Computing Science and Mathematics
University of Stirling

Smart contracts on Hyperledger Fabric

Yordan Gospodinov

Supervised by Dr. Andrea Bracciali

Submitted in partial fulfilment of the requirements for the degree of
BSc with Honours in Computer Science

April 2019

”If you are not making someone else’s life better, then you are wasting your time”
-Will Smith

Abstract

This project was undertaken to research blockchain technology and its implications in enterprise fields. The new technology is capable of securing information, while also certify for its integrity inside the network. Hyperledger Fabric is specifically made to answer the requirements enforced by the business.

An opportunity arised for certain research due to the security and integrity provided by blockchain. The idea for what exactly is persons identity and can it be made more explicit and secure. What's more, can an identity be assigned quickly and used worldwide.

The scope of this project was to research the blockchain framework Hyperledger Fabric and the smart contract, called chaincode in this permissioned network. With the knowledge gathered an attempt of self-sovereign identity prototype is made.

The field of blockchain is still in its infancy and is developing with a rapid growth. At the start of this project, Fabric was at version 1.2 and the project kept its development consistent with it. However at the time of handing in of the dissertation, Fabric is already at version 1.4.

Smart contracts are representing the business logic in a blockchain network. In Hyperledger Fabric they are called chaincode. It runs on top of the blockchain to implement the desired interaction with the ledger. It can be written in two ways, at the time of writing this dissertation. The first is using only Fabric and writing low-level code (golang recommended). The second, which is used in this thesis, is through Hyperledger Composer. Composer is an open development toolset and framework to make developing blockchain applications easier. It creates assets, participants and transactions through high-level code, that is compressed into a single file and installed on top of Fabric blockchain network.

Attestation

I understand the nature of plagiarism, and am aware of the University's policy on this. I certify that this dissertation reports original work by me during my University project except for the following (adjust according to the circumstances):

- The technology review in section **??** was largely adapted *www.software-review.org/article9815.html*.
- The code discussed in section **??** was created by Acme Corporation (*www.acme-corp.com/JavaExpert*) and was used in accordance with the licence supplied.
- The code discussed in section **??** was written by my supervisor.
- The code discussed in section **??** was developed by me during a vacation placement with the collaborating company. In addition, this used ideas I had already developed in my own time.

Signature: **Date:**

Acknowledgements

First and foremost I would like to thank to Dr. Andrea Bracciali for introducing me to blockchain. What's more I am grateful for his patience, guidance and support, without which I would not have been able to grow into what I am today. Thank you.

I am grateful to my parents for giving me the opportunity to study in University of Stirling.

I am grateful to my classmates for helping me out, when I could not understand something, and just for being around, it was fun.

I would like to thank the library for having the nice quiet, fourth floor, and for always buying the books I requested.

Contents

List of Figures

Chapter 1

Introduction

Recently a new technology emerged into the world that is called blockchain. This is a distributed decentralized database of blocks constructed of transactions. The reason why it is very interesting is that it has the potential to change many aspects of life. It is an opportunity for better authenticity and security of data.

Hyperledger Fabric is a blockchain framework designed specifically to be easily adoptable by different business use cases.

The main two objectives of this project are: to research how the smart contracts in Hyperledger Fabric work; to make a prototype of self-sovereign system with the gathered knowledge.

1.1 Background and Context

Information has always been one of the most valuable assets a person could have. Through times information was traded in many different ways, from barter to monetization. Recently the information about an individual has become a great selling point, because it can be used in variety of fields, from science to business. However, the collection of this data is becoming a problem.

As individuals, our identities are, to some extent, not ours anymore. If we cannot certify who we are, we became no one in the eyes of business and government. Needless to say that had we lost all of the documents that certify our place in the company, city, country, Earth, we would be in a big trouble. [2]

Another approach to critical and private information is how it is being used live. Whenever we want to identify ourselves somewhere, the usual document for identification would be either an ID or a passport. Here is the problem concerning all information on this document. It turns out that whenever a person wants to prove his or her existence, the party that requires this identification, can take and keep a record of all sensitive data on that document. In some countries this may be illegal. This data then could be used for a wicked purpose. [1]

Furthermore whenever a person is signing in to receive some kind of certificate, whether that would be a school or an academy, they are leaving sensitive data with this company. In most countries, whenever a person starts living in a city, he or she has to identify himself/herself to the council. In the end, there are a lot of institutions that keep sensitive data for an individual. This is a problem, because

institutions and businesses have different levels of security. An attacker only needs to pick the easiest target and a great deal of sensitive data will be exposed.

I believe all of these problems are just subproblems of bigger challenges - what is an identity today and how to be able to give private access to our data. The solution could provide us awareness for a better control of our own data, as well as to be able to share only what is exactly needed to provide to those companies and institutions.

1.2 Scope and Objectives

The scope of this thesis will involve Blockchain technology and what is digital identity. This project focuses on Hyperledger Fabric. This is a permissioned blockchain modular framework, especially developed for businesses.

Being a modular framework, a lot of the scope will involve around resolving how customizable Fabric can be. To be personalized is of an essence for the creation of a good system. The other main features to be examined are the scalability and usability of this blockchain framework.

The knowledge build up from the research will be implemented in a prototype program as a final part of the project. A model of self-sovereign program with a focus on the security and authentication provided by the blockchain framework. This work aims to present advantages of the decentralizing element that can save resources and protect the personal data of the end-user. Last but not least, the project will focus on how the ledger is making the whole system trustful, thus none of the parties needs to worry about being cheated.

The objectives of the project include:

- Understanding how Hyperledger Fabric work;
  - Installing all prerequisites;
  - Installing Hyperledger Fabric;
  - Running a simple network with 2 organizations;
  - Learning how to add more parties into an already running system;
  - Trying out how the chaincode (smart contracts) work;
  - Trying to install and control newly added chaincode on a running system.
- Building a fully functional Fabric blockchain with several different parties;
- What an ID is and identity and how it is defined in the digital world;
- Deeper understanding of self-sovereign identity, what it is and how it should/could be best defined in a blockchain platform in order to be used genuinely and without misappropriation;
  - Trying out different configurations on Fabric;
  - Trying out different chaincode functions, to find out the best for the use case.
- Building a prototype of self-sovereign identity system.

1.3 Achievements

Summarise what you have achieved.

1.4 Overview of Dissertation

Following the introduction are six chapters explaining the choice of technology and development process. They will discuss the reasons behind selecting Fabric as technology of interest and some of its applications, stressing on the case about securing the identity of a person. In other words keeping the private data private.

The first few chapters are giving background and insight into what blockchain, Hyperledger Fabric and Composer are, while also giving some examples of working Fabric systems. Thereafter is the chapter about the process of developing a prototype of self sovereign identity. Reasons and graphical representation of the workflow will illuminate the design choices.

Lastly, the dissertation will conclude with evaluation of the work done and provide ideas for future development of the prototype.

Chapter 2

State-of-The-Art

This chapter is going to touch on the technical part of distributed ledger technologies, blockchain and cryptography. Thereafter a few examples will be overviewed that are constructed with Fabric blockchain.

2.1 Technical Background

2.1.1 Distributed Ledger Technologies

Ledgers have been in use of the humanity since ancient days. Their medium has been clay, wooden tally sticks, stone, papyrus and paper. They served its purpose as a one-side record-keeping tool. But this also brings concerns around who is going to validate this one-side register. Later in 15th century, the Italian mathematician Luca Pacioli became the first person, recorded, to publish a paper on the double-entry bookkeeping [9].

However, even when all of the parties have their own records of a particular deal, someone, or even a group of the participants, may cheat and keep a different record, from the original. Thus taking advantage over the people who are trying to trade fairly and honorably.

A DLT is a concurrent system, referring to a database that is consistently shared and synchronized across multiple machines/nodes in a network. It allows transactions to be monitored by multiple actors, thereby making a cyberattack more difficult. The participants at each of those machines can access the recordings shared and can keep an identical copy of it. Since it is a distributed ledger, any changes made on it are then reflected and the change is done to all the nodes holding a copy of it [8] . However, in order to know which entry should be spread, and which is/are the correct ledger/s the system has to have a consensus among all the peers and reach a final solution.

Consensus

In general, a consensus algorithm is a process in computer science used to achieve agreement on a single data value among distributed processes or systems. Consensus algorithms are designed to achieve reliability in a network involving multiple unreliable nodes. Solving that issue known as the consensus problem is important in distributed computing and multi-agent systems.[18]

Information Sharing

DLT has the ability to maintain tamper-resistant records and the arrangements could be designed to allow participants to have read-only access to certain parts of the common ledger. This even if it limits the users options, it still gives visibility which in turn stresses the integrity of the system, since you are being able to see the supply chain of a particular asset or its history. At the same time, however, since not all of a service providers transactions concerning customers might be on one or more ledgers, certain regulatory requirements could be difficult to meet by simply providing access to a ledger.[18]

2.1.2 Blockchain

Blockchain is a new technology that represents several ideas that are now able to work together. In its core, this high tech is decentralized database. Moreover, due to the asymmetric (public - private key) cryptography, every peer has an unique identity. Whenever a peer adds data into the blockchain, everybody in the network can see his or her public address as an initiator of this transaction. Since everyone participates in this database, no duplication of data is made, hence no redundancy.

In the blockchain each blocks header includes a hash of the blocks transactions, as well as a copy of the hash of the prior blocks header, hence blockchain. In this way, all transactions are sequenced and cryptographically linked together. This mechanism keeps the ledger data very secure. Even if one node hosting the ledger has been tampered with, it would not be able to convince all the other nodes that it has the correct data, because the ledger is distributed through a network of independent nodes. **[Fabric ledger doc]**

Blockchain is a linked list of blocks and a block is a group of ordered transactions. It is a distributed database on which once a data has been put, that data cannot be changed. Another unique feature is that there are specific rules, which can put data into the block. These rules, protocol, are made so that there could be no conflicts with data that is already in the database. The data is locked on to an owner. Finally, the nodes agree upon the state of the blockchain.[19] It is important that in different blockchains the consensus can be different as well. Thus, two blockchains can have different unique features.

An important note is that a blockchain network can be *permissioned* or *permissionless*.

Permissioned blockchain

This type of network means that only the ones with permission can enter the network. The consensus can be more or less a variation of Proof-of-authority, where selected nodes endorse and agree between each other of the state of the blockchain. In this case, the trade off is that the system is not as decentralized, however the transactions are much faster and cost-effective.

Permissionless blockchain

Everyone can join in the network. Perfect examples of such systems are Bitcoin and Ethereum. Typically the consensus they execute at the moment is called Proof-of-Work. This mechanism allows every node to participate in a fair contest to mine the next block. The winner gets either Bitcoin or

Ether respective to the network. This type of consensus and availability to enter the network is giving the blockchain its most famous feature - being decentralized.

Cryptocurrency

Cryptocurrency is a digital asset, medium of exchange in the network. It is created and stored electronically in the blockchain by using encryption techniques to control the creation of monetary units and to verify the transfer of funds. The most important features that cryptocurrency possess are: it has no intrinsic value - you cannot redeem it for a raw material; it has no physical form; its supply is not determined by anyone but the creators of the respective blockchain. [6] An example of a working blockchain system with a cryptocurrency can be seen on figure 1.

A peer makes a transaction. This transaction is then taken upon consideration whether it is valid or not. The decision is made by all nodes or just the ones that have been given permission to validate transactions. Upon reaching the conclusion that a transaction is valid, then it is wrapped up with many more, or in some cases alone, in order to create a block. Two things happen from the last event. First, a transaction is being completed. Second, in permissionless blockchains, the one to win the competition, to mine the newly created block receives a reward.

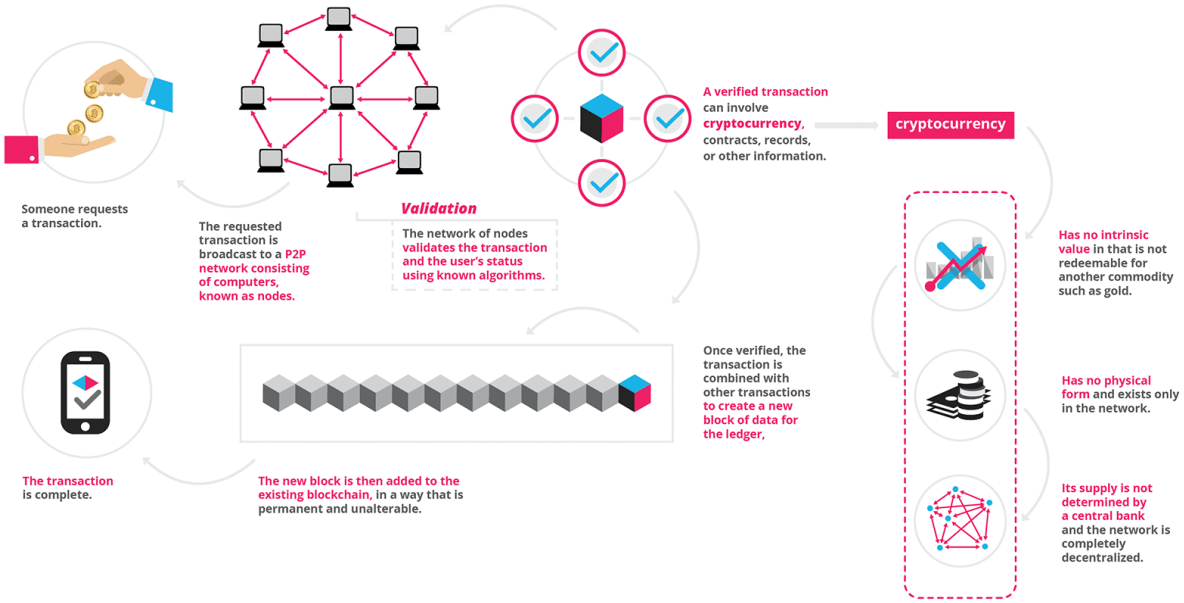

Figure 1: An abstract of a blockchain system [5]

2.1.3 Cryptography

Cryptography is the study and practice of protecting information by encoding messages or data with the use of transformation techniques. When the message is received by the correct party, they have the necessary means to decode and consume the message. Encryption method is the algorithm to be used to convert plain message into a meaningful random characters. The method would use some

sort of a key to do the encryption. In order for the receiver to make sense of this rambling they would need the correct key to decrypt the message.[11] Cryptography is widely used to protect information and communications. Three of the objectives that this field concerns itself with are:

- Confidentiality - the information is kept secret or presented in a confusing manner from anyone whom it was unintended;
- Integrity - the information cannot be tampered with without the change being detected;
- Authentication - the sender and the receiver can confirm each others identity, thus they can establish the origin of the information. This is easily done with asymmetric cryptography, and harder with symmetric. Because symmetric cryptography is easier to break and interfere with the communication.

When the same key is used for encryption and decryption, its called symmetric cryptography. When there are two different keys needed, one for encryption and the other for decryption - that is asymmetric cryptography. The Asymmetric encryption is more secure, but relatively slower. [21]

2.2 Successful projects made with Hyperledger Fabric

2.2.1 Altoros

Altoros is a software company that delivers different solutions. One of the problems their customers have is issuing bonds. The customer, Russia's National Settlement Depository (NSD), wanted a system that allows automate bond placement and accounting with blockchain, while minimizing risks of reconciliation and ensuring transparency. The reason they chose Fabric is for its support of confidential transactions and resilience in the production environment. [3]

What they did was to customize Fabric as needed for the different roles and actions. They set up four different channels so the communication,data transferring, between the peers and the NSD could be safe and secure. Every channel has its own chaincode (smart contract) that is basically the logistics behind the given channel.

One of the challenges they had was that the REST API was still in development. Fortunately, this is not the case anymore. Another challenge is that Fabric does not support cross-channel transactions. [4]

The benefits of choosing Fabric are:

- Faster transactions compared to the traditional solution, where a lot of data exchanging has to be done through a middleman. Thus, not only making it faster but also cheaper.
- Minimizing fraud in a secure trusted network. The permissioned feature does not allow for anyone that does not meet the requirements to monitor whats happening into the world ledger. Whats more because of the non cross-channel transactions, a peer could observe only the channels he is using. And even when he or she is inspecting another peers transaction, because of the encryption, he or she would not get any valuable information.
- Reduces expenses of the bond issuer by making the process faster and simplified

2.2.2 Verify.Me

SecureKey is a company providing identity and authentication provider for simplified access to online services and applications. They are using trusted providers such as banks, telcos and governments to make their clients assert identity information and connect to critical online services with digital credentials.

After the government of Canada recognized their problem sending private data to a citizen, they asked for a solution. SecureKey responded to this call in collaboration with IBM with a blockchain based solution. It is a mobile app, that allows the user to connect different types of services providing only specific data. So what happens is the user connects to the blockchain through the phone. Then, it connects with the service actors. It is important to note that in the phone there are only pointers to the data and not the data itself. Whenever a person is sharing his or her identity with the new service he or she can see exactly what information is asked to be provided. [23]

The SIM card is used as an anchor of trust. Since the system is private and permissioned blockchain, only trusted actors like banks and government can write on it. Upon losing or breaking the phone, the creators reassure that is easy to recover whats lost. Again, here one of the main reasons to choose Fabric for the development of this service is mainly - the adaptability of the platform and the zero-knowledge proof supported concept. [20]

The benefits of using Fabric are :

- Data integrity
- Security and resiliency
- No central database or honeypots
- No central point of failure
- Cannot track user across relying parties; privacy of the data
- Cost efficient due to simplifying the process

Cons:

- New - open standards needed

2.2.3 TradeLens

TradeLens is a company founded by collaborative work of Maersk and IBM. Maersk is an integrated container logistics company working on improving the supply chain area. The idea is to make the shipping process cost-efficient, faster and in respect to accessing the needed documents - simpler.

For this task, the collaboration is combining their technical and specialized knowledge to build a system on top of Hyperledger Fabric. What they created is a network, that tracks the supply chain - the documents needed for starting a shipping process, the deal that is made, the location of the containers.

To participate, a user has to pay a price to enter the network. Still it is not confirmed what the requirements are. However, once a user decides to enter he will experience something way different from the usual way of things. Due to the blockchain technology, a user can check a block on the

blockchain to track the location of the container or any other process involved. The usual way for this simple task would be to request this information from a middleman. TradeLens are saying they can reduce the paperwork and the need of a mediators, saving lots of time and money in the process. [15]

It is important to be mentioned that TradeLens is not fighting the frauds. If a user input false data at start,that seems to be correct to the endorsement parties, the system wont be able to catch it. So the network helps to have less fraud, but it is more of a side effect rather than main function.

Another great use of this system is that, according to the World Trade Organization, simplifying the supply chain will not only reduce costs, but also help developing countries to increase their export by more than 30% . [16]

2.2.4 BitNation

2.2.5 E-Residency

2.2.6 AIA Group

Chapter 3

Technology

This chapter is going to cover information about three essential technologies. Docker is creating virtual environment where Fabric nodes are runnig. NPM (node package manage) is used to access Hyperledger Composer command line interface. VS Code is a text editor that supports two vital extentions, one for docker and one for Hyperledger Composer, which greatly improve the experience of the developer.

3.1 Docker

Docker and Docker Composer are essential for the developing of this project. This technology is being used to run Hyperledger Fabric. Different parts, modules, of the system are mounted on Docker containers. All of those containers know about each other and intercommunicate. This system is also known as Fabric.

Docker containers are similar to a virtual machines. Alike resource isolation and allocation benefits, however, containers are more portable and efficient since they virtualize the OS instead of hardware.[10] Figure 2 shows an abstraction of where Docker containers take place in the software architecture when running.

3.2 NPM

3.3 Visual Studio Code

Visual Studio Code for short VS Code, is an open source powerful, yet lightweight, text editor. It is available for Windows, Linux and macOS. The text editor comes with built-in support for TypeScript (Angular), JavaScript and Node.js. One of the many features is that it has a built in terminal in the GUI, making simple command calls easier than ever.

Released only 4 years ago, it is doing such a great job that in a Stack Overflow survey, the editor has been chosen as most popular in 2018.[22]

Whats more important is that there are a lot of great extensions for this source code editor. The ones immensely important for this project are Docker, Hyperledger Composer and Beautify.

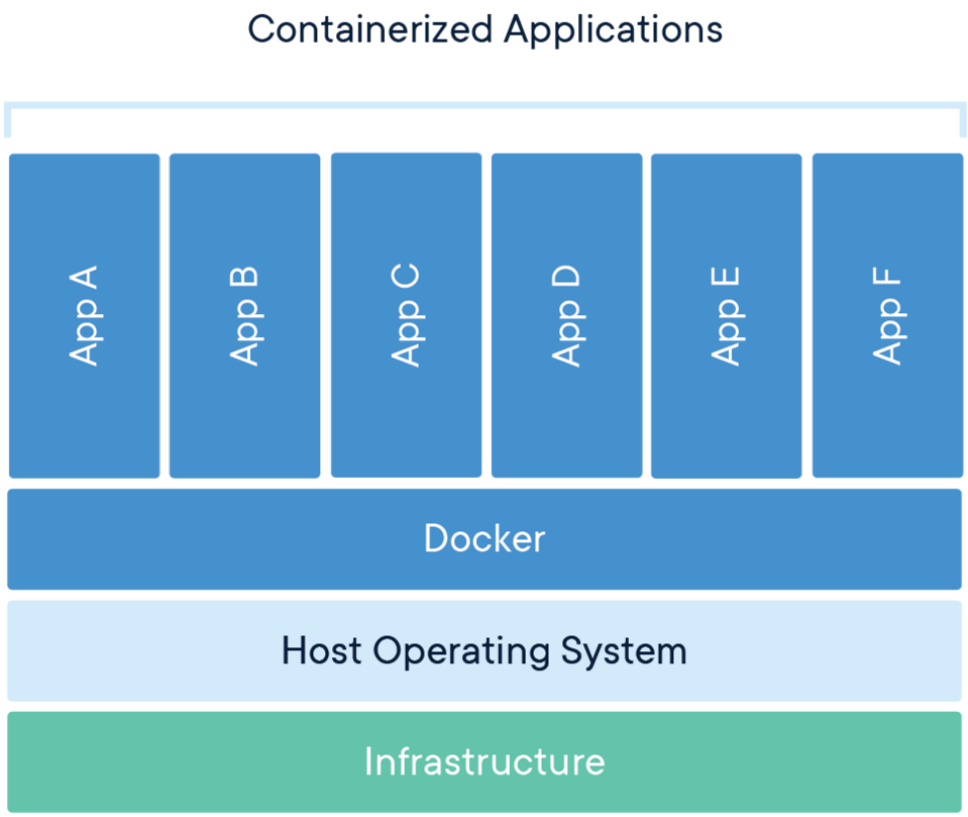

Figure 2: Containerized applications [10]

The docker extension is providing enormous help. Not only one can see the all images, and containers that are created or running on the machine, but also one can manage, control and inspect the images or containers. To be able to monitor the state of all Fabric nodes, which are mounted on the docker containers, saves a lot of time. The build in terminal is used when invoking or inspecting what happens in a specific container. The contrary to this method, is having to manually search and type commands through the terminal in order to find and inspect the specific container.

Hyperledger Composer extension is validating the structure of the business model files, discussed in section 5.4 . The validating goes through the .cto, model, file. The model file contains the structure of the business model. So from time to time, there could be errors, just because you are modifying one of the other files, without having the model file open. As a rule of thumb, while working on your network, its best if you have open all files of that system at once.

Having this extension is giving you the freedom to write your code in a good environment, where you can see whats happening, in comparison to playground, online tool to run business networks

discussed later in section 5.1. Once the code is written, testing it in the composer playground is very good idea.

Chapter 4

Hyperledger Fabric

Permissioned blockchain

A blockchain where the peers need to meet certain requirements to enter the network where he or she can perform certain actions is called permissioned. These systems are more attractive for the business and enterprise because they are faster and more cost-effective. Another feature that is appealing for those clients is the role system. That way actors, that all of the companies trust, can be the endorsers, the ones to validate the transactions. The feature could also be used to classify different players into respective roles. Which can give a particular system a better clarification and simplicity around executing different tasks.

It is not as decentralized system as the permissionless blockchain, however the tradeoff is acceptable enough for businesses to prefer it. The processes of Anti-Money Laundering and Know Your Customer require that service providers can confirm a peers legal identity and give clearance to make a transaction. The adoption of these processes in permissionless blockchain would be wrong, since they can illuminate who this peer is, thus breaking the promised anonymity. On another note, a permissioned blockchain can have larger volume of transactions per given time compared to a public one.

Whats more, many prefer permissioned blockchains for supply chain. Since only the peers inside the blockchain can see what is happening on the path of a material to its final destination. And the tracking data can travel much faster, due to the simplified verification and less peers.

Hyperledger

Hyperledger is a group of open source projects focused around cross-industry distributed ledger technologies. Hosted by The Linux Foundation, collaborators include industry leaders in technology, finance, banking, supply chain management, manufacturing, and IoT.

4.1 Fabric overview

Fabric is one of those open source projects. It is a modular distributed ledger, which makes it highly customizable and adaptable to a variety of ideas and restrictions. The main scope of this

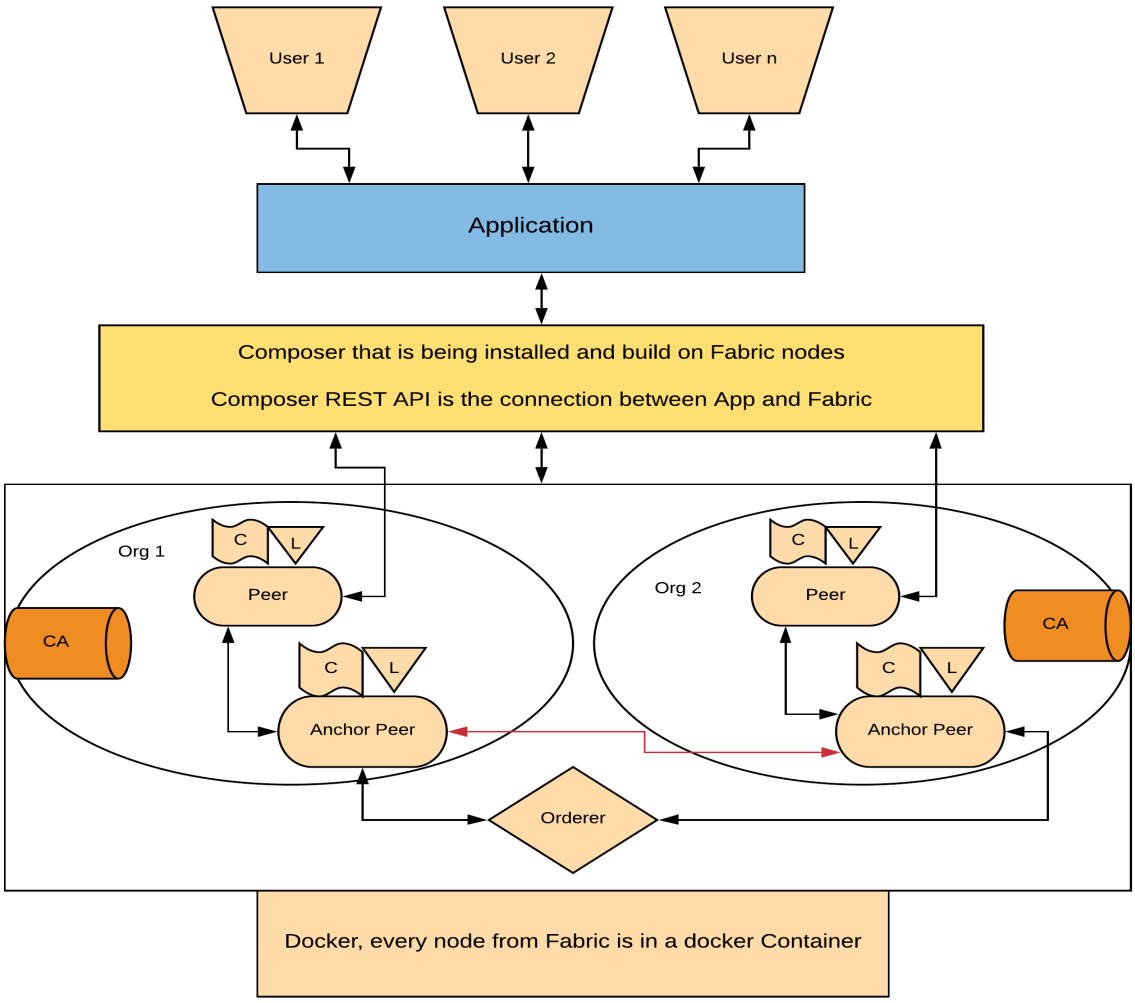

Figure 3: High-level overview of Fabric system

undergraduate project is to test how functional and useful Fabric can be in different business and science situations. Is it making some of the use cases in those fields cheaper and more secure?

The feature which makes Fabric the perfect choice is that it can create different communication channels between different peers. Some of those channels could be for contract making between a supplier and a buyer. If a supplier has a favourite customer, he or she may give an exclusive deal. However, if everyone see this exclusive deal, then the business of the supplier would break down. Thats why this exclusive deal could exist in a confidential channel, one that only the two of them can see.

This Hyperledger project is a preferred platform mainly because of its adaptability to different use cases. One interesting feature, and main reason for the self-sovereign use case, is that Fabric supports zero-knowledge proof (ZKP). What this means is that it allows a peer to identify itself without having to show any private data. This gives authority to ZKP to offer anonymous authentication for clients in their transactions. [16]

The act of communication between different peers from different organizations (or groups) is through channels. These channels can be public or confidential. The communication inside works based on the chaincode, the smart contract. All of the logistics and functionality of a new blockchain application is based on its smart contracts. That is why they are extremely important and main object of interest in this undergraduate project.

4.2 Essential Fabric elements

Hyperledger Fabric is set by organizations that want to setup a consortium. Every organization is constructed by several type of peer nodes (or just peers). All that these peers require is appropriate configuration and cryptographic materials like certificate authority (CA) and endorsement policy.

Committing peer node

The normal peer node or just peer, is a vital part of the Fabrics network. This node is holding instances of the ledger, thus they commit the new blocks when received. Usually in production, multiple peer nodes are created. This imposes redundancy, but it also creates a no-single point of failure for the system.

Endorsing peer

A special kind of commiting peers. They are important in the transaction flow and the consensus. When transaction is processed for verification, the endorsing peers are taking it and simulate what can happen if the transaction is added to the ledger. For that purpose, endorsing peers also have an instance of the chaincode in order to run the transaction proposal. If the simulation went well, and there were no problems with the current state of the ledger, the endorsing peer signs the transaction as validated. The endorsement is done by the committer nodes against the endorsement policy, which is specified when the chaincode is deployed. That means that while some channels will require majority of endorsing peers to run the transaction proposal, other could be happy with just a single endorsing peer.

Orderer node

It is pivotal for the consensus mechanism and transaction flow. It is responsible for consistent Ledger state across the network. Once the endorsing peers are done with the validation of a transaction, they send it to the orderer node. The orderer node is taking all transactions and then puts them into order, then batches them into blocks. Thereafter, the block are sent to the commiting peers via the anchor nodes.

The orderer node neither executes the chaincode nor holds a copy of the blockchain. However, the ordering service (multiple nodes) are implementing specific ordering algorithms to decide what to do with the responses given from the endorsing peers. More about them in the consensus section.

Anchor peer

The anchor peer is the connection of the organization with the network. If there is no anchor peer in the organization, then this organization cannot connect to any other organization. Whats more important, it is the link between the orderer node and the ledger inside the organisations. If there is no mechanism to send or receive transactions then the whole organisation will become obsolete. Thus having setup several anchor peers, just in case of some unfaithful crash, is the safe bet.

CA

Uses the BCCSP to generate cryptographic material for Peers, Orderers and Users, that certifies them and allowing access to the respective network. CA provide dynamic certificate lifecycle management capabilities such as registering, revoking and erolling users via the REST (representational state transfer) API.[17]

MSP

Responsible for all the cryptographic operations - issuing, verification, signing. Every organisation has local specification of MSP with local specification of CA.

Cryptography in Fabric

Fabric uses BCCSP, the Blockchain Cryptographic Service Provider. It is a creation of IBM and it offers implementation of cryptographic standards and algorithms. Written in golang, mainly to be used on Hyperledger Fabric.

BCCSP is designed to be pluggable component into the fabric network. It can mount different public key cryptographic standards, depending on the clients requirements. Supports both symmetric and asymmetric encryption. Fabric needs both simple encryption/decryption and signed. One the key for the preferred algorithm is retrieved BCCSP exposes encryption and decryption methods to be called. The sign is obtained by the key as well, thereafter sign and verify methods are exposed to be used. [12]

Certificates and PKI

Every organization in Fabric has a membership service provider (MSP) which is pluggable interface to support variety of credentials architectures. In other words, it takes what the BCCSP provides. The MSP is responsible for the identity and the authenticity of respective organizations peers and users. By default, the security implementation is Fabric-CA. Public key infrastructure that writes certificates defined by x.509 standard. MSP and CA will be explained in more detail in section 4.2.

4.3 Fabric's Ledger

Consists of two distinct, though related, parts - **world state** and a **blockchain**.

4.3.1 World state

The world state is a database that holds the current values of the assets in the ledger as ledger states. The ledger states are usually expressed as key-value pairs, though there is some flexibility in this regard. The world state changes frequently because of the CRUD operations applied on the network. Needless to say, only validated transactions are able to change the ledger.

The world state is created with the premise of faster transactions. Instead of traversing the entire blockchain to calculate the current value of the asset, a program can just take it from the world state.

The world state is a NoSQL database. It provides rich set of operations for the efficient storage and retrieval of states. Fabric can be configured to use different types of db that will answer to the requirements of the network. Usually Fabric will be either with LevelDB for simple networks and CouchDB for more complex networks.

In order to keep track of the changes in the WS, a counter called version number is incremented every time there is a change. This counter is checked whenever the state is updated to make sure that the current state of an asset matches the version at the time of validating the transaction. This ensures that the world state is changing as expected, meaning there has not been a concurrent update. [14]

4.3.2 Blockchain

The blockchain is with its defining qualities - immutable sequence of blocks, each of which contains a set of ordered transactions. Every new transaction is being validated or rejected. The successful transactions are batched into blocks and appended to the blockchain - enabling you to understand the history of changes, which result into the creation of the WS. On figure 4 is a representation of the ledger

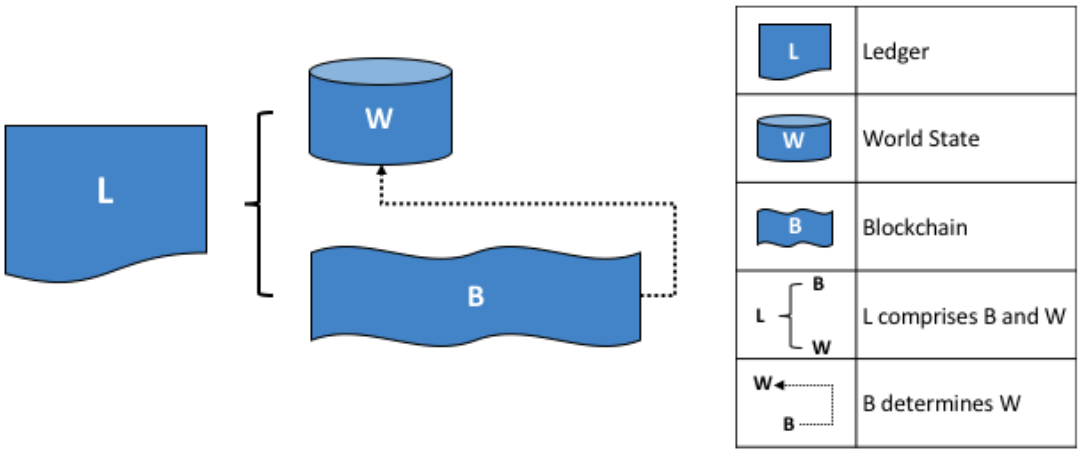

Figure 4: An abstract of the Fabric Ledger [14]

Physically, the blockchain is always implemented as a file, in contrast to the world state. This is a reasonable design choice as the set of operations on the blockchain data structure is heavily biased

towards small limited number. Appending to the end of the blockchain is the primary operation, querying is currently infrequent operation because of the WS.

Blocks and their structure

Figure 5 shows how the blockchain is structured and their graphical representation.

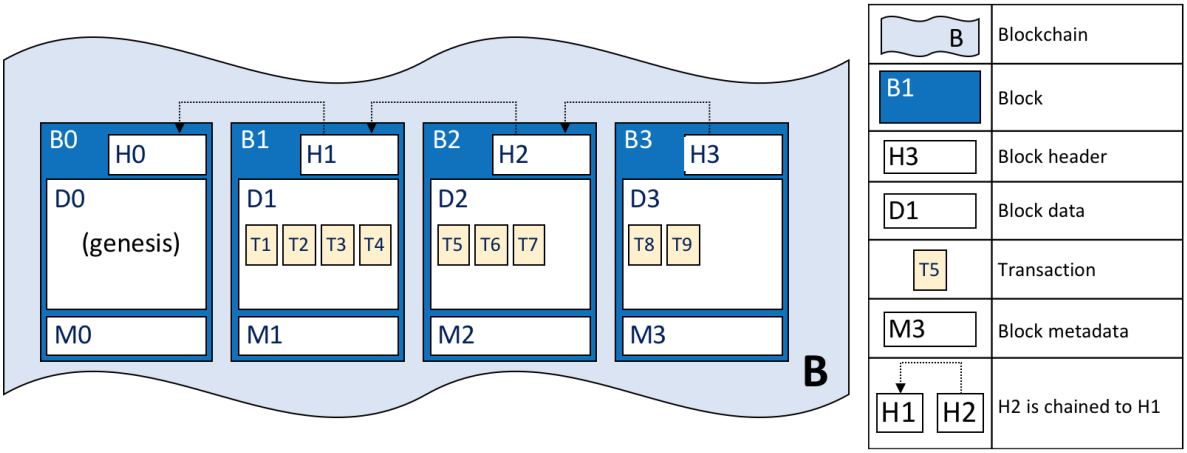

Figure 5: An graphical representation of the block [14]

- Block header - consists of three fields:
  - Block number - integer, at 0 is the genesis block, increased by one for every new block that is appended to the blockchain
  - Current block hash - hash of all the transactions contained in the current block
  - Previous block hash - copy of the hash from the previous block
- Block data - contains a list of transactions arranged in order of appending.
- Block metadata - contains the time when the block was written as well as, the public key, certificate and signature of the block writer.
- Transactions
  - Transaction header - captures the essential data about the transaction like name of the chaincode and its version.
  - Signature - is being generated only by the users private key. The field is used to check that the transaction details have not been changed.
  - Proposal - encodes input parameters supplied by the user via an application to provide them to the chaincode which creates creates the proposed ledger update. But before the

proposed transaction is being added to the blockchain it first has to be verified and validated by the *endorsing peers* which are discussed in the following subsection.

- Response - captures the before and after values of the world state, as a read-write set. Its the output of a chaincode (smart contract), if the transaction is successfully validated, the response will be applied to the ledger to update the world state.
- Endorsements - Although there is only one transaction response, there are multiple endorsements from different *organisations*.If there are not enough endorsements, specified in the transaction verification process, the response would not match the needed number and the transaction will be rejected as invalid and will not update the world state.

4.3.3 Channel

In Ethereum or Bitcoin, when someone joins the network, they are connecting to the blockchain. That being just one ledger starting from genesis block. Every participant has all blocks or just the headers, however, the important note is that everybody keeps information from the same blockchain.

Hyperledger Fabric, contrary to the blockchains above, can have several different ledgers in one network. Since it is build with the premise of business, having several ledgers in a network is beneficial. It is possible, because the ledger is owned by a channel. A channel is like a communication channel, it is the mechanism by which different organizations are interacting between each other. Those interactions are append on the immutable ledger.

Whats more, every channel is having its own business logic, chaincode installed. So a network, depending on the use case, can have one channel with all business partners, one channel with all end users, and several other channels with specific business partners and end users. And all of them can have different chaincodes or the same, depending on the case and requirements of the system.

All peers from a channel have the same ledger. Whats more a peer can have multiple ledgers,hence multiple instances of different chaincode. However, those ledgers are completely different and cannot interact between each other.

4.4 Fabric Consensus

The consensus in fabric is broken out into three phases: Endorsement, Ordering and Validation. At the end of the section, figure 6 is representing the whole process.

4.4.1 Endorsement

Endorsement is driven by policy (m out of n signatures) upon which participants endorse a transaction.

This phase starts with client application sending transaction proposal. A transaction proposal is an action that will change the state of one or more assets in the ledger. The endorsing peers are taking this proposal and simulate it into the network. The transaction has been signed with the result of the simulation. Depending on the policy, which is defined upon installation of the chaincode, there will be a number of signatures needed before the transaction to proceed into the block. If the policy is not fulfilled then the transaction is going to be rejected.

At this point, nothing in the ledger is changed. Many different transaction proposals can be taken and put against the current ledger to check out whether the transaction is going to be valid or rejected. While it is done with the premise of scalability, this parallel validation is introducing one problem.

If there are two transactions being validated that update the state of the same asset, then they will be voted both valid from the endorsing peers. However, once the transactions are being sent to the next phase, the ordering, one of them will fail. Upon batching the transactions they are put into order. Depending on that order, what happens to be second transaction is going to fail.

4.4.2 Ordering

Ordering phase will get the endorsed transaction and agrees to the order to be committed to the ledger.

The ordering phase or service, consists of cluster of orderer nodes. They are receiving the transactions. Batch the transactions into a block. An important event occurs here, that once a transaction is put into the block, it it said to be final. This means that the position of the transaction in the ledger is immutable. The order is consistent and strict. Finally, the blocks are distributed to the committing peers.

Note, the order of the transactions is not necessarily as in the order of the transactions send to the node. This is important, because once the batch is done, some of the transactions may be rejected. It can happen due to the fact that two transactions may have tried to change the same resource, and only one change can happen per block. When transaction proposals enter the network, the endorsers are simulating them in a parallel manner, against the current ledger. So, both transactions will be looking at the same state of the asset they want to change. Both of them can be valid, however, the first that gets to be put in the order batch is the one that is going to make the change in the blockchain and world state. The second is going to be rejected.

Batches are defined mainly by two factors. The first is the time to wait before a block is being generated. The waiting starts after the first transaction is received and can finish on the time specified or before that. For a block to be generated before the end of the time set, the blocks size had to reach its limit. If the configuration is done so that 10 transactions can be put into a block, and 2 seconds to be time set, then there are two cases. First - have a block in 1 second with 10 transactions. Second - have a block with less than 10 transactions after 2 seconds. The configuration can be found in confingtx.yaml. Will be discussed in more detail in the configuration section. (12 min [20])

So in order to effectively avoid invalid transactions due to two transactions updating the same resource, the configuration of generating the blocks have to be well-thought for the specific system.

There are several types of ordering mechanisms implemented in Fabric. They are all pluggable, so the engineers of the network can try with one of them and then go to the other, simply to see which one will be most efficient for the case:

- SOLO - involves a single ordering node, single point of failure. It is very fast, but unreliable for real data, which makes it the perfect mechanism in developing stage.
- Kafka - based on Apache Kafka, high-throughput, low-latency platform. Crash fault-tolerant solution.

- PBFT - practical byzantine fault tolerant mechanism. It is both crash fault and byzantine fault tolerant, meaning it can reach an agreement even in the presence of malicious or faulty nodes. Slow, but secure.

According to a IBM paper [7], PBFT is the most used one in production, however, due to the pluggable design, depending on the network, it can be changed with Kafka or a future solution.

Usually, since the first and the third steps are always the same, people would refer to Fabric consensus just as the name of the ordering mechanism.

4.4.3 Validation

Validation - takes a block of ordered transactions and validates the correctness of the result. The moment a committer node receives the new block, it starts checking every transaction and the transaction result from the endorsing peers. This check is against the endorsement policy. Here is the moment where if two transactions are trying to change the one asset, the second fails. However, instead of returning the whole block, the faulty transaction is just being labeled as *invalid*. The committer updates the ledger. Lastly, asynchronously returns to the user/app that the transaction is successful or not.

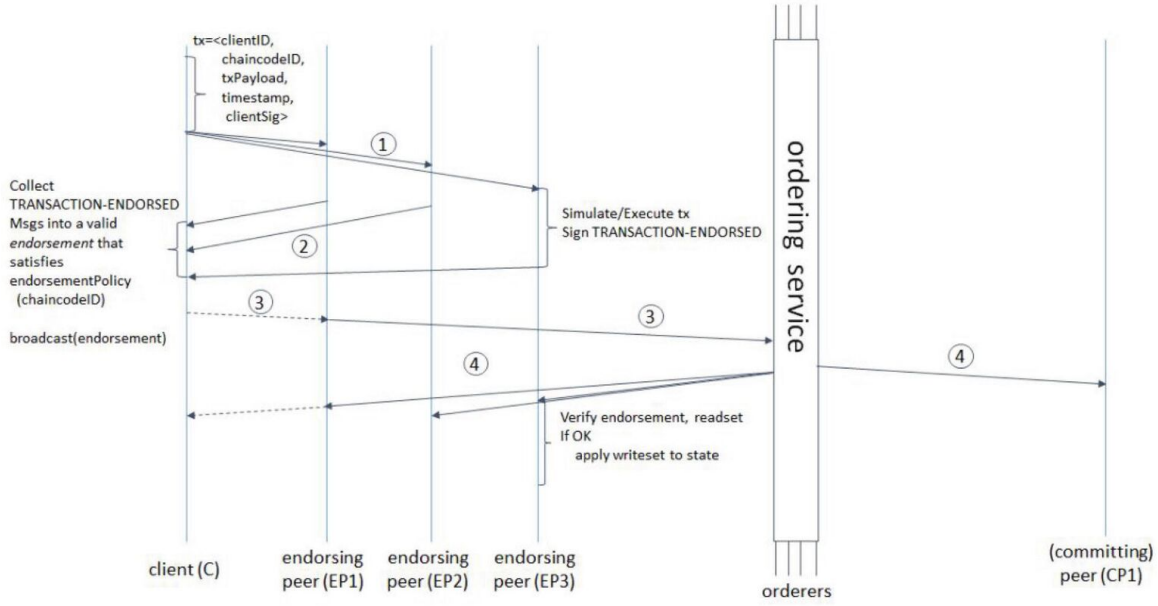

Figure 6: An graphical representation of the transaction flow [13]

4.5 Fabric's configuration

When setting up Fabric, there are two very important files. It is out of the scope of this project to show how one can generate custom network. However, it is critical that the reader is introduced to those two configuration files and see the important bits and pieces.

The two files are crypto-config.yaml and configtx.yaml (If you follow the user manual and download the git repository, then the files should be located at /fabric-samples/first-network/).

Yaml is a human-friendly data serialization standard for all programming languages. In Hyperledger Fabric it is being used for all configuration files.

4.5.1 Crypto-config

The crypto-config.yaml holds the information about the configuration of the orderer nodes and the organizations. It does not matter where the physical machines are located, as long as they know which organization they belong to in the network and have the respective certificate to operate with.

```
OrdererOrgs:
  # ---------------------------------------------------------------------------
  # Orderer
  # ---------------------------------------------------------------------------
  - Name: Orderer
    Domain: example.com
    # ---------------------------------------------------------------------------
    # "Specs" - See PeerOrgs below for complete description
    # ---------------------------------------------------------------------------
    Specs:
      - Hostname: orderer
# ---------------------------------------------------------------------------
# "PeerOrgs" - Definition of organizations managing peer nodes
# ---------------------------------------------------------------------------
```

Figure 7:

The first configuration is about the orderer nodes. The name specifies the name of cluster of orderer nodes. Different clusters can be of use for different channels. Depends on the use case. In this file, there is one cluster.

The domain is where this entity will run. This is what other nodes will try to find in order to connect to the orderer. It is not necessary to be a top-level domain, like a web page. Depending on the use case, the network could have a local private domain space that could be used in the context of the network.

Certificates are bind to specific domains. So if an anchor node does not have the correct certificate, it would not be able to connect to this orderer node. Security measure that prevents outsiders to invoke information from this orderer.

The hostname specifies the name and creates a node. In the picture above, there is only one node, called orderer. Left with one node for the orderer, is dangerous, as it is a single point of failure. For testing purposes it is okay, however in real-life system, there should be a lot more. The name by which other nodes are going to find this one is by the schema, first the hostname then the domain - orderer.example.com

The next part is about the peer organization. The name and domain serve the same function.

The template is about the different nodes in the particular organization. The count variable sets with how many nodes will the organisation start. In this case there will be two peers from this organization. Later on, peer nodes can be introduced or deleted, depending on the requirements and what

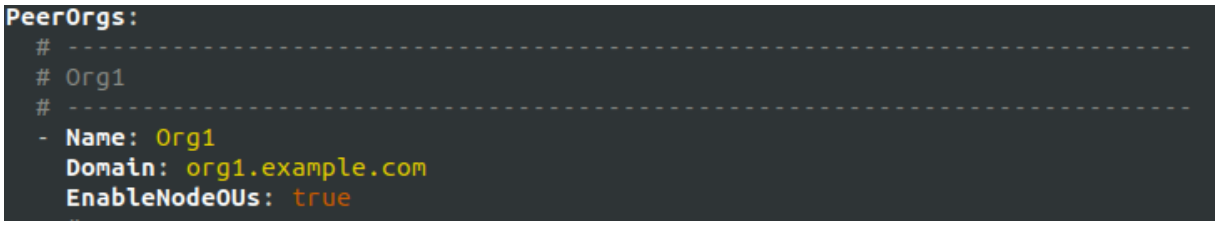

Figure 8:

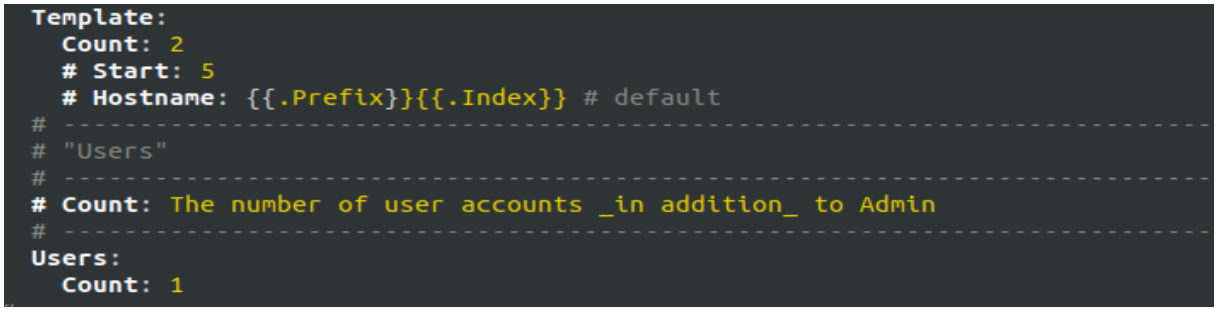

Figure 9:

would be the most beneficial occasion. Important note is that every peer is having its own certificate. The name of the node will be peer0.org1.example.com.

The users section is about how many users, other than the admin, should the network start with. If the system is about a lot of people, then this variable would not matter. The Certificate Authority can dynamically introduce new users and nodes to the system.

4.5.2 Configtx

Configtx.yaml holds information about the organisations and the orderer in a bit more detail in regards to input and output. It is the configuration about the genesis block on a channel.

In figure 10 you will see Name, ID and MSP. The three variables are regarding the cryptography and accessibility. The ID is defined by the membership service provider. It is associated with all the crypto-materials - certificates by admin, ca and tls. The MSP ID is verification tool in the backend to verify the users certificates.

Specific section about the Anchor Peer - selecting which node is going to be the anchor peer in an organization. These nodes will be able to see each other by host name and communicate via port. This is how a connection between two or more organizations is configured.

Specific setting for the orderer. The orderer service is specified under the variable OrdererType, which is SOLO, not good for production, but good for testing. Under the addresses are specified all nodes from the cluster, which are going to batch the transactions into block. In this default configuration that is only one node.

Next in line are the configurations about the block size and finalization. BatchTimeout is the variable holding how many seconds should the ordering service wait before it batches all available transactions into a block. The only case where transactions are going to be batched earlier is if the

```
Organizations:

    # SampleOrg defines an MSP using the sampleconfig.  It should never be used
    # in production but may be used as a template for other definitions
    - &OrdererOrg
        # DefaultOrg defines the organization which is used in the sampleconfig
        # of the fabric.git development environment
        Name: OrdererOrg

        # ID to load the MSP definition as
        ID: OrdererMSP

        # MSPDir is the filesystem path which contains the MSP configuration
        MSPDir: crypto-config/ordererOrganizations/example.com/msp

    - &Org1
        # DefaultOrg defines the organization which is used in the sampleconfig
        # of the fabric.git development environment
        Name: Org1MSP

        # ID to load the MSP definition as
        ID: Org1MSP

        MSPDir: crypto-config/peerOrganizations/org1.example.com/msp

        AnchorPeers:
            # AnchorPeers defines the location of peers which can be used
            # for cross org gossip communication.  Note, this value is only
            # encoded in the genesis block in the Application section context
            - Host: peer0.org1.example.com
              Port: 7051
```

Figure 10:

max number of transactions for a block has been reached.

BatchSize has three variables MaxMessageCount, which is defines the top limit of transactions can a block have. AbsoluteMaxBytes and PrefferedMaxBytes are defining the size of the block.

Since endorsing peers are simulating the transaction proposals on the current state of the ledger, the finalizing of transactions is utmost importance. As I already mentioned, valid transactions may be rejected because the ledger is not updated frequently enough. In the perfect case, the system would be able to take enough transactions for small amount of time, so that the blockchain would always be updated with the latest actions and state changes.

BatchTimeout and BatchSize are extremely important. If not set correctly they can break the system. On the other side, if they are set with most care about the system and how many transactions there would be, it could make the system faster and invalid-transactions resistant. It is case specific what would be the best strategy for their configuration. The architect should always have in mind the blocks to be generated fast enough and to be the correct size. If the blocks are not going to the transactions number limit, then they would be just wasting the resources allocated.

In the profiles section are the final configurations for the genesis block of the channel and the participants of this network.

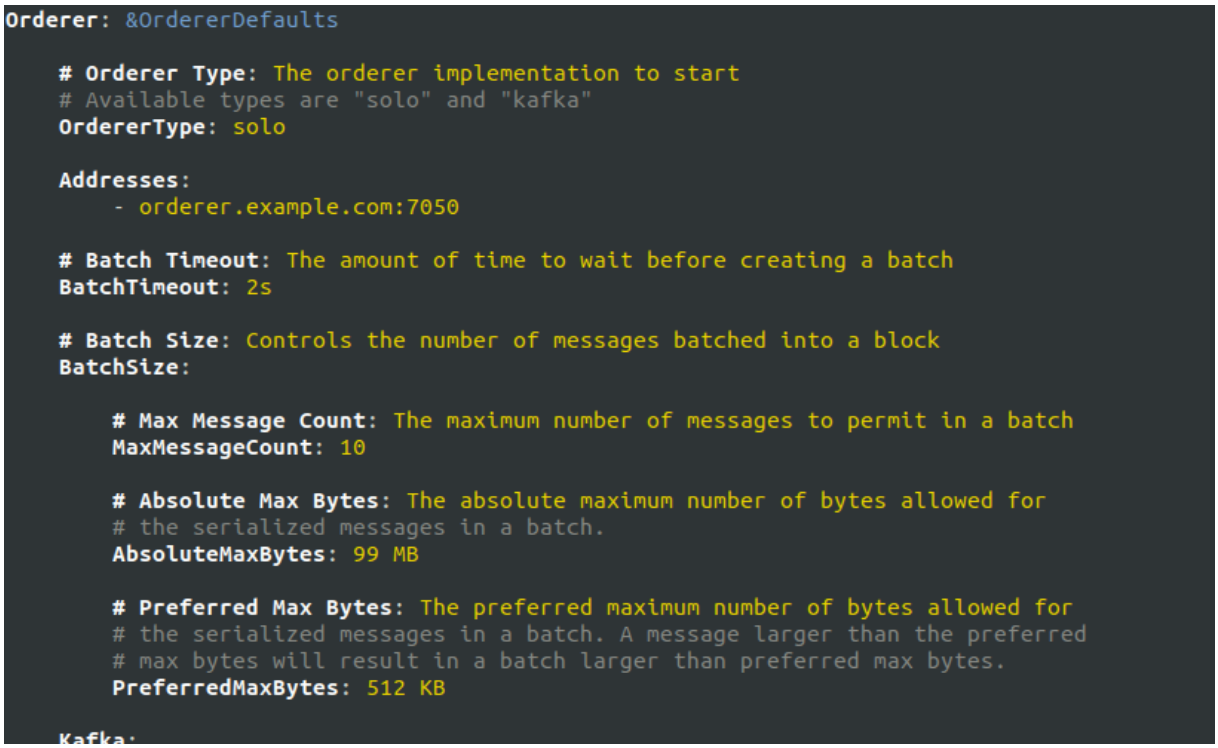

Figure 11:

Configurations about the genesis block are set before the channel. The settings are defining the ordering system, the participants in the blockchain. Setting which organization are going to enter the consortium, later to be given in the channel configuration. Consortium in this context is which organization will be in a channel that is being served by the orderer.

Lastly is the channel definition, where the architect would specify which consortium(s) is going to be used, and which organizations are going to be initially in the particular network.

```
Profiles:

    TwoOrgsOrdererGenesis:
        Capabilities:
            <<: *ChannelCapabilities
        Orderer:
            <<: *OrdererDefaults
            Organizations:
                - *OrdererOrg
            Capabilities:
                <<: *OrdererCapabilities
        Consortiums:
            SampleConsortium:
                Organizations:
                    - *Org1
                    - *Org2
    TwoOrgsChannel:
        Consortium: SampleConsortium
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Org1
                - *Org2
            Capabilities:
                <<: *ApplicationCapabilities
```

Figure 12:

Chapter 5

Hyperledger Composer

5.1 Overview

Hyperledger is an open-source application development framework. The tool aims to simplify the creation of Fabric blockchain applications. Composer lets the user to establish a system without having the knowledge of Go programming language or low-level details about blockchain.

Composer creates blockchain business networks which are compressed into .bna files. They are composed out of three main files - model, logic, permissions - that are discussed in section 5.4.

One can test out the .bna network without Fabric nodes, just to see if it is going to work at all. The online tool https://composer-playground.mybluemix.net can provide an environment for testing, fixing and fast deployment of the fixed system. Another great functionality of this online tool is that it can export the system as .bna file. Instead of going through mundane process of manually creating the connection and compression of all files into business network, one can compress them through the playground.

There are two things to keep an eye on when exporting the business network. First is that all files of the system should be present in the composer-playground. Second is, when one is installing the .bna file onto Fabric, the command needs a version of the system. It can be found on the define window on the left bar From: previous version To: current version . The composer cli command requires the current version to be provided.

Unfortunately, IBM are not going to continue further development on the project, due to the fact that Composer and Fabric drifted away from each other, and are not completely compatible anymore. [**?**]

5.1.1 Composer CLI

Composer command line interface, with primary command in the terminal composer, is used for administrative, development and operational tasks.

This project has been tested and developed with composer cli 20.2 and composer-rest-server 20.7, since crs 20.2 breaks upon using one of the functionalities (multi-user crs).

5.2 Business Network Cards

In order for a user to start using the system, they need some sort of identification. This identification can be created from the system they are going to be using, or import from another one. The importing from another can happen only if the requirements for participant match.

When someone wants to enter a blockchain network, a public and private key-pair is generated. The user then can interact with the system with those two keys. Usually whenever someone wants to send a message to the user, he or she will take the users public key and encrypt the message with it and send it to the user. Then the user will decrypt it with his/her private key. On the other hand, if the user wants to create something and put it to everyone, he will encrypt that creation with his/her private key and will put it open in the network. If someone wants to see what this new message is, they have to take that users public key in order to decrypt the creation and see it. Among other things such as security, this method also proves authentication and integrity of the message.

In the world of Hyperledger Composer, this credentials pair is called business network card. It holds the keys, certificates and connection profile to a particular system. One user can have multiple business cards for different networks. In some for being an end user, for others - business partner.

BNC (blokchain network card) provides all of the information that is needed to connect to a BBN (blockchain business network). You can only access a BBN through a valid BNC. A card contains an Identity for a single Participant within a deployed BN.

Can be stored on : File system, RAM (embedded runtime), database and cloud storage. The safest is local file system.

5.3 Composer Historian

Specialised registry which records successful transactions, including the participants and identities submitted them. The historian stores transactions as HistorianRecord assets, which are defined in the Hyperledger Composer system namespace.

The historian registry is a Hyperledger Composer system-level entity. To refer to the historian registry as a resource for access control the historian must be referenced as: org.hyperledger.composer.system.HistorianRec

Note that: All participants must have the permission to create HistorianRecord assets. If a transaction is submitted by a participant who does not have the permission to create such, the transaction will fail.

5.4 Composer programming

Hyperledger Composer is using several files in order to create a compressed one that would be installed on Fabric.

5.4.1 Model file

Model files contain the structure and types of the system. The modeling language is object-oriented. It supports inheritance, concepts, enums and different types of variables. Because of the

inheritance, the resource child will take all properties and fields required from the extended resource parent and will add additional from its own definition.

The most important part of a model file is the namespace - defined at the start of the document. In all references to the resources of this particular file, the namespace will be present as part of it. If the file does not contain a namespace, it will be unusable

In Composer the system can be modelled by introducing participants, assets, transactions and events.

- Participants - usually can be thought of as a person or some institution.
- Assets - can use as model everything of value
- Transactions - have built in variables for transactionID and timestamp
- Events - have built in variables for eventID and timestamp

The primitive types supported in the language are:

- UTF8 encoded String
- 64 bit numeric value Double
- 32 bit signed whole number Integer
- 64 signed whole number Long
- an ISO-8601 compatible time instance, with optional time zone and UTZ offset DateTime
- Boolean.

Within the model one can also have arrays and relationships. Reference to an asset or participant is indicated by –¿ and are called relationships.

```
transaction confirmDiploma{
 --> Person owner
 --> Diploma diploma
 --> HighSchool hs
}
```

As seen on figure 5.4.1 is an ordered set of the objects name and the instance created. The full name of the newly created resource will be namespace.Resource#nameOfTheObject for example org.ssidentity.Person#examplePersonID.

5.4.2 Logic file

The Logic file is where the business logic is written. The programming language used is TypeScript Angular framework. Unusual characteristic is that you have to define the resource and parameter in the comment section before using it in a normal function.

```
/**
 * To confirm the diploma
 * @param {org.ssidentity.confirmDiploma} cd
 * @transaction
 */

function confirmDip(cd){
if(cd.diploma.owner.personID === cd.owner.personID){
   return getAssetRegistry('org.ssidentity.Diploma')
            .then(function(confirm){
               cd.diploma.diplomaStatus = 'Confirmed';
               return confirm.update(cd.diploma);
               }).catch(function(error){
                   throw new Error (error);
               });
   }else{
       throw new Error ('This is not the person that graduated');
   }
}
```

5.4.3 Access control language file

ACL provides declarative access control over the elements of the domain model. The rules are consecutively executed. Which means that if the first rule is to grant permission to resources to any action, no matter what rules follow, all resources will be grant full access. ACL is constructed by a single file with fixed name - permissions.acl.

The file have rules to restrict the users of action, rather than to permit such. If the file does not exist, then there is no restriction in the system for any resource or user. The rules have an action: allow or deny

Rule controls permission to CRUD operation on resource(s). Unfortunately, each rule can use only one resource. Meaning that in order to give permission to one participant to function with several resources, a rule that allows CRUD operation for each specific resource have to be written.

Simple rule control access to namespace, asset or property of an asset by a participant type or participant instance.

Conditional rule takes boolean JavaScript expression evaluated at runtime to ALLOW or DENY access to the resource by the participant.

Chapter 6

Self-sovereign identity prototype

The main goal of this thesis was to see if Fabric blockchain is going to be up to the task of improving peoples identity. Securing the private information while proving authenticity on the limited data provided to the institutions that require it.

6.1 Design

The prototype is designed to be somewhat close to the reader. Representing a system designed to replicate the real actions that would be done in Bulgaria. The entities included are five participants and three assets, has six transactions and can emit two event.. Person, high school, driving school, university and customs, being participants and diploma, university diploma, driving licence - assets. Transactions - confirmDiploma, confirmDrivingLicence, erollInUniversity, graduateUni, and events - waitingDiplomaConfirmation and waitingDrivingLicenceConfirmation.

The network administrator is going to add only those participants that are meeting the necessary requirements. The system later can be configured to accept participants that are added from another, highly trusted, participant.

Class diagram will be included here

Participants and assets in the system are designed as:

Person

Has an unique string, personID, that identifies it in the system. In this model the name, age, phone, email and gender are also defining qualities of the person. This participant represents the main end user. He/she can execute the transactions to create their own diploma and driving licence, that then have to be confirmed by the respective participant.

High school

Can be referenced as the administrator of the high school. Since every student will have to know the particular high school's ID. An assumption is made that this will be easily accessible information for the students of the respective school. In the system this entity only confirms the presented document.

Driving school

Can be referenced as the administrator of the driving school. This entity represents those institutions or businesses where one only need to present specific information that is in the form Yes, I have/ No, I do not have. In Bulgaria, in order for a person to start his/her driving lessons, they have to have a certificate for completion of elementary school and to be at least eighteen year of age.

Customs

Can only see all participants and assets. Do not have the possibility to create, change or update any resource.

University

Can be referenced as the administrator of the driving school. The entity represents those institutions that would require more private information. Unlike the driving school, university needs to compare the grade a person have with the one in the conditional offer given. To simulate the common practice, the conditional offers are given to people applied to enroll. So when a person request to enroll in the university, the respective institution will create a transaction. It that will enroll him/her only if that person has a grade equal or more to the one written as part of the transaction.

Having a transaction like that would give some hints about the persons grades, good enough to enroll or not, but ultimately does not provide the exact information. It stays private.

Diploma

For better security and faster transaction flow, the person, student, is creating his/her own diploma and sends it to the high school for confirmation, instead of the other way around. High schools can only see and have permission to update the students diplomas, where the high schools ID has been referenced.

Example : if John Doe creates a diploma and references Williams Highs unique ID, then WH will be able to see John's diploma. However, if John references Wollas High, then there are three possible cases.

First, if there is no such school, the transaction will be rejected.

Second, there is such school and they see that he is not a student, they will decline the confirmation.

Third, they confirm, but is the wrong student. This is a violation and Customs are going to demand explanation. The blockchain can be scanned for the necessary facts of this violation. Then the network admin can remove the asset. If the violation has been intentional, the participant high school may be banished as well.

This is possible due to the permissioned nature of the system. In case of malicious intents in the network, the administrator will remove any participant.

Driving licence

In the system, similar to the creation of diploma, the person is creating his/her driving licence and sending the digital papers to be confirmed. In contrast to the creation of diploma, in order for the person to be able to submit a driving licence transaction, there several checks that must be passed.

In the transaction, the user has to provide his/her own id, his/her diplomas id and a string for id for the licence. When the transaction proposal is submitted, three checks are made, before adding the transaction to the ledger, if passed successfully.

The first is a comparison between the provided personID and the owner of the provided diploma. Simple check to prevent people from trying to certify their documents with another persons certificate.

The second is a comparison between the persons age and respective countrys law. In Bulgaria, the minimum age is eighteen.

The third, and most important, is a check if the diploma is with status confirmed. The importance comes with the fact that it is not needed for the driving school to see all of the information on the diploma or the diploma at all. Thus securing the users private data. The system is guaranteeing the authenticity of the authorized asset. Since all participants are screened before allowing them to enter the system. They are assumed to be trusted parties, until the customs catch a malicious one and throw that party out of the system.

University diploma

6.2 Implementation

6.3 REST Interactions

Chapter 7

Conclusion

7.1 Evaluation

If you do not have a separate chapter on testing, explain here in detail how you went about systematically testing your system. If appropriate, also include end users in your testing. Summarise your main results, and explain how you have advanced the state-of-the-art. Stand back and evaluate what you have achieved and how well you have met the objectives. Evaluate your achievements against the objectives stated in section 1.2. Demonstrate that you have tackled the project in a professional manner.

7.2 Future Work

Elaboration on ”IBM left Composer, because it deviated from Fabric structure”.

Future plans - the same system but constructed solely on Fabric.

Explain any limitations in your results and how things might be improved. Discuss how your work might be developed further. Reflect on your results in isolation and in relation to what others have achieved in the same field. This self-analysis is particularly important. You should give a critical evaluation of what went well, and what might be improved.

7.3 Further readings

The following books and documentation will improve your understanding of blockchain, Hyperledger Fabric and Hyperledger Composer.

- ”Enterprise blockchain development with hyperledger Fabric and Composer” by Ernesto Lee and Sudip Ghosh
- ”Distributed systems: principles and paradigms” by Andrew S. Tanenbaum, Maarten van Steen
- ”Blockchain for dummies” by Manav Gupta

- IBM Red Books series - ”Developing a Blockchain Business Network with Hyperledger Composer using the IBM Blockchain Platform Starter Plan”
- ”Mastering Ethereum: Building Smart Contracts and DApps Book” by Andreas Antonopoulos and Gavin Wood Ph.D.
- Fabric’s documentation - https://hyperledger-fabric.readthedocs.io/en/latest/
- Composer’s documentation - https://hyperledger.github.io/composer/latest/introduction/introduction.html

References

[1] S. Alboaie and D. Cosovan. Private data system enabling self-sovereign storage managed by executable choreographies. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 83–98. Springer, 2017.

[2] C. Allen. The path to self-sovereign identity. *LIFE WITH ALACRITY BLOG (Apr. 25, 2016), http://www. lifewithalacrity. com/2016/04/the-path-to-self-soverereignidentity. html*, 18, 2016.

[3] Altoros staff. A Blockchain-Based Platform for Automating Bond Issuing Worth 10M. [Online]. Available: *https://www.altoros.com/portfolio/blockchain-based-platform-automating-bond-issuing-worth-10m*.

[4] Altoros staff. Technical demo of NSD application with Hyperledger . [Online]. Available: *https://www.youtube.com/watch?v=NfNOT6WmRR4*.

[5] Blockgeeks staff. What is Blockchain Technology? A Step-by-Step Guide For Beginners. [Online]. Available: *https://blockgeeks.com/guides/what-is-blockchain-technology/*.

[6] Blockgeeks staff. What is Cryptocurrency: Everything You Must Need To Know! [Online]. Available: *https://blockgeeks.com/guides/what-is-cryptocurrency/*.

[7] C. Cachin. Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*, volume 310, 2016.

[8] Christina Majaski. Distributed Ledgers Definition. [Online]. Available: *https://www.investopedia.com/terms/d/distributed-ledgers.asp*.

[9] COTI Staff. Ledgers over the years - from ancient Egypt to blockchain, DAG and beyond. [Online]. Available: *https://medium.com/@COTInetwork/ledgers-over-the-years-from-ancient-egypt-to-blockchain-dag-and-beyond-47924175cb97*.

[10] Docker staff. What is a Container. [Online]. Available: *https://www.docker.com/resources/what-container*.

[11] Dr Deepayan Bhowmik. CSCU9YS: Cryptography-I. Lecture from University of Stirling 2018.

[12] V. V. P. A. S. T. V. Elli Androulaki, Angelo De Caro. Blockchain crypto service provider.

[13] Hyperledger Fabric staff. Fabric Architecture. [Online]. Available: *https://hyperledger-fabric.readthedocs.io/en/release-1.4/arch-deep-dive.html*.

[14] Hyperledger Fabric staff. Ledger. [Online]. Available: *https://hyperledger-fabric.readthedocs.io/en/latest/ledger/ledger.html*.

[15] IBM staff. Maersk and IBM Introduce TradeLens Blockchain Shipping Solution. [Online]. Available: *https://newsroom.ibm.com/2018-08-09-Maersk-and-IBM-Introduce-TradeLens-Blockchain-Shipping-Solution*.

[16] B. Li, Y. Wang, P. Shi, H. Chen, and L. Cheng. Fppb: A fast and privacy-preserving method based on the permissioned blockchain for fair transactions in sharing economy. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1368–1373. IEEE, 2018.

[17] A. N. Mencias, D. Dillenberger, P. Novotny, F. Toth, T. Morris, V. Paprotski, J. Dayka, T. Visegrady, B. O'Farrell, J. Lang, et al. An optimized blockchain solution for the ibm z14. *IBM Journal of Research and Development*, 62(2/3):4–1, 2018.

[18] D. C. Mills, K. Wang, B. Malone, A. Ravi, J. Marquardt, A. I. Badev, T. Brezinski, L. Fahy, K. Liao, V. Kargenian, et al. Distributed ledger technology in payments, clearing, and settlement. 2016.

[19] Nolan Bauerle. What is Blockchain Technology? [Online]. Available: *https://www.coindesk.com/information/what-is-blockchain-technology/*.

[20] SecureKey CIO Andre Boysen. How blockchain is changing digital identity. [Online]. Available: *https://www.youtube.com/watch?v=EQ5PGPIjrtI*.

[21] SSL2Buy staff. Symmetric vs. Asymmetric Encryption What are differences? [Online]. Available: *https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences*.

[22] Stack Overflow staff. Developer survey results 2018. [Online]. Available: *https://insights.stackoverflow.com/survey/2018/*.

[23] Verify.Me staff. Website of the company. [Online]. Available: *https://verified.me/*.

Appendix A

User Manual

A.1 Step one

The key software and technology used for the creation and development of this project is:

- OS: Ubuntu 16.04 Xenial 64 bit
- Hyperledger Fabric - modular blockchain framework
- Hyperledger Composer - a tool to create abstract blockchain application, that can then be run on Fabric.
- Docker and Docker Composer
- Visual Studio Code