

Universidade do Minho
Escola de Engenharia

Departamento de Informática

Laboratórios em Engenharia Informática



Uma criptomoeda para micropagamentos utilizando
HyperledgerFabric

Bruno Machado, A74941

João Alves, A73542

Conteúdo

1	Introdução	3
2	Motivação	4
3	Ferramentas e plataformas open-source de tecnologia blockchain	6
3.1	Ethereum	7
3.2	OpenChain	8
3.3	Stellar	8
3.4	Corda & BigChainDB	9
3.5	Hyperledger	9
4	Hyperledger Fabric	10
4.1	Valores	10
4.2	Chaincode	10
4.3	Funcionalidades da Ledger	11
4.4	Transações	11
5	Criando blocos	13
5.1	Modificando o Chaincode	13
5.1.1	initledger	14
5.1.2	createPeer	15
5.1.3	queryPeer	16
5.1.4	transaction	16
5.2	Implementação de segurança	18
5.3	Utilização da moeda	18
6	Criação da Wallet	19
6.1	Implementação	21

7	Trabalho Futuro	23
8	Conclusões	24
9	Bibliografia	25

Capítulo 1

Introdução

No âmbito da Unidade Curricular de Laboratórios em Engenharia Informática do curso Mestrado Integrado em Engenharia Informática da Universidade do Minho, foi-nos porposta a realização de um projeto tendo em vista a criação de uma criptomoeda com os seguintes pressupostos:

- Avaliar as várias ferramentas e plataformas open-source de tecnologia blockchain (com ou sem smart contracts) que permitem implementar uma cripto-moeda.
- Determinar qual a mais adequada para micropagamentos.
- Desenvolvimento de um protótipo de uma cripto-moeda com wallet e micropagamentos

Assim neste relatório iremos descrever os passos a as decisões tomadas no desenvolvimento deste projeto de modo a conseguirmos cumprir com os pressupostos descritos anteriormente.

Capítulo 2

Motivação

No período inicial deste ano a Bitcoin, Blockchain e as Criptomoedas foram temas dominantes nas notícias, tanto o seu uso como valorização, passando até pelo uso da Blockchain para além das criptomoedas. Estes são temas que tinham despertado o nosso interesse ainda antes de terem chegado às notícias e tendo a oportunidade de participar num projeto que envolvesse Blockchain e a criação da nossa própria Criptomoeda foi algo que nos chamou a atenção desde o início e um projeto no qual decidimos entrar.

No whitepaper da Bitcoin publicado em 31 de Outubro de 2018 temos o seguinte excerto:

"A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution."

A ideia inicial da Bitcoin passa por tornar possível a realização de transacções monetárias (incluindo as de pequeno valor) sem a necessidade de uma entidade central, porém com a valorização da Bitcoin nos últimos anos, surgiram dois grandes problemas.

O primeiro desses problemas tem a ver que com esta valorização as taxas pagas aos "Miners" por cada transacção aumentaram consideravelmente pelo que para realização de pequenas transacções este custo acaba por tornar-se proibitivo e ser um fator limitante. O segundo tem a ver com o próprio "Mining" para a verificação dos blocos, com o aumento do valor da moeda, a recompensa dada aos "Miners" é cada vez maior pelo que surge um interesse económico-financeiro em realizar o mining da moeda.

Como vemos na figura 2.1, o consumo de energia relacionado com a Bitcoin está a atingir valores considerados preocupantes, pelo que soluções que sejam mais conservativas do ponto de vista ambiental têm sido procuradas.

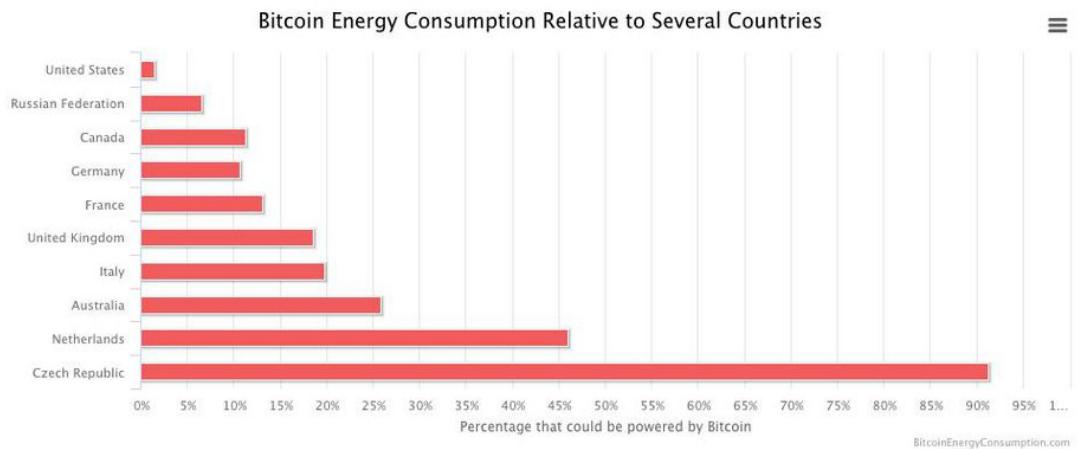


Figura 2.1: Bitcoin energy consumption compared to countries from digiconomist.net/bitcoin-energy-consumption

Tendo em conta os fatores acima mencionados, este projeto pareceu-nos deveras interessante porque busca implementar os valores originais da Bitcoin, ou seja, uma moeda apta a micropagamentos, sem a necessidade de uma entidade central, porém tentando usar soluções que vão de encontro aos problemas atuais da Bitcoin, sejam não ter custos adicionais para o utilizador, nem ter um custo de validação de transações tão elevado como o da Bitcoin.

Capítulo 3

Ferramentas e plataformas open-source de tecnologia blockchain

A primeira parte deste projeto passava por um processo de análise de plataformas e ferramentas open-source de tecnologia blockchain para que pudesssemos escolher a(s) que mais se adequava a implementação e criação da criptomoeda.

O nosso processo de análise de plataformas e ferramentas passou por pesquisa, participação em workshops de blockchain, tentar compreender qual das plataformas seria a melhor para utilizar para avançarmos com o nosso projeto.

As plataformas que numa primeira abordagem decidimos como passíveis de investigarmos foram:

- Ethereum
- HydraChain
- BigChainDB
- Corda
- Quorum
- OpenChain
- Stellar
- Hyperledger

3.1 Ethereum



Figura 3.1: Ethereum Alliance

À semelhança da Bitcoing, Ethereum é uma rede de blockchain distribuída e pública, apesar de existirem algumas diferenças técnicas entre as duas, a diferença mais importante está no propósito e capacidades do Ethereum que são bastante diferentes.

A Bitcoin é uma aplicação particular da tecnologia Blockchain que consiste em ser uma moeda eletrónica que permite pagamentos ”peer to peer”; e, enquanto que a blockchain da bitcoin é utilizada para rastreamento da posse da moeda digital, a blockchain de ethereum centra-se em correr pequenos ”exertos de código” denominados por ”smart contracts” de qualquer aplicação descentralizada.

Smart contract é uma expressão que é usada para descrever código que permite facilitar transações monetárias, de conteúdos, propriedades, ações ou outros items de valor, quando corrido na blockchain um smart contract é como se fosse um programa autónomo que vai ser executado quando as condições específicas para ativação do mesmo são conseguidas; uma vez que estes correm na blockchain eles correm extamente como programado sem que exista a possibilidade de interferência ou fraude.

Todas as blockchains têm a habilidade de processar código, porém a maioria de uma forma bastante limitada, é aqui que o ethereum é diferente uma vez que em vez de permitir um set de operações limitadas, ethereum permite aos programadores criarem quaisquer operações que eles queiram, assim é possível criar aplicações de uma multitude de maneiras diferentes para lá de algo que tenhamos visto antes.

Um exemplo disto é a famosa aplicação criptokitties que utilizando as capacidades da blockchain de ethereum e os smart contracts é uma espécie de jogo de gatos colecionáveis (o sucesso deste jogo foi tal que chegou a abrandar a própria blockchain do ethereum).

Ethereum tem a sua própria linguagem que é o solidity em que são escritos os smart contracts, o jogo CryptoZombies disponível em <https://cryptozombies.io> é um bom começo para quem quiser começar a programar em solidity e explorar melhor as capacidades da Ethereum blockchain.

Na figura acima vemos vários grupos e empresas que fazem parte da ethereum alliance, porém para aquilo que pretendíamos inicialmente, uma moeda sem custos associados às transferências para os utilizadores o ethereum está longe de ser a melhor solução uma vez que a todas as transações está associado um custo, e também porque sendo este um projeto de âmbito académico o custo de termos um simples protótipo da nossa moeda a correr na blockchain de ethereum seria algo que não poderíamos suportar, assim, tivemos de deixar de fora ethereum assim como plataformas baseadas em ethereum nomeadamente, **HydraChain, Quorum**

3.2 OpenChain

Openchain é uma tecnologia open source de um ledger distribuído.

Apesar de o Openchain ser escalável e trazer uma wallet integrada, o facto de a sua utilizar uma arquitetura cliente-servidor em vez de peer-to-peer não vai de encontro ao que pretendíamos pelo que decidimos à semelhança do ethereum não a utilizar.

3.3 Stellar

A Stellar lumens, é praticamente uma versão lançada que pode ser considerada a versão mais conhecida do que este projeto permite implementar, uma vez que tem um custo por transação quase negável para o utilizador e permite transações de valores muito baixos, porém a plataforma da stellar, permite-nos usar a sua coin, ou seja, stellar lumens como tokens de transações e implementar projetos e aplicações que utilizem as mesmas porém não é uma open-source que permita implementar uma criptomoeda, como é pretendido neste projeto, assim podemos ter a stellar lumens como uma espécie de modelo de produto final de algo que queremos obter porém não podemos utilizar como plataforma para desenvolver a nossa moeda.

3.4 Corda & BigChainDB

Estas plataformas ao contrário do que sucedeu com as anteriores acabaram por ser descartadas não por características das mesmas mas acabaram por não ser muito exploradas devido a termos começado uma pesquisa mais a sério da Hyperledger tendo estas ficado para segundo plano, no entanto de salientar que a ausência de global broadcast na Corda também era algo que não pretendíamos, assim como a pesquisa inicial da BigChainDB não revelou mostrar que esta fosse adequada ao nosso projeto, uma vez que à semelhança de outras plataforma OpenSource visava uma integração e utilização da sua blockchain ainda que utilizando tokens e não nos pareceu apropriado à criação de uma nova moeda.

3.5 Hyperledger

A plataforma que acabamos por escolher foi a HyperledgerFabric, o facto de ser um projeto robusto desenvolvido pela Linux Foundation e pela IBM, foram os factores que nos levaram a interessar pela mesma.

Esta plataforma possui imensa documentação e tutoriais do funcionamento da mesma e como criar novas implementações utilizando a plataforma, porém tais factos só por si não seriam suficientes para nos levar a optar por este caminho. Porém, para além destes factores o facto de o método de validação das transações ser através de consenso, dispensando assim o "Mining" para validar as transações, permite ir de encontro aos pontos que pretendíamos no desenvolvimento desta moeda nomeadamente a redução de custos da transação para o utilizador e temos uma moeda mais amiga do ambiente. O facto de existir bastante documentação do funcionamento da mesma e como criar novas implementações utilizando a plataforma e uma vez que o método de verificação das transações é por consenso vai de encontro a tudo o que pretendemos fez com que esta fosse a nossa plataforma escolhida para desenvolvemos o nosso projeto. As suas características e potencialidades encontram-se descritas no capítulo seguinte.

Capítulo 4

Hyperledger Fabric



A Hyperledger fabric é uma framework de implementação de blockchain e é um dos vários projetos da Hyperdeger da Fundação Linux.

Desenhada para ser uma fundação para desenvolver soluções com uma arquitetura modular, permite que os componentes sejam ”plug-and-play”.

4.1 Valores

Os valores representados podem ir desde dinheiro até contratos e representação de propriedade intelectual, a Hyperledger Fabric permite a capacidade de modificar estes valores utilizando transações em *chaincode*

Estes valores são representados como coleções de pares chave-valor em que o estado e as transações são guardados na ledger, estes podem ser representados em Binário ou num formulário JSON.

4.2 Chaincode

O *chaincode* é software que vai definir os valores representados na ledger, como as transações vão modificar esses valores. É o *chaincode* que vai criar e defenir as regras

para ler e alterar os valores pares de chave-valor ou outros estados de informação da base de dados. O *Chaincode* vai executar contra o estado atual da ledger através de propostas de transações, esta execução irá resultar num set de chaves-valor a serem escritos que podem ser submetidos a network e aplicados à ledger de todos os utilizadores.

4.3 Funcionalidades da Ledger

- Query e updates da ledger utilizando valores baseados em chaves
- Queries que apenas lêm resultados, podendo assim verificar-se o histórico de proveniência dos dados
- Transações consistem em chaves/valores que foram lidos e escritos no *chaincode*
- Transações conêm assinaturas de cada utlizador que a aprovou e são assim submetidas
- Antes de um bloco ser acrescentado, é realizada uma verificação que garante que os valores lidos não mudaram desde a execução do *chaincode*
- As transações são imutáveis uma vez que sejam validadas e submetidas
- Um canal ledger contém as definições de configuração do bloco, listas de controlo de acesso e outra informação pertinente
- Um canal contém um serviço de providenciação de membros que permite que os materiais criptográficos sejam derivados de diferentes autoridades certificadas.

4.4 Transações

Para que uma transação ocorra é necessário que o canal entre os dois utilizadores esteja aberto, o *Chaincode* esteja a correr e ambos sejam utilizadores da network.

O utilizador A pretende iniciar uma transação, uma aplicação suportando SDK, (no nosso caso utilizando Node) vai utilizar a API da Hyperledger Fabric para gerar a proposta de transação, esta é um pedido para a execução de *chaincode* para que os dados possam ser lidos e/ou escritos na ledger.

Os utilizadores que vão endossar a transação têm de verificar que esta está bem formada, não foi já submetida, devem verificar a assinatura/chave pública do utilizador que está a submeter a transação. Os utilizadores que aprovam a transação tomam os valores como

inputs das funções do *chiancode*, este é executado mas a ledger não é modificada, os valores gerados assim com as assinaturas dos endossadores são passadas como sendo uma resposta a proposta de transação.

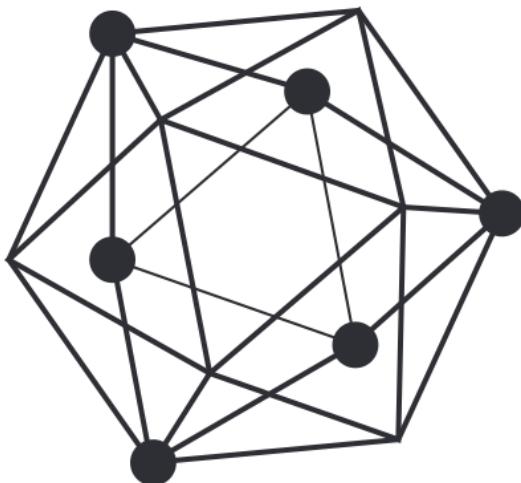
As propostas de transação dos vários endossadores são verificadas para ver se são iguais, e a aplicação verifica se esta cumpre a política de consenso definida para que a mesma possa ser submetida.

A proposta será convertida numa transação que vai conter os sets de pares chave-valores lidos/escritos e as assinaturas dos utilizadores que endossaram a transação.

Os blocos com as transações vão ser entregues a todos os utilizadores, as transações de cada bloco são verificadas para garantir que a política de consenso foi cumprida e que não houve alterações adicionais à ledger para além das efectuadas pelas transições validadas. Cada utilizador anexa o bloco à sua cadeia e por cada transação válida os sets de escrita são submetidos ao estado atual da base de dados. É emitido um evento que informa o utilizador que a transação foi adicionada à cadeia assim como se a mesma foi ou não validada.

Capítulo 5

Criando blocos



Decidida a plataforma a utilizar passamos então à implementação da moeda propriamente dita.

Uma pequena nota sobre o nome, uma vez que a criptomoeda criada se destina à execução de micropagamentos optamos pelo nome de Xxscoin uma vez que deriva de (Transfers extra small) e pode-se ler como "Axes" coin.

5.1 Modificando o Chaincode

Para podermos implementar a moeda tivemos de utilizando as ferramentas e packages providenciados pela hyperledger fabric fazer alterações ao chaincode para que a nossa moeda pudesse surgir

Inicialmente criamos o chaincode utilizando a linguagem **go**, porém devido a uma maior facilidade para gerar pares de chaves pública e privada do modelo RSA as assinaturas e verificar as mesmas acabmos por optar por desenvolver o chaincode também em node-js. As funções que implementamos no chaincode foram o initledger, createPeer, queryPeer e transaction.

5.1.1 initledger

```

async initLedger(stub, args) {
    console.info('===== START : Initialize Ledger =====');
    let peers = [];
    peers.push({
        value: '500000000',
        public_key: '-----BEGIN RSA PUBLIC KEY-----\n'+
        'MIIBCgKCAQEAhF6963IwAP8g2E7MC5Fr9yMQVkiuTFF5VT16pW4fhYcJtLEWw7N8BuS7GKVR\n'+
        '84vtJUsbNlgtbCpVnk8adS4DavtQvUdzBcBzIhpduU1P4iFQyIgpQN8yNAcXCiYbd0a1TVxL\n'+
        '4n8TU7LixWkzPyfPdhuCcMCvSEU7v1caDdYyYmZwz8Nk19fptPyn90t6FH6aHx2MYHOARTM5\n'+
        'cTs3kc0huUEDukPRjho971Hf2n5F2SChW7GdC21fH2Ix/Fj2U1Ae8HgzD2iYnwkDfKVPUhGb\n'+
        'wK4uj9K+IA/6Ftnom8oAexuJAwyktqUyaoSqTrZFSM4+26mTLbcTveVB8UVCvOrkwIDAQAB\n'+
        '-----END RSA PUBLIC KEY-----\n'
    });
    peers.push({
        value: '500000000',
        public_key: '-----BEGIN RSA PUBLIC KEY-----\n'+
        'MIIBCgKCAQEApq/4WP31aFm3kg12CgIJwDugpi3XgLHstPdVnAzPKIZuSJWinUdaXR04i8K7\n'+
        'hW4VJf/wrhRqv57gg0E6nbnD/RFuw/8cdjrjj2+N0uA7Iz8k3Dwq1bR7/IqDIM/Hxp8RzT9\n'+
        '2X57XMxxCpIU8K5EEczc+nDxY2zuVGbCXlqtH+6/VIGDHov1HveVKmyT9wAn4E/Uv2HV0xcK\n'+
        'TnKGHtpcN3kjzX0vs6NAGJqNixpp5fcQ8vxDUt1MLr2Kco+g6hzq72sRQIdgx0kpNt8B51Z\n'+
        '1/JbtKkaMyxuZRdwzHk/0f1FAk/RmdJFloiyEnhCj0wp5pSGe3G36syCxCPfmD5FGQIDAQAB\n'+
        '-----END RSA PUBLIC KEY-----\n'
    });

    await stub.putState('MIIBCgKCAQEAhF6963IwAP8g2E7MC5Fr9yMQVkiuTFF5VT16pW4fhYcJtLEWw7N8BuS7GKVR\n'+
    '84vtJUsbNlgtbCpVnk8adS4DavtQvUdzBcBzIhpduU1P4iFQyIgpQN8yNAcXCiYbd0a1TVxL\n',
    Buffer.from(JSON.stringify(peers[0])));
}

```

```

    await stub.putState('MIIBCgKCAQEApq/4WP31aFm3kg12CgIJwDugpi3XgLHstPdVnAzPKIZuS
JWinUdaXRo4i8K7'+
'hHW4VJf/wrhRqv57gg0E6nbnD/RFuw/8cdjrjj2+NOuA7Iz8k3Dwq1bR7/IqDIM/Hxp8RzT9',
, Buffer.from(JSON.stringify(peers[1])));

    console.info('===== END : Initialize Ledger =====');
}

```

Esta função será responsável por inicializar a ledger com os dois utilizadores iniciais (que neste caso representam os membros que desenolveram o projeto) tendo cada um inicialmente o valor de 500000000coins, sendo que a cada um dos utilizadores iniciais criados inicialmente é associada uma chave pública.

5.1.2 createPeer

```

async createPeer(stub, args) {
    console.info('===== START : Create Peer =====');
    if (args.length != 1) {
        throw new Error('Incorrect number of arguments. Expecting 1');
    }

    var peer = {
        docType: 'peer',
        value: 0,
        public_key: args[0],
    };

    var temp = args[0].slice(31);
    let key = temp.substring(0,145);
    let nova = key.substring(0,72) + key.substring(73,145);
    await stub.putState(nova, Buffer.from(JSON.stringify(peer)));
    console.info('===== END : Create Peer =====');
}

```

Esta função será responsável por inicializar a criação de um novo peer da rede, sendo que deve receber como argumento a chave pública deste novo membro. Um utilizador

tem como chave uma fração da sua chave pública nomeadamente 144 caracteres, a esta chave está associado o seu valor de XXSCoin e a sua chave pública completa. Estes dois atributos são guardados num objeto JSON.

5.1.3 queryPeer

```
async queryPeer(stub, args) {
  if (args.length != 1) {
    throw new Error('Incorrect number of arguments');
  }
  let peer = args[0];

  let peerAsBytes = await stub.getState(peer);
  if (!peerAsBytes || peerAsBytes.toString().length <= 0) {
    throw new Error(peer + ' does not exist: ');
  }
  console.log(peerAsBytes.toString());
  console.log(peerAsBytes + "-----");
  return peerAsBytes
}
```

Esta será a função que irá verificar o ”saldo” atual de um dado utilizador.

5.1.4 transaction

```
async transaction(stub, args) {
  console.info('===== START : Transaction =====');
  if (args.length != 4) {
    throw new Error('Incorrect number of arguments. Expecting 4');
  }

  let peerAsBytes1 = await stub.getState(args[0]);
  let peer1 = JSON.parse(peerAsBytes1);

  let peerAsBytes2 = await stub.getState(args[1]);
  let peer2 = JSON.parse(peerAsBytes2);
```

```

let signature = args[3];

const verify = crypto.createVerify('SHA256');
verify.write(peer1.public_key);
verify.end();

if(verify.verify(peer1.public_key,signature,'base64') == false){
    throw new Error('Assinatura não verificada'+ signature)
}

if(parseFloat(peer1.value) < args[2]){
    throw new Error('Balanço insuficiente' + peer1.value);
}
if(parseFloat(args[2]) < 0.0000000001){
    throw new Error('Transferência não suportada: valor demasiado pequeno');
}
peer1.value = parseFloat(peer1.value) - parseFloat(args[2]);

peer2.value = parseFloat(peer2.value) + parseFloat(args[2]);

await stub.putState(args[0], Buffer.from(JSON.stringify(peer1)));
await stub.putState(args[1], Buffer.from(JSON.stringify(peer2)));
console.info('===== END : Transaction =====');
}
};


```

Esta é a função mais importante que foi implementada e como em cada blockchain a transação acaba por ser a unidade elementar da mesma, sendo que a blockchain é nada mais que um histórico distribuído e ordenado da realização de transações.

Como podemos verificar cada transação terá de receber quatro argumentos, sendo eles a chave pública parcial do primeiro e do segundo elemento (é utilizada uma chave pública parcial por motivos de eficiência e gestão do espaço na ledger), identificando assim os intervenientes na transferência, sendo que o terceiro argumento será o valor da transferência e como quarto elemento será uma assinatura com a chave privada da chave pública completa do primeiro utilizador.

De notar que para expandir a política de consenso da ledger seria aqui que deveríamos intervir de modo a implementarmos a política pretendida e de modo a verificar que a mesma é conseguida.

Para ser uma moeda específica para microtransações estipulamos o valor mínimo de transferência como sendo 0.0000000001 XXSCoins, uma vez que assim permite transferências de valores extremamente pequenos.

5.2 Implementação de segurança

A assinatura existente nas transações permite verificar a segurança das mesmas. Foi assinado um argumento extra para as transições. Foi assinado com a chave privada a chave pública completa do utilizador que faz a transação.

De salientar que apesar que apesar de ser utilizada uma chave pública parcial, no chain-code tem-se acesso à chave pública completa de modo a verificar a assinatura e garantir que foi mesmo aquele utilizador que efectuou a transferência.

Para criar os pares de chaves pública/privada usando o sistema RSA, utilizou-se uma biblioteca de node.js denominada keypair.

5.3 Utilização da moeda

A partir do momento que é inicializada a ledger, temos a moeda Xxscoin em circulação, sendo que o número de moedas em circulação é estipulado desde inicio, isto é pretendido uma vez que eliminamos o mining e a moeda pretende apenas representar transações de valores pequenos, existindo uma quantidade muito grande de moedas e permitindo fazer pagamentos com valores muito reduzidos das mesmas. Podem ser adicionados novos utilizadores, a qualquer momento, saber-se o saldo de qualquer utilizador desde que se conheça a chave pública do mesmo e fazer transações entre cada dois utilizadores registados na ledger.

Para que todas estas operações fossem mais fáceis implementamos uma wallet que descreveremos no capítulo seguinte.

Capítulo 6

Criação da Wallet

A wallet foi criada utilizando a framework electron que permite utilizar linguagens web como javascript,css,html para criar aplicações desktop.

A wallet desenvolvida tem 4 vistas principais: uma vista geral, uma vista para dar load a chaves privadas e públicas, uma vista para criar os par de chaves e finalmente uma janela para poder ser feita as transações.

1. **Vista Geral** A vista geral é a janela que aparece quando um utilizador abre a aplicação, nesta existem dois campos não podem ser escritos onde aparece a fração da chave pública que serve como key na blockchain, e o valor de XXSCoin associado a essa key neste momento.



2. **Vista de Create** Esta janela aparece apenas com dois botões, um para criar um par de chave que são escritos em dois ficheiros distintos e um botão de back que volta para a janela principal.



Figura 6.1: View da criação da wallet

3. **Vista de Load** Na vista de load existem dois campos de texto com o seu respetivo botão de browse de ficheiros que permite selecionar um ficheiro das diretórias existentes no computador. Depois de selecionar os ficheiros das respetivas chaves esta informação é utilizada para preencher os ficheiros de chave pública e privada da wallet.



4. **Vista de Transferência** A vista de transferência apresenta-se com dois campos de texto, uma para inserir a chave para a qual se pretende enviar XXSCoin e um segundo campo para inserir a quantidade de XXSCoin que se pretende enviar. Depois de ser preenchido os dois campos vaste carregar no botão send para enviar a respetiva quantia de XXSCoin.



6.1 Implementação

Para conseguir as funcionalidades inerentes a uma wallet é necessário consultar as blockchain e alterá-la constantemente. De seguida iremos explicar as funcionalidaes e a implemtação de cada vista.

1. **Vista Geral** Na vista geral são carregados através do jquery as informações contidas em dois ficheiros a chave.pub e o balance.txt que contêm a informação atualizada sobre a chave pública utilizada e o balanço atual da conta respetivamente. Esta informação é atualizada de 10 em 10 segundos de maneira a que o utilizador tenha sempre a informação o mais atualizada possível sem sobrecarregar a aplicação. De 10 em 10 segundo é também feita uma query à blockchain com a chave pública atual escrevendo esse valor para o ficheiro balance.txt. Esta funcionalidade é obtida através da importação do modulo presente no ficheiro connection.js.
2. **Vista de Create** Na vista de create ao carregar no botão "create" são criadas um par de chaves publico e privado através da biblioteca keypair, que são escritos nos ficheiros para a chave pública e privada da wallet. De seguida é tratada a chave pública criada e usado o script newPeer.js para criar um novo peer na blockchain com balanço de 0.
3. **Vista de Transferência** Na vista de transferência é utilizado o script transaction.js para fazer transferênciia de XXSCoin para outro utilizador. Este script recebe 4 argumentos a chave pública parcial do utilizador actual, a chave pública parcial do utilizador para o qual pretendemos enviar XXSCoin, a quantidade de XXSCoin

e uma assinatura feita assinando-se com a chave privada do utilizador atual a sua chave pública total. Este argumento são enviados para o chaincode onde é verificada a assinatura e caso seja verificada é subtraído o valor ao primeiro utilizador e somado ao segundo, concluindo-se a transferência de XXSCoin.

Capítulo 7

Trabalho Futuro

Tendo em conta o trabalho desenvolvido, como trabalho futuro pretendíamos ter a moeda ”Live”, isto é ter servidores remotos com o nosso chaincode e utilizando a wallet por nós desenvolvida sermos capazes de efectuar e realizar transações em tempo real.

Uma possível melhoria a fazer como referido no capítulo Construindo Blocos seria a de expandir a política de consenso da ledger, mediante as necessidades e o número de utilizadores que fosse surgindo.

Capítulo 8

Conclusões

A política de consenso parece-nos uma boa alternativa ao ”Proof-of-work” uma vez que retira custos das taxas de transações e não tem os custos energéticos da mesma sendo portanto mais amiga do ambiente, porém devido à sua própria natureza, uma entidade com recursos suficientes, pode conseguir ter nodos suficientes para efectuar transações sem que estas tenham qualquer validação por parte de outra entidade o que pode permitir que esta manipule as transações para seu benfício, porém uma possível solução passa por não permitir que tal suceda ou implementar uma política em que transações de uma entidade tenham sempre de ser validadas por um nodo de outra entidade.

A HyperledgerFabric é uma ferramenta óptima para quem quiser começar a trabalhar em Blockchain. Sendo uma ferramenta open-source da Linux, com todos os tutoriais e documentação que a acompanham, o facto de ser modular com o modelo de plug-and-play, tornam-na eficiente para todo e qualquer trabalho que se pretenda desenvolver em Blockchain.

De salientar que consideramos que cumprimos praticamente todos os pressupostos iniciais, tendo identificado uma plataforma open-source para desenvolvimento de uma moeda especializada em micropagamentos, a criação de um protótipo da moeda, assim como a disponibilização de uma wallet.

Tendo sido este o nosso primeiro trabalho realizado em Blockchain, estamos bastante satisfeitos com o que conseguimos desenvolver e implementar e por todo o conhecimento extra que adquirimos ao longo deste processo, sentindo-nos capazes de desenvolver projetos ainda que pequenos em blockchain ou de fazer parte de uma equipa que pretenda desenvolver projetos maiores.

Capítulo 9

Bibliografia

<https://bitcoin.org/bitcoin.pdf>

<http://hyperledger-fabric.readthedocs.io/en/release-1.1/>

<https://www.ibm.com/blockchain>

Blockchain for dummies, Manav Gupta, John Wiley & Sons, Inc.

<https://www.npmjs.com/package/keypair>

<https://www.bigchaindb.com>

<https://csrc.nist.gov/CSRC/media/Publications/nistir/8202/draft/documents/nistir8202-draft.pdf>

<https://www.blockchain-council.org/blockchain/list-of-best-open-source-blockchain-platforms>

<https://digiconomist.net/bitcoin-energy-consumption>

<https://ethereum.org>

<https://github.com/HydraChain/hydrachain>

<https://github.com/corda/corda>

<https://www.openchain.org>

<https://www.stellar.org>

<https://www.jpmorgan.com/global/Quorum>

<https://www.cryptokitties.co>

<https://cryptozombies.io>