**Spécification Interface fichier**

# Solène CHATEAU - Arthur DESOUCHES

# Jérémy GERMANICUS - Théo SEASSAL



Département MIDO

Année universitaire 2017 – 2018

M2 IF

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Modifications** |
| V 1.0 | 15/02/2018 | Création du document |

# I – Description

## Contexte

À la fois monnaie cryptographique et véritable alternative aux monnaies traditionnelles, Le Bitcoin présente une innovation majeure en proposant une nouvelle alternative au modèle financier actuel.

Le nom Bitcoin regroupe un système de paiement, une monnaie et une infrastructure de notariat électronique, capable de protéger l'intégrité des données qui y transitent.

Bitcoin est indépendant de tout organisme, banque, banque centrale ou état. Entièrement libre et transparent, il est une infrastructure communautaire et open-source. Le grand apport du Bitcoin est de nous offrir ce « livre magique » avec la technologie blockchain.

En appréhendant les concepts et mécanismes du Bitcoin, nous nous efforcerons de mettre en place un système de paiement peer-to-peer à l'image du système Bitcoin inventé par Satoshi Nakamoto dans ce cours de programmation par composants.

L’interface fichier est le premier composant constituant la blockchain. C’est lui qui définit sa représentation.

## Schéma des interactions avec les composants connexes

Composant 1 :

**Fichier**

Composant 5 :

**Vérificateur de bloc**

Composant 4 :

**Hacheur**

## Organisation

### 3.1- Fichier

Le fichier sera au format .txt. Il contiendra un ensemble de blocs et chaque bloc contiendra des transactions.

Une transaction sera un message avec une structure uniformisée. Une transaction enregistre une opération de transfert entre 2 utilisateurs. Chaque transaction est unique et non modifiable. Elle possède donc :

* Un identifiant
* Le numéro de bloc ou elle a été validée
* Une date
* L’identifiant de l’initiateur de la transaction
* L’identifiant du destinataire
* Le montant

Une structure de donnée pouvant être privilégié pour le format de la transaction peut être le JSON :

{

"TXI": [{

"Id": "1",

"nBloc": "1",

"nUTX0": "2",

"nTx": "3",

"signature": "az1"

}],

"TX0": [{

"Id": "1",

"nBloc": "1",

"nUTX0": "2",

"nTX0": "3",

"clé\_publique": "az1",

"hash": "az1"

}],

"TXM": {

"Id": "1",

"nBloc": "1",

"nUTX0": "2",

"nTX0": "3",

"clé\_publique": "az1",

"hash": "az1"

},

}

### 3.2- Bloc

Les transactions sont enregistrées dans des blocs. Les blocs sont eux-mêmes chainés les uns aux autres pour former la blockchain. Un bloc est composé de deux parties : une en-tête et un corps. L'en-tête permet de restituer le jeton d'horodatage. L’entête est une empreinte numérique qui authentifie un bloc comme un maillon de la blockchain à l’aide de diverses informations. Le corps contient l’ensemble des transactions effectué sur ce bloc.

Un bloc possède un identifiant, un hash, le nombre de transactions, son code de hachage et celui du bloc précédent, une date de création, la version du hash, la somme des transactions et une liste des transactions.

{

"Bloc":{

"hash" : "az1",

"nonce" : "1",

"previous\_hash" : "az2",

"num" : "1",

"TX": {

"TXI": [{

"Id": "1",

"nBloc": "1",

"nUTX0": "2",

"nTx": "3",

"signature": "az1"

}],

"UTX0": [{

"Id": "1",

"nBloc": "1",

"nUTX0": "2",

"nTX0": "3",

"clé\_publique": "az1",

"hash": "az1"

}]

},

"TXM": {

"Id": "1",

"nBloc": "1",

"nUTX0": "2",

"clé\_publique": "az1",

"hash": "az1"

}

}

}

## Fonctionnalités

### 4.1- Lecture du fichier

Cette fonction effectue une lecture du fichier, elle lit l’ensemble du fichier et identifie chaque bloc. La fonction prendra en paramètre le chemin d’accès au fichier et retournera la liste chainée des blocs.

### 4.2- Insertion d’un bloc après un bloc donné

Cette fonction a pour objectif d’insérer un bloc dans la blockchain et donc dans le fichier. Elle prend donc en paramètre le bloc à insérer et l’identifiant, ou le hash, du bloc précédent le bloc à insérer. Elle fera appelle au composant 5 Vérificateur de bloc pour l’insertion et la vérification de sa conformité.

### 4.3- Recherche d’un bloc

Cette fonction recherche un bloc dans la blockchain et renvoie le bloc trouvé. Sinon elle renvoie une erreur. Elle prendra en paramètre l’identifiant du bloc cherché ou le hash du bloc.

### 4.4- Vérification de tous les blocs

Cette fonction utilise la fonction de lecture du fichier pour identifier tous les blocs puis utilisera le composant 5 Vérificateur de bloc pour valider les blocs identifiés. Si le fichier n’est pas conforme elle renverra un message d’erreur.