Programmation par Composant

MCHICH Mohammed , MACHROUHI Ayoub , PISTIS Ipani

Mars 2019

Version	Date	Commentaire
1.0	11/03	Version initiale
2.0	21/03	Dernieres Modif

Ce document a pour but de rassembler les spécifications concernant le composant 2 portefeuille/wallet du projet de classe

1 Introduction

Un portefeuille est le composant qui stocke les clés privées et publiques et interagit avec toute la blockchain pour permettre aux utilisateurs d'envoyer et de recevoir cette devise "numérique" et d'obtenir des informaions sur leur solde.

Un porte-monnaie detient aussi une clé publique U**cet clé publique est son adresse Bitcoin.**

Il permet ainsi d'insérer de nouvelles transactions signées dans le registre Bitcoin, rajoutant a cela le fait que n'importe qui disposant d'un portefeuille/wallet pourra envoyer de l'argent à cette adresse. Dans la partie qui suit

Figure 1: representation graphique du Portfeuille/Wallet du bitcoin

on detaillera plus le focntionnement de notre composant ainsi que les interactions avec les autres composants.

2 Fonctionnement du composant 2

Premierement, le composant portefeuille doit pouvoir connaître à n'importe quel instant le montant qui lui est attribué, Rappelons qu'un compte est rattaché à une clé privée et une clé publique. Le portefeuille doit retrouver toute les transactions qui lui sont destinées.

Deuxièmement le composant doit pouvoir composer des transactions sachant que le portefeuille peut avoir plusieurs compte (plusieurs clés privées et plusieurs clés publiques).

Une transaction est toujours définie par une clé privée de l'émetteur ainsi qu'une clé publique du récepteur et finalement un montant qui lui est associé.

2.1 Processus du composant

Le processus du composant se présente comme suit :

- Connaître la valeur du portefeuille
  - Lecture des blocs du composant 1 avec la fonction **Bloc getBlocs()**
  - On récupère les UTXO de chaque bloc lu avec **UTXO getUTXO(bloc)**
  - On récupère les montants disponibles avec **Double getMontant(publicKey)**
- Composer une transaction :
  - Une clé privée de l'émetteur de la somme à envoyer
  - Une clé publique qui désigne le récepteur des BitCoin
  - La somme des Bitcoin à envoyer.
- Pour réaliser une transaction il faut :
  - Recevoir la **clé privée**
  - Recevoir la **clé publique**
  - et le **montant** qu'on va envoyer

le schéma ci-dessous explique comment une transaction utilisant la blockchain est passée et sécurisée.

La transaction deviendra un bloc de la blockchain, le block sera haché, (le hachage est une sécurité lorsqu'on veut cacher des données ou les rendre incompréhensible pour un utilisateur non convié) pour tous les mineurs. Dès que les mineurs connectés valideront la transaction, celle-ci sera chiffrée avec la clé secrète, puis cette clé secrète sera chiffrée par la clé publique de l'utilisateur B (destinataire).

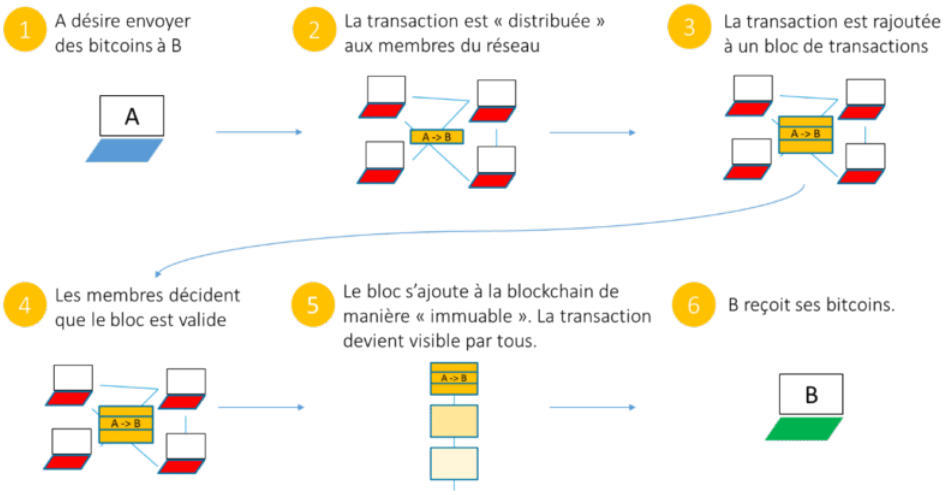

Figure 2: Resumé du processus du Bitcoin

Quand tout cela est fait la transaction maintenant validée va être concaténée avec la blockchain

2.2 Interactions avec les autres composants

hspace1cmEn ce qui concerne les interactions avec les autres composants : Dans un premier temps on fera appel au composant 6 pour signer la transaction avec notre clé privée.

Ensuite Envoyer la transaction au réseau, avec les informations suivantes : Emetteur, destinataire, montant disponible... au composant 5 pour la vérification et la validation de la transaction.

Enfin, Dès que tout le monde valide la transaction, elle est enregistrée sur un bloc dans le registre global des transactions.

- **Composant 1** : nous donne les blocs avec des numéros.
- **Composant 5** : il valide notre transaction.
- **Composant 6** : il signe la transaction.

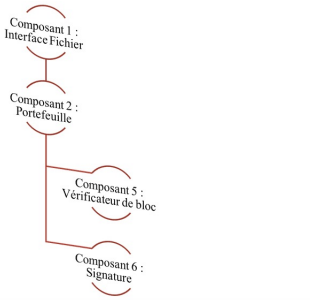

Figure 3: Schema d'interactions avec les autres composants

Avant de fusionner les composants on doit prendre en compte les informations suivantes :

- Les transactions forment une chaîne, les entrées des dernières transactions correspondent aux sorties des transactions précédentes.
  La clef d'Alice permet de créer une signature qui déverrouille les sorties précédentes, prouvant ainsi au réseau bitcoin que c'est elle qui détient ces fonds.

  Elle lie ce paiement à l'adresse de Bob, ce qui verrouille les fonds qui ne peuvent être utilisés que si Bob fournit une signature valable.

  Cela représente un transfert de valeur de Alice vers Bob.

  Cette chaîne de transaction, de Joe vers Alice puis Bob, est illustrée dans une chaîne de transactions, où la sortie d'une transaction est l'entrée de la transaction suivante.

3 Plans de test

Il s'agit d'une application constituée de 6 composants, et les interactions entre ceux-ci pouvant être complexes il va falloir choisir dans quel ordre réaliser ces tests d'intégrations. aprés une discussion avec les autres groupes le choix ci-dessous s'avere etre le meilleur.

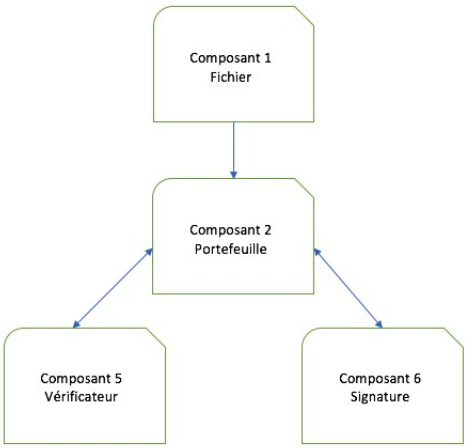

Figure 4: Architecture des intercations composant 2

pour pouvoir simuler au mieux le fonctionnement de notre composant On doit tout d'abord simuler les autres composants en l'occurrence, les composants ci-dessus (1, 5 et 6)
Composant 1: Nous donnons une blockchain avec l'hypothèse que les blocs sont valides
Composant 5 : On envoie des blocs valide et d'autres non pour simuler le retour du composant 5
Composant 6 :On transmet la transaction + clé privé et on attends la signature.