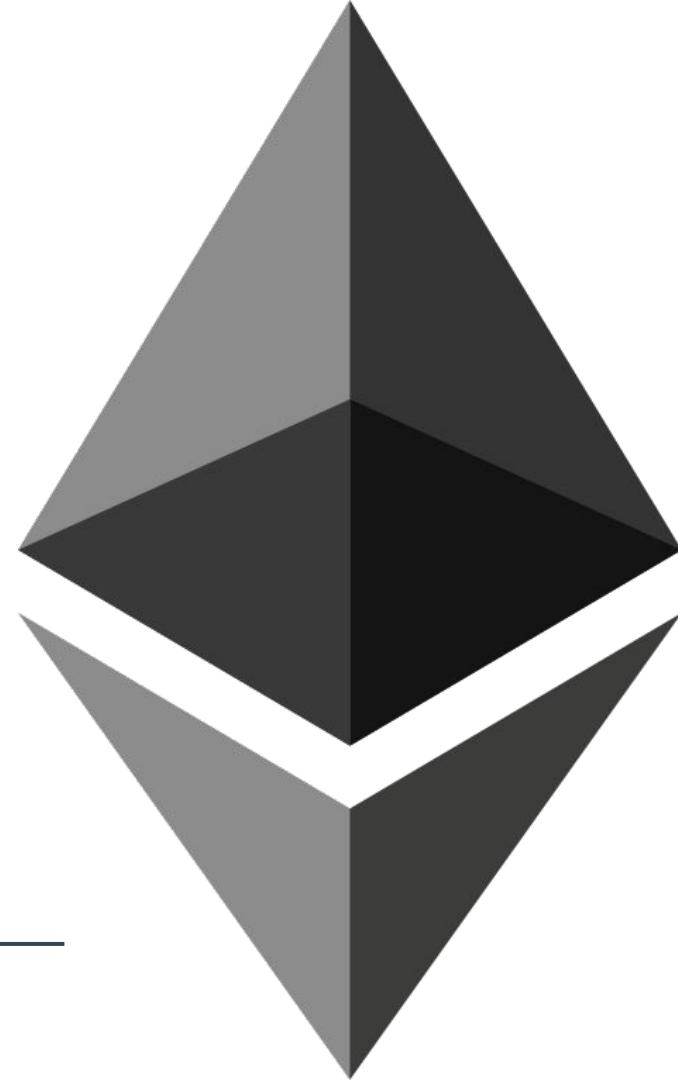
Hyperloop tube installation
Transportation systems
Expired 2 HYPERLOOP

Refer your network for a Quality Manager position (co-option reward : 1 500 €)
Aviation
Open 5 Anonymous

Ingénieur Functional Safety - Cybersecurity
Transportation systems
Open 0 Anonymous

Ingénieur Homologation
Transportation systems
Open 0 Anonymous

Origine et spécificités d'Ethereum



Machine de Turing et EVM

- Une machine de Turing est un modèle mathématique qui permet de résoudre théoriquement tous les problèmes du moment qu'elle possède le temps et la mémoire nécessaire
- La Machine Virtuelle d'Ethereum (EVM) est une machine de Turing
 - les smart contracts peuvent théoriquement résoudre tous les problèmes
 - l'EVM décode les smart contracts compilés en bytecode et déployés dans la blockchain à des adresses connues
 - le réseau Ethereum exécute les smart contracts avec les EVM des noeuds du réseau
 - Le résultat des EVM doit être identique sur tous les noeuds (déterminisme)

Mécanisme de consensus

- Le premier consensus utilisé par Ethereum a été le Proof of Work
- Des essais sont en cours pour le passage au Proof of Stake sur la blockchain principale (Casper V2 d'Ethereum 2.0)
 - La sortie est prévue en été 2019 (déjà reportée de nombreuses fois !)

Déploiement et utilisation d'un smart contract

- Le smart contract est codé dans un langage de programmation
- Le smart contract est ensuite compilé en bytecode reconnu par l'EVM et déployé sur la blockchain à une adresse donnée.
- L'ABI (Application Binary Interface) est l'interface qui permettra ensuite d'interagir avec ce smart contract.
- L'ABI contient :
 - Les noms des fonctions
 - Les types en entrée et sortie des fonctions
 - Les noms des événements et leurs paramètres

Langages de programmation d'Ethereum

- Actuellement, il existe trois principaux langages de programmation pour développer des smart contracts dans l'environnement Ethereum :
 - Solidity
 - Vyper
 - LLL
- Solidity est le langage officiel et le plus utilisé par la communauté
- C'est un langage original s'inspirant de Javascript, Java et Python principalement
- Solidity dispose du Compilateur Solidity (solc) pour compiler les smart contracts en bytecode reconnu par l'EVM
- Les 2 IDE les plus utilisés pour développer ses smart contracts en Solidity sont Remix et Truffle

Gaz

- Le gaz (gas) permet de récompenser les mineurs dans le consensus PoW lorsqu'ils prennent en charge une transaction.
- Lorsqu'une transaction change l'état de la blockchain, il faut indiquer le montant maximal de gaz que l'on est prêt à payer (gas limit)
 - Plus vous mettez une limite, plus vite votre transaction est traitée
 - Inversement, si la limite est trop basse, la transaction peut échouer car aucun mineur ne souhaite la traiter
- Le prix du gaz est la fraction d'ether que l'on est prêt à payer pour une unité de gaz.
- Le coût en gaz est le coût en gaz associé à chaque opération qui modifie l'état de la blockchain.

Gaz et smart contract

- Un développeur Solidity qui code des smart contracts qui seront ensuite déployés sur la blockchain publique doit faire attention à chaque instruction sinon son smart contract peut devenir très coûteux pour ses utilisateurs

Compte

- **2 types de compte**
 - Wallet possédé par un humain
 - Smart contract possédé par le code
- **Chaque compte de la blockchain Ethereum possède un propriétaire, une balance en ether et la possibilité de faire des transferts**
 - Cela signifie qu'un smart contract peut contenir des ether.
 - Cette propriété est à l'origine de nombreux usages et bonnes pratiques.
 - Comme un smart contract se possède lui-même, il faut lui préciser le propriétaire humain à la création si nécessaire (souvent le cas, par exemple pour avoir le droit de détruire le smart contract)

Transaction et stockage

- Chaque opération unitaire qui modifie l'état de la blockchain Ethereum est une transaction.
- Cela concerne :
 - Le déploiement d'un smart contract dans la blockchain
 - L'enregistrement d'une variable d'état dans la blockchain
 - L'enregistrement d'une variable locale d'une fonction étiquetée storage dans la blockchain
 - L'envoi d'Ether d'un compte à un autre
- Plus la donnée à stocker est grande, plus le coût de stockage est élevé.
 - Par exemple, il n'est pas envisageable de stocker des fichiers multimédia (IPFS à la place)
- Par contre, le paiement ne se fait qu'une seule fois, ie lors de la transaction.
- En plus des types d'instruction, ces notions de stockage sont à maîtriser dans votre architecture, à l'aide de design patterns, pour éviter des smart contracts hors de prix.

Crypto-monnaie

- L'ether est la monnaie qui sous-tend le fonctionnement d'Ethereum :
 - Crypto-monnaie pour échanger de l'argent entre 2 comptes tenus par un humain ou un smart code
 - nécessaire pour la gestion du gaz
 - L'ether est décomposé en plusieurs dénominations selon le fragment d'ether représenté
 - Le wei est l'unité de base (le nom vient de Wei Dai un crypto-activiste reconnu dans ce milieu)

Crypto-monnaie

<https://etherconverter.online/>

Wei	10000000000000000000
Kwei, Ada, Femtoether	1000000000000000
Mwei, Babbage, Picoether	1000000000000
Gwei, Shannon, Nanoether, Nano	1000000000
Szabo, Microether,Micro	1000000
Finney, Milliether,Milli	1000
Ether	1
Kether, Grand,Einstein	0.001
Mether	0.000001
Gether	0.000000001
Tether	0.000000000001
USD(at 134.442\$ p/ ether)	134.442
EUR(at 118.697€ p/ ether)	118.697

Récupérer des ethers

- Pour récupérer des ethers :
 - Etre un mineur
 - Echanger d'autres crypto-monnaies ou ses FIAT contre des ethers sur des exchanges dédiés
 - Utiliser Mist
- Nécessaire pour la blockchain Ethereum publique
 - Corollaire, pas besoin d'ether pour faire fonctionner une blockchain Ethereum privée !

Ecosystème Ethereum

- L'EVM et les smart contracts sont au centre d'Ethereum
- Mais comment sauvegarder les gros fichiers ?
- Des systèmes de stockage de fichiers ont été mis en place
 - IPFS (Interplanetary File System) le plus utilisé
 - Swarm celui intégré dans la stack Ethereum mais moins mature

IPFS

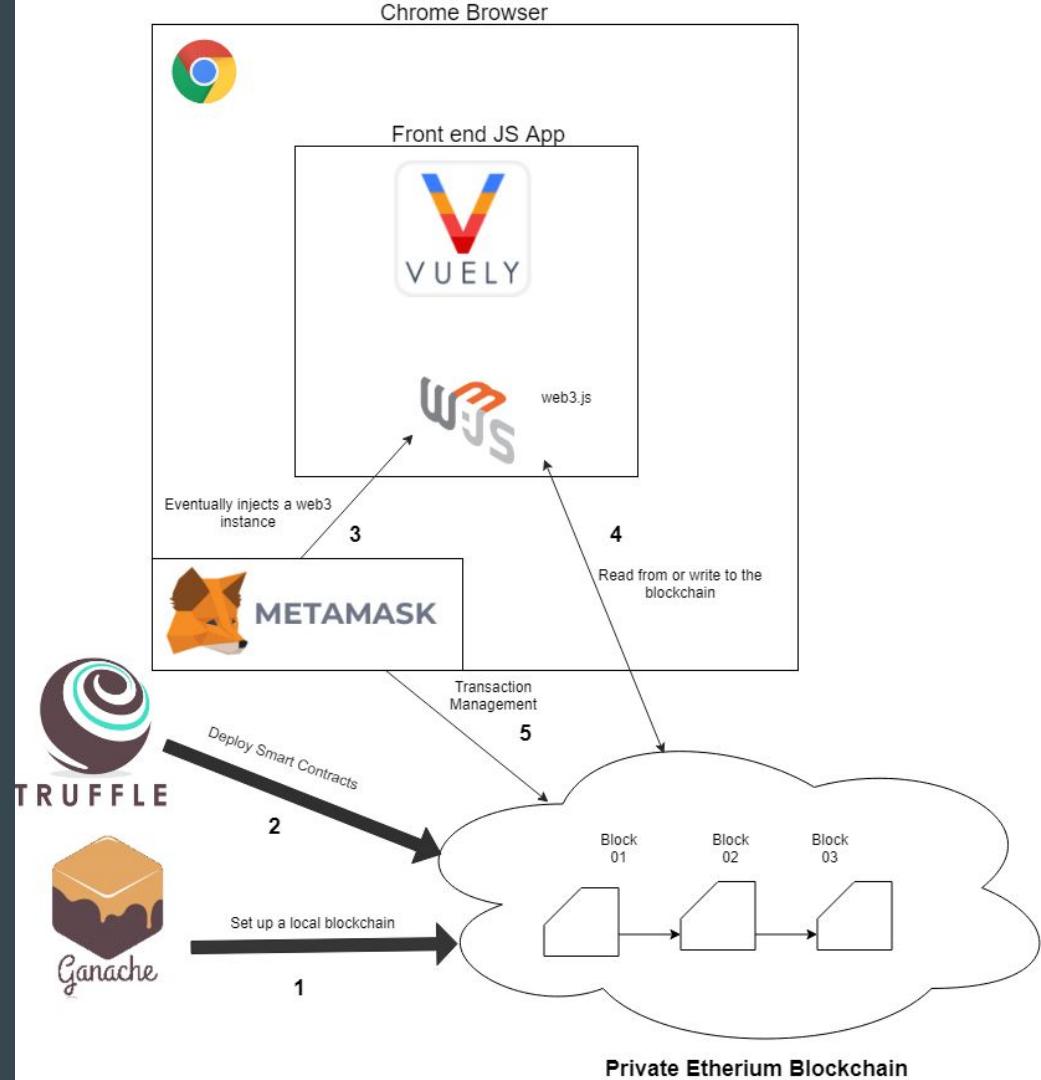
- Projet open source avec une communauté très active
- Réseau pair à pair pour stocker et partager des données
- Basé sur les concepts des protocoles P2P performants (Git, Bitcoin, BitTorrent)
- IPFS connecte tous les appareils avec le même système de fichiers (~ web)
- Ajout d'un fichier
 - Ajouter un fichier dans IPFS
 - Option : crypter le fichier avec la clé publique du destinataire
 - IPFS crée une valeur de hachage commençant par Qm
- Partage d'un fichier
 - Envoyer la valeur de hachage du fichier au destinataire
 - Le destinataire utilise IPFS pour télécharger le fichier
 - Option : Le destinataire utilise sa clé privée pour décrypter le fichier
- Utilisation de IPFS dans Ethereum : stocker la valeur de hachage du fichier crypté ou en clair

Exemple d'une Dapp : Elizeth

- Le premier chatbot de l'histoire : Eliza
- Simule un psychothérapeute qui répond à une question de son patient par une autre question
- Développé en Solidity avec la suite Truffle
- Le patient a la possibilité de payer le psychothérapeute à la fin de la séance en envoyant des ethers au smart contract d'Elizeth
- Le propriétaire du smart contract est le seul à pouvoir récupérer l'argent déposé sur le smart contract par les patients



Architecture d'Elizeth



Démo d'Elizeth en local

ChatBot

DID YOU COME TO ME BECAUSE YOU ARE HAPPY ?

ELIZETH

USER

Not really

ELIZETH

PLEASE TELL ME MORE.

ELIZETH

USER

I ate too much yesterday

ELIZETH

LET'S CHANGE FOCUS A BIT... TELL ME ABOUT YOUR FAMILY.

ELIZETH

USER

My brother is tall

ELIZETH

WHY DO YOU SAY THAT YOUR BROTHER IS TALL.

Submit

Pay what you want

Withdraw

Projet Hyperledger

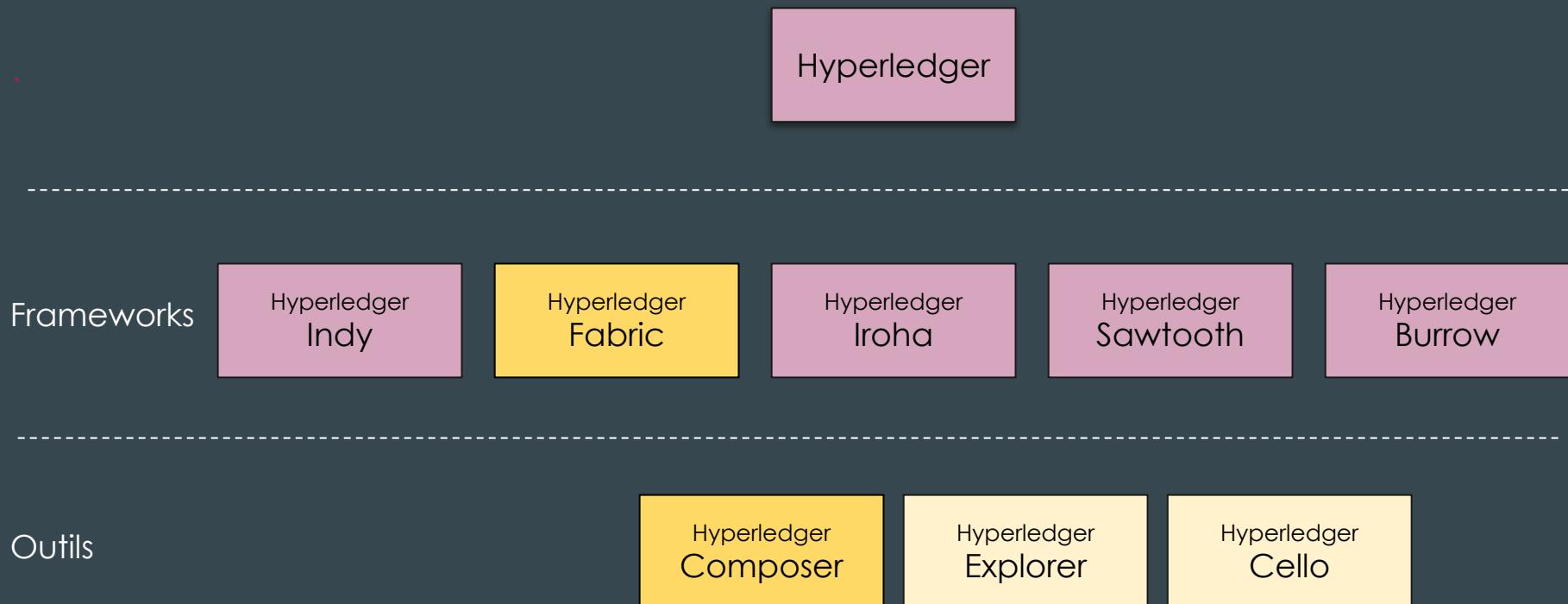
HYPERLEDGER



Présentation générale

- Hyperledger est un projet open source hébergé par la fondation Linux, conçu comme un framework autour des registres distribués d'entreprise
- Il ambitionne de répondre aux inconvénients perçus autour des premières blockchains
 - Débit limité
 - Confirmations de transaction lentes
 - Processeurs anonymes
 - Orienté autour des crypto-monnaies
 - Mauvaise gouvernance
 - Pas d'intimité
- Approche modulaire
 - Hyperledger est un incubateur pour de multiples projets avec une approche cohérente de la propriété intellectuelle et des normes de collaboration communautaire
 - Effort de collaboration open source pour faire progresser les technologies multisectorielles blockchain

Approche modulaire



Description des modules d'Hyperledger

- **Hyperledger Fabric**
 - Infrastructure applicative d'exécution pour la blockchain (nœuds homologues, nœuds de commandes, registres, autorités de certification), permettant l'exécution de smart contrat (chaincode)
IBM est le contributeur principal de ce projet
 - SDK Go, JavaScript et Java
- **Hyperledger Iroha**
 - Basé sur Hyperledger Fabric et orienté pour les applications mobiles
- **Hyperledger Sawtooth**
 - Infrastructure applicative d'exécution alternative à Fabric et supportée par Intel
 - SDKs Python, Go, JavaScript, Rust, Java, et C++

Description des modules d'Hyperledger

- **Hyperledger Composer**
 - Ensemble d'outils permettant de créer des réseaux blockchain d'entreprise, qui propose des abstractions centrées sur l'entreprise (modélisation des données des contrats, des participants et des contrôles d'accès), ainsi qu'un environnement de test WYSIWYG (playground)
- **Hyperledger Cello**
 - Orchestrateur pour le provisionning et le déploiement de l'infrastructure applicative Fabric
- **Hyperledger Explorer**
 - Pour afficher, invoquer, déployer ou interroger des blocs, des transactions et des données associées, des informations réseau et des chaincodes
- **Hyperledger Quilt (interopérabilité inter-registres), Hyperledger Caliper (benchmark), Hyperledger Burrow (contract machine), Hyperledger Indy (identité décentralisée)**

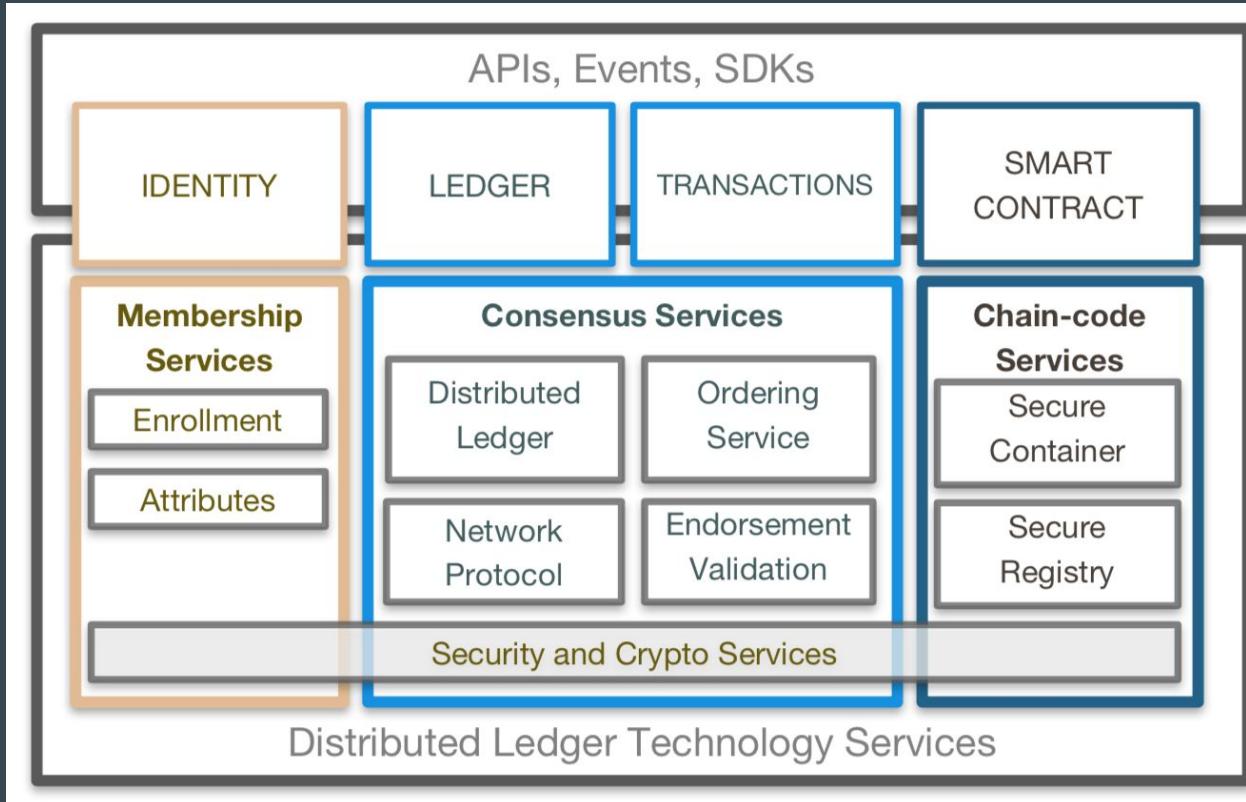
Mécanismes de consensus

- Hyperledger Fabric
 - Le mécanisme de consensus est conçu pour fonctionner comme un module
 - Il est donc à priori interchangeable
 - Il utilise actuellement Kafka, mais supportera à l'avenir les mécanismes de consensus Simplified Byzantine Fault Tolerance, Honey Badger, et Smart BFT
 - Le mécanisme mis en oeuvre avec Kafka est tolérant aux pannes mais pas Byzantine Fault Tolerant, on doit donc faire confiance à tous les approuveurs
 - Il est possède une latence très faible, une bonne mise à l'échelle mais une politique de confiance faible

Mécanismes de consensus

- Hyperledger Sawtooth
 - Le mécanisme de consensus est conçu pour fonctionner comme un module
 - Il est donc là aussi interchangeable
 - Les mécanismes de consensus existants sont Proof of Elapsed Time (PoET) et PoET-SGX
 - Ce mécanisme s'appuie sur des instructions CPU spécifiques aux chipsets Intel
 - L'ajout des mécanismes PBFT et Raft est planifiée à moyenne échéance

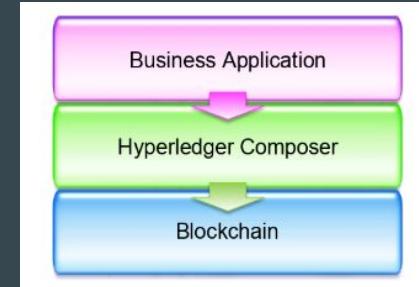
Architecture d'Hyperledger Fabric



- **Identité**
 - Appartenance, confidentialité et auditabilité de transaction
- **Registre & Transactions**
 - Registre transactionnel distribué dont l'état est mis à jour par consensus des parties prenantes
- **Smart Contract**
 - Chaincode
- **APIs, Events, SDKs**
 - Go, JavaScript et Java

Hyperledger Composer

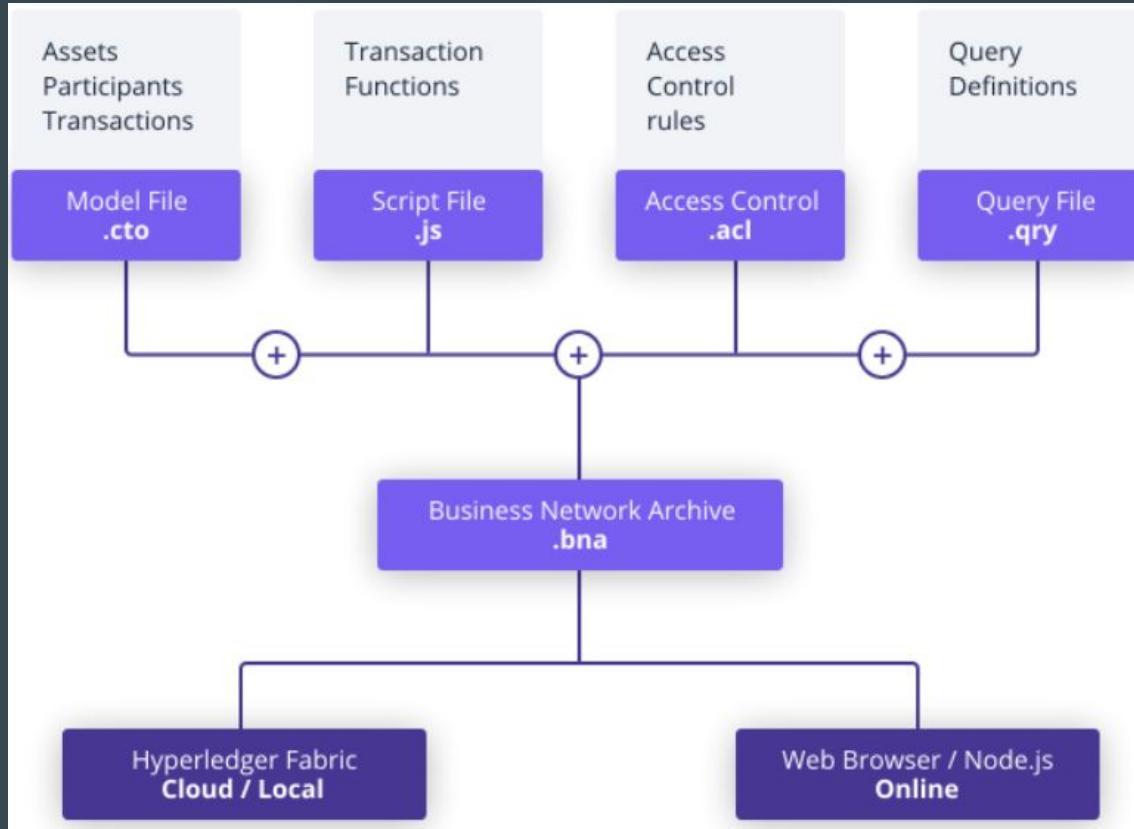
- **Constat**
 - Les smart contracts écrits en chaincode contiennent trop de code technique sans valeur ajoutée et ne sont pas assez portés sur la logique métier
- **Hyperledger Composer**
 - Suite d'applications d'abstractions de haut niveau pour les réseaux d'entreprise
 - L'accent est mis sur le vocabulaire métier pour la création rapide de solutions
 - Réduire les risques et augmenter la compréhension et la flexibilité
- **Caractéristiques**
 - Modélisation, tests et exposition via des API des réseaux d'entreprise
 - Les applications appellent les transactions de l'API pour interagir avec le réseau d'entreprise
 - Intégration avec les systèmes existants à l'aide des API REST



Concepts d'Hyperledger Composer

- Dans la vision Composer, la blockchain s'appuie sur les concepts d'entreprise suivants
 - Les réseaux d'entreprise (Business Networks) relient les entreprises
 - Les actifs (Assets) circulent sur les réseaux d'entreprise
 - Les transactions décrivent l'échange d'actifs
 - Les participants soumettent des transactions
 - Les contrats définissent les règles métier les transactions
 - Le registre (ledger) est le journal des transactions
 - Il contient donc les informations sur les actifs échangés et les participants liés aux échanges

Vue d'ensemble d'Hyperledger Composer



IDE d'Hyperledger Composer

The screenshot shows the Hyperledger Composer IDE interface with several numbered callouts:

- 1**: A red box highlights the "FILES" sidebar on the left, which lists project files: About, Model File, Model File, Script File, Script File, Script File, Access Control, and Add a file... Export.
- 2**: A red box highlights the "UPDATE NETWORK" section at the bottom left, showing deployment details: From: 0.0.5-deploy.0 To: 0.0.5-deploy.1, and a Deploy changes button.
- 3**: A red box highlights the "Test" tab in the top navigation bar.
- 4**: A red box highlights the main code editor area containing an ACL file named "permissions.acl".
- 5**: A red box highlights the status bar at the bottom right, which says "Everything looks good!" and "Any problems detected in your code would be reported here".

FILES

- About
README.md, package.json
- Model File
models/car-insurance-model.cto
- Model File
models/car-repairer-model.cto
- Script File
lib/car-insurance-script.js
- Script File
lib/car-repairer-script.js
- Script File
lib/permissionHelper.js
- Script File
lib/script.js
- Access Control
permissions.acl

Add a file... Export

UPDATE NETWORK

From: 0.0.5-deploy.0
To: 0.0.5-deploy.1

Deploy changes

3

4

5

ACL File permissions.acl

```
1 // Insurer part
2
3 rule ParticipantsSeeSelves {
4   description: "Let participants see themselves"
5   participant(p): "org.hyperledger.composer.system.Participant"
6   operation: ALL
7   resource(r): "org.hyperledger.composer.system.Participant"
8   condition: (r.getIdentifier() == p.getIdentifier())
9   action: ALLOW
10 }
11
12 rule InsurerSubmitInsuranceClaim {
13   description: "All insurers can submit a insurance claim when they are involved with"
14   participant(insurer): "org.car_insurance_network.CarInsurer"
15   operation: CREATE
16   resource(request): "org.car_insurance_network.SubmitInsuranceClaim"
17   condition: (insurer.getIdentifier() === request.claim.insurer.getIdentifier() && request.claim.status === "CLAIM_CREATED")
18   action: ALLOW
19 }
20
21 rule InsurerWriteMessage {
22   description: "All insurers can add a new message for a claim when they are involved with"
23   participant(insurer): "org.car_insurance_network.CarInsurer"
24   operation: CREATE
25   resource(request): "org.car_insurance_network.WriteMessage"
26   condition: (insurer.getIdentifier() === request.claim.insurer.getIdentifier())
27   action: ALLOW
28 }
29
30 rule InsuranceAdjusterMakeReport {
31   description: "All insurance adjusters can make a report for a claim when they are involved with"
32   participant(insuranceAdjuster): "org.car_insurance_network.CarInsuranceAdjuster"
33   operation: CREATE
34   resource(request): "org.car_insurance_network.MakeInsuranceAdjusterReport"
```

Everything looks good!
Any problems detected in your code would be reported here

Playground v0.20.4 Tutorial Docs Community

Hyperledger Cello

- **Cello est un service de fourniture de blockchain et d'exploitation de ses services**
 - Simplification des problèmes de déploiement et de gestion des réseaux blockchain
 - Surveillance et analyse des activités du réseau à plusieurs niveaux
- **Ses principales fonctions applicatives sont**
 - La mise en place et la gestion d'instances de blockchain à travers un tableau de bord
 - La planification des ressources parmi divers pools de ressources, y compris des serveurs et des clouds de conteneurs
 - La surveillance automatique de l'état de santé de la chaîne
 - L'analyse du statut, du journal et des données en cours d'exécution pour les chaînes
- **C'est l'équivalent sur le papier d'outils comme Rancher ou OpenShift pour les µServices**
 - La version actuelle est très décevante dans la pratique, car peu opérationnelle
 - La topologie du réseau proposée est fixe

Hyperledger Explorer

- **Vue du réseau**
 - Blocs, transactions, noeuds, smart contrats et statistiques
 - Administration
 - Provisionning des noeuds
 - Déploiement et mise à jour des smart contracts
 - Health check
 - Notifications et alertes

Démo d'Hyperledger Composer

https://composer-playground.mybluemix.net/editor

Web animaltracking-network Define Test admin

FILES

About README.md, package.json

Model File models/com.hyperledger.composer...

Script File lib/model.cto.js

Access Control permissions.acl

Add a file... Export

UPDATE NETWORK

From: 0.2.7

To: 0.2.8-deploy.0

Deploy changes

About File README.md

Animal Tracking Network

This is an Animal Tracking Business Network based on UK DEFRA government regulations (<https://www.gov.uk/animal-movement-england>). Farmers can move animals between farms/fields and the UK government farming regulator has visibility into the locations of all animals and all animal movements between farms.

This business network defines:

Participants Farmer Regulator

Assets Animal Business Field

Transactions AnimalMovementDeparture AnimalMovementArrival SetupDemo

Each Farmer owns a Business that is identified by a Single Business Identifier (SBI). A Farmer owns a set of Animals. A Business owns a set of Fields. A Field contains a set of Animals owned by the Farmer. Animals can be transferred between Farmers or between Fields.

To test this Business Network Definition in the **Test** tab:

Submit a `SetupDemo` transaction:

```
{ "$class": "com.biz.SetupDemo" }
```

This transaction populates the Participant Registries with two `Farmer` participants and a `Regulator` participant. The Asset Registries will have eight `Animal` assets, two `Business` assets and four `Field` assets.

Submit a `AnimalMovementDeparture` transaction:

Legal GitHub

Playground v0.20.8 Tutorial Docs Community