Ana Luisa Lima Rodrigues

Implementar Diplomas Universitários na Blockchain do Hyperledger Fabric

Belo Horizonte

2019/2

Ana Luisa Lima Rodrigues

Implementar Diplomas Universitários na Blockchain do Hyperledger Fabric

Apresentado como requisito da disciplina de Projeto Orientado em Computação do DCC/UFMG

Orientador:
Jeroen van de Graaf - Departamento de Ciência da Computação

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Belo Horizonte

2019/2

Resumo

A tecnologia Blockchain vem sendo muito discutida nos últimos anos, principalmente no que diz respeito às aplicações e escalabilidade. Em paralelo, a Blockchain corporativa Hyperledger Fabric surgiu para atender necessidades do ambiente corporativo. Com uma arquitetura baseada em sistemas distribuídos, possui um funcionamento específico e exige do desenvolvedor pré-requisitos e pesquisa para que seja implementada. Neste projeto foi escolhido um caso de uso real, o armazenamento de diplomas universitários, para demonstrar como a arquitetura e rede do Hyperledger Fabric são ao mesmo tempo complexas e impressionantes.

Palavras-chaves: blockchain, hyperledger, hyperledger-fabric, smart contracts, diplomas universitários, peers, ledger.

Abstract

Blockchain technology has been much discussed in recent years, especially with regard to applications and scalability. In parallel, the corporate Blockchain called Hyperledger Fabric has emerged to meet the needs of the corporate environment. With a distributed systems-based architecture, it has a specific functioning and requires prerequisites and research from the developer to be implemented. In this project, a real use case, the storage of university degrees, was chosen to demonstrate how complex and impressive the architecture and network of Hyperledger Fabric are.

Keywords: blockchain, hyperledger, hyperledger-fabric, smart contracts, university diploma, university degree, peers, ledger.

Sumário

LISTA DE FIGURAS

LISTA DE ALGORITMOS

1 INTRODUÇÃO

Qualquer profissional e usuário de internet que se mantém informado na rede deve ter ouvido falar em blockchain. Recentemente, essa tecnologia tem sido o centro de muitas discussões sobre segurança de dados e o futuro da internet como conhecemos. Há quem diga até mesmo que a blockchain é a promessa para revolucionar a forma como transferimos propriedade digital, seja dinheiro ou bens.

O termo blockchain ganhou notoriedade pública com o *boom* recente das criptomoedas, em especial o Bitcoin. Enquanto tecnologia, porém, a aplicação do blockchain vai muito além das moedas virtuais e já está sendo considerada para fins diversos e por empresas de todo tipo.

O Hyperledger é um projeto que enxergou essa oportunidade e resolveu levar os benefícios da tecnologia ao ambiente corporativo, e para isso desenvolveu vários *frameworks* e ferramentas para o desenvolvimento. Seu foco era principalmente desvincular a ideia de que blockchains estão diretamente ligadas à criptomoedas, muito pelo contrário, elas podem estar presentes em várias áreas.

O foco principal desse Projeto Orientado à Computação é estudar e pesquisar sobre as especificidades da arquitetura da Blockchain do Hyperledger Fabric e aplicação prática em um caso de uso real. Implementar diplomas universitários foi escolhido como exemplo de um protótipo já que é uma realidade do meio acadêmico e pode gerar questionamentos a respeito do sistema e processos burocráticos das universidades. Além do mais, como os documentos digitais estão cada vez ganhando mais força, sua aderência pode reduzir drasticamente a emissão de papel e contribuir para o meio ambiente.

A implementação será dividida em quatro partes, dentre elas: modelagem das transações; arquitetura e rede; API e *queries*. Em modelagem de transações serão definidos os principais Participantes, Ativos e Transações que irão ser registrados na Blockchain, de acordo com o modelo pré-definido do Hyperledger Fabric. Ao desenvolver a arquitetura, serão descritas quais as principais entidades envolvidas para a criação e validação das transações baseadas em sua modelagem. A rede então passará a funcionar, possibilitando que todas as entidades trabalhem

em conjunto para que a Blockchain funcione corretamente. O Hyperledger Fabric disponibiliza uma API REST, proporcionado uma interação com os dados através de métodos HTTP, além de ser possível realizar *queries* no banco de dados.

Vale constar que esse trabalho não tem o objetivo de demonstrar um passo a passo da implementação e nem servir como um tutorial, mas sim de explicar como é o funcionamento do Hyperledger Fabric na prática e quais benefícios e dificuldades sua implementação traz. Como ainda é um projeto muito recente, sua comunidade de desenvolvedores ainda é muito restrita e os materiais disponíveis para quem quer entender melhor sobre são escassos e de difícil compreensão, o que dificulta para que mais pessoas estejam dispostas a desenvolver novas aplicações e beneficiar diferentes áreas. Portanto, esse projeto pode contribuir para a aderência dessa tecnologia, principalmente no ambiente universitário.

2 CONTEXTUALIZAÇÃO E TRABALHOS RELACIONADOS

2.1 Hyperledger Fabric

O Hyperledger [1] é um projeto open source que surgiu em 2015 para promover a utilização da tecnologia Blockchain em vários setores, em especial o tecnológico, financeiro e logístico. Promovida e hospedada por The Linux Foundation, possui uma grande variedade de *frameworks* (atualmente, seis), cada um com seu objetivo específico, conseguindo atender à múltiplas necessidades relacionadas à modelos de consenso, gerenciamento de certificados e validação de identidades.

Além de disponibilizar esses diferentes *frameworks*, o projeto também disponibiliza ferramentas (atualmente, sete) que são úteis para o desenvolvimento das aplicações. Elas auxiliam, por exemplo, no gerenciamento das transações realizadas na Blockchain e na criação de *smart contracts*.

Como é permissionada, a Blockchain do Hyperledger garante que só pessoas autorizadas tenham acesso à rede e previne nós maliciosos através do protocolo *Byzantine Fault Tolerance*. Não possui uma entidade centralizadora que controle a rede, garantindo sua confiabilidade e autenticidade dos dados. Por fim, adequa-se ao ambiente corporativo por preservar o acesso aos dados ao mesmo em tempo que garante o funcionamento adequado da rede de acordo com as regras de negócios.

Dos seis frameworks disponibilizados, o Hyperledger Fabric [2] foi uma das primeiras propostas e a que recebeu mais investimento, em especial da IBM. Além disso, possui a maior comunidade ativa contribuindo para seu crescimento, com mais de 200 desenvolvedores desde os primeiros *commits*. Ele foi escolhido para o desenvolvimento deste trabalho, por que além de ser o mais desenvolvido, é o que possui melhor adaptação ao ambiente corporativo, já que consegue abranger diferentes cenários e objetivos. Com o slogan ”Permissionado com suporte a canais”, está sendo considerado a principal plataforma para desenvolvimento de aplicações

privadas em Blockchain e promete lançar cada vez mais funcionalidades para melhorar esse processo.

2.2 Aplicabilidade do Hyperledger Fabric - Everledger

Fundada em 2015 por Leanne Kemp, uma empresária australiana com conhecimento em tecnologias emergentes, negócios, jóias e seguros, Everledger é uma empresa focada em enfrentar os desafios econômicos, ambientais e societários do mundo real por meio de suas soluções que criam ecossistemas de confiança.

Uma de suas soluções, o *Diamond Time-Lapse Protocol*, é uma iniciativa de rastreabilidade construída sobre o Hyperledger Fabric. O objetivo é concentrar em um só lugar toda a cadeia de fabricação e movimentação de um diamante. Nela, todos os participantes da indústria, incluindo fabricantes, varejistas e consumidores, podem conhecer a história de um diamante desde a origem até o consumidor final.

A partir de pesquisas com esses públicos, a empresa descobriu que era uma necessidade da indústria possuir informações a respeito da vida útil dos diamantes e jóias em circulação, com o intuito de demonstrar autenticidade, transparência e procedência. Além disso, havia uma grande preocupação com a circulação dos ”diamantes de sangue”, o que era facilitado por documentos físicos que podiam ser facilmente fraudados ou alterados.

Portanto, o DTLP foi criado para garantir a veracidade e procedência dos diamantes cadastrados, informações que estão protegidas pela tecnologia Blockchain do Hyperledger Fabric. O protocolo está disponível para todos que desejam consultar[1] os diamantes.

2.3 Diplomas universitários no cenário brasileiro

A emissão de diplomas é uma prática que faz parte do dia a dia das universidades em todo o mundo, principalmente no fechamento dos semestres letivos. Geralmente emitidos em papel, certificam a formação superior do aluno e podem servir como porta de entrada para oportunidades profissionais.

As universidades responsáveis por emitir diplomas dos alunos concluintes dos cursos ofertados precisam mantê-los em sua base de dados para casos de roubo, perda e também para verificar a veracidade. Atualmente, em especial na Universidade Federal de Minas Gerais (UFMG),

[1] Pesquisar cadeia do diamante a partir do número de série: `https://diamonds.everledger.io/search`

essa checagem envolve muitas burocracias, como preenchimento de formulários e entrega de documentos, o que dificulta o processo.

A falsificação de diploma universitário é modalidade criminosa prevista nos artigos 297 e 304 do Código Penal Brasileiro cujo bem jurídico tutelado é a fé pública. Esse crime está cada vez mais comum, principalmente no Brasil, onde o índice de desemprego é alto [3] e as oportunidades de trabalho podem aumentar ao possuir um diploma.

Em fevereiro de 2019, a Universidade Federal da Paraíba foi a primeira do Brasil a emitir diplomas digitais desenvolvidos com tecnologia de registro e autenticação distribuída, baseada em Blockchain [4]. Dessa forma, os estudantes paraibanos poderão ter acesso aos seus diplomas de forma mais prática. Entretanto, essa iniciativa foi feita apenas pela UFPB e não envolve outras universidades.

O Ministério da Educação (MEC), como ação de inovação tecnológica, publicou duas Portarias [5] para a modernização do fluxo processual para emissão e registro de diplomas digitais nas instituições de Ensino Superior pertencentes ao Sistema Federal de Ensino brasileiro. A primeira Portaria [6], de número 330, foi publicada em abril de 2018 e foi a primeira medida na construção de uma estrutura legal para diplomas digitais. No entanto, essa implementação estava condicionada à publicação de ato específico do Ministro da Educação. Em março de 2019 foi publicada a segunda Portaria [7], de número 554, que além de estabelecer as especificidades técnicas para emissão e registro do diploma digital, também constituiu um marco para a contagem do prazo de 24 meses para que as instituições de Ensino Superior se adequem às novas regras.

O armazenamento dos diplomas digitais em uma Blockchain poderia tornar os acessos mais rápidos e práticos, principalmente entre universidades de diferentes estados que precisam realizar checagens para cursos de graduação e pós-graduação. Uma vez incluídos no Hyperledger Fabric, os dados dos diplomas não poderão mais sofrer modificações e apenas quem possuir credencial de acesso será capaz de acessá-los, provendo maior segurança e resistência à ataques maliciosos. Além disso, a realidade digital está transformando a forma de gerir documentos com o passar dos anos, tornando mais conveniente e prática sua verificação. Os principais impactos positivos dessa mudança estão nos aspectos sustentável e financeiro para as instituições, já que a quantidade de emissão de papel diminuiu drasticamente.

3 DESENVOLVIMENTO DO TRABALHO

3.1 Modelagem das transações

Blockchain registra transações enquanto os bancos de dados registram dados. Mas de forma similar ao segundo, a Blockchain precisa modelar essas transações, ou seja, definir quem são os responsáveis por elas e o que acontece quando uma transação ocorre. Por exemplo, se algo foi criado ou apenas modificado.

Como principal forma de organização do modelo de negócios, o Hyperledger Fabric possui seu próprio modelo de transações com base em ativos digitais. Um ativo digital é um bem ou direito que alguém possui e que pode ser representado digitalmente. Ativos digitais podem ser divididos em dois grupos: os tangíveis, como imóveis e hardware, e os intangíveis, como contratos e propriedade intelectual. Esses ativos, assim como no mundo físico, podem ser transacionados entre pessoas no mundo digital.

Os responsáveis por essas **transações** de **ativos** são chamados **participantes** e todas essas mudanças, sejam de criação de um novo ativo ou mudança de titularidade do mesmo, são gravadas no Ledger e não podem ser alteradas. De acordo com o Projeto em Computação I desenvolvido anteriormente, ”Um Ledger Distribuído é um ledger digital diferente do já conhecido pelos sistemas financeiros. Suas informações são guardadas em uma rede de computadores e qualquer mudança no ledger reflete simultaneamente em todas essas máquinas.”

No âmbito desse projeto, os ativos digitais serão os diplomas emitidos pelas instituições universitárias. Como essa emissão atualmente é de responsabilidade do Colegiado e o Estudante é quem possui o diploma, ambos serão incluídos como participantes na modelagem. A cerimônia que concede o grau acadêmico para o estudante juntamente com seu diploma é a Colação de Grau, portanto ela fará parte das transações, já que um novo diploma será criado para o estudante assim que colar grau.

A figura a seguir foi criada com base no modelo desenvolvido, com o objetivo de tornar

mais visual as entidades do sistema, seus respectivos atributos e como elas se relacionam.

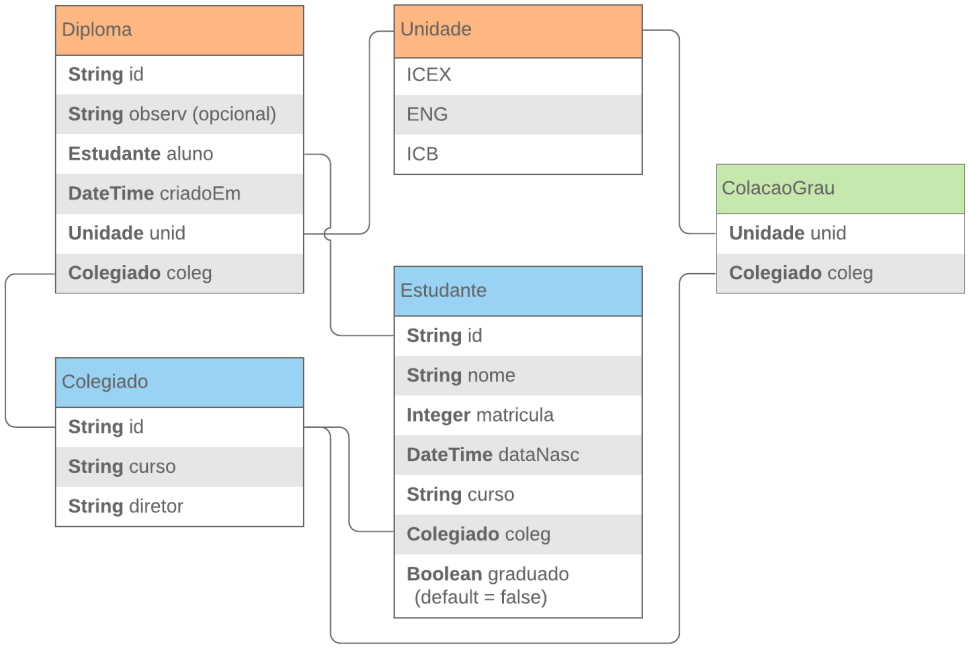

Figura 3.1: Diagrama da modelagem dos dados que compõem o sistema.

Representando os ativos do modelo estão o Diploma e a Unidade. O Diploma é um ativo que deve possuir um *id* para identificá-lo e seus atributos podem ser opcionais ou obrigatórios. Já a Unidade é um tipo especial, chamado de *enum*, que como o próprio nome diz é uma enumeração de itens que podem ser utilizados dentro das entidades principais, assim como foi feito em Diploma. Diferente do Diploma, Unidade não precisa ter um *id* que a identifique.

Os participantes da rede são compostos pelo Colegiado e pelo Estudante, que devem possuir identificadores. A entidade Estudante possui atributos que são importantes para o funcionamento correto do sistema, como por exemplo, a variável do tipo *boolean*. Ela representa se o estudante já está graduado ou não, para que dessa forma, seja possível checar se já concluiu seu curso. Além disso, também é possível atribuir à uma variável *boolean* um valor *default*, que neste caso será falso, já que o estudante é incluído no sistema quando realiza sua matrícula.

A Colação de Grau será o evento que possibilitará a mudança de estado do sistema, ou seja, que novos diplomas sejam criados e associados a seus respectivos estudantes. No Hyperledger, é possível definir um arquivo Javascript para executar um trecho de código quando essa transação for criada. O trecho pode ser visualizado abaixo:

```
1 /**
2  * Sample transaction processor function.
3  * @param {poc2.transactions.Graduation} tx Students Graduation
4  * @returns {poc2.assets.Diploma} All the diplomas.
```

```
5   * @transaction
6  */
7  async function graduation(tx) {
8
9      const allStudents = await getParticipantRegistry('poc2.participants.
           Student');
10     const localStudents = await allStudents.getAll();
11     const allDiplomas = await getAssetRegistry('poc2.assets.Diploma');
12     const localDiplomas = await allDiplomas.getAll();
13     var factory = getFactory();
14
15     const unit = tx.unit; //unidade da graduacao
16     const coleg = tx.coleg; //colegiado da graduacao
17
18     for (const student of localStudents) {
19         let randomId = getRandomInt().toString();
20         let createDiploma = true;
21       if (student.graduated == true){
22           for (const diploma of localDiplomas) {
23               if (student.id == diploma.owner.$identifier) {
24               //eh graduado e possui diploma
25                 createDiploma = false;
26                 break;
27               }
28           }
29             if (createDiploma == true && student.coleg.$identifier ==
                   coleg.id){
30                 var timestamp = new Date(new Date().getTime());
31                 var newDiploma = factory.newResource('poc2.assets', '
                       Diploma', randomId);
32                 newDiploma.owner = student;
33                 newDiploma.unit = unit;
34                 newDiploma.coleg = coleg;
35                 newDiploma.createdAt = timestamp;
36                 await allDiplomas.add(newDiploma);
37           }
38       }
39   }
40     return allDiplomas;
41 }
42
43 function getRandomInt() {
44     let min = Math.ceil(1);
```

```
45      let max = Math.floor(1000000);
46      return Math.floor(Math.random() * (max - min)) + min;
47  }
```

Algoritmo 3.1: Código em JavaScript da transação.

De acordo com a modelagem descrita anteriormente, há três classificações para um estudante:

- Finalizou a graduação e possui um diploma;
- Finalizou a graduação e está aguardando a cerimônia de Colação de Grau para recebimento do diploma;
- Ainda não finalizou a graduação.

Esse trecho de código tem o objetivo de percorrer a lista de estudantes cadastrados no sistema e identificar a segunda classificação da lista, ou seja, os estudantes que finalizaram a graduação e estão aguardando o recebimento do diploma. Neste caso, ao realizar a cerimônia de Colação de Grau, novos diplomas serão gerados para esses estudantes específicos.

A função getRandomInt() foi criada com o objetivo de gerar um *id* aleatório para identificar o novo diploma que será criado.

Para que fosse possível testar o modelo de transações sem que fosse necessária a implementação da Arquitetura e da Rede, que serão discutidas nas próximas seções, o Hyperledger Fabric disponibiliza um Playground, uma interface web que possibilita a criação de participantes e ativos e também a execução de transações. Com ele, é possível fazer a validação do modelo para que o sistema funcione corretamente.

Com o modelo criado e testado, a Arquitetura pôde ser implementada para que o modelo pudesse funcionar corretamente na Blockchain do Hyperledger.

3.2 Arquitetura e Rede

A arquitetura do Hyperledger Fabric, desenvolvida pelos projetistas da IBM, é ao mesmo tempo complexa e fascinante. Baseada em sistemas distribuídos, foi implementada (e está em constante evolução) de forma descentralizada e robusta, proporcionando a criação de diversas aplicações, das mais variadas indústrias, já que é totalmente adaptável.

É possível definir um modelo de trabalho e as regras de negócio em um sistema que pode ser escalável, ou seja, se no futuro for necessário desenvolver melhor a arquitetura, só é necessário adicionar mais elementos, e não refazer todo o sistema. Isso aumenta o grau de reuso e garante uma manutenção mais simples e barata, objetivos almejados pela maioria dos sistemas computacionais.

No Projeto Orientado em Computação I, desenvolvido anteriormente, as entidades que fazem parte da arquitetura foram explicadas detalhadamente juntamente com a relação entre elas. Como é uma arquitetura adaptável, ela foi desenvolvida com base no modelo de transações definido na seção anterior. Além disso, também foi descrito como essas entidades se relacionam e trabalham em conjunto para formar a rede P2P do Hyperledger Fabric. Segundo Peterson (2004, p. 503) "Frequentemente, atributos como descentralizado e auto-organizável são mencionados quando se discute sobre redes *peer-to-peer*, significando que os nós individuais se organizam em uma rede sem qualquer coordenação centralizada." A rede do Hyperledger Fabric é um exemplo de rede *peer-to-peer*, garantindo que as entidades que a compõem estarão trabalhando juntas, sincronizadas e sem um controle central.

Para exemplificar melhor o caso de uso e trazê-lo para a realidade das universidades, foram selecionadas três grandes universidades brasileiras: Universidade Federal de Minas Gerais (UFMG), Universidade Federal do Rio de Janeiro (UFRJ) e Universidade de São Paulo (USP). Como é muito comum que alunos prestem cursos em diferentes universidades, seria interessante criar uma forma de que possam checar esse diploma sem muitas burocracias. Com os diplomas armazenados na Blockchain, além de cada uma poder gerenciar os diplomas emitidos aos alunos, também poderão ter acesso aos diplomas umas das outras para checar veracidade e autenticidade. Dessa forma, atuarão como organizações independentes, mas poderão acessar informações relevantes dos alunos.

A figura 3.2 foi criada com base na arquitetura desenvolvida. Nela estão representadas as principais entidades que participam da rede e serão descritas suas funções e importância no sistema como um todo.

Cada universidade do modelo representa uma organização responsável por executar suas próprias transações, de acordo com o modelo de transações já definido. As transações do Hyperledger irão representar a criação de estudantes, diplomas, realização da Colação de Grau, etc. Todas as transações executadas são autorizadas a partir de um certificado. Desta forma, a rede consegue controlar quem tem autorização para executar determinadas ações. Neste caso, cada universidade irá possuir um certificado, possibilitando assim sua atuação na rede.

Cada organização irá possuir uma Autoridade Certificadora, identificada na figura por CA, e

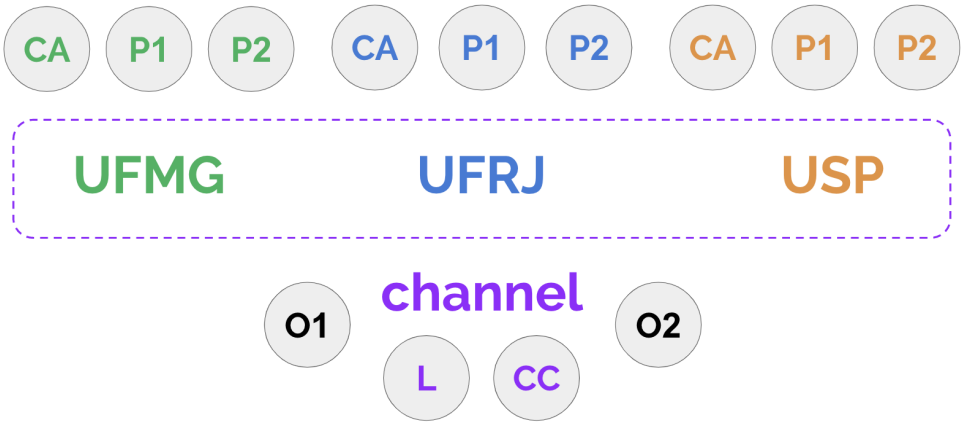

Figura 3.2: Rede P2P e as entidades do caso de uso.

dois Peers. A Autoridade Certificadora é responsável por gerenciar a distribuição e verificação de chaves públicas e privadas, que serão utilizadas para autenticação e ação na Blockchain. Seria possível implementar apenas uma CA para as três organizações, porém é recomendado que cada uma possua a sua para que o gerenciamento de chaves seja mais eficiente e também para possibilitar que cada uma implemente suas próprias regras de atuação na rede.

Os Peers, identificados na figura por P1 e P2, são responsáveis por receber todas as solicitações das organizações, ou seja, de execução de transações. Foram implementados 2 para cada organização já que essa quantidade consegue lidar com um número de transações bastante elevado e se um deixar de funcionar, o outro consegue suprir a demanda. Todos os seis Peers da rede trabalham em conjunto e estão sempre sincronizados, ou seja, possuem as mesmas informações no mesmo instante de tempo.

Além disso, essas organizações podem ser incluídas em canais. Cada canal é uma sub-rede privada de comunicação que é capaz de conduzir transações privadas e confidenciais entre essas organizações, possuindo seu próprio Ledger (identificado na imagem pela letra L) e Chaincode (identificado pelas letras CC). Um Ledger é um banco de dados digital que registra a transação de ativos, onde cada transação e seus detalhes são armazenados em uma base. Chaincode é o nome dado ao *smart contract* do Hyperledger e é um programa responsável por estabalecer as regras de negócios, que resumem-se em: termos comuns, dados, regras, processos, entre outros. A partir desses programas, é possível definir o quê as organizações da rede poderão fazer, ao quê terão acesso e como poderão se relacionar dentro dela. Utilizando o mesmo Ledger e Chaincode, torna-se possível que as três universidades tenham o mesmo modelo de negócios e os mesmos dados.

Uma vez que os Peers recebem as transações, elas precisam ser autenticadas e aprovadas no que diz respeito à autenticidade e veracidade. Os responsáveis por isso são os Orderers,

indicados na figura por O1 e O2. Assim como os Peers, eles também se mantém sincronizados. Em um ambiente de desenvolvimento, pode-se implementar apenas um Orderer, já que não ocorrerão problemas com segurança. Por outro lado, em um ambiente de produção pronto para ser utilizado pelos usuários, é necessário implementar no mínimo dois Orderers para evitar pontos únicos de falha. Se um Orderer parar de funcionar, por problemas do sistema ou por ataques maliciosos, o outro Orderer irá se manter e aguentará o volume de transações. Com as transações aprovadas pelo Orderer, elas serão então incluídas no Ledger em uma cadeia de blocos, formando a estrutura da Blockchain. Caso sejam refutadas, por exemplo se o certificado for inválido ou a operação requisitada não for permitida, o Orderer retorna para os Peers a resposta com o código de erro e os Peers retornam para o cliente.

Ao implementar mais de um Orderer em um sistema, para mantê-los sincronizados e trabalhando em conjunto é necessário implementar também o Kafka e Zookeeper. O Apache Kafka é uma plataforma de streaming distribuído que possui funcionalidades de *message broker* e capacidade de armazenar e processar os dados em um fluxo. Já o Apache Zookeeper é um serviço descentralizado para manter informações de configurações e nomenclaturas entre serviços distribuídos. O Kafka utiliza o Zookeeper para sincronizar as configurações entre os diferentes Orderers. De acordo com a documentação oficial do Hyperledger Fabric, o número mínimo de Kafkas em um sistema deverão ser quatro, para que sejam tolerantes à falhas. Já os Zookeepers, devem ser três, cinco ou sete, para evitar *Split Brain*.

Todas as entidades que foram citadas e descritas anteriormente são configuradas no Docker e cada uma possui um *container* do mesmo. As entidades que possuem quantidades maiores que 1, como Peers e Kafka, o número de containers corresponde à quantidade, ou seja, se existirem seis Peers, seis *containers* de Peers deverão ser configurados e inicializados. Esse exemplo pode ser visualizado melhor na figura 3.3.

```
ana@ana:~/Desktop/poc2-diploma/poc2-diploma/fabric-scripts$ docker ps -a
CONTAINER ID        IMAGE                               COMMAND                  CREATED             STATUS              PORTS                                                                                                 NAMES
47215a6cac12        hyperledger/fabric-tools:1.2.1      "/bin/bash"              2 minutes ago       Up 2 minutes                                                                                                              cli
5dbe03b9c622        hyperledger/fabric-peer:1.2.1       "peer node start"        2 minutes ago       Up 2 minutes        0.0.0.0:7051->7051/tcp, 0.0.0.0:7053->7053/tcp                                                        peer0.org1.poc2
6b3d1c9e7221        hyperledger/fabric-peer:1.2.1       "peer node start"        2 minutes ago       Up 2 minutes        0.0.0.0:9051->7051/tcp, 0.0.0.0:9053->7053/tcp                                                        peer0.org2.poc2
c3b06ae2af96        hyperledger/fabric-peer:1.2.1       "peer node start"        2 minutes ago       Up 2 minutes        0.0.0.0:12051->7051/tcp, 0.0.0.0:12053->7053/tcp                                                      peer1.org3.poc2
4ac9dac3e4da        hyperledger/fabric-peer:1.2.1       "peer node start"        2 minutes ago       Up 2 minutes        0.0.0.0:8051->7051/tcp, 0.0.0.0:8053->7053/tcp                                                        peer1.org1.poc2
5e85e5600692        hyperledger/fabric-peer:1.2.1       "peer node start"        2 minutes ago       Up 2 minutes        0.0.0.0:10051->7051/tcp, 0.0.0.0:10053->7053/tcp                                                      peer1.org2.poc2
2090850c6672        hyperledger/fabric-peer:1.2.1       "peer node start"        2 minutes ago       Up 2 minutes        0.0.0.0:11051->7051/tcp, 0.0.0.0:11053->7053/tcp                                                      peer0.org3.poc2
2d026930565b        hyperledger/fabric-orderer:1.2.1    "orderer"                2 minutes ago       Up 2 minutes        0.0.0.0:7050->7050/tcp                                                                                orderer0.poc2
99e429fb92dc        hyperledger/fabric-orderer:1.2.1    "orderer"                2 minutes ago       Up 2 minutes        0.0.0.0:8050->7050/tcp                                                                                orderer1.poc2
21e048a722d2        hyperledger/fabric-kafka            "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        9093/tcp, 0.0.0.0:32787->9092/tcp                                                                     kafka2
edc1d882ac37        hyperledger/fabric-kafka            "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        9093/tcp, 0.0.0.0:32786->9092/tcp                                                                     kafka1
b0c35196a88e        hyperledger/fabric-kafka            "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        9093/tcp, 0.0.0.0:32785->9092/tcp                                                                     kafka0
20b771c77ef0        hyperledger/fabric-kafka            "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        9093/tcp, 0.0.0.0:32784->9092/tcp                                                                     kafka3
23db2b7eef88        hyperledger/fabric-ca:1.2.1         "sh -c 'fabric-ca-..."   2 minutes ago       Up 2 minutes        0.0.0.0:7054->7054/tcp                                                                                ca.org1.poc2
f2a7237613c0        hyperledger/fabric-ca:1.2.1         "sh -c 'fabric-ca-..."   2 minutes ago       Up 2 minutes        0.0.0.0:8054->7054/tcp                                                                                ca.org2.poc2
2f60cf5c09b8        hyperledger/fabric-zookeeper        "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        0.0.0.0:3181->2181/tcp, 0.0.0.0:32783->2888/tcp, 0.0.0.0:32782->3888/tcp                              zookeeper1
2895a07b6d80        hyperledger/fabric-ca:1.2.1         "sh -c 'fabric-ca-..."   2 minutes ago       Up 2 minutes        0.0.0.0:9054->7054/tcp                                                                                ca.org3.poc2
ac8d239d4f45        hyperledger/fabric-zookeeper        "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        0.0.0.0:2181->2181/tcp, 0.0.0.0:32781->2888/tcp, 0.0.0.0:32780->3888/tcp                              zookeeper0
bbeed4dc28c9        hyperledger/fabric-couchdb:0.4.10   "tini -- /docker-e..."   2 minutes ago       Up 2 minutes        4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp                                                            couchdb
90137a6f2c32        hyperledger/fabric-zookeeper        "/docker-entrypoin..."   2 minutes ago       Up 2 minutes        0.0.0.0:4181->2181/tcp, 0.0.0.0:32779->2888/tcp, 0.0.0.0:32778->3888/tcp                              zookeeper2
f82da1289c37        hyperledger/fabric-couchdb:0.4.10   "tini -- /docker-e..."   2 minutes ago       Up 2 minutes        4369/tcp, 9100/tcp, 0.0.0.0:6984->5984/tcp                                                            couchdb1
```

Figura 3.3: Containers do Docker inicializados.

O motivo principal para cada entidade estar em um *container* diferente é a fim de evitar um ponto único de falha. Se o sistema sofrer um ataque malicioso, ao invés de atacar apenas um ponto, para causar danos ao sistema como um todo o responsável deverá atacar todos os

containers, dificultando sua atuação.

Pela observação dos aspectos analisados, entende-se que a arquitetura e rede funcionam de uma forma coordenada e descentralizada, dois dos principais motivos para o surgimento das Blockchains, que queriam eliminar a necessidade de uma autoridade centralizadora que controlasse toda a rede. Além do mais, foram construídas com base em sistemas distribuídos e utilizando tecnologias que aumentam a segurança do sistema.

3.3 API

Application Programming Interface, mais conhecida como API, é um conjunto de definições e protocolos usado em integração de sistemas. Possibilita benefícios como segurança dos dados e facilidade no intercâmbio de informações entre diferentes linguagens de programação, gerando economia de tempo de desenvolvimento e consequentemente, de dinheiro. Com ela, soluções e serviços podem comunicar-se entre si sem precisarem saber como foram implementados, focando apenas na troca das informações necessárias.

Uma API do tipo REST, da sigla *Representational State Transfer*, é uma abstração da arquitetura da *World Wide Web*. Ela usa o protocolo HTTP de forma explícita e representativa para se comunicar, ignorando os detalhes da implementação e a sintaxe de protocolo e focando nos papéis dos componentes e nas suas restrições.

O Hyperledger Fabric inclui em sua implementação um processo Node.js independente que expõe uma rede comercial como uma API REST. Dessa forma, é possível consumir os dados incluídos na Blockchain através de métodos HTTP, como GET, POST e HEAD. Essa interação facilita tanto a visualização de dados para novos aderentes da ferramenta, já que disponibiliza uma interface Web intuitiva, quanto a integração com outros sistemas que irão consumir os dados.

No ponto de vista do projeto em questão, cada organização do modelo (UFMG, UFRJ e USP) terá sua própria API REST disponível em determinada URL. Dessa forma, é possível filtrar as requisições por organização e também organizar melhor os dados. Vale constar ainda que por mais que possuam sua própria URL para requisições, os dados serão incluídos no mesmo Ledger, possibilitando que as organizações tenham acesso a todos os participantes, ativos e transações da base.

Na figura 3.4 pode-se visualizar a interface Web da API REST para a porta 3000, equivalente à uma organização. As entidades do projeto (Colegiado, Diploma, Graduation e Student)

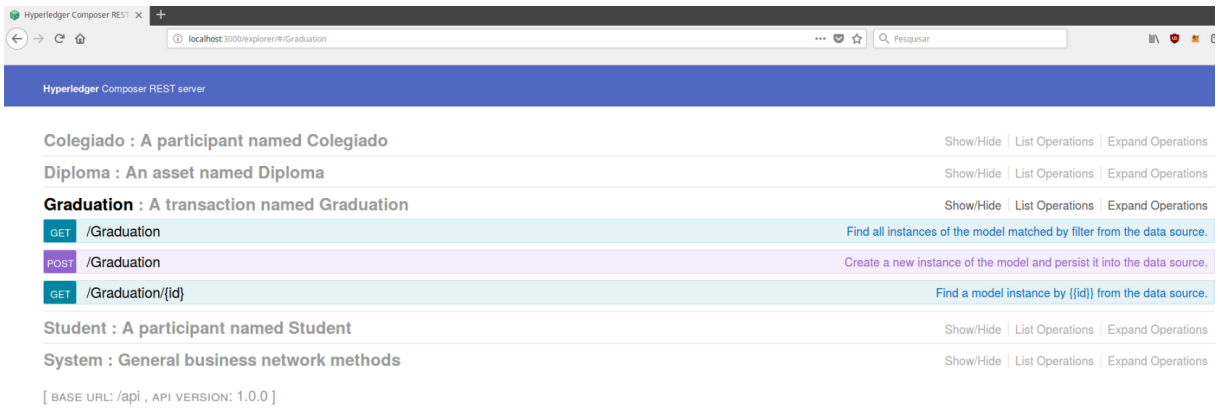

Figura 3.4: API REST rodando localmente.

podem ser expandidas para detalhamento dos métodos HTTP disponíveis, sendo que os mesmos podem ser executados no mesmo local. A aba *System* refere-se à configurações do sistema, como seu status e acesso ao histórico de transações.

Além disso, o servidor REST do Hyperledger possibilita algumas configurações adicionais para aumentar a segurança das transferências de dados entre diferentes aplicações. É possível ativar a autenticação de clientes, ou seja, os clientes que desejarem consumir os dados da API deverão se autenticar através de credenciais, caso contrário não conseguirão interagir com os dados. Além disso, pode-se aumentar a segurança utilizando HTTPS e TLS, muito recomendados para ambientes em produção. Dessa forma, todos os dados transmitidos estarão encriptados. Por último, ainda é possível customizar o servidor REST com mecanismos de autenticação adicionais que não são incluídos na versão disponibilizada pelo Hyperledger mas que são necessários para a aplicação.

3.4 Queries

O conceito de *Query* é muito conhecido quando se fala de banco de dados. Isso por quê ao incluir dados em uma base, é necessário consultá-los para diversos fins. Essa requisição da informação é a própria *Query*, uma linguagem de consulta que possibilita adicionar, remover e modificar dados em um banco.

Como o Hyperledger Fabric armazena transações em um Ledger, assim como os bancos de dados tradicionais é interessante interagir com esses dados. Para isso, ele possibilita a criação de *Queries* para consultas simples.

```
1 query getDiplomasICEX {
2   description: "Get all diplomas from ICEX"
3   statement:
```

```
4       SELECT poc2.assets.Diploma
5         WHERE (unit == "ICEX")
6   }
7   query studentRegistry {
8     description: "Get student by registry number"
9     statement:
10      SELECT poc2.participants.Student
11        WHERE (registry == _$reg)
12  }
13  query graduatedStudents {
14    description: "Get all graduated Students"
15    statement:
16      SELECT poc2.participants.Student
17        WHERE (graduated == true)
18  }
19  query graduatedStudentsCC {
20    description: "Get all graduated Students from Computer Science"
21    statement:
22      SELECT poc2.participants.Student
23        WHERE (graduated == true AND course == "Ciencia da Computacao")
24  }
```

Algoritmo 3.2: Estrutura das Queries.

O comando SELECT permite recuperar dados de um objeto do banco de dados, enquanto a cláusula WHERE permite ao comando passar condições de filtragem. Atualmente, o Hyperledger só disponibiliza esses dois comandos para consultas, o que restringe a obtenção de dados mais complexos.

Como exemplo, foram desenvolvidas quatro tipos de *queries*, interagindo principalmente com estudantes e diplomas da base. Como a interface web discutida na seção anterior dispõe do método GET, não foram desenvolvidas queries para retornar todos os estudantes cadastrados, por exemplo. A primeira, **getDiplomasICEX**, tem o objetivo de retornar todos os diplomas que foram emitidos pelo ICEX. A segunda, **studentRegistry**, recebe como parâmetro o número de matrícula do estudante e retorna seus dados com base nesse número. A terceira, **graduatedStudents**, filtra todos os estudantes graduados da base, ou seja, todos aqueles que já concluíram seu curso de graduação, independente de qual for. Por último, **graduatedStudentsCC** retorna todos os estudantes graduados no curso de Ciência da Computação.

4 CONCLUSÕES

Este trabalho teve como foco uma pesquisa aprofundada sobre o Hyperledger Fabric e como funcionaria uma aplicação prática de um caso de uso real. A forma que foi projetado exige um conhecimento maior do desenvolvedor sobre aspectos específicos dos sistemas distribuídos e do funcionamento de uma Blockchain, para que possa realizar um desenvolvimento mais ágil, coeso e seguro. Por serem dois assuntos extensos, e a Blockchain ainda ser um assunto recente, isso pode desanimar muitos desenvolvedores a terem um interesse em desenvolver aplicações com o uso do Fabric, o que pode justificar sua pouca aderência, principalmente no Brasil, onde a comunidade é quase inexistente.

Implementar diplomas universitários foi escolhido como tema já que é uma realidade do meio acadêmico e pode gerar questionamentos a respeito do sistema e processos burocráticos das universidades. Além do mais, como os documentos digitais estão cada vez ganhando mais força, sua aderência pode reduzir drasticamente a emissão de papel e contribuir para o meio ambiente.

Como um projeto de cunho tecnológico, acrescentou conhecimento considerável sobre as tecnologias emergentes do mercado, incluindo a ainda recente Blockchain. O Hyperledger Fabric foi construído levando em conta importantes estruturas dos sistemas distribuídos, reconhecidos na área de Ciência da Computação, reforçando ainda mais a ideia de que a computação do futuro é aquela não governada por entidades e organizações. Deste modo, contribuiu para um melhor aprendizado e consciência de que a tecnologia pode melhorar a forma como a sociedade realiza a troca de informações.

Os principais desafios encontrados no desenvolvimento deste trabalho foram a necessidade de alto processamento e a dificuldade de *debug*. Como a Blockchain realiza muitas operações matemáticas para formar as *Hashes* que unem os blocos, formando assim a cadeia de blocos que a caracteriza, exige um alto poder de processamento que não é suportado por máquinas usuais. No início do desenvolvimento, o projeto foi rodado em uma máquina com um processador Intel Core i5 com 2,3 GHz e 8GB de memória RAM, não sendo possível finalizar sua

execução. Finalmente, foi executada em outra máquina de processador Intel Core i7 com 1,80 GHz e 8GB de memória RAM e foi possível finalizar sua execução. Portanto, implementar o projeto em sistemas grandes com muitas requisições pode encarecê-lo e torná-lo inviável para as organizações.

Além disso, o processo de *debug* também foi um desafio. Os erros apresentados pelo Hyperledger não são bem descritos, não sendo possível identificar seus motivos e um tempo considerável foi perdido na busca e exploração dos erros em fóruns da internet. Como é um projeto que está ganhando cada vez mais espaço e sendo muito discutido nos fóruns globais, nos próximos anos devem ocorrer maiores investimentos para facilitar seu desenvolvimento e trazer mais adeptos à tecnologia.

Como futura evolução do trabalho, é possível uma implementação de um sistema completo, no qual as universidades fossem capazes de integrar a Blockchain ao seu sistema de gerenciamento e que toda a administração de diplomas fosse feita única e exclusivamente pelo Hyperledger Fabric. Dessa forma, seria possível testar a viabilidade do sistema como um todo e medir quais foram os efeitos na emissão e verificação de autenticidade dos diplomas.

O código-fonte deste trabalho pode ser encontrado no repositório do GitHub [8].

Referências Bibliográficas

[1] Hyperledger. Open Source Blockchain Technologies. https://www.hyperledger.org.

[2] GitHub. Hyperledger Fabric. https://github.com/hyperledger/fabric.

[3] Desemprego sobe para 12,7% com 13,4 milhões de pessoas em busca de trabalho. https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/24283-desemprego-sobe-para-12-7-com-13-4-milhoes-de-pessoas-em-busca-de-trabalho. Acesso em: 5 set. 2019.

[4] Primeiros diplomas digitais da UFPB serão entregues a formandos do Centro de Informática. http://www.ufpb.br/antigo/content/primeiros-diplomas-digitais-da-ufpb-ser\%C3\%A3o-entregues-formandos-do-centro-de-inform\%C3\%A1tica. Acesso em: 5 set. 2019.

[5] Mec publica portaria que regula emissão de diploma digital. http://portal.mec.gov.br/ultimas-noticias/212-educacao-superior-1690610854/74041-mec-publica-portaria-que-regula-emissao-de-diploma-digital. Acesso em: 5 set. 2019.

[6] Portaria nº 330, de 5 de abril de 2018. http://www.in.gov.br/materia/-/asset_publisher/Kujrw0TZC2Mb/content/id/9365055/do1-2018-04-06-portaria-n-330-de-5-de-abril-de-2018-9365051. Acesso em: 5 dez. 2019.

[7] Portaria nº 554, de 11 de março de 2019. http://www.in.gov.br/materia/-/asset_publisher/Kujrw0TZC2Mb/content/id/66544171/do1-2019-03-12-portaria-n-554-de-11-de-marco-de-2019-66543842. Acesso em: 5 dez. 2019.

[8] Ana Luisa Rodrigues. Repositório GitHub do Projeto Orientado em Computação II. https://github.com/anallr/HyperledgerFabric_Diploma.

[9] L. PETERSON. *Redes de Computadores*. Elsevier Editora Ltda, 2004.

[10] G. COULOURIS. *Distributed Systems Concepts and Design*. Addison-Wesley, 2005.

[11] A. TANENBAUM. *Sistemas distribuídos*. Pearson Education, 2007.

[12] ReadTheDocs. A Blockchain Platform for the Enterprise. https://hyperledger-fabric.readthedocs.io/en/release-1.4/.