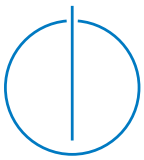
DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Information Systems

Analysis of Use Cases of Blockchain Technology in Legal Transactions

Ulrich Simon Stefan Gallersdörfer



DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Information Systems

Analysis of Use Cases of Blockchain Technology in Legal Transactions

Analyse von Anwendungsfällen der Blockchain Technologie in Rechtsgeschäften

Author:	Ulrich Simon Stefan Gallersdörfer
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	Bernhard Waltl, M. Sc.
Submission Date:	15.05.2017

I confirm that this master's thesis in information systems is my own work and I have documented all sources and material used.

Munich, 15.05.2017 Ulrich Simon Stefan Gallersdörfer

Acknowledgements

To begin with I would like to express my very great appreciation to my advisor Bernhard Waltl who supported me throughout my master thesis. I feel very fortunate to have worked with him in this area of research.

I would like to offer my special thanks to Elena Scepankova for her support.

I also want to thank Prof. Dr. Florian Matthes for his time, feedback and the opportunity to write this thesis at his chair Software Engineering for Business Information Systems (SEBIS).

Furthermore, I want to thank all interview partners for their time and support: Christian Sprecher @ Weblaw.ch, Ronan Sandford @ Etherplay.io, Glynn Bird @ IBM, Kai Jacobs @ SAP, Christoph Möslein @ Pulsar-Tax, Maximilian Möhring @ #Neuland, Marco Barulli @ bernstein.io, Vitus Zeller @ Starwings, Marco Spörl @ Jambit, Matthias Harzheim @ Ventum Consulting, Bernd Lapp @ SwarmCity, and the company BMW.

My special thanks are extended to Christian Sillaber and Daniel Resas for their support.

Abstract

The interest in blockchain technology of enterprises and startups is rising. The technology itself, up today mostly found in cryptocurrencies, promises to be a decentralized platform for storing data or transferring assets preventing any manipulation. The decentralized database cuts out a trusted third party (TTP), guaranteeing the integrity only with its underlying cryptographic promises. While cryptocurrencies clearly benefit from this technology, it is difficult to see the benefits and usages of this technology in other areas of interest. Varying industries are researching the potentials behind blockchain, proposing a range of different use case scenarios.

We give an insight into the technology itself behind cryptocurrencies and explain in detail, how the functionality of Blockchain is established and how it is set up. Upon that knowledge, different views describing the blockchain architecture are created, giving an overview about the technical layers, the roles, and its life cycle. The different views allow users and developers to comprehensively access the technology. Additionally, a blockchain ontology is created, explaining connections between single components within the network.

Furthermore, this thesis provides an overview of different use cases and proposes a topology. In this topology, use cases are classified in categories, showing the potentials of Blockchain technology. Additionally, we give a detailed description of existing parameters for the blockchain, explaining which influence they have on the overall network. With this, a mapping is facilitated between these categories and the different parameters, giving a detailed overview about the blockchain and its potentials. It informs about all varying abilities and enables decision makers to properly find and select use cases within this technology. In interviews with over 15 experts from different companies, an insight is given into the recent developments in this technology and the advancements of it.

Additionally, we prototypically implemented a use case, enabling lawyers to collaboratively create a contract in which all changes are recorded on a Blockchain. Thereby, Blockchain is effectively used to prevent manipulation of content or attribution to authors.

Contents

1. Introduction

1.1. Motivation

> Undoubtedly, its best applications are yet to come.
>
> *Don and Alex Tapscott*
> *about Blockchain Technology* [1].

Enterprises are more and more interested in a new technology called Blockchain which was first presented in 2008 by Satoshi Nakamoto [2]. The investments in Blockchain startups are growing, summing up to a total venture capital of 1.4 billion US$ [3]. Most startups are located in the areas of financial services, payment providers, and exchange platforms. This is hardly surprising, because the first main use case for this technology was the cryptocurrency Bitcoin which still exists until today. Cryptocurrencies experience high attention, because they allow their users to transfer money to other participants without a third party, without exchange rates, and even without stating their real name. Only a small processing fee has to be paid and the transaction is executed within minutes or hours (depending on several factors), even if the transaction happens between people located across the globe. Humans are using the currency for several different reasons: for normal day to day transactions, lack of trust in fiat money[1], for investment (as a part of diversification), but also (because of the pseudonymity) for illegal activities. Despite the usage, cryptocurrencies seem to be regularly used. The market capitalization of Bitcoin is almost about 20 billion US$, the daily volume amounts to roughly 200 million US$ [4].

One of the key features of Blockchain technology is to enable the transfer of digital assets (i.e. digital money) ensuring that the original owner does not keep a copy of it. Thinking of other digital assets like music or e-books, transferring them would mean that the original owner can keep the file. The duplication of assets should not happen, if a *real* transfer (e.g. a sale) shall be executed. Of course, digital money that could be duplicated would not be worth anything. The reason why transfers work for cryptocurrencies is the underlying technology: the Blockchain. It keeps record of all money and effectively prevents double-spending. The Blockchain technology enables people to build cryptocurrencies. To this day, over 600 different kinds of these exist [4].

The Blockchain itself can be thought of as a shared ledger which is distributed over all participating computers and servers, called nodes. The ledger keeps record of all assets stored in it and the history of them. With this, the assets can be traced back to their origin and can only be in possession of one entity. All participating nodes agree on one rule set which defines how the Blockchain is built up. With this rule set, one state of the blockchain can be determined, thus one state of the ledger is specified. An extension of the Blockchain results

[1] Fiat money is an object without inner value. It is used for exchange.

in the possibility to include new transactions in the ledger. Because of this decentralized infrastructure, the rule set, and the state of the ledger, it is not possible to transfer coins whilst keeping them in the original wallet.

With the rising interest in cryptocurrencies, the hype developed not only around the cryptocurrencies but around the underlying Blockchain technology, too. The community around this technology thinks of novel applications. One cannot only save data (the transactions) on the blockchain, but also code. Code stored on the blockchain results in so called Smart contracts. Smart contracts can be imagined as software in the blockchain which can be executed by anyone at any time. Using smart contracts, whole new use cases are possible and the mainly applied use case (cryptocurrencies) is not entirely in the main focus any more. The investment in blockchain technology rises and many companies look into the technology, hoping to find new use cases and to apply them within their enterprises. Different newspapers, blogs, books [1] and more write about the "revolution" of blockchain. Articles about varying use cases and implementations are published, among others in the health division [5], in governmental sectors [6], for land register and many more. The hype around the Blockchain technology arose.

Asking experts in the field about the blockchain, the opinions seem to deviate from the overall impression. For example, Gideon Greenspan, Founder of Coin Sciences Ltd., responsible for the Startup MultiChain, writes in one of his blog articles: "Blockchains are overhyped." [7]. He further says that they get a lot of requests of companies for support with their use cases with blockchain technology. In his article he states following:

> We're waiting to gain a clearer understanding of where blockchains genuinely add value in enterprise IT. [...] a large proportion of these incoming projects have nothing to do with blockchains at all. [...] you need to have a very clear idea of why you are using a blockchain.
>
> *Gideon Greenspan*
> *Founder of Coin Sciences Ltd.* [7]

Many companies want to build applications on top of blockchain, but do not have a deeper understanding of the technology, and even fewer apprehension of a meaningful use in their businesses.

The Blockchain technology is an important development for our society and economy. The key principles of this technology seem promising and the platform provides different key factors. However, we do not know the implications and impacts of this technology and how the Blockchain adds real value apart from existing developments. In this master thesis we want to find out what overall usages are possible, where the technology adds value, where possible risks occur and what requirements are necessary for a successful usage by analyzing use cases from existing companies and startups. However, this thesis does not address platform specific characteristics and details, but remains with overall concepts and properties of blockchain technology. Rather than focusing on a single implementation, we give neutral views and answers about the technology itself. Some aspects of current developments will be covered in the chapter 8, though. We think understanding the elements of Blockchain is crucial to a deeper understanding and development of the technology. Upon that, science and economy is possible to further facilitate concepts, software and products that will lead to a broader usage

of the technology. Revisiting Don & Alex Tapscott and the best applications: We provide a framework for finding them.

1.2. Goal and Research Questions

From the previous statement, the following research questions arise regarding to the subject to be considered.

Q1 What is the structure and the architecture of a standardized blockchain and its network?

For a solid understanding about blockchain and its setup, different architectural views are created in order to answer following sub-questions: In which tiers can the blockchain and its network categorized? What technical structure does the blockchain consist of? What are main components of the blockchain and how do they interact with each other? What are the main roles in the Blockchain network and how do they differ from each other? How does the life cycle of a Blockchain look like? We create five different views for the Blockchain: An architecture overview, a seven-layer application architecture, a Blockchain ontology, roles in the blockchain architecture and a view at the process of the life cycle of Blockchain.

Q2 What are fundamental parameters used in Blockchain technology?

The Blockchain technology is used in many different ways. The basic structure is adapted to the needs of the network and its use case, thus every implementation varies. With changing parameters, some networks have different functionality or features which make them unique or suited for a special use case. We describe possible parameters, their functionality in the Blockchain technology, their possible values and their impact on the overall network.

Q3 What are risks of applying blockchain technology?

The Blockchain itself is exposed to different risks and challenges. If a Blockchain is used, the participants should be aware of these risks and possible malicious behavior which endanger the functionality or integrity of the network. An overview about risks and challenges enable to individually assess risks for given use cases.

Q4 What are categories for the usage of Blockchain technology?

Blockchain is used differently across diverse industry sectors with varying benefits. A categorization for possible applications can help to understand how Blockchain can be used in a proper way and where it can add benefit. Use cases can be classified among these categories to enable the comparison of certain use cases.

Q5 What requirements emerge from the use case being implemented by blockchain technology?

The Blockchain itself has some properties beneficial or adverse for business models or use cases. They may lead to disadvantages which are not known in the first place. To prevent Blockchain technology being used in areas where drawbacks will likely occur, we want to give an overview about the properties and possible pitfalls.

Q6 What are fundamental principles in Blockchain technology?

As stated before, the underlying Blockchain technology used varies among different applications. But not only basic parameters influence the behavior of the network, but the Blockchain itself builds upon different principles which describe certain characteristics and define a specific allocation of parameters. Different principles may introduce new parameters to the system which have to be considered. A selection of some principles can lead to a new configuration of Blockchain, leading to a new application. One can establish links between the use case categorization and the Blockchain principles, leading to an easy approach for designing a Blockchain to a certain need.

1.3. Research Approach

The conducted research is based on a combination of two important research strategies in Information Systems (IS): Design Science Research (DSR) concentrates on creating and evaluating small IT artefacts for local problems, whereas the Grounded Theory Method (GTM) focuses on providing a reliable theory and input to the knowledge base about the field of research. The data basis for the research is based on an extensive literature study and semi structured expert interviews. The framework of DSR and GTM can be seen in figure 1.1 and 1.2.

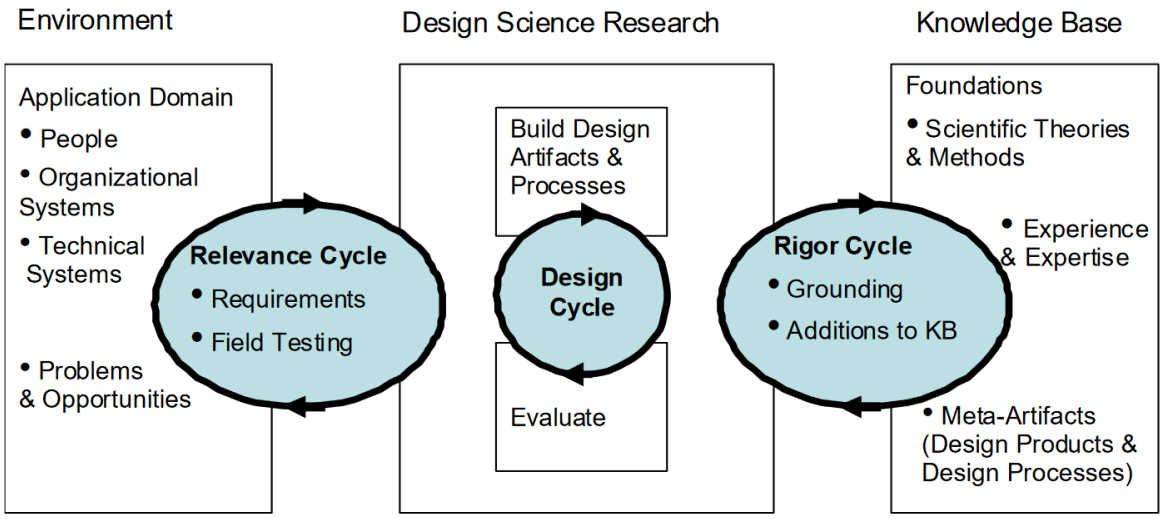


Figure 1.1.: Design Science Research Cycles [8]

1.3.1. Grounded Theory Method

The GTM rivets on the discovery or creation of grounded theory, adding information to the knowledge base. It focuses on the discovery of concepts and categories as well as the relationships between them [10]. In this master thesis, with GTM, main theories about the Blockchain, its structure, its principles and possible use case categories are stated. With data

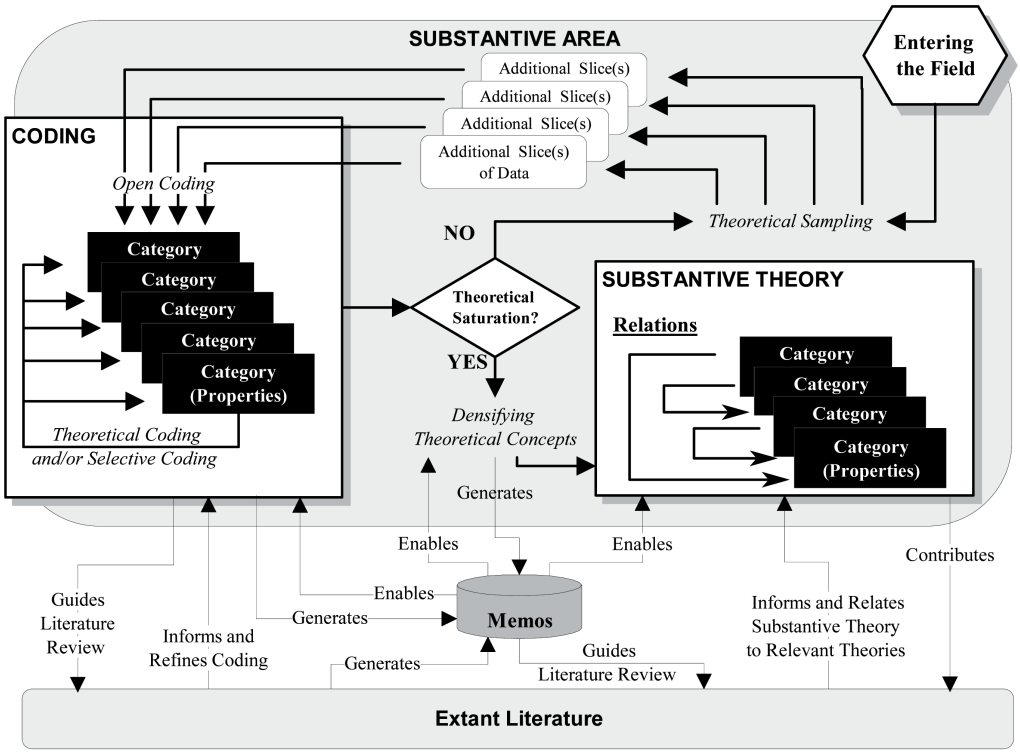


Figure 1.2.: Grounded theory's building process Expanded [9]

from different sources we are able to add significant information about the Blockchain to the knowledge base.

1.3.2. Design Science Research

DSR is a method often used in Information Systems. It focuses on creating and evaluating IT artefacts for specific problems [11]. In this thesis DSR is used as a second step in the process to build upon the reliable theory provided by the first step, GTM. By creating a prototypical implementation of a use case within a Blockchain network, an IT artefact is created.

1.3.3. Semi Structured Expert Interviews

Further, we are conducting interviews with diverse experts from the industry. Only companies or enterprises which have a strong focus on Blockchain technology or are known for doing Blockchain research (e.g. insurance companies) are considered. We also talk to startups with business ideas based on Blockchain technology. To get a further insight into the world of smart contracts, we talk to small projects building upon the Ethereum platform, so called ĐApps [12].

1.4. Outline

The remainder of this papers are structured as follows. Chapter 2 covers the foundations and concepts. This includes the technology overview itself, a definition of the term *use case* and the literature study. In chapter 3 one can find the classification overview, the architecture views and the parameters. In Chapter 4, the results of the interviews are shown and analyzed. The requirements for using the technology are extracted. Further, risks and challenges within the technology are determined. In chapter 5, selected use cases are analysed extensively, and the categorization of use cases is conducted. An overview about the Blockchain principles is given. In chapter 6, further thoughts about the implementation of the prototypical use case are described. Details about the architecture and used technologies, about the data model and the aspects of the implementation are portrayed. At chapter 7 we critically discuss our findings which consists of lessons learned from implementing the technology and a summary of answers to the research questions. In chapter 8 we give a summary of the thesis and an outlook on the possible future developments of Blockchain technology.

2. Foundations

In this chapter we give an overview about the foundations of blockchain technology. We start with the chosen research strategy, give a technology overview, and discuss the reference literature.

2.1. Research Strategy

As explained in the introductory chapter, this thesis builds upon two major IS research methods: Grounded Theory Method (GTM) and Design Science Research (DSR). In this short section we draw an overall picture about the major contents of this master thesis and how they are generated.

In information management [13] four different terms about types of information are used: tokens, data, information and knowledge. We use these terms to describe how the understanding of certain artefacts can be perceived. We leverage this concept to give a clear overview about which artefact is placed in the corresponding category, which method is used to conduct it, and from what prior artefact(s) it is derived. Therefore we structure all our sources, intermediate results and all final outcomes according to these categories. Furthermore, a fourth layer called application comprising implementation and utilization is added. This is required to show the usage of the generated knowledge. The graph can be seen in figure 2.1. All items marked with green dots describe artefacts generated by GTM, all orange dots reference artefacts generated with DSR.

2.2. Terminology

In this section, important terminology in this paper is covered.

2.2.1. Blockchain

To be able to write about and discuss Blockchain, one defines the precise meaning of the terminus "Blockchain". Different terms are widely used: Blockchain (sometimes written blockchain), Block Chain, Blockchain technology, Blockchain network, Blockchain infrastructure, Blockchain (eco)system, Blockchain S.A. (a company from Luxembourg)[1] and many others. Even Satoshi Nakamoto did not write about the term and its correct usage in his paper about Bitcoin [2]. There is a discussion about whether or not the Blockchain is just the data-structure, the

[1] `https://www.blockchain.com/`

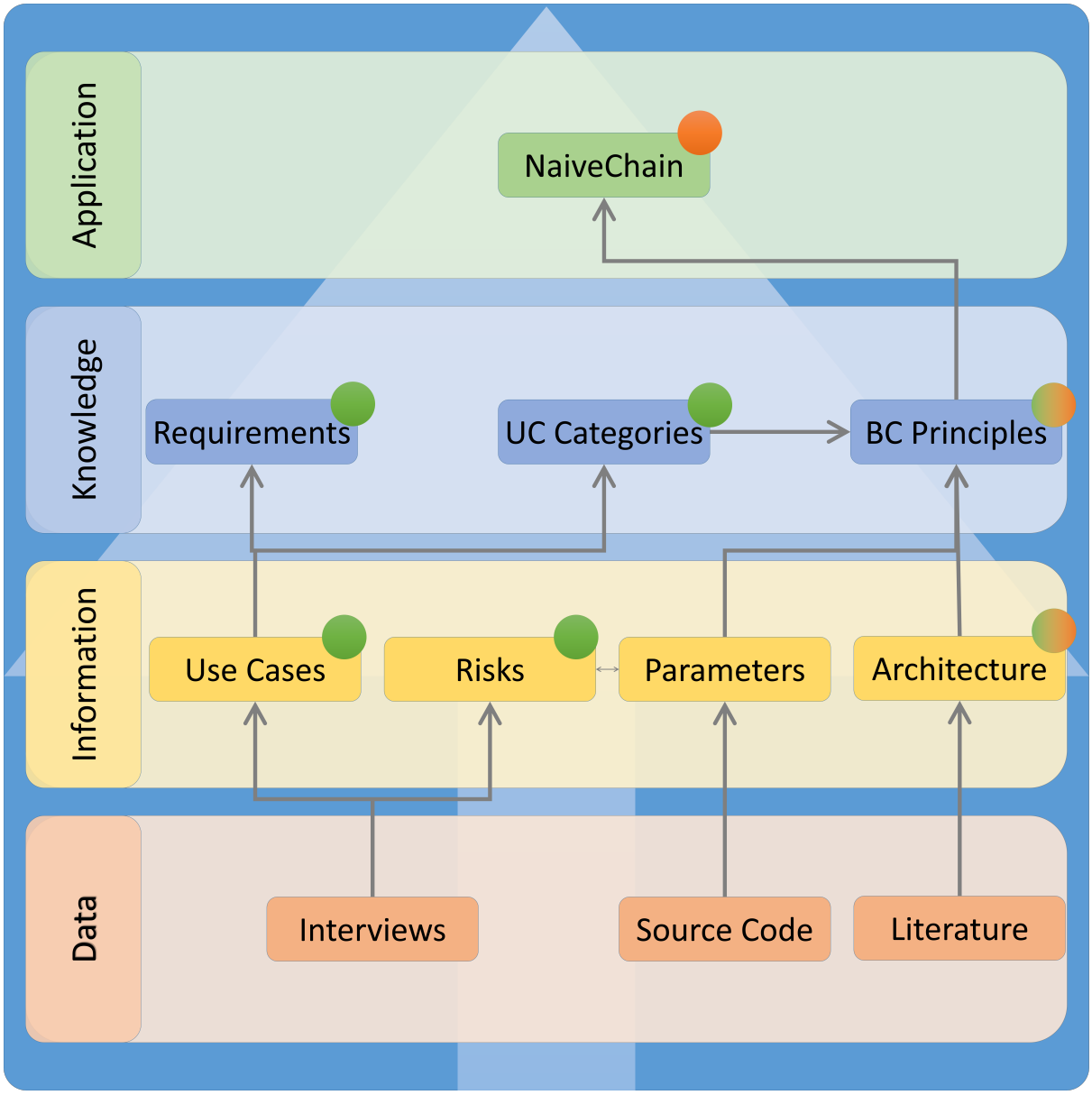


Figure 2.1.: Research Strategy

network itself containing the data-structure, or if it is only the data structure that exists within the Bitcoin network.

Leaving the last suggestion aside, arguments can be found for both sides. We do not want to evaluate the discussion or pick one side, but rather give the reader an overview about the used termini in this thesis and what was meant by using either one. If the term *Blockchain* is used, then it includes the network, the data structure and its contents and code. *Blockchain network* and *Blockchain system* is used interchangeably. If *blockchain* is written in lower case, only the basic data-structure without the network is meant. *Blockchain infrastructure* is the network with its used technology inclusive the *blockchain*, but without applications or contents running within it. The *Blockchain ecosystem* includes the *Blockchain* and surrounding businesses and third party applications that do not depend entirely on the *Blockchain* itself.

2.2.2. Use Cases

In this thesis we conduct a use case analysis. There are many different definitions for the term *use case* [14]. is aware of at least 18 different definitions, but also gives four dimensions in which these definitions differ.

Purpose	What is the purpose of the creation of the use case?
Contents	Are the contents plain prose or are they in a formal notation?
Plurality	Does a use case contain more than one scenario?
Structure	Is a collection of use cases formally structured, informally structured, or unstructured?

Table 2.1.: Dimensions of possible definitions of use cases.

Before applying these dimensions, the goal of the use cases and the use case analysis is described. All use cases are derived from personal interviews with experts of enterprises or startups. We show possible application ideas for Blockchain technology, describe how they work, what stakeholders are involved, and how the process looks. The analysis contains the classification of the use case in proposed architectural views, the use case categories and the used Blockchain principles. The use cases are used to derive information about potential application scenarios and to create the grounded theory. The use case analysis is to classify them into the theory.

We use the definition from [15] which says that *a use case describes an actor trying to achieve a goal by using the system*. The purpose of the creation of use cases is to build requirements, what the underlying technology has to do. Contents are written in plain prose and one use case only contains one scenario. The collection of use cases will be informally structured.

2.2.3. Legal Transactions

Legal Transactions are "*the means by which legal subjects can change the legal positions of themselves or other persons intentionally*" [16]. That means that (juristic) persons are able to create contracts, to terminate contracts, and to create or transfer rights to other legal entities. In this master thesis the described and analysed use cases are always included in some type of a legal

transaction, depending on the actual proposal. Usually, rights for usage are granted or rights are executed on behalf the customer.

2.3. Blockchain Technology

A blockchain can be viewed as a decentralized database in which information can be stored. This database is distributed across all participating nodes, resulting in a decentralized network. All nodes agree upon a certain set of rules, defining the allowed behaviour in the network and the structure of information to be stored. The rule set defines the purpose and the functionality of the Blockchain. In this section about the basics of Blockchain technology, most of the information is derived from [17] unless stated otherwise.

The Blockchain is designed that all stored contents are immutable. Every piece of information that was stored on the Blockchain can only be altered or deleted with tremendous expenditures (though depending on the size and integrity of the network) [18]. The immutability allows to create a ledger which keeps track of certain assets. With the rules enforced by the network, it is guaranteed that assets are not created at will and cannot be duplicated. This distributed ledger enables the creation and use of cryptocurrencies. Thus Blockchains are often synonymously referred as distributed ledger technology (DLT) for financial or governmental sectors [19].

A Blockchain can be used in different ways. These usages can be differentiated by the user group supervising the network. There are three types [20]: Decentralized, meaning the Blockchain is governed by everyone who participates; Hybrid or permissioned, meaning the supervisors are preselected; or centralized, thus only one entity sets the rules of the Blockchain. Additionally, the Blockchain can be public or private. Public means that anyone can access the network and read information from it and private means that the access to the chain is restricted. These different access restrictions of Blockchain result in different technical approaches, including some technique leaving out another. We give an overview about a common Blockchain used for cryptocurrencies, but do not stick to a specific implementation.

At first, cryptographic foundations are covered. Then, the structure and single elements of the Blockchain are covered, explaining how it is stored on one single node. After that, the network, its setup and its behaviour for ensuring the integrity are explained. Additionally, the method how new information is put into the network is introduced. In the next step, applications within the Blockchain are described. At last, Blockchain-typical phenomenons are shown.

2.3.1. Cryptographic Foundations

To fully understand Blockchain and its surrounding technologies, we first have to explain two basic concepts of cryptography and information technology: Hash functions and public-private cryptography.

2.3.1.1. Hash Functions

Hash functions are the foundation of Blockchain architecture. Hash functions allow to generate some sort of fingerprint of data. The generated fingerprint will not change if the data does not change, but if the data is altered, the fingerprint will change. Basically, hash functions take an input of arbitrary length and translate it to a string with fixed length. If the hash of a file is kept secure, the integrity of the file can be proven. Thereby these functions need to fulfil three properties, to be used with Blockchain technology. The first one is called **collision resistance**. It makes sure, that it is very unlikely that two random inputs generate the same output.

The collision resistance of a hash function is defined following:

$$\text{It is infeasible to find two inputs x and y s.t.: } x \neq y \text{ and } H(x) = H(y). \tag{2.1}$$

It is not guaranteed that no collision occurs, because there is no way to design a hash-function which fixed output length and arbitrary input length. The input space is larger than the output state, inevitably resulting in several different inputs sharing the same output.

The second property is called **Information Hiding**. Given the output of a hash function, there should be no way to retrieve the input. It guarantees that, if we are supplied with the output of $H(r||x)$ (r is a random number from a high min-entropy), we cannot efficiently compute x.

The hiding property of a hash function is defined as follows:

$$\begin{gathered}\text{It is infeasible to find } x\text{, given only } H(r||x),\\ \text{when } r \text{ is a secret value chosen from a high min-distribution.}\end{gathered} \tag{2.2}$$

This allows for an interesting application. You can commit to a secret value r by publishing the hash h of r. Later on, to prove that you were in possession of that value r, you can publish it. Anyone can check if $h = H(r)$.

The third property is **Puzzle Friendliness**. It is not obvious what this property is useful for, but it is later required for extending the Blockchain with additional functionality. It means that the output of the hash-function is uniformly distributed. Given a random input, every possible output has the same likelihood to be "hit".

The puzzle friendliness of a hash function is defined following:

$$\begin{gathered}\text{It is infeasible for every possible n-bit output value } y\text{, if } k \text{ is chosen from a high min entropy,}\\ \text{to find } x \text{ such that } H(k||x) = y \text{ in time significantly less than } 2n.\end{gathered} \tag{2.3}$$

This enables an application called *Search puzzle*. Almost every cryptocurrency uses a Search Puzzle. It works as follows: A target-area T of the output-area of a hash-function is selected. After that, every participant need to find a value k to add to a message x, such that $h = H(k||x)$ is within the range of the target area. Depending on how small the target area is, it takes more or less attempts to find k that will alter h in such way, that $h \in T$. For example, if the target area is 1% of the whole output-area, one will have to calculate 100 hashes on average to find such k. In the context of Blockchain, instead of using search puzzle, the term mining puzzle is used.

2.3.1.2. Public-Private Cryptography

Public-Private cryptography is used frequently. For example, every SSL encrypted connection relies of some sort of asymmetric cryptography. Unlike symmetric cryptography, asymmetric cryptography uses one key to encrypt data and another key to decrypt data. If certain conditions are met, it can be used not only to encrypt data, but also to sign data. To prove that someone is in possession of the private key, he has to "encrypt" some data with his private key and everyone can decrypt the message with the associated public key to check its sender's legitimacy. In Blockchain technology, this signing mechanism is used for authorizing transactions.

2.3.2. Structure of a Blockchain

After explaining cryptographic foundations, we give an insight into the structure of the blockchain.

2.3.2.1. Blockchain

The blockchain is the central object of the network. It is distributed among every client and is on every honest node identical. How these Blockchains are kept synchronous is explained in 2.3.3. The Blockchain contains many single blocks which are connected to each other. The first block in a Blockchain is called *Genesis Block*. It is defined by the developer of the Blockchain and can be filled with arbitrary data. Every node must have the same Genesis Block. Deviant nodes would not participate in the network. Each block has a height h and contains the hash of the previous block $h-1$. If any information is altered, the renewed computation of the Blockchain (while keeping the latest hash) will reveal the compromise. Therefore, if the hash of the last block of the Blockchain is kept, one can guarantee that any other node with the same hash has the correct Blockchain.

2.3.2.2. Block

A block is data which contains two objects: The block-header and a tree of transactions. The header contains everything important about the block: The version, the hash of the previous block, a timestamp, the hash of the tree and information about the search puzzle. The tree contains every transaction that the miner wants to include. Usually, every transaction (up to the block size limit) which is new in the network is included. In most cryptocurrencies (like Bitcoin and many Altcoins[2]) the tree for information storage is a Merkle-tree [2], as most altcoins base on the code of Bitcoin. It is easier to search for transactions within a Merkle-tree than other data structures [21]. To build a tree, two transactions are concatenated and hashed. The results of each two concatenations are again connected and hashed, until there is only one hash left. The last hash is called the Merkle-root-hash and stored in the block-header.

[2]Altcoins are alternative coins with a similar functionality to Bitcoin.

2.3.2.3. Transaction

Before a transaction is included in a block, it is send to all connecting nodes, to increase the probability to be included in the successive node. If someone wants to transfer some coins to another participant, he has to be in control of unspent transactions. Transactions are either completely spent or unspent. It is not possible spend only a part of a transaction. If it was possible, every miner had to check the complete blockchain for partially unspent transactions. To save time, everything has to be transferred. Funds that go beyond the granted amount are transferred back to the own “wallet”, issuing a new unspent transaction. There is no implementation of wallets in the blockchain of current cryptocurrencies[3], but one labels a collection of private keys "wallet". One can leave out wallet instances, as transactions already cover the functionality of it: Every transaction has an output which binds the amount of the output to a public signature. Only the participant with the matching private key to this public signature is possible to spend the output later on. Technically speaking, each output and input is written in a script-language. This means, only if $<$ code input $>$ $||$ $<$ code output $>$ ($||$ is concatenation) returns the value TRUE, the new transaction is valid. Therefore, there is no element as a wallet[4] in regular blockchains, because it is only the possession of a private key which grants you to use unspent transactions. This term is only used to make the system easier to understand. Transactions can consist of n inputs and m outputs $(n, m >= 1)$. This makes it possible to combine unspent transactions together into one transaction. This can be useful, if one transaction does not contain sufficient money to pay for an item. Notice, that it is not possible to trace every individual coin, but only transactions. With three inputs and two outputs, no matter of the amount, it is not possible to say which input did go to which output. Concerning the numbers, the sum of all inputs has to be larger than or equal to the sum of all outputs. This is to prevent creation of coins out of thin air. The difference between inputs and outputs are considered transaction fees which are paid to the miner who proposes the certain block. Even if coins cannot be traced per se, it is possible to trace transactions and owners as payments with two inputs from two different private keys are not very common to originate from different entities. This means that most transactions with two inputs belong to the same entity. Therefore, joined payments link two public keys together which means they belong to the same entity.

2.3.3. Network

After explaining the blockchain setup of a single node, we discuss how the network ensures the integrity of the network and adds new information to the chain. We describe the way of consensus of a decentralized Blockchain. We do not go into further details of centralized or hybrid Blockchains, as the applying consensus rules are basic: only blocks signed by a pre-defined authority are allowed.

[3]Wallet software exist for every cryptocurrency, but is not implemented in the core.
[4]There are wallet-based approaches, e.g. in Ethereum.

2.3.3.1. Consensus

Consensus plays a vital role in the Blockchain network. The goal is to achieve a joint state, meaning everyone agreeing on a certain state of the Blockchain. As in a distributed system, there is no central authority to decide which new blocks are valid and which are not. Every node has to decide on its own if it accepts a new block or not. The basic consensus algorithm is simple: The only valid chain is the chain that (1) contains the genesis block, (2) contains only valid blocks according to the network rules and (3) is the longest one. All other chains are not accepted. Longest chain means that the chain contains the most blocks.

From the research about the Byzantine Generals' Problem [22] we know that a distributed system can only work if at least 51% are honest nodes that want to participate in the network. The main idea is a process to choose one single node randomly from all participating nodes. This node is allowed to propose the new block. The possibility of a Sybil Attack[5] should be eliminated by this, as well. A method to choose randomly a node which *is* vulnerable to a Sybil Attack, is fairly easy to devise. Think of a tombola in which every participating node puts its ID in. A number is randomly chosen and this number is allowed to propose the next block. This would be easy to manipulate, because an attacker could create an arbitrary number of IDs and submit them all to the tombola. Creating new IDs is very easy which leads to the conclusion that creating an arbitrary amount of IDs is easy, too. The method has to be designed such that an attacker is not able to increase their chance without significant investments.

2.3.3.2. Mining

Assuming that the blockchain is static and no blocks could be added, the system would know which chain is the only valid one. The process of adding new blocks to the network is called mining. Mining is the essential process in all blockchain-based technologies. This process proposes new blocks to be added onto the blockchain. This area covers different topics, starting from the process itself, the puzzle to the mining rewards.

2.3.3.3. Process

Building new blocks is – from a technical point of view – fairly easy. All new transactions which are announced in the network are validated, stored and built to a transaction tree. Everything else, for example meta information, is stored, too. The hard part of creating a block is to find the solution to a mathematical puzzle. The details of this puzzle are covered later on. If a solution to this puzzle has been found, it is included into the block as well. After that, the new block is announced in the network. Other miners will validate the new block and, if everything is fine, they will include the new block into their local storage and will try to find new blocks on top of the recently announced block. Afterwards, the process starts again. If two blocks are proposed simultaneously, the rules number (1) - (3) stated in the consensus section do not seem to produce one single valid chain anymore. This means that there are two versions of the blockchain which are equally accepted, because they have the same length. For the moment, the mining-community splits because of a fourth rule, a so called fork. Rule (4)

[5] An attacker should not be able to increase his chances of winning by only generating many nodes

says that in case of a fork the blockchain of which the node heard first is the valid one. An example: Two nodes find simultaneously two Blocks, Block A and Block B. Everybody who heard of Block A first will try to find a new block on top of A. All others who heard first of Block B will mine on top of that one. The fork is resolved at that moment, when somebody finds an additional block on top of A or B. At that moment, the fork is resolved because all other nodes that worked on top of the "wrong" block, will disregard their current chain and continue working on the longest chain. The losing block is left alone, won't be considered anymore and is called an "orphan block" . Additional, the likelihood of finding two blocks at the same time depends on certain parameters, but is usually negligible [6].

2.3.3.4. Mining Puzzle

Knowing how new blocks are published in the network, the method of how to choose a random node of the network is shown. The Mining puzzle ensures that blocks are proposed by a random node in a pre-defined period of time (For example: On average, Bitcoin proposes every 10 minutes a new block). To ensure this period of time, the difficulty of the puzzle is adapted to the power of the whole network regularly (Bitcoin recalculates the difficulty every 2016 blocks). In between these recalculations the difficulty stays the same. There are several different ways this puzzle can look like, the two most known are described here.

2.3.3.5. Proof of Work

There are several ways the puzzle works. One of is the so called "Proof of Work" (PoW). To propose a new block, the hash of the block concatenated with a random value (called "nonce") has to meet a certain criterion. It has to be in a specific target-space which is a subset of the output-space. The complete output-space of a hash function can be represented as a number. The assumed hash-function is SHA-256. The output-space of SHA is 256 bits, therefore 32 byte or 64 hexadecimal numbers. The output-space ranges from $\overbrace{00...00}^{64x}$ to $\overbrace{FF...FF}^{64x}$ and the network agrees to a threshold for creating a new block. Every new block has to be below this threshold, to be accepted in the network. For example, in Bitcoin (in the year 2015) this number is so small, that only $1/(10^{20})$ blocks will actually meet this criteria. Mathematically speaking, $H(Nonce||Block) < Threshold$ must be fulfilled. If the function fulfils the third criterion (puzzle friendliness), one has to brute-force numbers.

Note that this can be considered as a huge waste of energy, because the functionality of the Blockchain does not depend on computation power. No matter if server farms are mining or just a mobile phone, the contents of the resulting blockchain would be (almost) the same. This huge investment in mining power ensures that no single participant could bring up more computation power than the rest of the network. A small device cannot guarantee this, but a big network can. A blockchain built by a mobile phone would be much easier to recalculate and to outrun (make a longer blockchain) than a blockchain which is built by massive mining power. This leads to a more secure blockchain, but it remains an open question if there are better alternatives.

[6] `https://blockchain.info/de/charts/n-orphaned-blocks`

2.3.3.6. Proof of Stake

Another set of methods for proposing new blocks is Proof of Stake. This approach can be divided up into three areas: "Proof of Work/Stake combination", "Proof of Stake" and "Proof of Deposit". The idea behind all of the methods is that the control over parts of the currency makes it easier to propose new blocks. The Proof of Work/Proof of Stake combination works as follows: Traditional Proof of Work can be used, but the puzzle to solve becomes easier if some "old" coins are used. Usage means that the age of the coins becomes reset. The older the coins, the easier the mining puzzle gets. A possible implementation is the cryptocurrency Peercoin. Proof of stake is the purest form. Nodes which control large amounts of the currency are given the easiest puzzle. In this case, age of coins is not taken into account. Proof of deposit requires the miner to freeze the coins they used for a certain amount of time. This can be seen as an opportunity cost as one is not able to spend the coins. These approaches have two major benefits compared to Proof of Work: Firstly, the footprint and the waste of energy are much lower. No miner gets a significant advantage of running big server farms for solving puzzles. The second advantage is that the nodes which have the most money in the system are the nodes which are most interested in the health of the currency and will likely behave honest. Nonetheless, this approach has a significant drawback. Firstly, it grows into a kind of centralization, because the miners with the most money in the system will likely get more money than everyone else. The second drawback is that attacks on the networks are cheap. Thinking of "Proof of Work", if someone wants to build a fork, the costs for mining are wasted if the attack fails. With Proof of Stake, if the attack fails, the coin remains unspent meaning the age is not reset, therefore the attacker did not lose much energy or money as he would in proof of Work. That said, Proof of Stake has not received so much scientific attention, because Bitcoin implemented Proof of Work and has a dominant role in cryptocurrencies. Nonetheless, Ethereum has announced to switch to Proof of Stake in summer of 2017 [23].

2.3.4. Applications

As we understand the setup, the network and the consensus of Blockchain technology, we look more deeply into applications in the network.

2.3.4.1. Scripts

As we see in the transaction section, we actually do not transfer money to wallets, but to scripts. A new input script has to return TRUE with a previous output-script, otherwise it is rejected by the miners. We add an arbitrary number of inputs and outputs, allowing interesting applications.

2.3.4.2. Escrow Payments

An escrow payment is typically used if two participants want to trade a certain value against a good. Usually, this is done using a third party which keeps the value and the good by himself until both of them are received by the third party. If everything is fine, the third party gives

the participants the other object. The participants who want to trade do not have to trust each other, but they have to trust the third party. In cryptocurrencies it is the same, the involved third party has to be trusted. Nonetheless, Cryptocurrencies have an advantage compared to traditional payments: The third party is only involved, if there is a dispute between creditor and debitor. If not, the participants can execute the transfer of the money without the escrow. Think of following scenario: Alice wants to trade a mug of Bob for one coin. The third party Charlie is involved in this transaction in case anything goes wrong. A multi-party transaction output is created which requires a two out of three signed input transaction to redeem. If everything is fine with the trade, Alice and Bob can sign a transaction to transfer the money to Bob. If there is any problem and Alice and Bob cannot agree, Charlie has to resolve this dispute by investigating and signing a transaction, either back to Alice or to Bob.

2.3.4.3. Smart Contracts

Smart Contracts are a highly interesting possibility to put computable contracts into the blockchain. The technology itself is not new, because software which transfers money as a reaction to certain events (stock prices, etc.) are widely spread amongst major banks and companies. Every system for high speed stock trading can be viewed as "computable contract". The advantage of implementing such contracts on a blockchain is that the execution of the contract is guaranteed and independent of any node. For example, it is possible that the earnings of a company are split equally amongst the founders every night. Unfortunately, the scripting language of most cryptocurencies (like Bitcoin and many altcoins) is rather limited. To build an arbitrary script, a Turing-complete programming language is needed. As mentioned in the reference literature, the cryptocurrency Ethereum supports smart contracts. It allows the user to create whatever he wants and let it be executed on the blockchain. The possibilities are almost endless, it is possible to bet, to build prediction markets, play a game of chess and many more. Nevertheless, the created software should not be highly complex, because the execution of contracts is not free of charge. Each execution step costs a certain amount of money, therefore large applications are not suitable to be stored on a blockchain. If a contract is stored on the blockchain, it is executed until the funds of this contract runs out. Therefore, it prevents users from building unnecessary large scripts or infinite loops.

2.3.4.4. Decentralized Autonomous Organizations

Thinking smart contracts further, it is possible to build so called decentralized autonomous organizations (short: DAO). The idea behind them is that organization-like structures can be built in the blockchain. It begins with the idea, that people can participate in a DAO by buying their shares in a ICO (initial coin offering). With that, they gain vote rights and can participate on the decision making within this construct. This organization can name people that act as CEO or developers and get paid out of it. These structures could be seen as individual legal entities, but will depend on the respective laws. Despite legal uncertainties, this seems to be a promising approach in Blockchain technology.

2.3.5. Blockchain Specific Phenomenons

After the introduction to Blockchain technology, all basic concepts and components are known. Additionally, one should keep in mind some special behaviour of this technology.

2.3.5.1. Trust in the System

The Blockchain is often called a trust-less system, because a user does not need to trust other participants. If he wants to use smart contracts or receive some coins, he can be sure that the system will behave the way it is developed. This is basically true, but only under the condition that the network behaves entirely correct. The user has to trust the developers of the system that their software has no major flaws and he has to trust the miners to keep up the integrity of the network. If malicious forces act in the network, the functionality of the network cannot be taken for granted, for example, if a hack appears and the underlying software needs to be upgraded which leads to the next phenomenon.

2.3.5.2. Forks in the Blockchain

If the underlying software is changed in any way, a fork happens in the Blockchain. Forks happen during the process of mining (and are resolved within minutes to hours), but they also happen if the rule set of the Blockchain gets an update. Two sorts of fork can happen: a soft-fork and a hard-fork. The difference between these two forks is, that they differ in whether the rules for validity of a single block are expanded or restricted. Additionally, they differ in the consequences for the network.

A soft-fork occurs, if the rules are more limited than before. For example, the rules could say that only transactions with under 10 coins are allowed. The new software only accepts blocks that have this limitation, the old software does not care whether the transactions have this limitation or not. If the majority of the network uses the new software, the change will be adopted by all nodes and only one blockchain will exist, because the old software will always change back to the longest blockchain which is maintained by the new software. If the majority of the network uses the old software, two chains will exist until the new software wins the majority.

A hard-fork occurs, if rules are expanded. For example, the new software allows additional content in the block. In this case it is reversed: The new software accepts all blocks, but the old software only accepts its own blocks. If the majority of the network accepts the new software, a fork will definitely occur, otherwise it is possible that only one chain exists.

2.3.5.3. Scalability of the Blockchain Network

Independent of the count of nodes or the used computational power, the speed and the throughput of the system stays the same and is only defined by its design. The problem is that every node has to do the same computational work, because everybody needs to be sure that all the blocks he received are valid and the results from this computation match

the proposed transactions or datasets. Therefore, the machine produces the same results, no matter how many nodes are connected. This is a major difference compared to traditional database systems which are much more faster and scale with their used resources. There are some advances in the area of Blockchain scalability [24], but all approaches have some side effects on the technology itself, meaning that some functionality of the original Blockchain draft is lost.

2.3.6. Blockchain as a Platform for Digital Twins

A digital twin is a digital reflection of a physical object. The properties of the object are represented in the digital twin; sensors are often used to keep the image up to date and accurate, such that one can rely on the data of the digital twin. The digital representatives are used for different purposes, such as 3D modeling, monitoring, or diagnostics.

A blockchain offers a platform for digital twins. As some blockchains not only represent coins but other assets, these assets become a representative of real-world objects. As the Blockchain technology does not provide the possibility to update information in real-time (only with a delay), the Blockchain is not suited for real-time applications. Whether or not real-time applications are used, the situation for digital twins in the Blockchain remains difficult. Considering models of (industry-) machines: The physical machine is in secure location (as no unauthorized personnel is allowed to access it) and the connection between the sensors measuring the machine and the server computing the model is secure, as it is in a secure network. Thus, every participant can be trusted: the sensors, the network, and the server. However, what now about Blockchain? Considering the case, using the Blockchain does not bring any advantages, as there is no possible trust lost (no one will tamper with the data). Therefore we need a scenario more suitable for blockchains. For the sake of the example, we consider a more visionary case in which Blockchain technology is proposed: Trading cars over the Blockchain or proof of possession of vehicles. How would this look like? Of course, for every car there has to be a digital twin in the Blockchain. The ownership of the digital twin represents the ownership of the car. To claim the ownership, the car would have to be connected to the Blockchain actually to be able to validate for ownership. The owner of the digital twin would use his private key to proof the ownership and the car would open up and allow the start of the engine. But how actually would the linkage between the Blockchain and the car look like? Is the digital twin in the blockchain tied to some serial number in the car or does the car manufacturer just create the digital twin and hand it over to the buyer? Not only technical questions arise in the previous scenario, but also practical and even legal questions. Gets it easier to steal cars as only the private key has to be obtained? Is it possible to replace the "blockchain-module" in the car for theft? Is it sufficient to proof possession via Blockchain?

As we see, the answers to these questions are not trivial, in fact, become more complex as more information is stored about the physical object. If the physical object changes, how does one transmit the "status update" into the Blockchain? How is it ensured that the data inside the Blockchain represents the data of the real object? The problem is that the Blockchain cannot ensure the data that is coming from "offchain" and the danger exists that wrong data is recorded inside it. The creation of bonds between the digital twin and its real twin remain an open question in the research area of Blockchain technology.

2.4. Reference Literature

Some relevant literature and approaches cab be found with respect to various topics. In this section, literature are categorized, among with their possible advantages and drawbacks.

2.4.1. Architecture

Literature about the Blockchain or Bitcoin architecture can hardly be found. in *Towards Reference Architecture for Cryptocurrencies: Bitcoin Architectural Analysis* [25] Israa Alqassem gives a brief introduction about Blockchain technology, proposes a Blockchain Transaction Domain Model, in which main components of Bitcoin software are shown. It gives an insight on how the software is built up and what components it consists of, but does not give a high level overview about the network and its components, but only examines a single client. An architecture is needed that does not only consider cryptocurrencies, but the blockchain as a whole system.

In a very recent paper, *Towards a Blockchain Ontology* [20], Joost de Kruijff proposes different ontologies for blockchains and blockchain transactions. On three different layers, a datalogical layer, an infological layer and an essential layer he proposes each an ontology for blockchain and transactions. Additionally, he compares the blockchain with traditional transaction systems on each layer and emphasizes the differences between them. However, an infological ontology for Blockchains is missing.

Regarding the processes of Bitcoin and Blockchain more graphical representations can be found. For example, Joshua J. Romero et al. [26] provide a clear picture of the process to create transactions and how they are validated inside the network. However, they only consider one functionality used in the network, but do not give an insight into the different process stages of Blockchain technology.

2.4.2. Parameters

Information on manipulable parameters and their implications regarding Blockchains can hardly be found, as it might not be relevant to the users, as they cannot be changed anyway. Only for creation a new chain or forking an existing one, Blockchain parameters become relevant.

Multichain exploits the first scenario (creating new chains): They provide an infrastructure which allows the creation and execution of single Blockchains by design and pre-defined parameters. Customers of Multichain can select relevant parameters and start a new chain with the previously chosen configuration. For the sake of documentation, they provide a list of all configurable settings of a chain [27]. However, they only describe the parameters, but do not provide additional information about implications. As they only describe their own platform, the parameters cannot be generalized, but give quite a good insight into possible parameters.

2.4.3. Use Case Categorization

Fraunhofer FIT gives a good overview [28] about the recent approaches for classifications of Blockchain applications. They apply the categorization of William Mougayar [29] on the status quo of the blockchain startup market, conducted by AngelList [30]. Mougayar proposes a categorization in four different categories: Infrastructure and Plattforms, Middleware Services, Applications and supplementary works. The advantage of these classes is that they can be separated very well from each other, giving the opportunity to classify them easily. But the problem with this approach is that it only classifies amongst the layer of the architecture of Blockchain and does not consider the actual purpose of the Blockchain. It only gives an overview about the market and its developments, but does not help to understand what use cases or capabilities the Blockchain has.

In its paper *Robust, Cost-effective Applications Key to Unlocking Blockchain's Potential Credit Benefits* [31] Moodys gives an overview of 25 generalized use cases for Blockchain technology. They only categorize them in four different classes and sort them after usage in an industry sector. Financial Institutions, Corporates, Governments and Cross-industry are named. They only give a very high level view about these use cases. A categorization amongst function of the Blockchain needs to follow.

2.4.4. Frameworks

For a better understanding of this technology, three different frameworks are considered which are briefly described. Bitcoin, Ethereum and ZCash.

Bitcoin [2] is the first and most frequently used cryptocurrency to this day. It introduced the main concept of Blockchain and helped to draw the attention to both topics. It uses a simple Proof of Work mechanism which generates new coins within the system. The amount of coins is limited to 21 million coins, thus no inflation can occur. The used language is Bitcoin-script which is very rudimentary and only provides basic functionality such as single payments or multi-party payments, so called escrows. Besides that, Bitcoin is sometimes used to store small information, but was not designed for that.

Ethereum is a cryptocurrency [32], too. The major difference to Bitcoin is that Ethereums selected a Turing-complete language, which means that any program can be executed on it. With that, it introduced the concept of smart contracts, allowing to provide services or to build DAOs. It also implements another Proof of Work mechanism which will be replaced in the future with Proof of Stake [23]. The average blocksize is much smaller, additionally transactions are stored in another way than in Bitcoin. After an attack in 2016 [33] and a successive hard-fork, the currency split up into Ethereum and Ethereum Classic. Ethereum had some flaws which were fixed. Ethereum Classic kept the old code with the potential bug.

ZCash [34] is a novel approach for a cryptocurrency. Its main feature is that transactions are not pseudonymous, but completely anonymous. It uses a zero-knowledge-proof in order to keep all transactions a secret. It is interesting, because it is the first currency which allows completely anonymous behaviour in a currency.

An overview about all reference literature can be viewed in table 2.2.

Area/Author	Description
Architecture	
Alqassem and Svetinovic	Main components and structure of Bitcoin implementation, does not provide a high-level overview.
de Kruijff and Weigand	Proposes on three different layers a Blockchain ontology, comparison between Blockchain and traditional transaction systems.
Joshua J. Romero	Insights into transaction- and block-creation of cryptocurrencies.
Parameters	
Multichain.com	Describes possible starting parameters for Blockchains in its own, Bitcoin-like ecosystem.
Use Case Categorization	
Mougayar	Categorization into four vertical categories. Does not give an insight into possible applications.
Williams	Overview about different use cases ordered by industry sectors.
Frameworks	
Nakamoto	Bitcoin, first and most used cryptocurrency. Does not provide much additional functionality besides transactions.
Wood	Ethereum, first cryptocurrency which supports smart contracts and many other features.
Sasson et al.	ZCash, truly anonymized cryptocurrency, implemented via zero-knowledge proof.

Table 2.2.: Reference Literature

3. Blockchain Classification

3.1. Architectural Approach

As shown in the previous chapter, Blockchain technology consists of complex mechanisms. Generally speaking, there are only few visual representations of the Blockchain, its structure and its processes. It is important to comprehend the setup of the technology, because one can only understand many properties of the system if the underlying theory is understood.

We therefore build five distinct views on the Blockchain. Analogously to typical Enterprise Architecture Views [35] we want to give a high level overview about the technology, not an insight into the details of the low level implementation of a client. We think that the Blockchain infrastructure is built similar to Enterprise Architecture Models. The Blockchain has to be described in a *basic* version, as we want to prevent influences from certain implementations.

With these views we are able to categorize and better describe the use cases later on, because different use cases involve different tiers of the platform, different roles and different processes. Therefore, a neutral view is needed.

3.1.1. Analysis of Existing Components

To further derive architectural views, one has to look more closely at the application infrastructure of Blockchain. In chapter 2 we described the basic functionality, the scripting within the network and the network itself were described. In the foundation part we intentionally left out those applications that access the Blockchain and utilize it in order to provide some further services. These applications do not belong to the Blockchain directly, because they are not crucial to the technology or its functionality, but belong to the eco-system. Therefore there are three major areas in which applications or software are used in the network. First, the software that runs on the nodes and establishes the network and its integrity itself. Second, the software that runs within the blockchain, so every script, every smart contract would belong to it. And third, applications that utilize the Blockchain for service provisioning. From a high level architectural point of view, it is possible to further split up two of these areas. First, one has to differentiate between software that provides access and software that utilizes the access to the blockchain. An access provider, for example, would offer APIs for communicating with the network or would provide the storage and administration of private keys. Software utilizing the access add value outside the blockchain ecosystem that only arises due to the integration. Software outside the Blockchain ecosystem generates profit from accessing the blockchain technology. The second distinction we make is between the network itself and the software that runs within the network. The design of the network, public or private, protocol and size do not influence the functionality of the software establishing the blockchain. Yet, it

influences the possible usages of the overall system, but the node itself is not dependent on the fashion of the underlying network.

With these five tiers we create the first architectural view which are shown in the next section.

3.2. Different Views on Blockchain

3.2.1. Blockchain Architecture Overview

The aim of the first view is to provide a high level overview over the whole architecture of Blockchain. It enables to see where abstractions occur and where specific elements are located in the overall view. The view is depicted in figure 3.1.

It has two axes: The *architecture scope*, from high level domain specific to low level technical, and the *architecture zoom*, from largest to smallest granularity. The architecture consists of five tiers in the y-axis: “Business”, “Business Service”, “Application”, “Infrastructure Service” and “Infrastructure”. These tiers match the description in 3.1.1. Each tier is split into three columns, describing the granularity from high to low. We describe each tier and explain why we categorize some aspects in it. Starting with the infrastructure: This tier contains the hardware and the network on which the Blockchain software is deployed and executed. The network itself is needed for communication and integrity assurance. The layout of the network (centralized or decentralized) is defined here. In the network of a Blockchain, different stakeholders use different hardware, depending on the configuration of the system: Miners or nodes use varying hardware and have alternative strategies to best fit their needs. A network also has to agree on several variables: Algorithms like hash functions or public-private cryptography have to be defined upfront to provide a functional network. There are more variables which the network agrees upon which we leave out for the sake of clarity and overview. Blockchain parameters are discussed in section 3.3.

Continuing with the infrastructure service: It contains the very core of the Blockchain. Building upon the hardware, it contains the Blockchain and its components, e.g. single blocks. Also, the assets, tokens, or information which are stored on the Blockchain are included. That said, single blocks contain a block header and data trees. In this representation we leave the “Puzzle-Mining Agreement” (known as Proof of Work or Proof of Stake) out as the consensus algorithm defines the validity of transactions, assets, and the data itself. Additionally, the algorithm often controls the supply of currency. Zooming in further, infrastructure service contains single data-transactions and possibly their respective source code.

The application tier is the largest tier in this view. We think that the highest potential of Blockchain technology exist on this tier. It contains all software or assets that are stored and executed in the network. Every interaction with the system (concerning new information) happens on this tier. The central application and the basis for all other applications is *persistent data storage*. Because of the design of the underlying infrastructure services, an alteration of once inserted data is neither possible nor desired. Thus, the integrity of information is guaranteed. Other applications build on top of the persistent data storage such as “Cryptocurrencies” itself (generally speaking distributed Ledgers), “Asset Transfer”, "Finding Meta Consensus" or service provision. We distinct between "predefined" applications and user created applications.

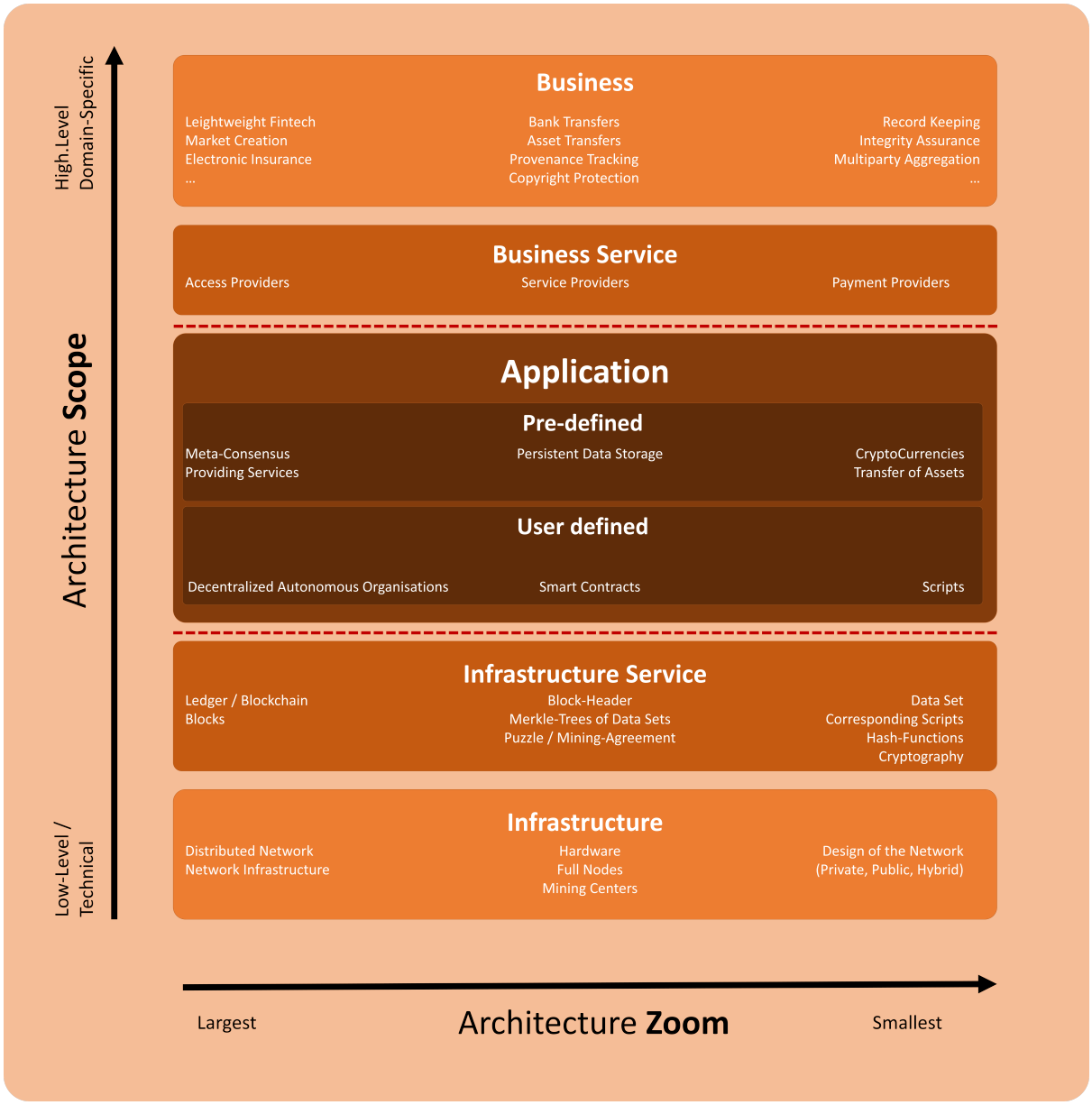


Figure 3.1.: Blockchain Architecture Overview

Predefined applications only allow one to use a given set of actions, for example transfer of money or asset tracking. Cryptocurrencies only offer only very few actions such as executing transactions. If the user is allowed to define applications on the blockchain itself, many different applications are possible. These user created scripts can be distinguished in three different areas, depending on their size and complexity. Small scripts take care of basic tasks. Smart contracts are applications that enable users to interact with it and execute tasks. If smart contracts are linked together and interact which each other, in order to provide a "business", one refers to them as decentralized autonomous organization (DAO).

In the business service tier there are three important participants. Access providers offer a connection or an API to the Blockchain network itself for further usage. Service providers enable a convenient usage of wallets to store private keys for different purposes, such as cryptocurrencies. Payment providers play an important role to connect other financial streams to the network such that businesses do not have to deal with the currency itself, but are able to accept payments in the respective currency.

The top tier "Business" contains possible business models. We do propose four use cases for Businesses derived from an article of Gideon Greenspan [36]: Lightweight financial systems, provenance tracking, inter-organizational record keeping and multi party aggregation. We added different use cases which suits in the category, but the field of possible businesses is big and contains many different use cases. For that, we decided to include an selection of use cases.

3.2.2. 7-Layer Application Architecture

The second view provides a seven-layer application architecture, giving an insight in technical details of the first view. The “Infrastructure Service” tier of figure 3.1 is split up into seven layers, helping to better understand the position of single artefacts in the architecture. The *Infrastructure Service*-Tier is the heart of Blockchain technology, because it covers the essential parts which ensure the integrity of the data within the network. For simplicity reasons, in this description we refer to the Infrastructure Service-Tier as application. The view can be found in figure 3.2.

The seven layers are divided in three areas of focus: interaction, domain and data, from top to bottom. It helps to understand the focus of each layer and how transitions between them happen. Those seven layers are “Technical Interaction”, “Abstract Interaction”, “Domain Interaction”, “Domain Service”, “Domain Data”, “Abstract Data” and “Technical Data” (top to bottom).

Four arrows behind the tiers explain different flows in the application. The first arrow explains the *Error-Flow* between the layers. Errors can happen on every layer at any time (always depending on the implementation), but the errors only affect the same layer they originate from or layers above. The *State-Flow*-arrow describes the propagation of states within the application. State changes influence every layer in this view, because changes in underlying or overlying layers require a change of state in the corresponding layer. The third arrow covers the *Data-Flow* which is bidirectional. The fourth flow shows the *Action-Flow* which shows that only underlying layers are affected by actions. At any layer, one can only access functions

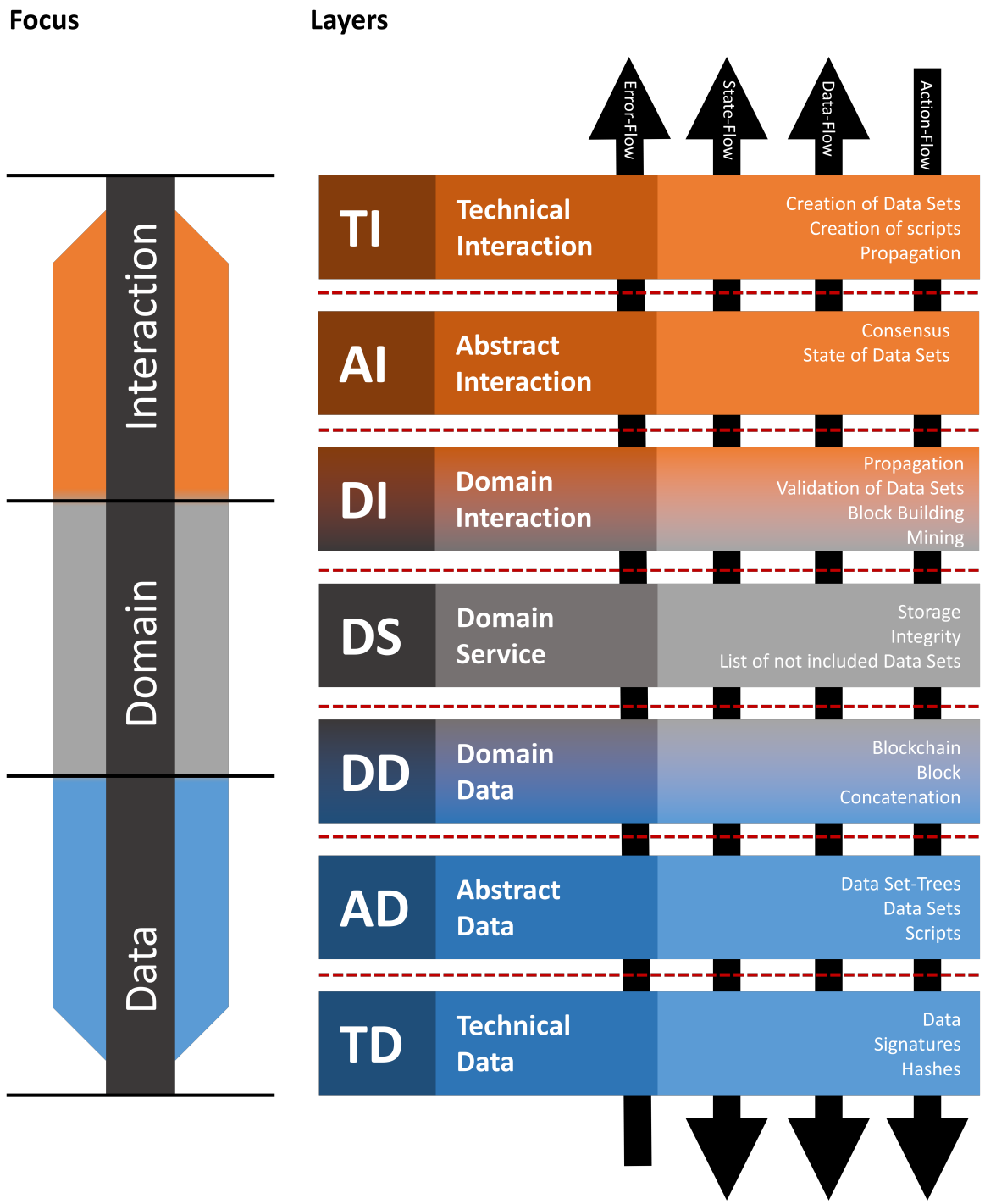


Figure 3.2.: Blockchain 7 Layer Application Architecture

from lower layers, therefore, the action flow goes from top to bottom.

We continue with the specific details of the layers: Starting at the “Technical Data” layer: The basic resource of blockchain technology is the data itself, e.g. any value stored in it. Two special types of values are hashes which compute a unique fingerprint of arbitrary data (like of a block or a dataset) and digital signatures which ensure proof of ownership and enable valid creation of datasets.

The "Abstract-Data-Layer" contains basic structures of the blockchain. Datasets, trees of datasets, and scripts are built upon values, signatures, and hashes. A dataset-tree is a structure which comprises elements in such way that in the end one hash can be generated out of it which is later on used for storage reasons. Elements can either be preceding hashes or the datasets itself.

The "Domain-Data-Layer" contains blocks and the Blockchain itself. One could include these in the “Abstract-Data-Layer”, but we think the blockchain itself is a significant part of the domain, especially the concatenation of blocks plays a huge role in the domain. Blocks usually contain the dataset-trees, and thus every information that should be included in it. The concatenation of the Blockchain describes how single Blocks are linked together. Normally, the hash of the previous block is stored in the current block, but in general one can think of other ways to establish a linkage.

Crucial elements of the functionality are included in the “Domain-Service-Layer”. Depending on the functionality of the node, different elements play a vital role. The storage and the storage management secure the validity of existing data and incoming new data. It allows the node to validate new blocks and depend on the information contained within the block. Additionally, with the inclusion of new blocks, it is able to update the list of not included datasets. These are datasets that are propagated in the network, but are not included in a block yet.

The “Domain-Interaction-Layer” contains additional features for interaction between nodes within the network. Mechanisms, like validation of new transactions, block creation, puzzle solving and propagation of new blocks are handled here.

The higher level of consensus is placed in the “Abstract-Interaction-Layer”. Its main concept is that the longest chain in a system is the valid chain. This cannot be part of the previously described layers, as the answer to the question of consensus is not solely a technical question, but a logical question. The state of datasets is an abstraction of the storage and are therefore integrated in this layer. In cryptocurrencies, single datasets represent single transactions. However, from a single transaction one cannot derive whether or not a transaction output is spent. Only the computation of all datasets arises a complete overview of all unspent transactions. Therefore, the state of datasets can be viewed as the overall state of all wallets. This element is not required in a Blockchain, but is contained in most of them.

The “Technical-Interaction-Layer” deals with questions of how new transactions and new scripts are built. Also, the propagation to the network is handled here. Additionally, one validates, if these transactions are permanently stored in the blockchain, and thus if they are executed properly.

3.2.3. Different Roles in the Blockchain Architecture

There are different actors in the blockchain system. Some only participate once, others need to constantly validate and insert information in the network in order to support the blockchain function properly. We categorize roles and actors in the system in a more general way as depicted in figure 3.3. To provide a better understanding of the roles, we give examples for the corresponding role in cryptocurrencies. Additionally, the coloured bars show in which tiers an actor is active.

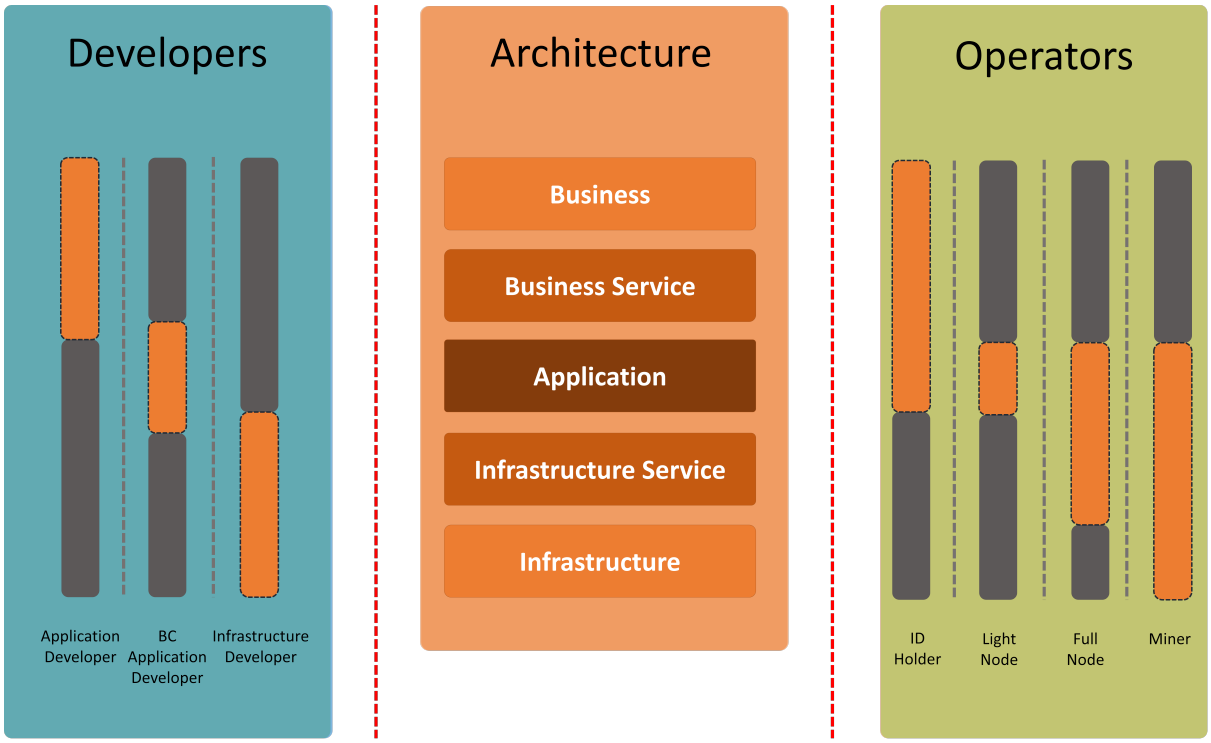


Figure 3.3.: Different Roles in the Blockchain Architecture

As mentioned before, the actors split up in two groups: operators and developers. This might seem trivial at first, because almost every software system can be split up in this way, but in a Blockchain network the two groups split up in several different operators and developers which execute unique tasks. Operators are keeping the system alive and populate the blockchain. The blockchain itself is controlled by its community, because different roles are of varying significance in the system.

3.2.3.1. Developers

The developers themselves are the designers and creators of the software and the system. They ensure functionality and correctness of the software, maintain the code and propose new extensions to it. This group can be split up into three different developer roles: the core-developer, caring about the underlying system, the script-developer, concerned with the software executed on the platform, and the software-developer who creates the software to interact with the system.

The core-developers have access to repository which contains the most important software for the blockchain. Other developers which create software for this specific system, build upon the code of the core-developers. Therefore the core developers have influence on every detail of the software and with that on the network and its corresponding applications. Most of the time, the core-developers are also the founders of the blockchain. This gives them very much power at the beginning. Due to the decentralized design of the network, the power shifts to other actors when the network is running and growing. Everybody can fork the code of the network at any time, and if the network itself is not happy with its core-developers, they will find new developers they trust.

The script-developer care about the software which is executed on the Blockchain. As described in the foundations section and figure 3.1, the Blockchain network is capable of executing scripts. The scripts are created with languages specific to the blockchain, to provide the individual functionality. The software itself can be very basic, for example BitCoin-Script[1] consists of only few commands. On other platforms, the language can be more complex. For example, Ethereum offers a scripting language with is Turing-complete, allowing the script-developer to write any software he requires. Therefore, the importance and the power of the script-developer depends on the capability of the scripting language itself.

The software developer creates software on top of scripts and network. He takes care of the software which enables users to communicate and work with the network more easily. For example, he builds the software which enables users to send money, keep track of own wallets and many more. He is not bound to one blockchain, but can support many systems. These applications are not essential to the system and the system does not depend on the software.

3.2.3.2. Operators

Operators interact with the system continuously and at will. They create datasets or communicate with the system itself. We suggest four different kinds of operators for a generalized blockchain: The IH (ID holder), the LN (Light Node), the FN (Full Node) and the miner. It is difficult to draw a clear line between the types of operators, keeping in mind it should match a general Blockchain scenario. Therefore, available roles depend on the specific system.

The ID-holder is the entity possessing private keys in the network. Private keys usually allow to act as an identity and access all resources which are associated with this identity. In the case of cryptocurrencies, private keys enable the entity to transfer coins to other ID-holders within the system. Every subject which wants to profit or interact with the system needs to be an ID-holder. This key mast be kept secret to retain the rights and assets in the future. By signing a transaction with the corresponding key, the transaction becomes valid and is recorded in the blockchain, making it immutable. ID-Holders can be users, miners, merchants, anybody who has assets in the system.

As the light node does not interact with the network directly, it does not need an own key. The main task is to provide knowledge: Knowledge, if some dataset is stored in the Blockchain or not. Thus, the Light Node needs to know what to look for. It assumes that a dataset will be stored in one of the next blocks, thus inspecting the next blocks for the respective dataset.

[1] the language used in the Bitcoin network

Because of the way blocks are usually created, it is usually very easy to check whether or not a block contains some dataset[2]. Also, checking the validity of the dataset in the block is fairly easy. Therefore, the effort of this procedure is very low in comparison to validating an entire block. The Light Node trusts the network that the chain consists only of valid blocks, minimizing its own work. A light-node could easily be implemented on a mobile phone or small devices. In cryptocurrencies this is useful for actors who want to know if a transaction has happened or not, for example merchants or regular users.

In contrast to the Light Node, the Full Node validates not only single datasets, but whole blocks. It interacts with the network by providing blocks, managing transactions and validating blocks, but it does not add additional information to the network. Checking a whole block is more time-consuming than checking a single dataset. It takes more computational power, because every block contains a large number of datasets. Furthermore, more storage is utilized, because the whole blockchain is stored in order to provide other nodes with blocks and to ensure the integrity of the complete blockchain. In theory, this could be implemented on a smart phone, but it is questionable as it would drain the battery very fast. All miners are also Full Nodes, but there are also use cases in which one only needs a Full Node. For example, if someone just wants to help to ensure integrity of the network, but does not want to participate in mining due to profitability. Full Nodes are not often used, as the use cases for it are rare. Nonetheless, they are part of the system.

The miner is responsible for the creation of new, valid blocks. In these blocks, only valid datasets are included. As stated previously, every miner is a Full Node. To be able to build new valid blocks, the miner has to know the whole blockchain and its content. The miner also ensures that only valid blocks are in the network. If it detects a wrong block, it does not add the block to his local blockchain. He is therefore able to produce valid blocks and earn a reward. (This is the only reason why he participates in the network.) In almost every case, a miner is also an ID-Holder. In order to get a reward (and not somebody else or no one), he has to include an address (to which he owns the private key) in the new block. Theoretically, the miner does not have to include his own ID. In private Blockchains without a reward for miners, a miner without an ID is also possible. The miner has the highest effort of all users. In most cases miners use specialized hardware which solves the mining puzzles faster than traditional computers.

3.2.4. Blockchain Ontology

The goal of the third view is to provide a deeper understanding of the single components of the blockchain ecosystem and how they are interconnected. Only considering the other views, it is unclear how the different parts of the blockchain influence each other. This view resolves the uncertainty. It can be found in 3.4.

We divide the ontology in seven different fields: “Applications”, “Data Structure”, “Methods”, “Network”, “Mining”, “Variables” and “Stakeholder”. Each area contains several different elements which are interconnected with other elements in the same area or in other areas. Starting with the “Stakeholder”-Area: It contains stakeholders of a typical Blockchain architec-

[2] As mentioned before, (Sorted) Merkle-Trees are a commonly used approach to store information. They enable a fast lookup to check if certain information is contained or not.

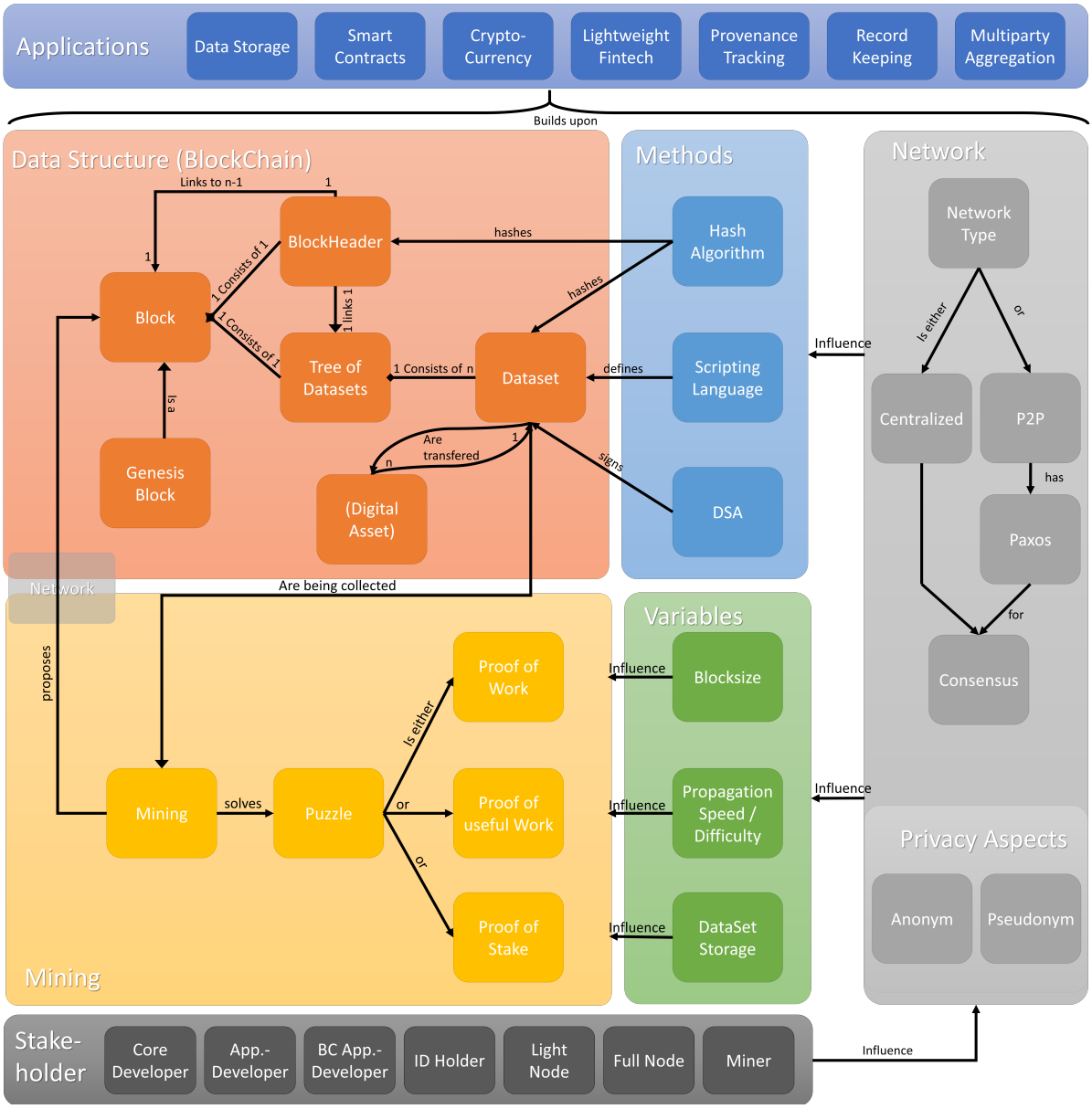


Figure 3.4.: Blockchain Ontology

ture, the roles presented in figure 3.3. They influence the network differently. The network itself makes decisions. Either it is centralized or decentralized, resulting in different methods for establishing consensus amongst the network. Additionally, it can be public, meaning everybody can participate in this network, private, and thus only accessible by authorized nodes, or shared, meaning only some parts are exposed to the network while others are kept private. We leave out the distinction in public, private, and shared blockchains for the sake of clarity. The network itself influences two further areas. These are “methods” and “variables”. Methods include Hash-Algorithms, the scripting-language and its power, and the digital signature algorithm (DSA). The network also decides upon variables like blocksize and propagation speed which indirectly determine the dataset speed of the entire network. It also chooses how datasets are stored on the network. Mining is the process of adding new blocks to the blockchain. It collects new datasets, builds a block out of it, solves a mining puzzle and then propagates the block over the network. The puzzle can either be “Proof of Work”, “Proof of Stake” or “Proof of useful Work” (see chapter 2), the later one not commonly used[3]. The Blockchain itself is built up of many blocks, starting with the first so called genesis block. Each block consists of a block header which links to the previous block and a tree of datasets, containing all the datasets relevant for this block. A dataset is used to transfer digital assets or coins from one public signature to another. In the graphical representation, digital assets are noted in brackets, as it is also possible that a Blockchain does not store digital assets but only information. “Applications” build upon all these elements. We use some examples from the previous views.

3.2.5. Blockchain Life Cycle

The fifth view displays the life cycle of the Blockchain. It helps to understand the various steps in the life cycle of a Blockchain and explains the work in the system, and which steps are executed repeatedly. The view is depicted in figure 3.5. The view does not cater for anomalies: e.g. if a blockchain gets redesigned or if a fork occurs as the forking process depends on the underlying consensus algorithm and is therefore difficult to generalize.

The life cycle consists of four layers: "process steps" which contain the different steps in the life cycle, "activities" which describe what happens in the respective step, "roles" which contain all participating stakeholders in the system, and artefacts which comprise the results of every activity. The Blockchain has four life cycle steps: "design", "implementation", "execution" and "wear out". A blockchain implementation can be executed multiple times. For simplicity reasons, we do not include this in the view.

In the first step, Design, infrastructure developers make major important decisions about the new Blockchain. What is the purpose of the Blockchain and how is it used? What is the unique selling point (if it is a public blockchain) and how does it differ from other Blockchains? How should the mining process be designed and how are other parameters set? All these decisions influence the functionality of the Blockchain and it is very hard to change these parameters later on. Usually, these changes result in some sort of fork, as one can see in 2.3.5. Infrastructure developers (responsible for the software design process) enforce the design decisions in the next step.

[3]Many other mining-puzzle exist, but left out of clarity.

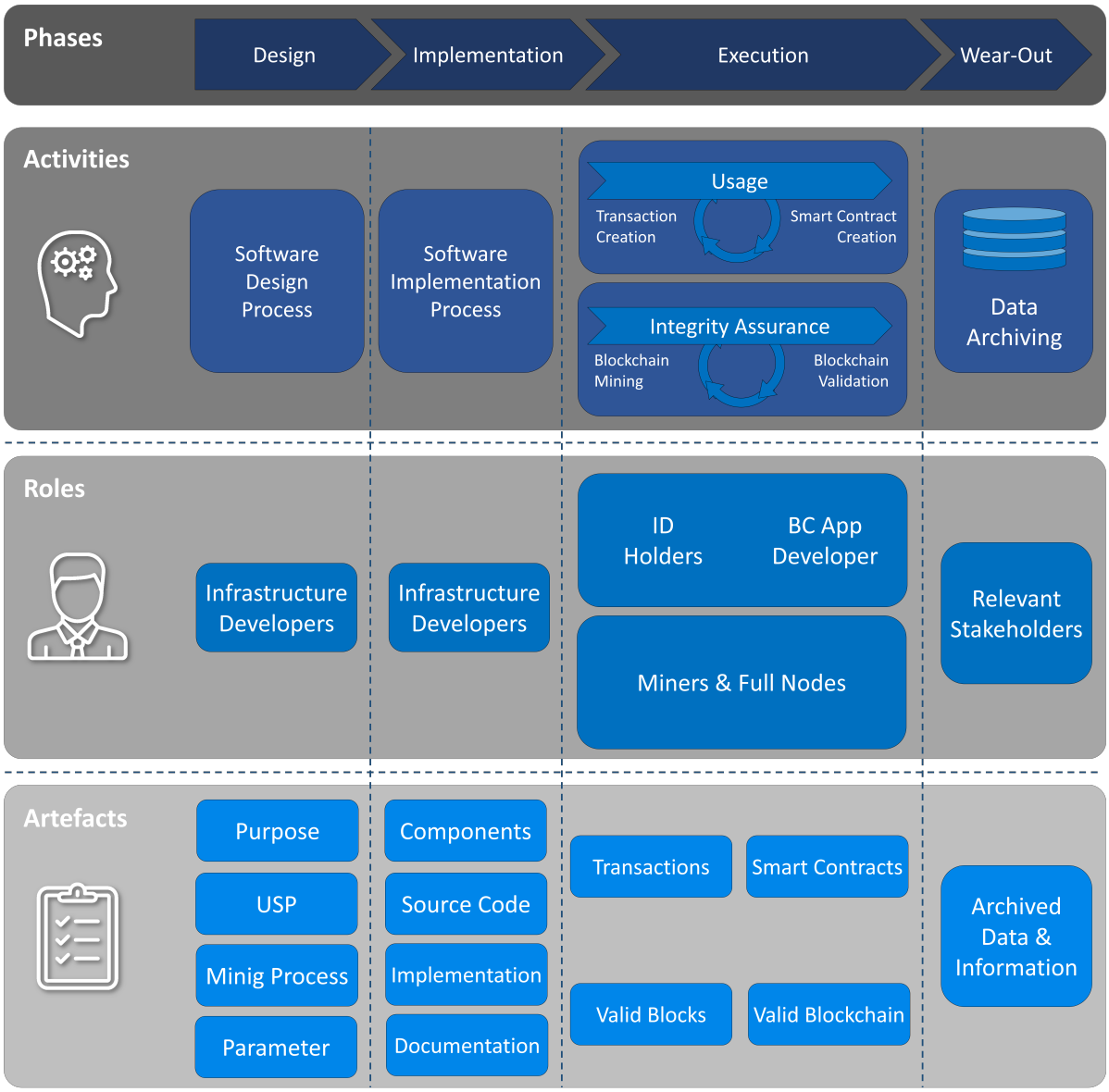


Figure 3.5.: Blockchain Lifecycle

In the second step, the infrastructure developers implement the Blockchain. They have to decide, if they use existing software (such as Bitcoin or Ethereum) or create their own Blockchain from scratch. Further details of design and implementation are not shown, because they always depend on the approach of the developers, and it is irrelevant to the life cycle of the Blockchain. The artefacts are also shown: Components of the software, source code, the finished implementation, and (eventually) a documentation.

In the third step the Blockchain is execution. Two major activities take place: Datasets are created and are inserted into the Blockchain. Furthermore, developers write blockchain applications. A Turing-complete language allows for more complexity. If the Blockchain only provides basic syntax, predefined software is used for repetitive tasks (for example transactions). These tasks are executed by ID holders and by Blockchain application developers, roles that can be found in figure 3.3. The second major activity is assuring integrity by full nodes and miners which validate new datasets and blocks that enter the network. Additionally, miners create new blocks and thus expand the blockchain.

The fourth step is called "Wear-out". If the chain gets too old or other reasons occur[4] why operators stop using it, the average creation time of a new block will increase until no blocks are created any more. This may be self-explanatory, but draws the attention to the fact that sooner or later every blockchain will eventually die. Stakeholders, who are interested in the stored data should archive the data for later access.

The user group "regular" application developer which can be found in figure 3.3 is left out on purpose: The Blockchain itself does not depend on applications that are built on top of it, application developers are therefore not required in the life cycle.

3.3. Blockchain Parameters

After presenting five views about architecture of a regular Blockchain, we extract the parameters responsible for the configuration from literature and evaluate their implications. Parameters strongly depend on the design of the Blockchain and on the goal it is pursuing, but it is important to know what fundamental factors exist that influence the behaviour of the blockchain. An exhaustive list of all parameters in Blockchain technology cannot be given, but the most important ones are covered. The parameters are sorted as the corresponding concepts occur in the foundations sections. Dispensable parameters that do not influence the functionality of the network are left out. Additionally, a list of the parameters is provided.

3.3.1. Cryptography

The cryptography of blockchain technology builds upon two different functions: The hash function and the digital signature algorithms. The parameter is the concrete choice of the used method.

Various implementations of hash functions exist, but only a few of them make sense in this context. The most popular hash functions are SHA-1, SHA-2 and SHA-3. Older implementa-

[4] E.g. a loss of trust could lead to a user decline.

tions like MD5 are insecure, because they are vulnerable to collisions, meaning two different inputs generate the same hash [37]. Additionally, SHA-1 was recently [38] discovered to be error-prone as well, therefore it should not be used anymore. The choice of the hash function does not influence the system per se, but the usage of common hash functions could appeal miners from popular coins (with the same hash function) starting to mine the new Blockchain. Additionally, the durability of the hash function has to be kept in mind. Historically, all hash functions were proven to be insecure sometime, it is therefore save to assume that current hash functions like SHA-2 or SHA-3 will be broken in the future. Future vulnerabilities are not threatening the security of the Blockchain, if new hash-algorithms are adopted via a hard-fork sufficiently early.

For public-private cryptography, two major implementations exist: RSA (Rivest, Shamir & Adleman) [39] and ECC (Elliptic Curve Cryptography) [40]. The selection of a method is not as important as the selection of the used key length. These methods differ from each other in the underlying mathematical concept, but provide the same functionality for a Blockchain. ECC requires smaller keys for the same security [41], but uses more computational power. Yet both factors do not matter in the Blockchain environment. The community assumes that both methods are secure, but with increasing computational power it might be possible that keys are bruteforced. Unlike hash functions, the public-private cryptography algorithm cannot easily be replaced, as private keys are tied to assets. If the public-private cryptography gets exchanged in the Blockchain, one cannot guarantee that assets linked to a private key of the old algorithm will be assigned to the same owner with the new cryptographic method. The owner would have to react and transfer their assets to addresses using the new algorithm.

3.3.2. Basic Parameters

The type of the Blockchain is the most basic parameter. As introduced in the foundations chapter, there are three ways: decentralized, hybrid or centralized. One can build the blockchain private or publicly accessible. These decisions are important to start with, because they influence the choice of other parameters or even make them obsolete. These parameters can be seen as two basic rights. The fashion of the network defines who is allowed to write blocks and add transactions. The second parameter is which users have access to the network.

If a Blockchain is publicly available and decentralized managed, the system has to be robust and needs self-governing functions such as elaborated consensus and mining algorithms. On the other hand, if the blockchain is private or the rights to add information and blocks are bound to some sort of entity, there is more trust between all actors (e.g. in a company) and some security measurements are obsolete. When deciding on these parameters, one should keep in mind the target audience.

3.3.3. Structure

The design of the Blockchain defines most of its structure. One can think of different ways to interconnect blocks or to build up a blockchain, but changes in this fundamental design will influence the network and the system, leading to questions on how the consensus and mining algorithms should look like. We are not aware of any approaches in the direction of creating

Blockchains with data structures other than connected lists. The layout of how transactions are stored into a block is modifiable. Normally, Merkle-trees are used to calculate the hash of all transactions. Additionally, Ethereum proposes the usage of Merkle-Patricia trees which enable a faster lookup whether or not a block includes a transaction [42]. The influence of this decision is low, but one has to decide on that to determine the exact notation and setup of a single block.

A more important parameter which is also in active discussion in the Bitcoin network is the size of one block. There is no overall recommendation on how big one block should be, but Bitcoin for example uses a maximum size of 1mb per block as of May 2017. It defines the maximum throughput of the system along with other parameters. This can not only limit regular transactions as in a cryptocurrency, but can lead to a Blockchain being overrun with load slowing down the system. On the other hand, to big blocks can "clog" the system, because they need to be transferred to all participating nodes. One has to determine the allowed block size very carefully. This also depends highly on the use case and the fashion of the network. In a private blockchain, an unlimited blocksize is no problem, as private operators have to use systems and network infrastructure supporting such large blocks.

An additional parameter that only applies to cryptocurrencies or any blockchain managing some sort of asset is how the transactions are stored. There are two approaches to facilitate that: Either transactions are recorded or the current state of all entities is stored. If only transactions are stored, a node has to validate the complete blockchain to know which entity owns which assets. The information is needed to validate new blocks and transactions, because the node cannot otherwise know, if an asset is transferred somewhere else. This process is highly time consuming, especially for new nodes entering an older blockchain. A different approach is to store the current state in the block. With this, a new node only needs to obtain the the latest block to understand the state and to validate blocks. Yet, this approach requires additional storage in a block. Both approaches can be combined together.

3.3.4. Network

In the network, one can change the most relevant parameters.

Another important parameter is the mining algorithm. There are many different algorithms which are suited for different purposes, but the main two are Proof of Work and Proof of Stake. These algorithms can be adapted to fit given needs. A problem of PoW is, that specialized hardware (so called ASICs[5]) is able to provide high hash rates at low cost, making it infeasible for other participants with regular computers or servers to participate as a miner. A countermeasure to this ASIC-usage are PoW-algorithms which consume a large portion of memory [43]. With a high memory requirement, ASICs are more expensive and slower, allowing computers and normal servers to participate in the network.

Besides the block size parameter, another parameter influences the throughput of the network: The block propagation time. It sets the difficulty of the mining algorithm such that on average one block is created in a given time frame. Depending on the mining algorithm, the time can be 15 seconds (Ethereum [44]) or 10 minutes (Bitcoin [2]). The optimal time frame depends on

[5]See A.2

the corresponding use case. If one builds a private or centralized Blockchain, one does not have to set the mining algorithm along with the block propagation time.

Another parameter is the recalculation of the difficulty. Yet, it does not tremendously influence the functionality of the Blockchain. During a small period of time, the computational power of the system changes only slightly. Thus, one should not recalculate the difficulty too often. One the other hand, a long period of time between recalculations makes the network vulnerable to attacks aiming at setting difficulty as high as possible: If suddenly a majority of participants of the network shuts down, the Blockchain significantly slows down until the block propagation time is adjusted. On the other hand, too many blocks might spawn until the block propagation time is adjusted to the increased computational power.

3.3.5. Applications

There exist multiple parameters for applications built on top of the Blockchain. They vastly depend on the goals of the Blockchain. We focus on the general Blockchain application and its used language and take a look at cryptocurrencies as a special application.

The used and supported language restricts the applications that can be built and executed on the Blockchain. As stated in the foundations, most cryptocurrencies use a very simple language which only supports basic concepts and operations. By adding some commands, one can significantly increase their functionality. If the language is Turing-complete, any code is possible. One also has to keep in mind that more complex languages require some sort of mechanism which takes care of possible manipulation or spam attacks which wastes computation power of all participating nodes. Such attacks already occurred in Ethereum [45].

3.3.5.1. Cryptocurrencies

If the application of the Blockchain is a cryptocurrency, additional parameters become relevant. Furthermore, creating a new currency requires economic considerations which we do not cover in this thesis.

The first question is how much money should be available in the network and how the amount of money changes over time (inflation vs. deflation). We focus on the main parameters. Is there a reward in the first block of which only the creator of the Blockchain benefits? This might not always be a good idea, as it can discourage people to use the Blockchain, because the developer "cheats". One also has to devise a mining reward to incentivise miners adopting the new Blockchain. Will it decrease (as in Bitcoin) or will it stay the same (as in Ethereum) or does it change differently?

To get an oversight of the most important parameters, one can view table 3.1.

Area	Description	Possible Values
1. Cryptography		
1.1 Hash-Algorithm	Generates hashes for blocks and datasets.	SHA-2, SHA-3, or other
1.2 Asymmetric Cryptography	Responsible for creating signatures & creating identities.	ECC[40], RSA[39] or other
2. Basic Parameters		
2.1 Type of Blockchain	Defines which entities are allowed to propose new blocks or datasets.	Centralized, decentralized, hybrid
2.2 Audience	Defines which entities are allowed to access the network.	Public, private
3. Structure		
3.1 Dataset Storage	Defines the way datasets are stored.	Merkle-trees, Merkle-Patricia-trees, other patterns
3.2 Block size	The maximum storage requirement for a single block.	Few kb to unlimited
4. Network		
4.1 Decentralized		
4.1.1 Consensus-Algorithm	Defines how the network finds new blocks.	PoW, PoS, other algorithms
4.1.2 Block propagation time	Defines the average time between two blocks.	15 seconds to any desired
4.1.3 Difficulty recalculation	The time between recalculation of difficulty.	Any desired (BTC:˜2 Weeks)
4.2 Centralized		
4.2.1 Authority	Entity which is allowed to generate new Blocks	Public key of an entity
5. Applications		
5.1 Language	Used programming language	Simple to Turing-complete

Table 3.1.: Blockchain Parameters

4. Expert Interviews

We establish a solid knowledge foundation about Blockchain technology in the preceding chapters. After creating different architectural views, we conduct semi-structured expert interviews and give an overview over the answers.

4.1. Preparatory Work

At first, we give an insight to the motivation for interviews, the development and structure of the interview guideline, the selection of interview partners and corresponding branches. Afterwards, we show the guiding questions.

4.1.1. Idea

The main idea behind the semi structured expert interviews is to get detailed information about a variety of use cases. We retrieve additional information of these use cases to apply GTM to generate theories about use case classification, Blockchain principles and requirements for the deployment of Blockchain technology. Additionally, we want to know the opinion of our interview partners: How Blockchain technology is understood, what problems and challenges occur when working with the technology and what important risks and downsides arise with the technology. Additionally, we want to know what hopes and expectations people have about the technology.

4.1.2. Development of Interview Guideline

The interview guideline is split in four different parts. At first, we want to know the background of the interview partner. Who is his employer, which position does he hold, and what are his responsibilities in the firm.

In the second part, we want to learn about their knowledge on Blockchain technology, knowledge about specific frameworks and development expertise, to tailor the questioning to the specific knowledge of the expert.

In the third part, we specifically talk about his or her use case. We are interested in the underlying problems that the stakeholders face and how they solve the problems with the usage of Blockchain technology. Further, the experts describe the progress of the development as well as challenges and difficulties that occurred during it. Regarding the setup, we want to know which underlying frameworks they use or if they create blockchain technology from scratch.

Lastly, we talked about Blockchain technology in general. We want to know the expert's opinion on the most important risks and downsides of Blockchain technology. Furthermore, we ask questions about the reasons for implementing a use case with Blockchain technology, specifically what requirements use case should fulfil to be suited for a Blockchain. After that, we ask the experts what needs to be done to further improve Blockchain technology and about their hopes for the future development of Blockchain.

4.1.3. Selection of Interview Partners

We select experts in five different areas: Enterprises which are known for the implementation of Blockchain technology, young startups which use the Blockchain in a disruptive manner, small companies which created so called Đapps[1], researchers in the area of Blockchain technology, and lawyers. The range of knowledge among the experts differs widely, but we conduct a sufficient number of interviews to underpin our theories. At their own request, some experts are not named. The distribution of the interviewees can be found in table 4.1.

4.1.4. Branches

Because we do not want to extract differences between single branches or develop a specific understanding about one branch, the branch of the interview partners plays subordinate role in the selection process. Some experts come up with different use cases and professions, therefore we count their use cases as two different units, as the second part of the interview is conducted for each use case individually.

Branch	Percentage	Number
Enterprises	22.2 %	4
Startups	22.2 %	4
Đapps	16.6 %	3
Experts	33.3 %	6
Lawyers	5.5 %	1
Sum	100 %	18

Table 4.1.: Branch Distribution

4.1.5. Questioning

In following, we show a list of questions we ask during the interview.

1. Preliminary Questions
    a) Are we allowed to record this interview?
    b) Do you want to stay anonymous or are we allowed to use your name and the name of your company?

[1]Đapps are small Applications built on top of the Ethereum framework.

2. Background
    a) What is your firm, how many employees does your firm employ and in which sector is your company active?
    b) What is your position within the firm and what are your responsibilities?
3. Knowledge
    a) What is the Blockchain for you?
    b) What is your understanding about it?
    c) Which frameworks do you know and how experienced are you with it?
    d) Do you have any specific development experiences with this technology?
    e) What benefits does the Blockchain offer?
4. Use Case(s)
    a) What is the problem that you want to solve?
    b) How is this problem solved by using Blockchain technology?
    c) How far are you in the process of your development?
    d) Do you use an underlying Blockchain Framework? Which one?
    e) What features do you use of this technology?
    f) What are the challenges that you faced throughout your process / face now?
5. General Blockchain
    a) What are the most important risks using Blockchain technology? How can they be minimized?
    b) What are downsides of Blockchain technology?
    c) Where do you see the most work to be done within the technology?
    d) What are your hopes about the future of Blockchain technology?

4.2. Overview of Answers

The questions were answered by the selected experts. The answers are not supposed to be evaluated empirically. We give an insight about how experts see the technology, what they understand about it, and furthermore what their use cases are. The answers are split in three sections: Talking about the topic Blockchain, about the use case and their description, and third, risks and downsides of the technology. Should the occasion arise, we will quote experts.

4.2.1. General Views about the Blockchain

In this section, we write about the answers of the experts for questions number three.

We want to know how the Blockchain is understood and what it signifies for the person. Every

expert we talk to is at least basically familiar with the technology, but of course the range of the understanding of the technology is large. The experts described Blockchain technology among others as *distributed database*, *decentralized platform that ensures integrity*, or *decentralized neural network*. Three factors that all experts claimed to be important are 1) decentralization, 2) integrity assurance and 3) transfer of some form of assets. In some cases, experts declared the platform as trust-less, as participants do not have to trust a 3rd party.

It is interesting that the focus of the experts is on assets and values, though the Blockchain can be used for other purposes as well. Some experts, especially involved in the area of Ethereum, mentioned the functionality of smart contracts which was not referred to as a critical part of the technology, but a very important functionality for the Blockchain ecosystem.

An interesting aspect mentioned by one experts was following: A Blockchain considered as database. This leads to the perception that the Blockchain itself is in competition with traditional databases. However, the interpretation of a Blockchain as a database is misleading, data is stored in it, but the data does not represent information, but actual assets and values. Therefore it is not a database, but provides a trust layer for a pool of data.

> The Blockchain is a worldwide distributed computer, a real distributed computer. Not a decentralized system, not a centralized system, but a real distributed system. A characterization as a distributed database, as it is often done today, wouldn't do justice to it.
>
> *Marco Spörl, jambit GmbH* (translated)

The description of the Blockchain Marco Spörl gives, is simple and understandable. Additionally, it implies that many characteristics of computers can also be found in Blockchain technology. For example, the terminus computer implies that no matter how much people own the computer, it cannot run faster as it is built[2]. That is exactly the same with Blockchain in which speed is limited by design.

When asked about the used platforms, most of the experts answered that their main focus is in the usage of smart contracts and therefore Ethereum. However, the opinions about this framework are torn. Some experts say that the platform is easy to understand and to use. Additionally, writing code is not difficult and allows for complex programs executed on the Blockchain. Other experts say that the platform itself is not very well developed and changes are implemented very fast. Since the DAO hack [33], the platform has lost trust of users.

Other platforms are barely used. One developer designed his software in such way that the implementation does not depend on one specific implementation, but runs on every possible cryptocurrency or Blockchain approach, if there is a type of transaction model. There is one green field approach in which a Blockchain is designed from the ground up without using any other platform. Unfortunately, we did not have the opportunity to talk to an expert who is familiar with the platform Hyperledger.

[2] A computer can only access its own resources and gets only faster, if the hardware is upgraded. Same holds for the Blockchain, as every node in the system does the same calculations as all other nodes.

4.2.2. Advantages

We asked the experts about the advantages of Blockchain technology and give an overview about the answers. The advantages are ordered into three different categories: Advantages in the development of applications and smart contracts, advantages in the usage of the system, and economic advantages. We provide an overview about the advantages in table 4.2.

4.2.2.1. Development

It is straight-forward to develop applications which facilitate Blockchain technology or smart contracts. There are several reasons for this: The Blockchain is not suited for the computation of large problems and complex methods, as it would use too much computation power and therefore cost too much. Therefore, the applied functionality and offered methods have a low footprint and are designed in a convenient way. The second reason is that the offered language to create smart contracts is designed such that it can be used by a large audience[3].

> The development for the Blockchain is fairly easy. Our developers were almost disappointed how little they had to work. The platform is well designed, such that the required contracts are not complex.
>
> *Bernd Lapp, CEO of Swarm City, about Ethereum* (translated)

Additionally, the required infrastructure investment for smart contracts is very low. (Almost) everything can be handled by the platform itself, the transfer of money, the execution of the contract, and the resulting outputs. Of course, the required infrastructure always depends on the chosen use case and its complexity, but with a minimalistic approach, Blockchain technology can replace a lot of infrastructure which is needed usually.

> The interesting thing about [a use case] is, that you can run it without any infrastructure. The website is just some HTML and Javascript, hosted on GitHub. If someone buys the service, I don't have to pay PayPal money to transfer any money around, I don't have to store the data, I don't have any service running that operates the application. It runs without any interaction from my side.
>
> *Glynn Bird*

4.2.2.2. Usage of the System

Compared to a traditional system, using a blockchain system brings along several advantages: The Blockchain system uses a distributed ledger which is immutable and therefore does not allow any forgery. One can make sure that once some information or a transaction is stored in the system, it cannot be undone.

Additionally, when using smart contracts, Blockchain technology has following advantage: The trust in the functionality of the platform implies trusting that every smart contract behaves exactly in the way it was designed and developed. The source code of a smart contract is

[3] One has to state that the only serious platform for smart contracts is Ethereum, which allows the usage of Solidity, a Javascript-like programming language.

available to anyone and the functionality of it is accessible. This enables users to check if a smart contract behaves like claimed. After that, he can decide, if he wants to use the contract or not. Especially if it is about special business cases, the users demand to be able to look into the code of the smart contract, for example gambling on the Internet.

On the Blockchain, money can be transferred faster than with traditional payment providers. The transaction usually happens within the next blocks (depending on the usage of the network), and no other providers are required. This increases the comfort for using the technology, because users do not depend on the vendor to support their preferred payment provider. In addition, the user does not have to register at the service with an email or other information, as he is always authenticated with his public ID which is used for the service.

4.2.2.3. Economic Advantages

Using no third party payment providers, payment fees are not necessary any more. The users and the developers do not have to pay a fee, if the money gets transferred via different providers or exchanged to foreign currencies. That enables the provider to offer his service at low cost.

Besides making payment providers superfluous, (almost) any other third party can be replaced, too. Often, additional intermediaries are required to reach potential customers with the offer for some service. This especially holds for all kind of platforms which bring together seller and buyers, for example Uber or AirBnB. With Blockchain, these platforms could be replaced and designed in such way that the revenue for the vendor of a good is prioritized.

Advantage	**Description**
Development	
Easy creation of smart contracts	The creation of smart contracts is easy as the language is designed for the purposes of Blockchain technology.
Little required infrastructure	The blockchain itself provides access to smart contract. Hardly any third parties needed.
Usage	
Immutability of data	All content stored in the Blockchain cannot be changed after insertion.
Speed	Transactions can be executed within minutes to hours.
Trust	The functionality of a smart contract can be looked up. The Blockchain guarantees that it is executed in that way.
Economic Advantages	
Cheap	The usage of cryptocurrency costs hardly any fee compared to traditional payment providers.
Replacement of third parties	Blockchain is designed in such way that costly third parties can be eliminated.

Table 4.2.: Selected Advantages of Blockchain Technology

4.2.3. Risks

We discussed risks of Blockchain technology with the experts. However, we do not want to go into deep technical details why and how these risks can occur, but aim at giving an overview about the experts opinion on if and how they are going to affect single blockchains or whole Blockchain ecosystems. The list presented here is not exhaustive, because not every risk we got to know of was us told by experts. As before, we structure the information we received from the experts. The technical risks are categorized along the process view shown in 3.5. Additionally, risks in the relationship between Blockchain and the surrounding environment are categorized. If a risk applies only to public or private Blockchains, we will add the information in the text. An overview can be found in table 4.3 and table 4.4. An intermediate result of this questioning was given at the IRIS 2017 in Salzburg as part of a panel discussion[4], the slides can be found online [46].

4.2.3.1. Technical Risks

In this section we describe the technical risks in design, implementation, execution and wear-out of Blockchain.

Risks	**Description**
Design	
No scalability	Blockchain technology is not able to scale with increased computation power.
Varying speed	The speed of the blockchain is not constant, but varies.
No settlement finality	The longest chain, thus the status is only probabilistic. Information is never 100% fixed.
Privacy risks	The Blockchain is open and transparent, but this can lead to privacy issues.
No true randomness	The Blockchain is deterministic and therefore randomness cannot be generated. Workarounds have downsides.
Implementation	
Bugs in Blockchain	Bugs threaten the security of the system, leading to a flawed Blockchain.
Bugs in Smart Contracts	Bugs in smart contracts can lead to the loss of money. As they are public, bugs can easily found.
Execution	
51%-Attack	51%-Attacks allow to rewrite of the Blockchain history.
Waste of Energy	Some consensus-algorithms (like PoW) can result in wasting energy.
Centralization	The longer the Blockchain runs, the more power in the network gets centralized as participation gets more expensive.

Table 4.3.: Selected Risks of Blockchain Technology Part 1

[4] goo.gl/YZlMpl

Risks	Description
Law & Governments	
Uncertain legal questions	Governments do not provide clear frameworks for conducting business with Blockchain.
No central responsibility	As no one is responsible for the system, the government has no options to influence the contents of the network.
Governmental influence	Governments could create backdoors in the system to get hold of certain identities.
Transparency vs. privacy	It is not clear how user information is protected in the Blockchain and how issues are resolved.
Blockchain Community	
Future development	Future developments of single Blockchain ecosystems can go wrong, if a community driven approach is chosen. Different participants can have different ideas, tearing community and network apart.
Industry	
Hype	The hype around Blockchain can result in frustration about the technology as it cannot match the expectations.
High demand of single platform	Industry only using one platform can result in higher prices and longer waiting times, rendering the platform unusable for some business cases.
Technical Progress	
New developments	New developments can result in the breakdown of cryptographic functionality, resulting in a broken Blockchain.
End Users	
Low interest	Because of the complex technology, users fear to adapt to it, resulting in low usage numbers.
Volatility of Prices	The prices of cryptocurrencies highly vary. Users do not want to risk loosing their money.

Table 4.4.: Selected Risks of Blockchain Technology Part 2

Design The first major flaw in the design of a Blockchain is the non-existent scalability. We gave a detailed explanation about the phenomenon of the non-scalability in the chapter 2, but this disadvantage results in a risk. Almost every expert we talked with said that it is a risk for use cases, if the Blockchain is not capable of scaling. They say the Blockchain is not capable of handling certain use cases. Major companies which trade goods or have a high-volume throughput cannot use Blockchain for their services or products, as the Blockchain cannot compute the inputs. It takes more and more time until datasets are inserted in the Blockchain system. Theoretically, the system could recover in times of low throughput, but the main problem persists: The technology cannot cope with a higher demand than its design. Anyway, experts are sure that a technical solution exists. There are different approaches, but often in exchange for other advantages [47].

Talking about scaling: It is always the question whether the speed of the system can be adapted or not. Not only that it cannot be adapted by increased computation power (but only by design), the speed is not constant, but it varies. New transactions or datasets are only included in the Blockchain when a new block is generated. In a public blockchain this happens at a given average speed, the creation time is probabilistic around a predefined value. In blockchain systems (e.g. in Bitcoin) it happens to take multiple times as long as defined [5]. Additionally (in case of public PoW blockchains), attackers are able to slow down the overall network at their proportion of computation power. E.g. if the attacker has 10% of the hash-rate, he can slow down the network by 10%.

There is no guarantee that a transaction included in the longest chain will remain there forever, as the solution with a PoW-algorithm to the Byzantine Generals Problem is probabilistic [48]. It it possible (but very unlikely) to build a longer chain that does not include some transactions. This is in contradiction to the Settlement Finality Directive which states that once a transaction happens, it must be final and not recoverable [49]. Banks are required to execute transactions and to prevent roll backs, in order to prevent reclaims. This could become a legislative risk for banks and other financial institutions.

Regular Blockchains are open and transparent by design. Every dataset, every information contained in the network can be read. This leads to the problem that every information which is stored in the Blockchain can be used for other objectives than checking the integrity of the network. This threatens the privacy of the Blockchain users. Therefore, after submitting information in the network, they lose control over it. Even if cryptographic standards are used to encrypt information, one cannot ensure that no one will break the encryption using more advanced methods several years later.

With its openness and transparency, the whole system is deterministic, meaning that all participants can understand and validate every process, every calculation and information. This contradicts the idea of randomness. In normal computers or server systems, randomness is generated by so called pseudo-random-number-generators (PRNG) which take many different variables into account: E.g. the number of opened processes, used memory, maybe even user input. These variables are used to facilitate random seeds[6] to generate random numbers. However, if a PRNG is used in a system in which all inputs and parameters are publicly visible

[5] One can consider the overview of mined blocks at `https://blockchain.info`. For example, the time between block 464886 and block 464887 was almost 44 minutes.

[6] A starting value for the PRNG

to everyone, the output of the PRNG is not random anymore[7], but becomes deterministic. Therefore if Blockchain would use random numbers from PRNGs, they could be predicted. Another possible method would be that the node that proposes the new block is in charge of generating new numbers. This would bring up the problem that the node (and everyone else) could not prove whether the numbers were generated by a PRNG or if they were just fabricated. Methods for generating random numbers out of the Blockchain are difficult, error-prone and vulnerable to manipulation [50]. Still, randomness remains an important feature in software development, therefore the community has to think of solutions.

Implementation There is only one particular risk in the implementation of Blockchain, because the process of implementing depends on the software engineer. Bugs in software occur on a regular basis, as there is no feasible way to guarantee a bug-free software. The problems with errors in the code compared to regular software is that errors in Blockchain technology result in bugs that threaten the security and integrity of the entire platform or expose it to some attack vectors. As mentioned before, a prominent bug occurred on the platform Ethereum, the famous *DAO-Hack* which lead to the loss of almost 50 million US$ which were later on recovered through a hard-fork that rewrote the history of the chain [33].

Smart contracts are error-prone, too. Smart Contracts are written in software-code and can have bugs which could lead to the loss of the functionality of the smart contract or maybe even money. Either way, during the implementation process, security should be the highest priority.

Execution Different risks occur during execution. Malicious participants can attack the network. A first attack, slowing down the network, is mentioned as a design risk. Other attacks can be carried out, too. Our experts mainly named two attacks: The 51% Attack and the Selfish-Mining attack. The 51% attack allows the malicious node to rewrite history in any way he prefers, but he is not able to create new money (but he can take it from other miners) or steal it from other wallets, but he is able to blackmail participants. With the second attack – the selfish-mining attack – a malicious miner can gain a competitive advantage over other miners by playing unfair.

Another risk which only applies to Blockchains with a Proof of Work consensus algorithm is the energy waste of this method. The experts stated that this algorithm is only computed to ensure integrity, but does not generate anything useful besides hashes which fulfil some requirement. Huge amounts of energy are wasted which could be saved. Yet, the experts are not aware of any superior consensus algorithm.

The centralization of mining power is another risk. This risk only applies to public decentralized blockchains. As the participation in the network becomes more expensive as time goes by, smaller participants drop out and only powerful participants can afford to stay. The price for participation depends on two factors: the blockchain gets longer, therefore more storage capacity is needed and the computation of new blocks becomes more complex, therefore more computation power is needed to stay profitable.

[7] In fact, PRNGs are never random. The randomness of the output depends only on the entropy of the source seed.

4.2.3.2. Environmental Risks

The Blockchain ecosystem interacts with other systems. Within these relationships risks can occur.

Law & Governments Legal questions arise with the usage of Blockchain and Smart Contracts. Lawyers will have to address numerous legal risks regarding this technology. These legal questions are therefore not within the scope of this thesis. Nevertheless, we present some questions by experts which are currently unanswered. For example, some experts asked the question how taxes should be paid with cryptocurrencies, because there is no way to directly pay taxes in the respective cryptocurrency. Additionally, it is not settled where the added value is created and if it falls under federal law of one country. Other experts worry about the question whether a smart contract is sufficient to actually agree on a legal transaction. Of course, the answer depends on the country.

Another question which involves governments is the relationship between anonymity, privacy and transparency. Blockchain systems can be configured in such way that every single of the three "goals" can be fulfilled, but not together in one single network. Should fully anonymized Blockchain networks even be allowed or does the state always need some access to the data? How can the users data be protected, if a fully transparent Blockchain is used? Governments will likely interfere with the technology when the general interest rises.

A decentralized Blockchain is not owned by one party, but by every participating node. That brings up the question of who is responsible for the network, when no one (or everyone) owns it. This question becomes relevant, when some malicious content is stored on the Blockchain. For example, a smart contract can be programmed such that committing a felony is rewarded. It is important to know what methods governments can implement to prevent malicious scenarios, as it would certainly affect the blockchain and its capabilities.

Users own a key-pair which represents the identity that owns assets on the Blockchain. However, law enforcement could be interested in knowing which public key belongs to which person to get hold of the person. It is possible that the usage of such systems is only allowed, if the identity to a some key is deposited at some governmental department.

Blockchain Community The community around single Blockchain or cryptocurrency implementations consist of many people with different capabilities, interests and plans for it. Therefore future updates, e.g. for Bitcoin, are heavily discussed. If the project is entirely community-driven, future development is more risky, because there is no central figure or person who leads the way in some direction. If there is a central person, future developments might be easier to convey. However, two risks persist: First, the community could decide on a dangerous or adverse path, and second, the community could split and with it the Blockchain and all assets, resulting in two co-existing platforms. This forking weakens the value of the network and may result in a trust-loss.

Industry The Gartner Hype Cycle [51] shows that Blockchain is at its peak of the hype. Experts which offer consulting in the area of Blockchain have a similar impression. Companies

which only recently first heard of the technology enforce use cases which may not profit from Blockchain at all. The technology is in danger of being wrongly perceived, leading to a lower interest in and maybe abandonment. Therefore, the technology should be understood before being adopted.

Another risk in the industry is that they might use a Blockchain which receives a very high interest. Per se, it is not a bad thing to use one chain that will certainly be around in the near future and is perceived as a good platform, but as the demand in this particular platform rises, the prices for the currency rise. All fees, transaction fees or execution fees[8], will rise with the hype around it. While the increasing price might be good for investors and early adopters, it can also kill entire use cases, because the execution of the software is too expensive. For example, the execution of more complex smart contracts could become too expensive or the price for a single transaction is that high, such that small transactions are not profitable any more. The value of regular fiat currency changes, too, but the volatility is not as high as in cryptocurrencies.

Technological Progress No one can reliably predict the technological advancements. But as it hopefully will bring new and better functionality to the Blockchain, it will also likely bring increased computation power. If the computation power rises far enough it becomes feasible to attack the cryptographic foundations of Blockchain, the whole system could become insecure and the assets stored in it can be stolen. Open blockchain protocols have to keep abreast of the technological advances and cannot stand still concerning their own development. The process will likely happen not immediately, but the process is slow and continuous.

End Users The people interacting and using the platforms are end users. End users have different levels of knowledge, resulting in a risk: The user does not adapt to the latest technology. The interviewed experts think that this can happen mainly for three reasons: The entrance into the network is very complicated, getting assets in the network (e.g. bitcoin in exchange for real money) requires often a process of registration and revealing one's own identity. This can discourage people to use the platform or to transfer money to the system. The second reason is that the usage of and interaction in the network is complicated. Standard-browsers, as they are common today in every computer and mobile phone, do not support the processes of Blockchain networks or smart contracts. Participants have to use specialized software which are not commonly used. And the third reason is that the users are frightened of the technology, because they do not know the implications of it. Regular users are used to functionality of *Password recovery* or *Fraud prevention* in which the bank helps getting stolen money back. Without this functionality, users might perceive the usage as too risky.

Another risk for end users who want to use the system on a regular basis is the volatility of the prices of the currencies. Normally, the prices are not bound to any value or centralized institution, therefore their value is only determined by supply and demand. It is therefore difficult to plan ahead, because the user does not know how much his money is worth in the near future. Users should be offered a way to exchange their cryptocurrency directly into known fiat-currency, in order to give the user more control over his own wallet and risk

[8]These are fees that have to be paid in order to execute Smart Contracts. In Ethereum it is called gas.

attitude.

4.2.4. Extracted Use Cases

We describe the use cases discussed with the experts. First, we describe the general purpose of the use case. Then, we describe the involved roles and how they interact with each other. If we are allowed to, we denote the company and website and describe the current project advancement. In the chapter 5 we select use cases and classify them according to the proposed architecture, the use case categories and the Blockchain principles.

We organize all use cases according to their used platforms. First, we describe all use cases which facilitate the Ethereum platform and smart contract features, then we cover all use cases which require an own Blockchain implementation (or rely on alternatives to Ethereum). Afterwards, we discuss the use cases which are independent from a specific implementation. Most of the described use cases could either be implemented in Ethereum or with an own Blockchain implementation. However, this classification is derived from the platform the experts chose.

A list of all use cases can be found in table 4.5.

4.2.4.1. Use Cases in Ethereum

Ethereum is the most commonly used platform for projects in Blockchain technology, as it supports a Turing-complete language.

Lottery Blockchain technology enables the user to bet on certain outcomes. In this use case the user is allowed to play the lottery. He can buy a ticket and bet on a number between 1 and 1000. Weekly, a random number is drawn and the winner receives the jackpot deducting a fee for the operator.

There are two roles involved in this use case. The gambler and the operator. First, the operator has to set up a smart contract which provides two functions: Betting on one number in exchange for Ether and drawing of a random number to derive the winner and transferring the prize. The operator has to manually execute the weekly drawing, as the system cannot automatically execute the smart contract. The random number originates from an external oracle, because the Ethereum platform is not capable of generating random numbers by itself. Second, the gambler uses his money to bet on a certain number. After the draw, the process starts all over. If no one guessed the right number, the contract retains the money for the next drawing.

The use case is a (fully functional) proof-of-concept, but the corresponding website was shut down in the meantime (March 2017). However, different proposes running lotteries within Blockchains exist [52].

Land Registration In Ethereum it is very easy to generate new unique tokens to represent some value or asset. In this use case the user is allowed to possess and trade tokens which

represent some kind of property. This helps to prove the possession of property to prevent fraud and expropriation (especially in countries without any land register or rigged land registers).

Three different roles participate in this use case. The developer of the smart contract, the owners of tokens, and some form of governmental representative. The developer of the smart contract issues tokens which represent some land (or some other space). It is important that the tokens can be split (such that the owner can sell or transfer parts of property), merged (to revert the process), and transferred (in exchange for some other value, i.e. Ether). Assuming that every owner already had his tokens which represent his property, the system would be fully functional and everybody could prove what they possess. In fact, the process of launching is difficult. The idea is that the whole space is issued in one token, given to the highest representative and split up in districts, handled to their representative, and split up further until every token is in the possession of the right person. Of course, this process is error prone, but this would be the only feasible way.

The use case is fully functional and should be used in Colombia, but the process is stuck, because of the lack of cooperation of governmental agencies. The project can be found on Github at `https://github.com/danmermel/landregister`.

Competitive Gaming Platform Most gaming platforms provide some sort of high score allowing people to submit their own score to the system to compete against each other and determine best player. Problems occur if cheaters and hackers are able to submit the high score they want, reducing the fun and competitiveness for all other players. To prevent cheating, one can create games connected to a Blockchain. Additionally, the user does not have to pay for the complete game, but just for one run of the game. The platform collects the money and distributes it at the end of the period among the winners.

In this use case there are three different roles. The gamer who wants to play a game (and maybe win a prize), and the developer of the game and operator of the service. The operator provides a smart contract which starts the game with a random seed when a user sends money to it. The seed serves as random value for the game (to ensure that every iteration is different), but also to recalculate the high score later on. The player plays the game and receives some high score, additionally all his inputs are recorded and submitted along with his high score and seed. The server computes if one can achieve the high score with the seed and inputs. If this is the case, the high score is submitted to the smart contract. The best user will receive the price later on. The operator has to execute the smart contract manually to disburse the money.

The software is in the beta-stage and almost fully functional. The result can be seen at `http://etherplay.io`. After the interview in February 2017, the platform included a way to allow third party developers to connect their own games. The creators plan to add more games in the future.

Competition Platform Some companies provide the opportunity to participate in competitions, for example to take the best picture of some occasion, to write a text, record an audio, or produce a video. The people who submit their content have to trust the provider that the vote is fair and the prize is paid out at the end. With a competition platform it is possible to

provide trust both in the correctness of the voting and in the pay out of the price.

In this use case, four roles are involved: The platform provider, the contest creator, the content submitter, and the voter. The platform provider creates smart contracts which allow to create competitions awarded with a price. The contest creator uses these contracts to create their own contest or competition and to set the rules for it. The money for prices must be put aside. Content submitters are now able to propose new content. Afterwards, the voter is now able to vote for his favourite and might have to pay some sort of fee, if the contest requests it. It is technically prohibited for the same person to vote twice and the fee prevents malicious votes. After the end of the contest, the submitter with the most votes wins the price. The platform operator takes shares of every transaction.

The implementation of the use case can be found at `http://call4contest.com/`. The website is fully functional and the team focuses on increasing its user base.

Service Platform Ethereum is not only suited for single platforms with specialized services. It is possible to create "meta"-platforms. This use case needs some imagination, because nothing comparable exists within in the regular market. The platform (in this description, we will refer to it as *meta-platform*) offers people to create their own platforms (referred as *user-platform*) that enable brokerage. The user-platform itself brings together buyers and sellers which have interest in some special product or service. These services, for example, could be flat-rental services (e.g. AirBnB) or taxi-like services (e.g. Uber).

There exist five different roles for this use case: The meta-platform creator, the user-platform creator, the buyer or seller (users), and additional service providers. The meta-platform creator develops smart contracts which support the creation of such user-platforms and the user-platforms are created by their respective developers. The users are able to offer or request a service in the respective user-platform and exchange the good or the service, while the user-platform developer takes care of the right implementation, conflict resolution, and rates. The meta-platform allows the creation of additional service providers. They provide additional services to different user-platforms. For example, an additional service could be the offering of an insurance policy for the possibility that the driver does not show up. The vision of the project is to create a whole ecosystem that lives on the Blockchain and enables the users to interact with different platforms.

The use case is developed and in the first beta-phase. It is implemented by Swarm City (`https://swarm.city/`) which collected over 3 million US$ in their first ICO. The first functional platform will go live in June 2017.

4.2.4.2. Use Cases within other or own Platforms

Energy Market Automation For simplicity reasons, we assume that the energy market consist of three players: Energy creators, energy consumers, and a middle-man who buys the energy from the power plants and sell it to end-users or industry. These providers buy all energy that is produced, which means that no "peer to peer" selling of energy is possible, for example if someone wants to buy the energy from the neighbour's solar panel. Additionally, there are no contracts that allow the customer to buy energy at the current rate from the energy

market. It would be an advantage for consumers to buy energy at the exact rates, to save money when energy is cheap. If the user cannot buy energy at the current rate he does not have an incentive to consume energy favourably (i.e. to stabilize the distribution network by using energy "off-peak"). Smart contracts and Blockchain technology are suited for this use case.

As our assumption the three roles remain: End users, producers and providers. Every producer of energy creates a smart contract which can be paid in order to receive energy. The price for the energy is determined by the smart contract itself and the overall demand. The end user can use hardware which accesses the Blockchain, enabling it to decide when to buy or sell energy. For example, electric cars contain a battery which needs to be loaded from time to time, depending on the usage. The car should load the battery when the price is very low. Often, the cheapest energy is available during the night. This might work the other way round, too. It could be possible for the car to return energy to the system, if this action is economically rewarding. Of course, other electronics could have access to the Blockchain. Additionally, the end-user would have full transparency over the provenance of his consumed energy and could make sure that only energy generated by renewable power plants is used.

The use case is a vision as it needs a lot of legal clarity, before further developments can continue. However, different companies work on using the Blockchain for energy markets, for example *GrünStromJeton* (`https://stromstunde.de/`) which basically constitutes an energy provider DAO.

Digital Identity Creation and Management The internet as we know it is a place where people can act in anonymity. Of course it is possible to track down people with some effort, but this often requires some form of legal procedure. There are websites on the internet on which users are required to authenticate to know which rights he might or might not have. Most of the time (webshops, information platforms, social networks), the provider trusts the information submitted by the user, because there is some other form of guarantee that the user behaves correct. For example, most web-shops do not act upon a new order before they received the money. With this they can make sure to only process valid orders. However, sometimes an official document is needed. For example, gambling sites need to make sure that their users are of age, requesting some form of identification. The identification is only needed for the sake of the minimum ages, but the document contains other sensitive data which is not required. With Blockchain technology, it is possible to only provide some information without revealing the other.

Three roles are involved in the use case: The certificate issuer (or authority, analogue to CAs[9]), the end user (certificate owner) and the provider of a service. The certificate issuer is a recognized entity on a Blockchain and identifies itself with a certificate which is bound to its identity. The end user is now able to approach the certificate issuer and request a certificate which contains all information the user wants the certificate issuer to confirm. The certificate is built up such that partial information can be revealed without revealing other information. The user is able to go to the provider of a service and to validate himself, if the provider trusts the certificate issuer. Different certificate issuers for different purposes are possible, for

[9]Certificate authorities (CA) are providers which issue SSL-certificates for websites, such that the communication between the user and the server is encrypted.

example a governmental CA which offers authentication for governmental services and private CAs, which are validated by the government.

The use case is a theoretical idea, but there are known approaches to create likewise platforms, for example the Cambridge Blockchain project (`http://cambridge-blockchain.com`).

Rights Management Digital signatures can be viewed as entities in the system of Blockchain. These entities can represent a person, a company, a process, a service or even real-world objects which are able to interact with other objects and their surrounding. The main idea of this use case is to restrict access to objects and only allow people access to it who are authorized to. The access can be managed by a third party which owns the corresponding object or by the object itself, requiring the user to contribute somehow to the preservation of the object. For example, one can think of an autonomous building which only grants access to people who participated in funding it or who are paying some sort of member fee. The building is now able to sustain itself by ordering and paying personnel that takes care of maintenance and cleaning. A more mundane usage would be to give access to company buildings and rooms according to the identity which is kept on the Blockchain. It would be very easy to keep track of rights of individuals, reducing the need for alternative management systems.

Three roles are involved in the use case: The owner of a property or the object itself, the system validation authority and the user. The owner of a property or the object itself would sign the user's public key with additional information which represents the rights of the user. When some condition is fulfilled (i.e. after some time) the right of the user would be revoked and he cannot access the entity any more.

The described use case underlies a non-disclosure agreement, we are therefore not able to provide more information on the progress or company.

Digital Access Management Not only physical places or objects need proper authentication before they are used, but digital content can also underlie protection. The use case pursues a platform-based approach to secure content and authorize people accessing it[10]. The main idea is the usage of so called "guards" which are services that provide the ability to check for a specific value of an attribute of the subject. For example, a guard can be a service which checks the location of the user and requires him to be in a 100 meter radius around a specific position. These guards can be combined to require different stages of authentication and authorization to implement more complex requirements.

There are four different roles in this use case: The platform provider, the guard creator, the owner of data, and the user who wants to access the data. The platform provider develops the solution on which certain guards can be developed, used, and combined. The owner of the data is able to purchase (either pay-per-use or time-based fee) the guards needed to protect his data from trespassers and to open it to allowed users.

The use case is (as of March 17) in closed-beta and cannot be accessed public. Further developments can be found at `https://keyp.io/`.

[10]This use case is similar to the use case mentioned before, but differs in the technical approach and secured object. Because of these reasons we write about this use case.

Self-organizing Co-working space Co-working spaces are a relatively young phenomena where entrepreneurs and young teams can rent small office spaces, usually facilities like kitchen, meeting rooms, and other resources are shared among all tenants. Thus, fair rates can be offered, but of course it has some disadvantages compared to own office rooms. However, small startups often use this space, especially because it provides the opportunity to exchange ideas and creativity between startups. The facility itself has to be managed by a firm which takes care of cleaning, organization of the rooms, and maintenance. To avoid costly administration (like supervised access to the rooms or a limitation of coffee machine), Blockchain technology could be used to manage little details of shared living and working in the co-working space. All organizational aspects (renting, room booking) are handled via smart contracts.

In this use case, two different roles are included: The provider which offers the co-working space, and the users. The provider enables the user to get in contact with smart contracts which allow them to use some predefined functionality in exchange for a fee. All required resources, water, internet, telephone, rooms, drinks, food, and many more could be processed over the Blockchain and smart contracts. The user would have full control over his expenses and could use the facility with very little delay without asking the owner and handling the payment process. The owner reduces his administrative cost and can take care about other more important tasks in his self-organizing co-working space.

The use case is only a vision and is not implemented yet. However, the founder of co-working space *Neuland* (`http://neuland.club/`) is planning to introduce these features in the near future.

Startup and Company Investment and Funding Platform Startups are founded frequently. One requires different resources for the success of a new venture: Knowledge, personnel, money, office space, and other factors. Traditionally, startups pitch their idea to some business angels or venture capitalists which, if the idea is promising, provide those resources in return for shares of the company or shares of profit. The process of defining this "giving and taking" requires legal advice to set up contracts which represent exactly what all participants agreed to. With Blockchain technology, these cooperations can be simplified and accelerated.

The use case involves three roles: The service provider, the startup or small company and the resource provider. The service provider connects the startups and the resource providers and works with them on individual contracts. Because the startup is placed on the Blockchain, shares can be given to the representative stakeholder and earnings are shared according to them. Shares can be sold and traded later on. The service provider supports both parties and, if he is interested, participates himself in the newly founded venture. The platform, because of its low footprint, allows small resource providers to participate, for example if only a small amount of money is provided or if a person wants to participate by giving his own knowledge and time.

The implementation is at a beta-stage and currently the founders seek for participating startups, experts and venture capitalists. The project can be found at `http://starwings.io`.

Automation of Reinsurance Contracts Insuring against risks involves complex business operations within insurance companies. They do not only provide insurances for private individuals or companies, they also take insurances at other insurance companies and specialized reassurance companies. This is required to circumvent insolvency in the unlikely event of many insurance holders suffering from damage and claiming their insurance premiums simultaneously. Therefore, the risks of events are spread across many insurance companies to lower the impact on a single company. All these agreements are textual contracts. Each contract has to be individually executed. These processes are expensive, take a long time and are error-prone. Therefore people think about modelling the contracts and their implications in a Blockchain ecosystem specifically designed for this purpose.

All reassurance companies which take part in this ecosystem have the same rights and obligations to the network. Therefore, this use case requires only one role. Every established contract is transferred to the Blockchain. They are set up such that contracts execute each other, such that if one contract is triggered, all other contracts which belong to this contract are executed, too. Trusted oracles are set up. They observe relevant information to the contract (appearance of some event) and execute the contract with some parameters, to initiate the regular process. Blockchain technology helps to automate and accelerate the process.

The described use case underlies a non-disclosure agreement, we are therefore not able to provide more information on the progress or company.

Micro Payments Payments for digital services and digital content occur frequently nowadays. Books can be purchased as E-Books, newspaper apps allow to buy the daily edition via in-app-purchases. However, the prices range from 1-10 Euros or Dollars. The range below one Euro is only used in very few cases, because the transaction and processing cost is mostly a fixed rate which would eat up all revenues. Cryptocurrencies allow the creation of profitable micro-transactions, because (depending on the network) the cost ranges from 0.1 cents (Ethereum) to 50 cents (Bitcoin)[11] which makes it feasible to use it. Furthermore, these digital currencies allow the user to commit to a payment without executing it. This allows the continued use of a service until the use stops. One does not have to publish single payments, but only one (circumventing "transaction spam"). For example, a newspaper could offer to read each article in the current newspaper for 10 Cents each. A user who stops after one article would publish his transaction with 10 Cent paid. However, a user who reads five articles would not publish five transactions, but only one. This allows to continuously use a service and to consume small contents. The provider does not have to rely on alternative earning methods like advertisement or membership.

In this general use case on micro payments, two roles are usually involved: The seller of contents or services and the buyer. The buyer would, as long as he consumes data or a service, sign transactions with the current accumulated fee and the seller would provide the data or service until the user stops signing transactions. Then, the provider stops providing and would publish the last transaction he received from the user. With this "off-chain" proceeding, double spending problems could arise. With escrow-payments and automated refunds it is possible to circumvent these problems.

[11] Block 464879 in Bitcoin has 2065 transactions, paying a fee of 1.677 BTC. At the exchange rate of that time the coins equals 2300 euros, resulting in average over one euro per transaction.

The use case is a generalized description of different approaches by companies which want to stay anonymous. However, Matthew Green and Ian Miers wrote about an approach for micro payments in decentralized currencies [53].

4.2.4.3. Use Cases independent from specific Platforms

Intellectual Property Management Managing intellectual property (IP) in real world is a complex, error-prone and expensive process. To properly register IP, one has to consult lawyers and notaries and publish information, to prove that an idea or concept was one's own idea. With Blockchain and the immutability of transactions and contents it is possible to verify that some specific information existed at a certain point of time. The basic idea is a so called proof of publication[12]. It can be facilitated using Blockchain technology, with a small difference: One does not have to publish the information itself, but only the hash of the information. Because of the pre-image-resistance property of hash-functions in cryptocurrencies it is not possible to retrieve the information from the hash [54]. If some IP-breach happens and someone uses it without any entitlement, it is possible to show the document (that computes to the hash) and the hash. This proofs its existence at that certain point of time.

The use case involves two persons: The platform provider and the user who wants to secure his intellectual property. The technical background is that the hash is contained in a transaction made by the platform provider. That said, the user is able to include the hash by himself, but the platform provides easy-to-use features and takes care of different tasks. It provides an additional backup space for the documents, so the document (which has been "published") can be found if needed. Furthermore, it supports the publication not only from single documents, but whole projects. It enables the user to secure complete project folders and enables them to reveal the concerning information and its proof without revealing any other documents from the project.

The application is in private-beta and its public-beta is planned for mid 2017. The project can be found at `https://bernstein.io`.

Process and Service Automation Several different companies offer services like web hosting. Companies build software which allows them to manage many customers which all use the same service. Most of the work is automated, because this enables the companies to reduce their costs to a minimum and increase profits through the scalability of their offering. These processes and services are automated and it is possible with Blockchain technology, too. [13]

Most systems or process-steps that occur in some form of process can be instantiated as an own and self-acting identity on a Blockchain. They can exchange information and data without the possibility of manipulation. Therefore, most use case ideas centre around the idea of placing elements of an accounting software on a Blockchain.

[12] Proof of publication describes the publication of information at a certain point of time to the public, in order to verify its existence at a certain point of time.

[13] It is therefore not a use case which requires the usage of a Blockchain, but because many companies presented it as their **Blockchain use case**, we chose to include it in this list.

Use Case	Status	Project (URL)
Ethereum		
Lottery	Discontinued	n.a.
Land Registration	Proof of Concept	https://github.com/danmermel/landregister
Competitive Gaming Platform	Beta	http://etherplay.io
Competition Platform	Beta	http://call4contest.com/
Service Platform	live in June '17	https://swarm.city/
In other or own platforms		
Energy Market Automation	Vision	https://stromstunde.de/
Digital Identity Creation and Management	Vision	http://cambridge-blockchain.com/
Rights Management	n.a.	n.a.
Digital Access Management	Closed Beta	https://keyp.io/
Startup and Company Investment and Funding Platform	Idea	http://starwings.io/
Self-organizing Co-working space	Idea	http://neuland.club/
Automation of Reinsurance Contracts	Idea	n.a.
Micro Payments	n.a.	n.a.
Independent from specific platforms		
Intellectual Property Management	Public beta mid '17	https://www.bernstein.io/
Process and Service Automation	n.a.	n.a.

Table 4.5.: Blockchain Use Cases

4.2.5. Use Case Requirements

In the interviews we talk with experts about the requirements that use cases have to fulfil to be suited for the usage in a Blockchain. We learnt that it is difficult to determine how suitable a Blockchain is for some use case, as the technology is highly diverse and can be structured in various ways. However, we propose six different measurements to estimate the fit of a use case, as there are no barriers that keep a developer from implementing a use case with a blockchain. All measurements have to be considered to check if a Blockchain is suitable. Furthermore, one has to consider all risks and downsides of the technology, as they can lead to a exclusion criterion for Blockchain technology.

4.2.5.1. Number of Nodes

The first measure is the number of planned nodes in the network. This number can be as low as one, but there exist no upper-bound. This number is important, as it defines the

decentralization of the system. The more nodes participate, the more nodes and computation power are required to manipulate the network, concerning both private and public blockchains. One has to differentiate between two numbers: The number of users and the number of nodes, as they are not the same. With this number we mean nodes, as a single node used by many people does not guarantee integrity or other security goals. Of course, the system requires a minimum number of participants[14], but in general the number of nodes at least equals (or is higher than) the number of participants.

4.2.5.2. Trust in Participants

The blockchain platform is often considered a trust-less platform. This is not entirely true, as we describe in chapter 2, because faith has to be put into developers and miners. However, a use case is more suited for a Blockchain system, if the participants do not trust each other (fully). If everyone would trust all other participants in the system, why would such a system be considered in the first place? Other (cheaper) implementations can be considered for usage, as the trust the Blockchain offers is not required. The system fills the gap of missing trust between the parties that want to work together, but do not trust each other entirely. It provides the technology, such that manipulation is just not possible (under regular circumstances), therefore the data in the chain is correct and can be trusted. If trust is mapped to a scale from "full trust" to "no trust at all", "full trust" is not considered for a blockchain project. This also explains, why we consider 2 as the minimum number for a blockchain, as a single person does not have a reason to mistrust himself. The other side however, could be highly suitable. Thinking of cryptocurrencies, every participant does not trust other participants, as everyone behaves to gain the most revenue out of the system. The less trust in participants, the more suitable Blockchain technology becomes.

4.2.5.3. Transparency

Transparency in Blockchain system affects three different areas: The transparency of money, the transparency of ownership, and the transparency of logic. By design, all information is publicly available, thus, one can see which identity transfers how much money to other participants. Considering smart contracts, the complete source code is available as every node in the network has to be able to compute it, therefore it lays open to everyone. The technology itself provides some methods for advanced privacy, such as zero-knowledge-algorithms which provide privacy regarding identities and money. However, we are not aware of smart contracts of which the source code does not have to lie open[15]. Considering transparency in general, if a use case requires a high level of privacy, Blockchain might not be the ideal solution. If the use case does not care about transparency or it is even required, a blockchain might be suited for the use case.

[14] We consider this number to be 2

[15] In fact, the code does not lie open, if the developer decides not to publish it. Despite that, the software is available in a compiled version. With big effort, it is possible to understand and rebuild the software.

4.2.5.4. Form of Data-Usage

What does the software do with the data? We make a distinction between read-intensive and write-intensive systems, as this consideration is relevant for the usage in a Blockchain. If data is written frequently, a Blockchain might not be suitable for two reasons: First, the Blockchain is slow in comparison to other approaches, as it has to distribute all data among all clients. The risk about scalability (see 4.2.3) applies. Second, the cost for writing in public blockchains can be very expensive [32]. However, if the application reads very much data, Blockchain can be considered. However, one can consider a write-intensive usage as a disqualifier for the utilization of Blockchain technology.

4.2.5.5. Digital Twin

We write about the problem of the digital twin in chapter 2 before. The existence or creation of digital twins in the use case might be a hurdle for the use of Blockchain technology. We discuss different use cases which involve the creation of digital twins before in this chapter. However, these use cases we describe work with digital twins that change very rarely, for example real estate. We consider the digital twin requirement as a scale from "real-time update" to "not existent". If a use case does not employ a digital twin, it can be implemented using Blockchain. A common example without digital twins in cryptocurrencies. The more frequently the digital twin is updated, the less suitable the use case is for implementing it with Blockchain as digital twins are write-intensive and as the Blockchain system has to be fed. As it is a transition from offchain to onchain, problems can occur regarding the quality of the data, because this oracle has to be fully trusted. A real-time digital twin is a disqualifier for Blockchain as the technology does not offer real-time information input.

4.2.5.6. Complexity

Complexity is a well suited indicator for the fit of Blockchain technology for use cases. Considering cryptocurrencies, the applications within the network are highly trivial. They basically describe a flow of coins in the form $A \rightarrow B$. Smart contracts require more complexity, as they allow for more functionality. For example, the lottery smart contract enables users to gamble on the outcome of a random number. However, as the software becomes more complex, the blockchain becomes less and less suitable for the use case because of several reasons. The execution of smart contracts costs money[16], as the miners have to execute the software. As every (full) node has to do this computation, it is obvious that the computation has to be simple. A extensive usage of resources disqualifies the Blockchain as platform for the use case.

[16] In a public Blockchain as Ethereum.

5. Synthesis of Blockchain Information

After giving an overview about the answers and use cases collected in the expert interviews, we are able to generate grounded theories about the applicability of the Blockchain technology in general. First, we describe the approach and results of a use case categorization. Afterwards, we present the structure of Blockchains, linking the use case categories to Blockchain parameters, creating a theory about Blockchain principles. Lastly, we select five distinct exemplary use cases and analyse them according to architecture layer, roles, and process. We classify them along the proposed use case categorization.

5.1. Use Case Categorization

A goal of this thesis is to understand how the functionality of Blockchain can be generalized and categorized. In this section we show how we approach this goal, explain difficulties in developing a categorization and describe our results.

5.1.1. Approach at Categorization

When starting to investigate the area of Blockchain, the main question is: What is the technology capable of? As we know, Blockchain gained popularity as cryptocurrencies are build upon this technology. Thus, it is obviously possible to transfer digital money in a decentralized system, but society does not know what to do with it besides that functionality. People came up with different ideas to benefit from the properties of Blockchain technology such as trust, speed, or pseudonymity. Additional features, such as smart contracts, were proposed. They led to more confusion, as it is not clear which needs they fulfil. We require therefore a categorization giving insight into the complete functionality of the Blockchain. As stated in chapter 2, there exist different approaches to categorize use cases in the ecosystem. However, they focus on business cases which address different needs in the ecosystem, for example provide payment services or developer tools. The proposed categorization by Mougayar is quite similar to our approach at the Blockchain architecture in chapter 3, but that is exactly what we do not want to do, because it does not answer our question about the functionality of Blockchain. For us, it is important to give a categorization for use cases (or business cases) which harness Blockchain technology to facilitate their ideas, thinking of what Blockchain can be useful for, not the other way round. So the question arises how one can create such an categorization. The goal is that should fulfill three requirements: 1) it should apply to all use cases which use Blockchain (therefore comprehensive), 2) it should be specific and precise (therefore separate between different functionalities), and 3) it should be complete (leaving no functionality out). We apply the Grounded Theory Method to generate a theory which fulfills the three requirements. We

start with the foundations from literature, read about different approaches of use cases and use case categories, and retrieve additional slices of information from the semi-structured expert interviews, until a theoretical saturation is reached.

5.1.2. Segmentation and Uniqueness

With the Grounded Theory approach we are able to propose a categorization. We start by looking at a use case and try to categorize its functionality. The first use case was Bitcoin, therefore we derived a category "cryptocurrencies". However, when considering additional use cases, the meaning of the category "cryptocurrency" changes. Considering the platform Ethereum: It is not clear whether one can assume that all tokens are coins (and therefore currency), as it is easily possible to create additional assets [55], requiring a category like "tokens". This leads to the first problem in this categorisation, as we have to carefully select the categories in between two extremes: a) selecting categories which fit a variety of use cases and therefore are highly generalized and b) selecting categories which fit only few use cases but therefore are highly specialized. These approaches either result in a very low number or a very high number of categories and do not bring an advantage to scientific research, because the first one is too generalized (representing core "functionalities" as trust) and the other one is never complete. We decided to steer a middle course, keeping in mind that this approach involves the risk of generating categories of different granularity, e.g. that one category unifies more use cases than another. Despite beign aware of this risk, this approach will generate (numerically) unbalanced categories, because some functionality of the system provides more opportunities than other functionalities, as we discuss in chapter 7.

Digging into the functionality of the Blockchain, we found that many features of Blockchain depend on each other or the combination of it leads to another functionality. Technically speaking, the same platform with the same implementation can be used for varying use cases, some complementary but others contrary. A single use case can utilize different functionalities at once which leads to a problem: It is not possible to categorize single use cases into a single category of functionality. To circumvent this issue, categories do not exclude each other, but we define three peculiarities (form 1 = "hardly any or no usage" to 3 = "primary goal of use case"). The single characteristics are defined in own terms to make a distinction clear and comprehensible. Now it is possible to assign a vector to every use case that represents the value of a functionality, leading to a "fingerprint" of a use case. If use cases are classified along this taxonomy, it is possible to measure the distance (and therefore the similarity) between single use cases, enabling to look for blockchain principles which we do in chapter 5.

As we classify our first use cases, we discover that it is possible to classify use cases as well as platforms with this characterization. As we discuss use cases and their classification, we do not consider the platform itself, but what the use case needs or utilizes. Otherwise, every use case which builds upon the Ethereum platform were categorized the same.

5.1.3. Overview

We give a detailed description of every category, their peculiarities and typical use cases of the category.

5.1.3.1. Creation of Markets

The first category is called *creation of markets*. A market, from economic perspective, is a system in which goods or services are exchanged for money. Typically, supply and demand regulate the price for a good or service. Depending on the market design, everyone can participate by buying or offering goods. In Blockchain technology, it is also possible to create markets. Most Blockchains already have some sort of money implemented, allowing people to transfer money to others. One can not only represent money, but it is possible to create any form of asset on the Blockchain. With these assets it is possible to represent any good; these goods can be exchanged for the money that exists on the Blockchain. But not only goods, but services or work can be offered, too. If the platform allows some form of computational language, it can provide atomic services. This means that the exchange of the money with the service happens in one step and is guaranteed.

In practice, the creation of markets might be the most common use case. All cryptocurrencies are a market, as the currency underlies supply and demand. Thus, there exists an equilibrium price, and one can trade cryptocurrencies for other currencies (e.g. fiat currency). The complete platform Ethereum provides the functionality to create markets. Obviously, Ethereum as a cryptocurrency can be perceived as a market, too. Furthermore, Smart Contracts and ĐApps are able to create a market within Ethereum. A Đapp takes money (the cryptocurrency) for exchange with a newly created token for the buyer. Energy markets which are created on the Blockchain are markets, trading energy for money. All investments in companies or ICOs (initial coin offering) handled via the Blockchain result in supply and demand, creating a market.

When can a use case be considered as a market? We define two different functionalities for a market creation: 1) The trading of assets and 2) the creation of new asset classes. The creation of new assets itself is not considered, as it is often the case that the currency is created as a side product of mining. A creation of asset-categories applies, if a new Blockchain is implemented to create a new token or asset. Therefore, creating new cryptocurrency can be considered with level 3, thus a strong focus on creation of markets.

The manifestations of *creation of markets* are following:

Level 1 Low / No focus on *creation of markets*: No usage of markets

- New tradable asset-categories cannot be created.
- No transfer of assets is possible.

Level 2 Medium focus on *creation of markets*: A market is used

- New tradable asset-categories cannot be created.
- The transfer of assets is possible.

Level 3 Strong focus on *creation of markets*: A market is created

- New tradable asset-categories can be created.
- The transfer of assets is possible.

5.1.3.2. Tracking and Provenance

Creating of tokens enables another category of use cases, *tracking and provenance*. In this scenario the tokens often represent some form of (real world) asset. The concept of ownership or location is modelled with public-private cryptography. Every possible owner or location is assigned a private key to which the token is transferred if the real world object changes its owner or place. This allows to check the history of an object and to follow it back to provenance, knowing who created the object. This can be applied in all forms of collaborative work in which goods are produced jointly. Private Blockchains can be used for supply chain management in order to create a trust layer to support faster and easier tracking. Another benefit arises from allowing private users to access this chain, in order to prove the origin of fair traded components. One can prove possessing an asset.

A typical use case derived from an expert interview is the land registry as seen in 4.2.4. This platform provides detailed information about how land was acquired, who possessed it, and who owned it first. Another use case is intellectual property management: One can prove the existence of information at a certain time and if the Blockchain is designed in such way that IP can be transferred, one can reconstruct the origin of ideas.

This category is often combined with the *creation of market*-category, because both use the same underlying principles (tokenisation). Additionally, it takes some effort to enable the creation of markets without the ability to track the respective assets, this therefore occurs frequently.

We define only one functionality for this category: If tracking is used or not, as there is no technical difference between tracking and provenance: If tracking is possible, the assets can be traced back until creation and, if provenance is known, it is also possible to track the path of ownership.

The manifestations of *tracking and provenance* are following:

Level 1 Low focus on tracking and provenance: No tracking

- It is not possible to track any assets or ownership.

Level 2 Medium focus on tracking and provenance: Some tracking

- It is not possible to track single assets, but transaction chains.

Level 3 Strong focus on tracking and provenance: Focus on tracking

- It is possible to track single assets and their composition back to the origin.

5.1.3.3. Autonomous Entities

Public-private cryptography allows to create new identities which can act on the Blockchain. Using cryptocurrencies people own wallets and can buy items with their money. They can approach any other participants in the system and trade with them. Not only natural persons can participate in this platform, but (as in the real world) one can think of other forms of entities: Companies (or legal person under private law[1]) and autonomous machines. Companies can solely live on a Blockchain. We are able (with Turing-complete programming languages) to

[1]German judicature

create software which embodies a company and its typical components, from shareholders to voted chief executive officers. However, we are not aware of autonomous machines taking part in every day life, because handling a complex world is too difficult for a machine, so machines are used only if clear interfaces exist. For example, there exist trading algorithms which buy and sell stock market shares. The Blockchain can be viewed as a more advanced interface. Of course, the internet as we know it can also be perceived as an interface, but interactions between users and service providers are not standardized, requiring software adapted for the individual use case. However, in Blockchain three fundamentals are standardized: 1) payments, 2) identification, and 3) service fulfilment. It is possible to use these three fundamentals to create entities that only exist and "live" on the Blockchain. Such entities are able to possess, trade, buy, or sell goods or services autonomous. Connecting these autonomous entities with hardware, one can think of hardware-provided services. We describe the possible use case hereafter.

Typical use cases for this category are all sorts of smart contracts, as they create an independent entity which works without the creator operating it. It depends on the single use case if all functionality is executed autonomous or if there is some functionality which has to be executed manually. For example, the lottery use case (see 4.2.4) could be designed autonomously. At current stage, the operator has to trigger the drawing, but it is possible to design a smart contract for a lottery which is capable of autonomously drawing numbers. Connecting with hardware, it is possible to create rooms or buildings which sustain themselves, requiring every visitor to pay a fee. With the earned money the entity would be able to pay personnel, just as described in the use case *Rights Management* in chapter 4.

We differentiate between two functionalities: The programming language and the autonomous interaction. A Turing-complete language enables user-defined operations and more complexity in the system. The second functionality is if the use case actually facilitates the blockchain to create an autonomous entity.

The manifestations of *autonomous entities* are following:

Level 1 Low focus on autonomous entities

- The programming language is not Turing-complete.
- Autonomous entities are not used or created.

Level 2 Medium focus on autonomous entities

- The programming language is not Turing-complete.
- Entities are used or created.

Level 3 Strong focus on autonomous entities

- The programming language is Turing-complete.
- Entities are used or created.

5.1.3.4. Information Storage & Retrieval

The category *information storage & retrieval* can be perceived as highly general, as every blockchain somehow stores data and information. For this category it is not important to know if data is stored, but what purpose for. As we know from chapter 4, Blockchains are not designed to work at the same speed, yet provide the same functionality as relational databases. However, there are use cases in which Blockchain is favoured for information storage. These use cases require functionality which relational databases cannot give: Trust in a decentralized system without a central operator. This applies to situations in which data or information is shared among participants, but the integrity and availability of the files is crucial to the network and the use case.

One can think of different use cases facilitating information storage and retrieval. A use case described by one of the experts is the management of intellectual property. The use case to save the hash of important documents to later on retrieve it in order to proof its existence at a certain point of time relies on it. Clearly, the Bitcoin blockchain is not designed for such applications, but it works nonetheless. Other use cases which require storage of information is the *Digital Identity Creation and Management* use case. Personal information is stored on the blockchain, validated by either a publicly recognized institution or an autonomous service, in order to reveal it later on to proof certain properties.

We separate the three levels according to the usage of the data. The first level describes the storage as it is necessary for transactions or service provisioning (in smart contracts). The second level applies for information storage in which data is lateron used again for calculations. For example, a lottery use case (the storage of the chosen numbers) would apply. The third level applies as data is stored for retrieval later on, as it can be used in other systems or provides a sort of backup.

The manifestations of *information storage & retrieval* are following:

Level 1 Low focus on information storage & retrieval

- Information is only stored for the immediate execution of transactions or services.

Level 2 Medium focus on information storage & retrieval

- Information is stored as intermediate results which are used in later calculations.

Level 3 Strong focus on information storage & retrieval

- Information is stored for the purpose of retrieving and publishing it later.

5.1.3.5. Meta-Consensus

Consensus is a fundamental functionality of Blockchain technology, as all nodes have to agree upon the same chain [2]. In this use case category it is not about the consensus reached by all nodes in the network, but operators utilize the platform for finding and reaching a consensus about some topic without manipulation or fraud. Public keys belong to a entity, therefore all actions committed with this public key can be linked to this entity, whether or not the entity behind the key is known. A blockchain or a smart contract could be set up in such way that it is possible for the single entities to record their own opinion on some topic. With a

standardized approach it is possible for a group of users to agree on one decision[2].

What use cases belong to this category? Certainly, all kinds of votes: It is possible to hold elections on a Blockchain. A government issues a private key to every citizen. The Blockchain is designed to allow that everybody can vote only once and no linkage between public key and vote is created. This approach has several advantages over current voting procedures, as it is fast, votes are counted automatically and everybody can participate in the network and calculate the results by himself. Another possible use case which is closely linked to another use case category is the shareholders' voting in DAOs. As a new DAO is created, all shareholders find consensus about investment or possible staffing in this new firm.

For the description of this category, we have to think about other types of consensus in Blockchain technology. Reaching consensus about the state of a blockchain does not influence any use case, as it is the main goal of every Blockchain to reach consensus and prevent disagreement, therefore it is not further considered. However, there is another type of consensus that commonly appears: Multi-party transactions. If the word consensus is interpreted in a strict sense, multi-party transactions would fall under the definition of it, as it is defined as a "general agreement" [56]. Multiple parties agree that a certain transaction should be created and executed in some way, therefore it is a form of consensus. But as we defined in the previous footnote, we draw the line between the type of consensus of multi-party transactions and of shareholders voting or election in which all participants are required to have the same opinion. Elections and shareholders voting leads to results, whether or not all entities share the same opinion. A multi-party transaction can only be executed if all participants agree.

The manifestations of *Meta-Consensus* are following:

Level 1 Low focus on Meta-Consensus

- A meta-consensus is not possible or required.

Level 2 Medium focus on Meta-Consensus

- A meta-consensus can be reached between a low number of entities; every single entity has to participate actively and with the same goal to reach consensus.

Level 3 Strong focus on Meta-Consensus

- Meta-consensus is the main goal and can be reached without all entities participating or sharing the same opinion.

5.1.3.6. Communication

Public-private cryptography is widely used for secure communication between two participants. 27 % of the top 10.000 websites worldwide offer encryption [57], PGP [58] applies RSA encryption for text e-mails, every secure connection is established via cryptography. Instead of using other means of transportation, one can use a blockchain for transferring messages between two or more entities. Relying on Blockchain would solve the problem of repudiation[3], as every message is recorded on the chain and cannot be deleted afterwards. Blockchain

[2] In this case, consensus does not mean that all participants have the same opinion about a topic. They only agree that the result of the referendum or vote counts and is carried out

[3] A person is not able to take a message back or to deny its existence.

per se cannot replace any kind of communication, as the technology would be too slow for instant messaging. However, there exist valid use cases for the category of communication. Especially for security goals (confidentiality, integrity, availability, authenticity, non-repudiation, accountability) Blockchain technology provides a solid foundation.

We are not aware of many communication use cases, even though Ethereum includes its own communication protocol called whisper [59]. The use case we propose and implement gives an insight about how use cases could look like (see 6). The use case allows two or more lawyers to collaborate on the draft of a new contract. The current state of the document is always saved in a new block enabling full transparency about the creation process and the responsible person. With its decentralized design it is easy to add additional servers which serves as backups. Of course, such systems would be only private accessible, but the advantages cannot be denied.

The manifestations of *Communication* are following:

Level 1 Low focus on Communication

- Entities do not communicate.

Level 2 Medium focus on Communication

- (Autonomous) entities and users interact and communicate.

Level 3 Strong focus on Communication

- Blockchain technology is used for communication between humans.

5.1.3.7. Identity Management

Blockchain networks require to provide an identity in form of a key-pair. Activities are assigned to those identities, such that no action can be manipulated or faked. This attribution is a core functionality of Blockchain, as is renders the network and its content transparent and comprehensible. Usually, one can create such identities at will and a single user can possess multiple identities in a network. However, the network and its applications can impose requirements on an identity to access a service. The management of authenticated identities is not a core functionality of the technology, but is important in some environments. For example, the technology has to ensure that only the owner of a smart contract has the possibility to spend his earnings, no other identity should be allowed to access this functionality. Another possible question is how a Blockchain can connect the real-world identity of a person or a company to a public key in the Blockchain. This is a requirement, if the technology is used for legal transactions in which participants want to know the person they trade with.

From our expert interviews, the following use cases mainly belong to this category: The self-organizing co-working space has some legal requirements: When the owner rents out space to some person or entity, he has to know the name of the person for his own safety in case of damages. If he wants to get this information validated via Blockchain, a specialized service is needed which provides and validates the respective information. This leads us to exactly the use case *Digital Identity Creation and Management*.

The manifestations of *Identity Management* are following:

Level 1 Low focus on identity management: No authentication

- There are no requirements on the identity to participate in the use case.

Level 2 Medium focus on identity management: Authentication

- There are some requirements on the identity to participate in the use case.

Level 3 Strong focus on identity management: Real world authentication

- The identity has to be linked to a real world identity.

5.2. Principles

As we discussed in the previous section 5.1, the Blockchain can be used in various different ways to accomplish goals. However, similar concepts underlie different categories, leading to the question what influences the categories have on the underlying parameters of the technology. After proposing a categorisation for the use cases utilizing Blockchain, we are able to propose a grounded theory about principles of the Blockchain. One can perceive Blockchain principles as components or elements of Blockchain technology which are required to facilitate the use of some functionality. They define the existence and configuration of parameters, enabling the usage of Blockchain in different ways. The goal is to create a list of principles allowing one to think of different combinations of principles in order to build new Blockchains with different functionality.

5.2.1. Differentiation

We have to differentiate between Blockchain principles and another related term: The software design pattern. While the software design pattern is perceived as a general reusable solution which solves a commonly known problem[60], the term "principle" lacks a clear definition. We define the term principle as a description of a given functionality. As we write about Blockchain principles, we neither provide a best practice nor a solution, just a description of components and their functionality. We describe what the principle does and not how it achieves it. If there are common approaches to some principles, we will name these approaches without any evaluation or comparison.

5.2.2. Derivation

The principles are derived from different sources. We look at different use cases and their categories and find similarities to build up a theory about the principles. Additionally, we look at platforms and how approaches are implemented and derive principles from that. The proposed list is not exhaustive, as future developments will add additional principles and functionality we cannot foresee. To find Blockchain principles one has to start with a foundational concept on which all other principles can build upon. The Minimum Viable Blockchain can serve as a theoretical approach.

5.2.3. Minimum Viable Blockchain

We define the Minimum Viable Blockchain (MVB) as a theoretical concept which comprises all traditional blockchain approaches. To do this, we give a short recap about hard and soft forks from chapter 2: Hard and soft forks allow to extend or limit the functionality of a Blockchain. Both terms imply the relationship between the former blockchain and the new blockchain, as one chain includes the other. For example, the sets of rules a and b define a blockchain each (α,β), such that all α are a soft fork of β and all β are a hard fork of α. Lets assume that A is the set of all α, and B is the set of all β. As the rule set a is more restrictive than b, and all α are a soft fork of any β, we know that $A \in B \wedge B \notin A$. The MVB is a set of rules m and the corresponding set of all blockchains M which fulfil m such that $\Omega \in M$ with Ω the union over all possible blockchains for all rule sets. Thus, m defines fundamental rules which must be fulfilled by all blockchains. It is uncertain if such a Minimum Viable Blockchain can exist without losing core functionality (such as the concatenation of blocks) of the technology. For the sake of this approach we define the MVB as a blockchain in which blocks of arbitrary data are connected with the single rule that the longest chain is considered the right one. Of course, a blockchain designed in that way cannot productively be deployed, but it lays the foundation for the principles we describe in the next section.

5.2.4. Overview about Principles

To give an insight in the Blockchain principles we first want to structure them according to the proposed architecture in chapter 2. We separate all principles into three tiers: The *network-access & participation* tier (representing infrastructure and infrastructure service), the *data* tier (representing infrastructure service) and the *operation* tier (representing application). The minimum viable Blockchain is placed on all three tiers, as it does not obey to any restrictions.. After describing the tier, we give a linkage to possible parameters. As we already introduced the MVB, we continue with the network-access & participation tier.

5.2.4.1. Network-Access & Participation Tier

In the network-access & participation tier we define all approaches to limit the access to the underlying network to a specific user group or the entire population. Depending on the selection of principles it is possible to define the Blockchain as public, private, or permissioned.

Secured Access Principle The secured access principle defines that only authorized nodes can participate in the network. A node has to somehow authenticate itself, either by knowing a secret or by performing a challenge response authentication. Authenticated nodes can enter the network and read all its contents, however is not allowed (yet) to publish data or blocks. All other nodes are excluded from the network and can therefore not participate.

The secured access principle is used in private Blockchains. Therefore, it can be linked to the parameter 2.2 (audience of the network).

Secured Block Creation Principle The secured block creation principle restricts the access to creating new blocks. Nodes or entities get restricted analogously how the access to the network is restricted. The same methods can be applied with different parameters to be able to distinguish between the right to enter the network and to create new nodes. This principle is independent from the secured access principle, as the creation of blocks could also be restricted in a public network. However, if both principles are applied, a node has to have both rights to publish blocks as he has to read preceding blocks to create new blocks.

The secured block creation principle is a basic principle used either in centralized or hybrid blockchains. Therefore, it can be linked to the parameter 2.1 (type of Blockchain) and 4.2.1 (entity which is allowed to generate new blocks).

Average Block Creation Time Principle The average block creation time principle determines that new blocks should be created at a given speed. This supports the system, if the secured block creation principle is not applied, preventing block spam and sybil attacks. A well known implementation would be the Proof of Work or Proof of Stake consensus algorithm.

This principle is only used in decentralized blockchains, as otherwise this principle is not required. It therefore links to three parameters: 4.1.1 (consensus algorithm), 4.1.2 (block propagation time) and 4.1.3 (difficulty recalculation), as these are required for this principle.

5.2.4.2. Data Tier

All principles on the data tier influence the structure and usefulness of all contained data of the blockchain. By choosing one, one defines the functionality.

Identity Principle The identity principle ensures that all data that is put into the system belongs to some identity. Thus, every information can be accounted to a single entity. It depends on the actual implementation, but usually public-private cryptography is the central part of it. Without this principle it is possible that information can be stored anonymously. However, the identity principle is the foundation for other principles as we see next.

This principle can be linked to the parameter 1.2 (algorithm of asymmetric cryptography), as it is required to handle identities.

Tokenization & Transaction Principle The tokenization & transaction principle is fundamental to all cryptocurrencies. It enables the creation and transaction of assets. One has to create a ledger in order to keep track of the assets and their transfers. Utilizing this principle, one has to devise how assets can be created as they are worthless if they can be created at will. Possible ways are to link the creation of new assets to the creation of blocks or previously define all assets in the first block. However, this principle requires the identity principle as foundation, otherwise it is not possible to create a link between owner and asset.

This principle cannot be linked to a parameter from our list. However, possible parameters could be the data format in the Blockchain, as some sort of ledger would be required. As we are not aware of other data formats, this parameter was left out.

5.2.4.3. Operation Tier

The operation tier covers all principles which are needed for the execution of software in a Blockchain system. Instead of connecting every single principle from this tier to single parameters, we connect this tier to parameter 5.1, as it defines the functionality and extent of the available programming language. The three principles depend on this single parameter.

Execution Principle The execution principle defines how the blockchain stores and executes software or source code and how the source code interacts with other functionality of the system. As we know, blockchain technology allows to define the software language defining how complex software can be. There are various ways execute software, one of the common known approaches is the usage of Ethereum and the Ethereum Virtual Machine. The functionality of the network is restricted to the given language. A Turing-complete language is also required for the creation of DAOs.

Communication Principle As we have multiple identities in the Blockchain network, some use cases require to deliver messages only meant for a single recipient. Entities within the Blockchain can access messages from other entities to further compute data. Messages can be exchanged between entities as smart contracts or users that interact with them. The message format has to be defined, but the content usually remains transparent in the Blockchain. It requires the execution principle, if these single entities are autonomous.

Oracle Principle The execution of software in the Blockchain sometimes requires additional information from outside the blockchain ("offchain"). However, no principle covers this functionality. The approach is to create the possibility to fetch data from external sources without putting it in the network manually. This principle requires the execution principle and the communication principle, as otherwise the entities could not talk with the oracles. As before, Ethereum is to our knowledge the only platform that supports oracles.

5.3. Use Case Analysis

Based upon proposed architectural views, parameters, conducted expert interviews, use case classification and Blockchain principles we conduct an analysis of selected use cases. First, we present the selected use cases we chose from the interviews. Second, we classify according to proposed theories, explain our decisions and compare the results. Lastly, we give an overview about the analysis.

5.3.1. Selected Use Cases

We select interesting use cases which deviate from each other in order to cover the varying usages of Blockchain technology. To recap, we give a summary of every selected use case.

Principle	Description
Basic Blockchain	
Minimum Viable Blockchain	Theoretical construct as foundation for all other principles.
Network-Access Tier	
Secured Access Principle	Only authorized nodes are allowed to access the network.
Secured Block Creation Principle	Only authorized nodes are allowed to create blocks.
Average Block Creation Time Principle	Blocks are created at a certain speed.
Data Tier	
Identity Principle	All data is accounted to unique entities.
Tokenization & Transaction Principle	Enables the creation and transfer of assets.
Operation Tier	
Execution Principle	User-defined software can be executed in the Blockchain.
Communication Principle	User-defined software can communicate with each other.
Oracle Principle	User-defined software can fetch data from sources outside the Blockchain.

Table 5.1.: Blockchain Principles

The first use case is *Lottery*. Users are able to bet on a certain outcome. If they guessed right, they receive an monetary reward. The creator of the smart contract keeps a fee. The second use case is *land registration*. Real estate is managed in a Blockchain, enabling people to prove that they posses some property. Bootstrapping may be the hardest part. The third use case is *Energy Market Automation*. Supply and demand of energy is decentralized and take place in a Blockchain network. People are able to directly buy and sell energy without the need of third parties. Intelligent hardware (e.g. electric cars or fridges) can participate, too. The fourth use case is *Rights Management*. We assume that the building is fully decentralized and governs itself, giving access to its members for a fee. The fifth and last use case is *Intellectual Property Management*. Documents about IP are hashed and stored into a Blockchain to later on reveal its existence at a later point of time.

5.3.2. Integration into overall Architecture

We first discuss on which layers of the Blockchain architecture overview the use cases are implemented. There are five different layers: Infrastructure, infrastructure service, application, business service and business. Further details can be found in chapter 3.

The first use case, lottery, is only active on the layer "application", as it is only a smart contract. The smart contract is in the application layer, as it is a user-defined application in the Blockchain. The use case does not provide any additional functionality, as the communication

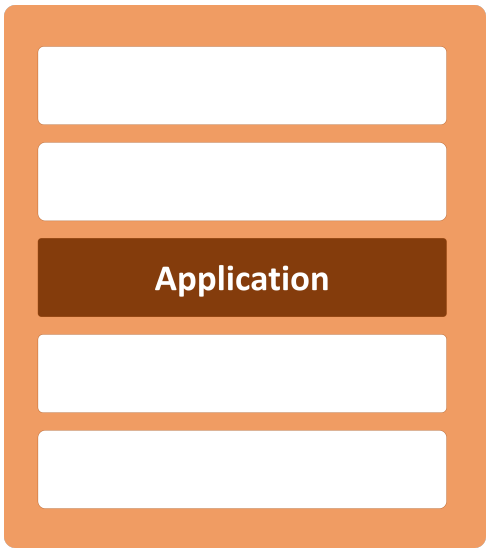


Figure 5.1.: Active Layers *lottery*

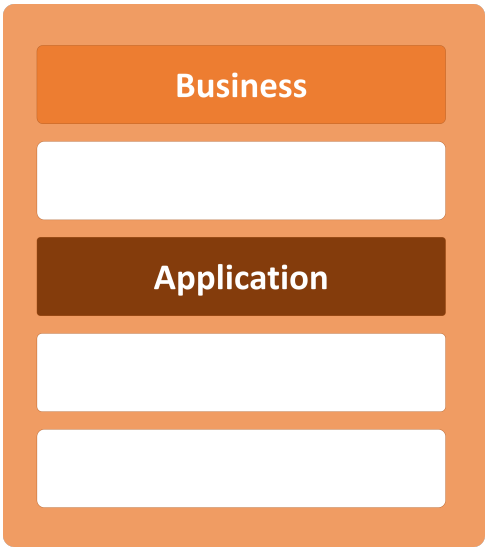


Figure 5.2.: Active Layers *land registration*

and the interaction with the smart contract can be performed by specialized browsers like Mist [61]. Underlying functionalities like integrity are provided by the Blockchain (in this case Ethereum) itself. The active layers can be viewed in 5.1.

The second use case, land registration, is operating on two layers: Application and Business. The use case does not only provide access to the smart contracts itself, but provides a frontend for which the land-registration can be conducted. As before, the smart-contract is active on the application layer. The frontend to register land or transfer it to other entities operate on the business layer, because the user does not have to interact with the smart contract directly. The Business Service layer is not required as the Ethereum Blockchain provides the access to the smart contract. The active layers can be viewed in 5.2.

The third use case, Energy Market Automation, stretches over all levels, as participants are involved in all Blockchain components because of the complexity of the system. The infrastructure itself has to be adapted on a decentralized energy network. The mechanisms for energy production, buying, selling, price calculation and consumption have to be defined in the layers infrastructure service and application. The Blockchain itself provides a service for retrieving information from the system and businesses build on top of it to connect cars and intelligent hardware with it. The active layers can be viewed in 5.3.

The fourth use case, rights management, is operating on the top three layers: Application, Business Service and Business. As the autonomous entity (the building) can live on existing Blockchains like Ethereum, one does not have to take care about the underlying blockchain and network. The entity lives on the blockchain as a complex form of an smart contract. To connect the real building with the smart contract, other layers have to be used. The building contains some functionality to access the blockchain and the smart contract, therefore it offers a form of business service. The use case is active on the business layer as it provides the possibility to interact with the entity without communicating directly over the network. The active layers can be viewed in 5.4.

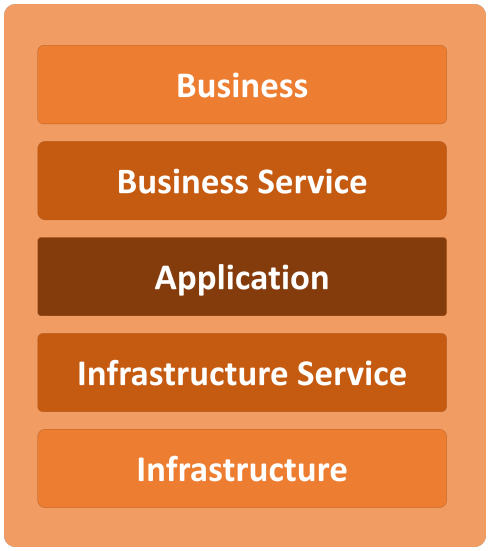


Figure 5.3.: Active Layers *energy market automation*

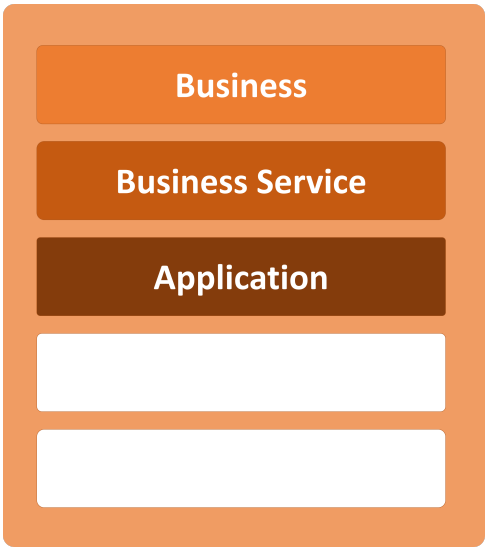


Figure 5.4.: Active Layers *rights management*

The last use case, Intellectual Property Management, stretches over three levels: Application, Business Service and Business. The operator defines the required transaction with its content and inserts it into the blockchain. Therefore the application tier is active. Additionally, they provide the frontend and the backend to upload the files and insert their hashes in the blockchain. Thus, they are operating on the business service and business level. The active layers are displayed in 5.5.

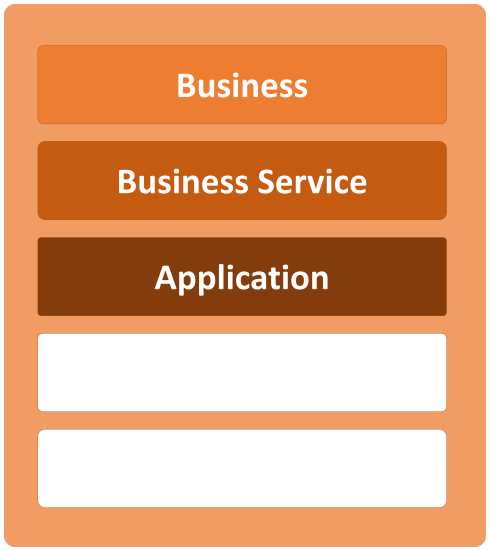


Figure 5.5.: Active Layers *intellectual property management*

5.3.3. Integration into Architecture Roles

After the first integration in the Blockchain architecture overview, we look which roles are involved in the selected use cases The roles in the network are separated in two groups: the developers and the operators. The developers comprise three roles, the application developer, the BC application developer and the infrastructure developer. The operators are divided up in an ID holder, a light node, a full node and a miner. Additional information about the roles is available in chapter 3.

The first use case, lottery, just involves two roles: BC application developer and ID holder. The player which bets on the outcome of the lottery is just an ID holder, but the owner of the smart contract is an BC application developer and an ID holder, because he has to be able to initiate the drawing and to receive his own fees. For that, he has to have an ID.

The second use case, land registration, requires the same two roles: BC application developer

and ID holder. In this use case, the software developer is only responsible for the smart contract, thus is a BC application developer. The other two roles described in the use case (land owner and government) are both ID holders, as they are just responsible for the transfer of tokens representing real estate. No additional roles are required.

The use case Energy Market Automation requires multiple different roles, as the use case is complex and comprises all architectural tiers. If such platform is designed from ground up, all three developer roles are required as they have different tasks in this new platform. Concerning the operators, it is hard to make a point which role is included and which not, as it depends very much on the design of the chain itself. As stated in chapter 2, miners ensure the integrity of the network. Thus, they are needed if a new platform is created. Of course, ID holders are required as an ID is needed if the entity wants to participate in the energy market. The need for light nodes or full nodes cannot be specified, because it depends on the concrete implementation.

The fourth use case, rights management, involves three roles according to the interview: The property, the system validation authority and the user. As recalled from the architectural overview, two developers are needed: The application developer covering business and business service tier (responsible for the software and hardware inside the propery, aka the system validation authority) and the BC application developer as he is responsible for setting up the respective smart contract. The user has to be an ID holder, in order to interact with the building and the smart contract, otherwise he were not be able to pay fees for the entrance.

The fifth use case, Intellectual Property Management, depends on three roles: Two developer roles and one operator role. One person can take over both developer-roles, but two different programs have to be implemented: The functionality within the Blockchain to store information as well as the system that allows the user to use the provided service. The user is involved as an ID holder as he is participating in the transactions on the Blockchain.

5.3.4. Integration into Use Case Classification

In this chapter we propose a classification for use cases and their functionality in Blockchain technology. As we cannot classify a use case directly to some class, we have to determine the manifestation of a use case class for each use case. With this approach a vector is generated for every analysed use case. We use radar charts to represent these vectors. We reason the decisions for every use case. Afterwards, we give an overview about all use cases in one diagram and explain similarities and differences.

Lottery The value for market creation is 1, as the use case does not introduce a new market or connects supply and demand. The value for tracking & provenance is a 2, as it is not required, but it is possible to track who won drawings. The value for autonomous entities is 3, as the provided service runs automatically and can be perceived as an own entity. The owner is only required to execute the drawing. The value for information storage is 2, as the information about the numbers that have been bet on are only required for the determination of the winner. The value for meta-consensus is 1, as the participants do not have to reach a consensus, because the winner is determined by a single party, the contract itself. The value

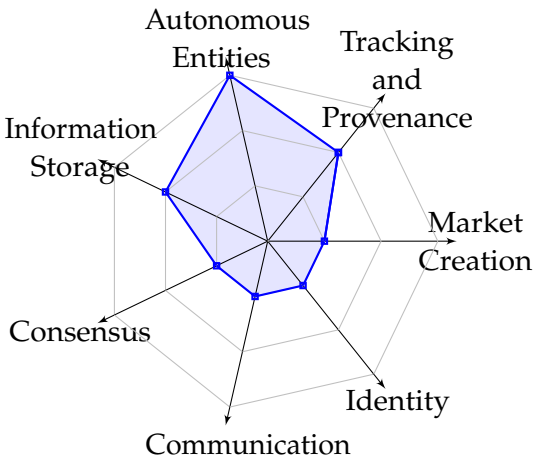


Figure 5.6.: Categorization Lottery

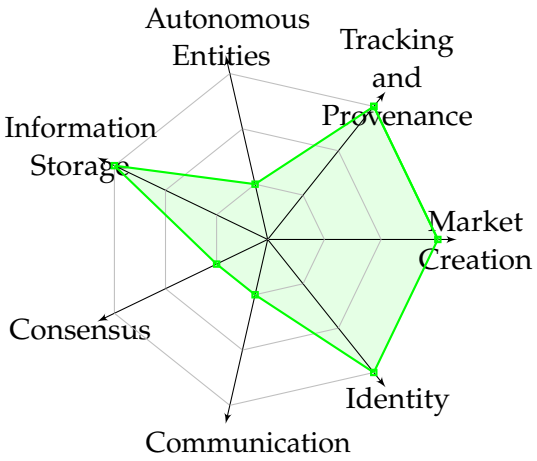


Figure 5.7.: Categorization Land Registration

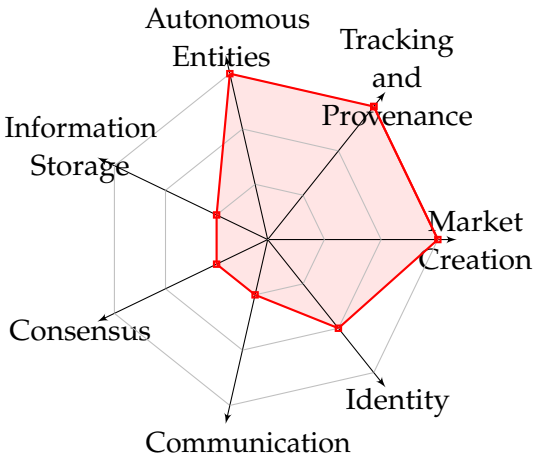


Figure 5.8.: Categorization Energy Market

for communication is 1, as no transfer of messages happen between the participants. The value for identity is 1, as no special requirements are imposed to participate.

Land Registration The value for market creation is 3, as this use case allows to create new forms of tokens, because property can be combined and divided. These new tokens can be traded, too. The value for tracking & provenance is 3. The use case allows to track how real estate changed over time and trace it back to its original owner. The value for autonomous entities is 1, as the smart contract does only provide a market. It does not provide any opportunity to interact with the contract itself. The value for information storage is 3, because information about real estate is used to later on prove the possession of it. The value for meta-consensus is 1, as there is no consensus to achieve, because the contract defines the possessions. The value for communication is 1, as no communicating happens between the owners over Blockchain technology. The value for identity management is 3, as it is very important to be able to account a digital identity with a real one.

Energy Market The value for market creation is 3, as the use case brings together supply and demand. The value for tracking & provenance is 3, as the use case supports the tracking of energy and finding out from what source the energy came. The value for autonomous entities is 3, as the power plants could be represented in the Blockchain by smart contracts which sell energy at a specific price. The value for information storage is 1, because the information is only stored for the sole purpose of executing transactions. The value for meta-consensus is 1, as there is no meta-consensus to reach, only agreements between two parties can exist. The value for communication is 1, as no communication between participants take place. The value for identity management is 2, because the identity plays partially a role in the system.

Rights Management The value for market creation is 1, as no new market is created or used. The value for tracking & provenance is 2, as it is not the main purpose of the use case to track, however in case of occurring damages tracking could be used to find the person responsible.

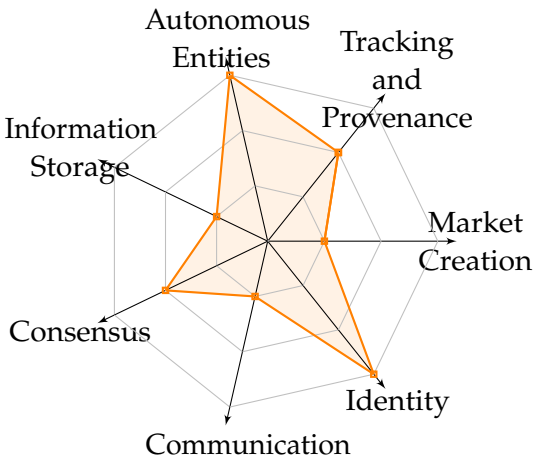


Figure 5.9.: Categorization Rights Management

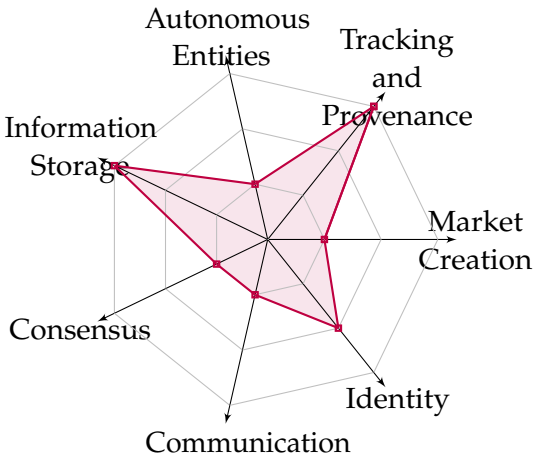


Figure 5.10.: Categorization Intellectual Property Management

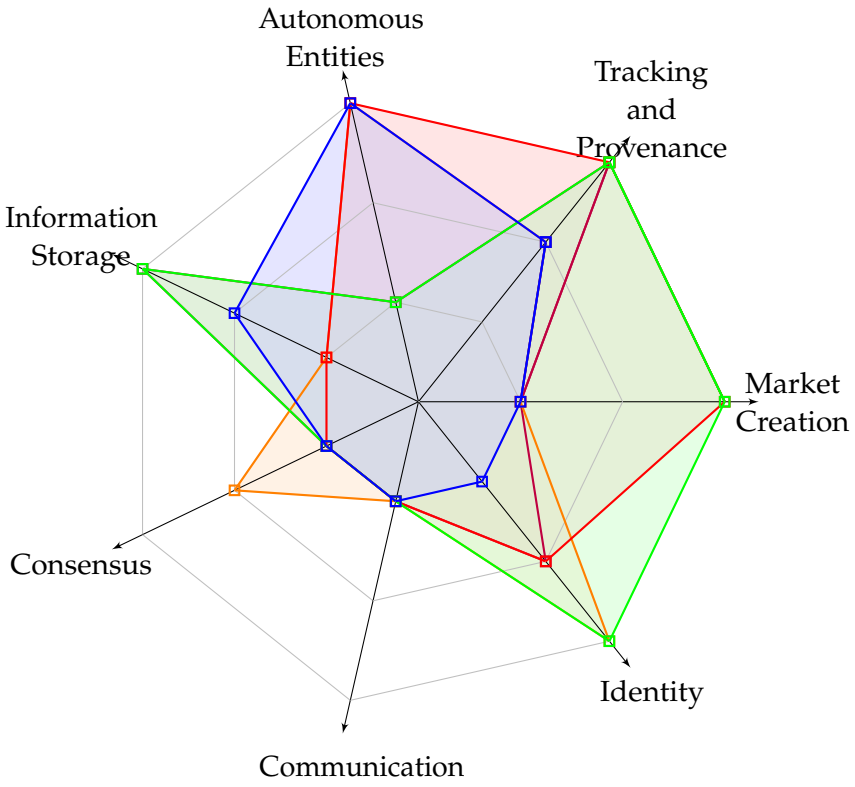


Figure 5.11.: All categorizations

The value for autonomous entities is 3, as the building or object acts independently and autonomously. The value for information storage is 1, as one does not have to store any information except data needed for the functionality. The value for meta-consensus is 2, as it is possible that members of the entity can vote on future decisions. The value for communication is 1, as no communication takes place. The value for identity management is 3, as only authorized users are allowed to use the facility.

Intellectual Property Management The value for market creation is 1, as no market is used or created. The value for tracking & provenance is 3, because the use case relies on the functionality of transparency, such that transactions can be retrieved years later in order to proof IP. The value for autonomous entities is 1, as the transaction stored in the blockchain is very basic. The value for information storage is 3, because the sole purpose of the transaction is to store information in the blockchain which contains the important data. The value for meta-consensus is 1, as no consensus has to be reached within the platform. The value for communication is 1, as no form of communication takes place. The value for identity management is 2, as the service provider has to be able to recognize the corresponding user and account the transaction to him.

In figure 5.11 all use cases can be viewed. The values are taken from table 5.2. This shows that the five use cases fit in the proposed categorization.

Category	Use Case 1	Use Case 2	Use Case 3	Use Case 4	Use Case 5
Market Creation	1	1	3	3	1
Tracking & Provenance	3	2	3	3	2
Autonomous Entities	1	3	3	1	3
Information Storage	3	1	1	3	2
Consensus	1	2	1	1	1
Communication	1	1	1	1	1
Identity	2	3	2	3	1

Table 5.2.: Overview Use Case Categorization

6. Prototypical Implementation

In this chapter we present the prototypical implementation of the selected use case with Blockchain technology. The goal is not to create software which can be used right away and includes all required functionality, but which applies the fundamental elements of Blockchain technology. The goals of the implementation are 1) to show how a use case can be implemented and executed by a Blockchain, 2) to give a framework of a *Minimum Viable Blockchain* on which other researchers can build upon, create own applications, and conduct own experiments and 3) to give an educational view into the mechanics of the technology and possible elements of it. Further, it helps evaluating theories of the chapter before. The chapter is separated in conceptual thoughts about the implementation, the design of the software and afterwards software technical aspects of this implementation. We refer to this use case as Collaborative Contract Creation. The source code is available online[1].

6.1. Conceptual Thoughts

At first, we describe the use case and how the application of the classification from chapter 5. Afterwards, we explain the infrastructure we used and how the setup and basic communication works.

6.1.1. Use Case

The scenario is as follows: Two or more lawyers represent two different clients who want to create a legal document. These lawyers want to communicate about the specific contract that has to be set up. All lawyers want to create the draft with the following requirements (analogue to IT-security goals [54]):

Speed The draft is created collaboratively and changes are recorded immediately.

Confidentiality All transmitted information can only be read by authorized lawyers.

Integrity All submitted information cannot be altered afterwards in any way. All changes are always visible.

Availability The system itself allows to add redundancy without any changes to the original system. Additional server capacity can join the network at any time to participate.

Authenticity All information stored within the system can be proven to be original and true.

Non-repudiation It is not possible to repudiate the creation of information once it is inserted in the system.

[1] `https://github.com/Basolato/naivechain_lawyer`

Accountability All actions executed within the system can be accounted to one entity.

Decentralization The structure of the network is decentralized. The functionality of a node does not depend on the functionality of another node or a central instance.

For this simple use case a private Blockchain can be considered. Both parties have to set up a node and connect them together to build up the network and provide the given functionality.

6.1.2. Infrastructure

We discuss if we use an already existing platform or develop a new platform. There are arguments for both approaches: If an already existing platform is used, one can focus on the implementation of the actual use case. Although, by creating a new platform we are able to design the underlying technology by ourself, gaining a better understanding about it. Additionally, it would allow us to design the platform in a way that enables potential future development. We decide against a greenfield approach, and rather use an existing platform, as spending a lot of time on implementation and not being able to create a platform for further research did not seem promising. Instead, we use an already existing platform with a very basic implementation. Our requirements were 1) easy introduction into the framework, 2) no additional functionality than a basic blockchain, 3) an easy to use interface, and 4) a high-level programming language.

After some research we found a 200 lines of code (LOC) implementation of a Blockchain called *Naivechain* [62]. Naivechain implements a basic structure in which Blocks between nodes can be exchanged such that it follows the rules of a typical consensus algorithm. The node keeps connections to other nodes, receives, and validates blocks without any additional algorithm like Proof of Work or Proof of Stake. The software does not further specify the structure of the contents of the block. Only five fields are required in the block: The block-height, a timestamp, the previous blockhash, the own hash and the attached data. Beyond that, no further features are implemented: It is not possible to tokenize, to write executable contracts, or to use basic language. The structure of the data is not further defined. The Naivechain does not implement any type of frontend, only offers a basic API.

The code is written in server-side Javascript using node.js. The node.js-Server connects to other servers which run the same software. Together they build up a decentralized network. Every single server can be accessed by HTTP-requests (GET- and POST-requests), e.g. getting a list of all Blocks, creating new Blocks, or receive a list of connected nodes. Because of the missing frontend, we create a website with AngularJS and Bootstrap.

The network is deployed locally on one machine, as the implementation requires only little computation power. It is possible to deploy the servers on separate machines, but the functionality does not depend on the used hardware or network setup. Therefore we choose to only use one machine.

6.2. Design

The Naivechain implementation is very plain. Therefore, the complete source code was restructured and object orientation was introduced. In the following section we explain the data model, the data flow, and the abstractions made.

6.2.1. Data Model & Propagation

As stated before, there is a strict separation between Frontend and Backend. However there are two connections: 1) the frontend communicates with the backend via API and 2) the frontend is served by the backend. The bottom line is that every request either is a request to the API or is to retrieve the source code for the frontend. We do not give a closer look into the frontend, because the frontend does not contain any logic and can easily be replaced by any other Angular frontend. We give a closer look in the setup of the backend, as it contains everything relevant for the Blockchain.

In the development of the backend we focused on three things: 1) high cohesion, 2) low coupling, and 3) the ability to easily exchange classes and functionality. The first property ensures that functions that are tied close together are placed in the same or a related class. The second property helps that the connection between classes is as small as possible [60]. With this, other researchers are able to reuse the code and implement other use cases on top of it. Single classes that are responsible for a certain functionality can easily be exchanged and the system changes accordingly.

The data model can be found in figure 6.1. As we can see, the backend is built up from different classes: `block`, `blockchain`, `contract`, `crypto`, `integrity`, `node` and `ruleset`. The functionality of each class is described as follows.

block The block class provides the basic element of a Blockchain: The single block. It stores the content of a block. It consists of the index or height, a nonce for the mining puzzle, a hash from the previous block, a timestamp of creation, the data, the status, information about the creator, the hash, and the signed hash.

blockchain The blockchain class is just the container for instances of blocks and instances of integrity-blocks. It only deals with the appending and retrieving of the information. The class itself does not define any rules or apply them, it is only the executing object.

contract All information and methods about the legal documents are stored in the contract class. It deals with all versioning of the document and how it is stored in single blocks.

crypto The crypto class deals with all cryptography in this implementation. In this case there are two main functions: 1) the definition of the hash function, so it can easily be replaced, and 2) the setup and definition of the cryptographic keys needed for the signing of the hashes.

integrity The integrity class is usually not required, because the implementation only creates blocks according to the rules. Invalid blocks are dismissed. Therefore it should not happen that a Blockchain contains invalid blocks, therefore a class which states the validity of a block is not required from a functionality perspective. We need it though,

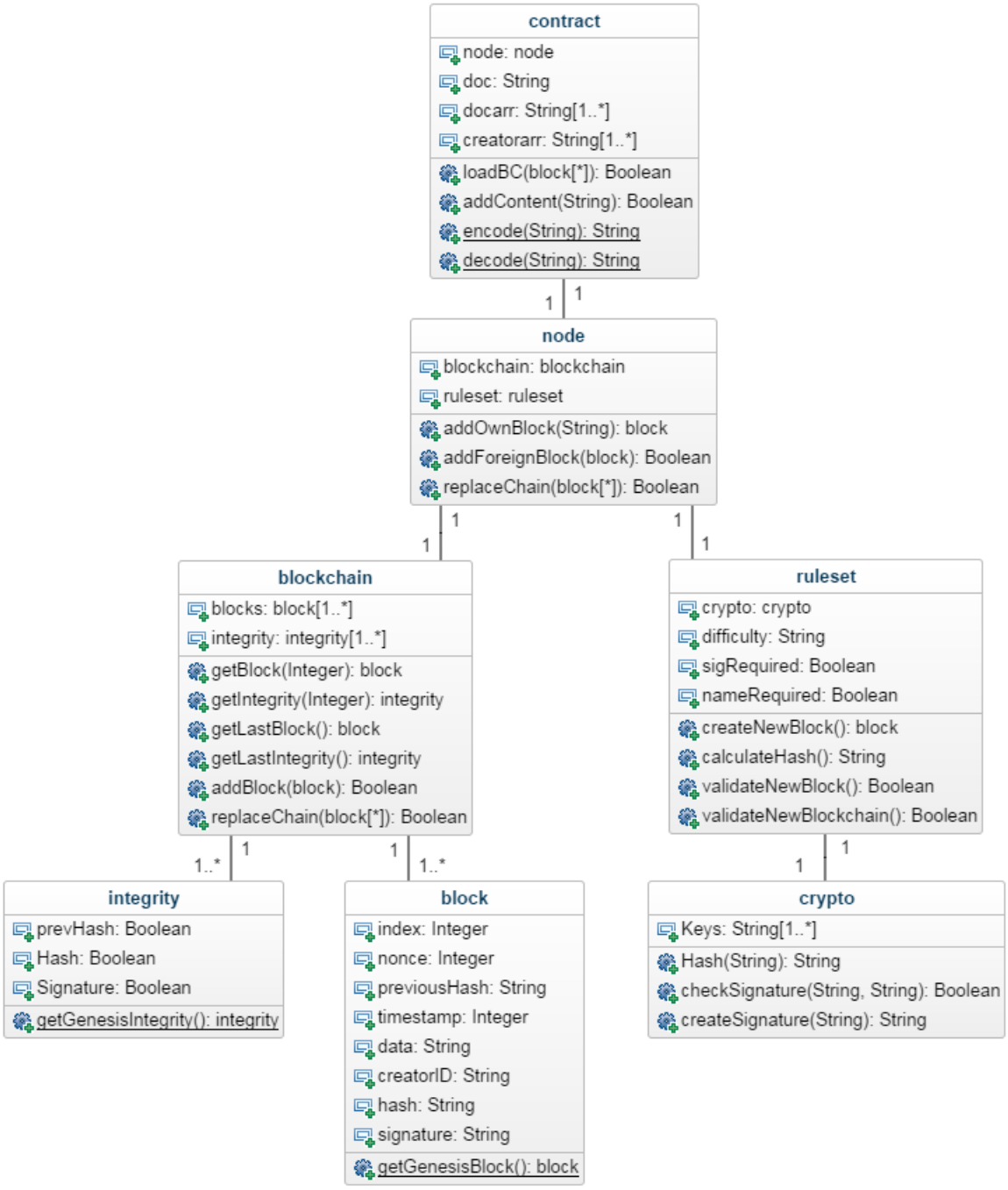


Figure 6.1.: The class diagram of the prototypical implementation

because we create an artefact which shows a possible way of data manipulation in the chain later on. The class is needed to communicate the integrity of a single block in the frontend, as one should not include validation in the frontend.

node The node is the acting object of the Blockchain. It includes two classes: The blockchain class and the ruleset class. With the help of these two classes it ensures that only valid blocks gets added to the chain and only valid blocks are created.

ruleset The ruleset class is responsible for the creation of blocks, especially for the calculations that have to be done to create a new one. Additionally, it validates the new blocks and checks if the signature matches the certain identity, ensuring that the block was created by the right entity. To facilitate this, it integrates an additional crypto library.

The API-Server enables us to use different commands. The commands, their impact and their returned JSON-objects are described in the following listing.

GET::/blocks The server returns a list of all blocks he has stored in a JSON-object.

GET::/integrity The server returns a list of the integrity of all blocks.

GET::/document The server returns the complete document, more precise, the contract that is contained in the Blockchain.

GET::/documentVersions The server returns all different versions of the document that are stored on the Blockchain. With this it is possible to see which changes were made by which entity.

GET::/info The server returns Information about itself, like the name, used ports, and additional data.

POST::/mineBlock The server receives information about a new block he builds and integrates in the Blockchain.

POST::/addPeer The server connects to an additional peer in the network. Information about the peer is specified in the POST-parameters.

POST::/updateDocument The server receives information about the new content which should be added to the contract.

POST::/Change This POST-command commands the node to manipulates a given block. This functionality is not found in normal Blockchain implementations. This is to show that blocks cannot be manipulated once they are part of the network. Further information can be found in 6.3.3.

6.2.2. Data Flow

We provide the data flow of the activity of content creation at two levels: The network level and the implementation level.

6.2.2.1. Content Creation - Network Level

At first, we take a look at the network level. The model can be viewed in figure 6.2.

We have four different actors in this model: Two lawyers (a and b) and two nodes (A and B). The lawyers both use browsers to access their respective node and both nodes are servers running our implementation of Naivechain.

At first, lawyer a creates new text and send it to his own node A. The node A creates a new block n+1 with the corresponding content and transfers it to node B. Node B validates the block (so the block obeys all rules). There are two cases: The block is valid, resulting in an accepted block, including in its own blockchain and transferring it to lawyer b. If the block is invalid, the block is rejected. As the acceptance or rejection of node B does not influence the output in lawyers a browser, there is no path back from node A to lawyer a.

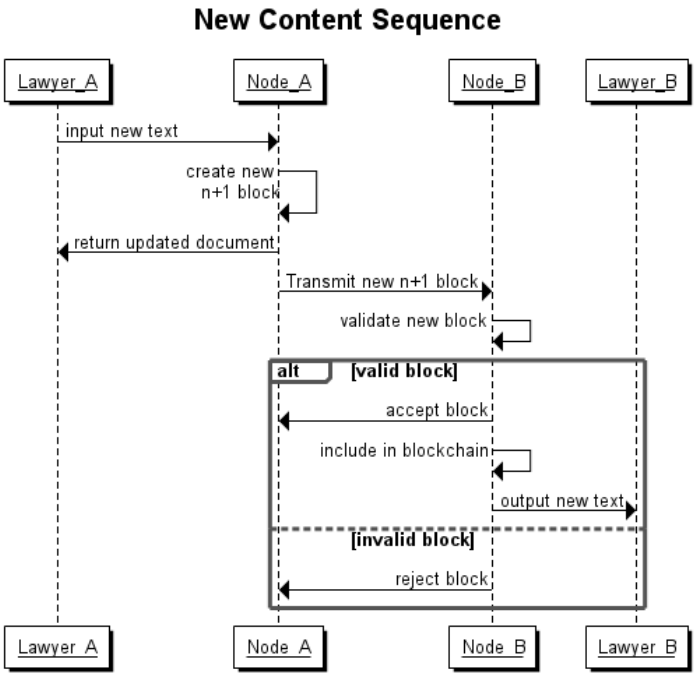


Figure 6.2.: Data flow of content creation at the network level.

6.2.2.2. Content Creation - Implementation Level

Instead of looking at the network level, we look specifically at the implementation level for a node that creates a new block with content. The model is depicted in figure 6.3.

We have seven different actors in this model: A browser which is responsible for the input of text, two interfaces for HTTP and P2P each and four classes: `contract`, `node`, `ruleset` and `blockchain`. We leave out the class `integrity`, as it is not available in normal implementations.

At first, the browser sends the new text to the HTTP interface. The interface forwards the new text to the class `contract` which includes it. Then it gives the order to create a new block with the content to the class `node`. The class `node` gives the text to the class `ruleset` which creates the block according to the current rules and gives it back. The `node-class` gives the valid block to the `blockchain` class. The block is included. After that, the `node` informs all other nodes about the new block. The `contract`-class returns the updated document to the

HTTP-interface and it returns it back to the browser.

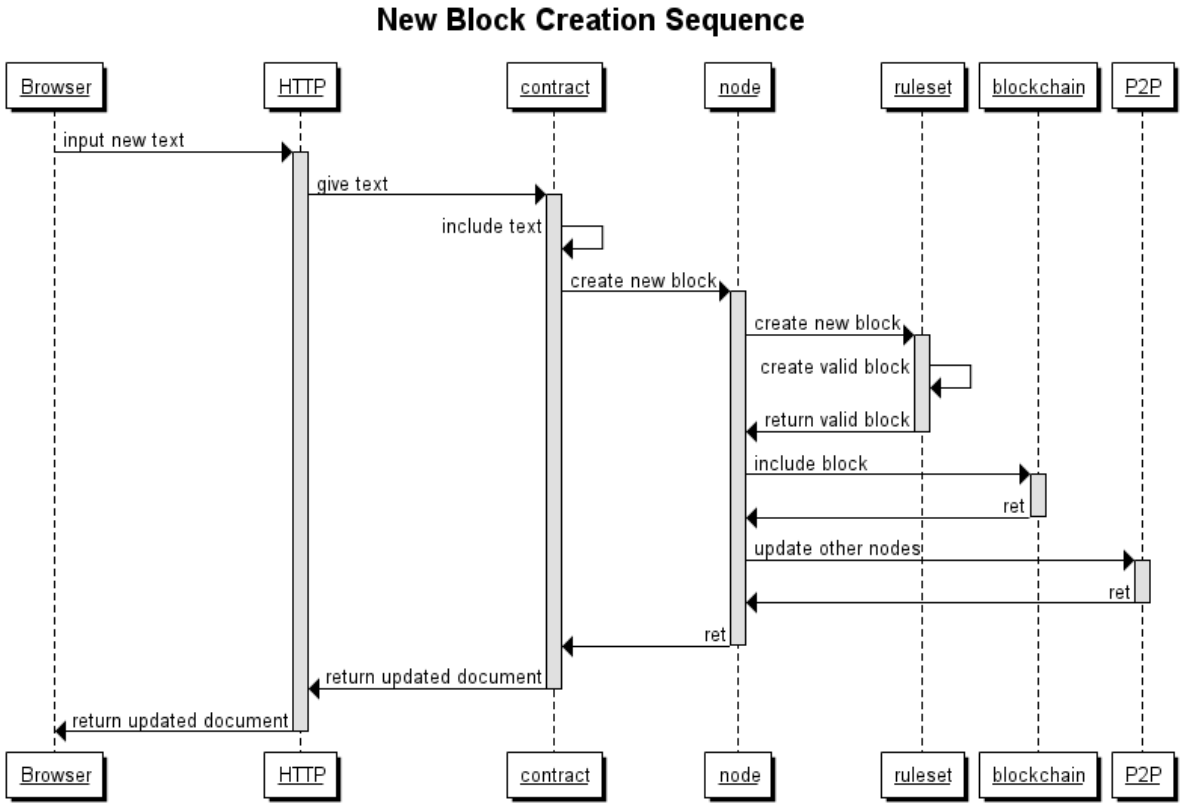


Figure 6.3.: Data flow of content creation at the implementation level.

6.2.3. Simplification of Design

A complete implementation of a Blockchain is not feasible or required to facilitate this use case. Therefore we made some simplifications in different areas of the software which we will cover here.

6.2.3.1. Blockchain Setup

The content of a blocks differ from blocks in traditional Blockchain systems.

First, instead of giving the opportunity to include many different datasets into one block, only one dataset per block is supported. Therefore, no merkle-trees or other structuring is used. The data is included in the block directly without any further data structure.

The hash of the block is included directly in the Block itself. This is for simplicity reasons, as it does not make a difference if the hash is stored outside or inside the Block. Technically speaking, one cannot include the hash of some block in itself, as it is impossible to fulfil $H(B) = h$ and $h \in B$ at the same time, if the hash function is collision-resistant. To solve that problem, only the hash of the included data is calculated.

6.2.3.2. Network

The network is decentralized. However, traditional services of a peer-to-peer network are not implemented (i.e. client discovery, automated connection handling, public key exchange). All connections which are made by the nodes are facilitated via starting parameters or manual addition of nodes. Thus, the underlying network infrastructure is very basic.

In chapter 2, we explain the rules for defining the only valid blockchain. In this implementation we deviate from one rule. Recap: The first rule states that every chain has to have the same genesis block, the second rule says that only valid blocks according to the network rules are accepted and third that only the longest Blockchain is the valid one. Naivechain has a rule, that is not typical for a normal Blockchain network: It checks the signature in the block. The block has to be signed by a entity which is known to the client. All other blocks (signed by unknown identities) will be dismissed. This is a necessary property for private Blockchains (like in this case), as we do not want any participants to create "anonymous" blocks without revealing their identity. An additional rule is that the hash has to meet criteria which we define as follows: It has to begin with three zeros. This can be seen as a classic Proof of Work approach, but differs in two ways. First, the difficulty of the puzzle does not adapt to the network speed and secondly, the difficulty of created blocks is not considered in deciding which chain is the longest.

Naivechain does not generate blocks automatically (unlike normal Blockchain technology). They are only created when the user inputs new content, as a block without any content does not serve any purpose[2]. Therefore, this functionality is left out.

In chapter 3 we specified different roles participating in the network. In the case of Naivechain, only two operator roles are involved and both are executed simultaneously, because one cannot create a block without a private key. Participants are therefore required to use both roles, ID Holder and Miner, at the same time, because it is impossible to only take over one role. Other roles like full node or light node are not required and therefore not used.

6.2.3.3. Blockchain Contents

The structure of the included data is simplified.

First, there is no language used inside the datasets. It is not possible to write any scripts in the Blockchain, because this would require a parser which understands and executes provided code. As this is not a goal (and not necessary for the goal), we choose a more simple approach. The data of the contract is just stored *base64*-ed[3] in the block. When the software retrieves the block, the content is base64-decoded and displayed on the website.

The data model is very basic. Think about the use case: People want to collaboratively work on one document or contract and of course there should be the possibility to change some words or even paragraphs to delete mistakes or add additional content within the existing

[2]For example, in Cryptocurrencies in every block a coinbase transaction is included. The miner receives a specific amount of money, so he is interested in creating a block whether or not data can be included.

[3]Base64 is a method for encoding data into ASCII-only characters. Without encoding, content can be written which would break the functionality of the blockchain [63].

document. A data-model like this does not require to just store additional text, but the changes that have been made since the last block would have to be stored. The required functions would have to function in following way: $Block^n = Diff\left(document^{t-1}, document^t\right)$ and $document^t = concat\left(Block^1, ..., Block^t\right)$. With these two functions, a more complex data model would be possible and the system would be more convenient to use.

6.3. Software-technical Aspects of Implementation

6.3.1. NaiveChain

The Naivechain implementation was created by Lauri Hartikka and published on Github [62]. His goal was to create a simple Blockchain which helps understanding the basic concepts and can be expanded by other software developers. The code basis itself only exists out of 200 lines of code and proposes a very simple, straight-forward implementation. As previously stated, the code was heavily adapted to fit our needs. Additionally, we provided a frontend for the Naivechain. In the section 8 we give an insight into future plans for the development of the Naivechain.

6.3.2. Node.js

Node.js is a server-side platform which uses Javascript v8 to create web servers. Because of its event-driven architecture, its memory footprint is very low compared to traditional approaches. It allows to easily implement simple web applications and is therefore widely used. The usage of Javascript for server-side development gives people an easy way to start developing with node.js. The strict separation between frontend and backend helps the development of multi-platform architectures, as only one backend (with node.js) has to be built and every frontend can be developed independently. Node.js can easily be extended with the Node Package Manager (npm) which contains modules suited for varying purposes.

6.3.3. Additional Features

In this part, we describe the additional features implemented in the Naivechain by us which are usually found in Blockchain technology.

6.3.3.1. Public-Private Cryptography

In regular Blockchains, public-private cryptography is often used in form of digital signatures to prove that a person who possesses the corresponding private key issued this transaction. As this implementation uses entire blocks rather than single transactions or datasets, the hash of the block is signed instead. If digital signatures are created with RSA (which we use), one has to keep in mind that asymmetric cryptography is very slow and the signature should be as least as big as the document itself. Instead, we sign only the hash rather than the entire document [54]. Because we do not want to create multiple hashes inside the block, the block

is hashed. Another risk could arise, because the signature in the block can be exchanged without modifying the hash, therefore later on new signatures could be generated. This is not a problem, as the identity (the name to the signature) is unalterable, therefore only the signature by the original creator is allowed. He would not gain any advantage if the signature is changed later on. This risk can be addressed in future implementations.

6.3.3.2. Proof of Work

We also implemented a PoW algorithm which is missing in the original Naivechain implementation. A short recap: The PoW algorithm is used as consensus algorithm to allow the network, independently from its computational power, to create new blocks in a predefined time window. The algorithm consists of three methods: Generating the hash, verifying the hash and adapting the difficulty to the network power. The first two methods are implemented, the third one cannot be implemented as that blocks are not generated automatically, thus there is no meaningful method to calculate the power of the network.

The source code to the function generating the correct hash can be found in code 6.1. The software increases `x` until the hash of the overall block matches a certain value. For simplicity reasons, the hash is not converted to a number and compared with the difficulty, but it is checked if the string of the hash begins with a certain character sequence. The limit for `x` is chosen in such way that it is very likely that valid hash is found.

```
#[..]
let x = 0;
while(true) {
    nextHash = Ruleset.calculateHash(nextIndex, x, previousBlock.hash, nextTimestamp, data,
        nextStatus, nextCreatorID);
    //If correct POW is found, break;
    if(nextHash.substr(0,this.pow.length) === this.pow) {break;}
    x++;
}
#[..]
```

Code 6.1: Source Code

The second method only validates the hash. Compared to the previous function, it does not need to compute a nonce, it just checks if the hash is correct and accepts the block. If the hash does not match the requirements, the block is denied.

6.3.3.3. Blockchain phenomenons

For educational purposes we included three sub-sites, with two types of attacks and a graphical representation of forking, as it happens on a regular basis in Blockchain networks.

51%-Attack The 51%-Attack is the commonly known attack around Blockchain (as already stated in the section 4). The displayed graph consists of rectangles with two different colours, (dark) blue for the user and red for the attacker. The user on the website is able to see how the

attack happens. He can generate new Blocks with his own colour and is able to see how his own blocks are not included in the longest chain, resulting in his information not included at all. A screenshot of the website can be found in figure 6.4. This is a simplified approach, as most users never create their own blocks. However, the methodology is the same, as the attacker controls the contents of new blocks.

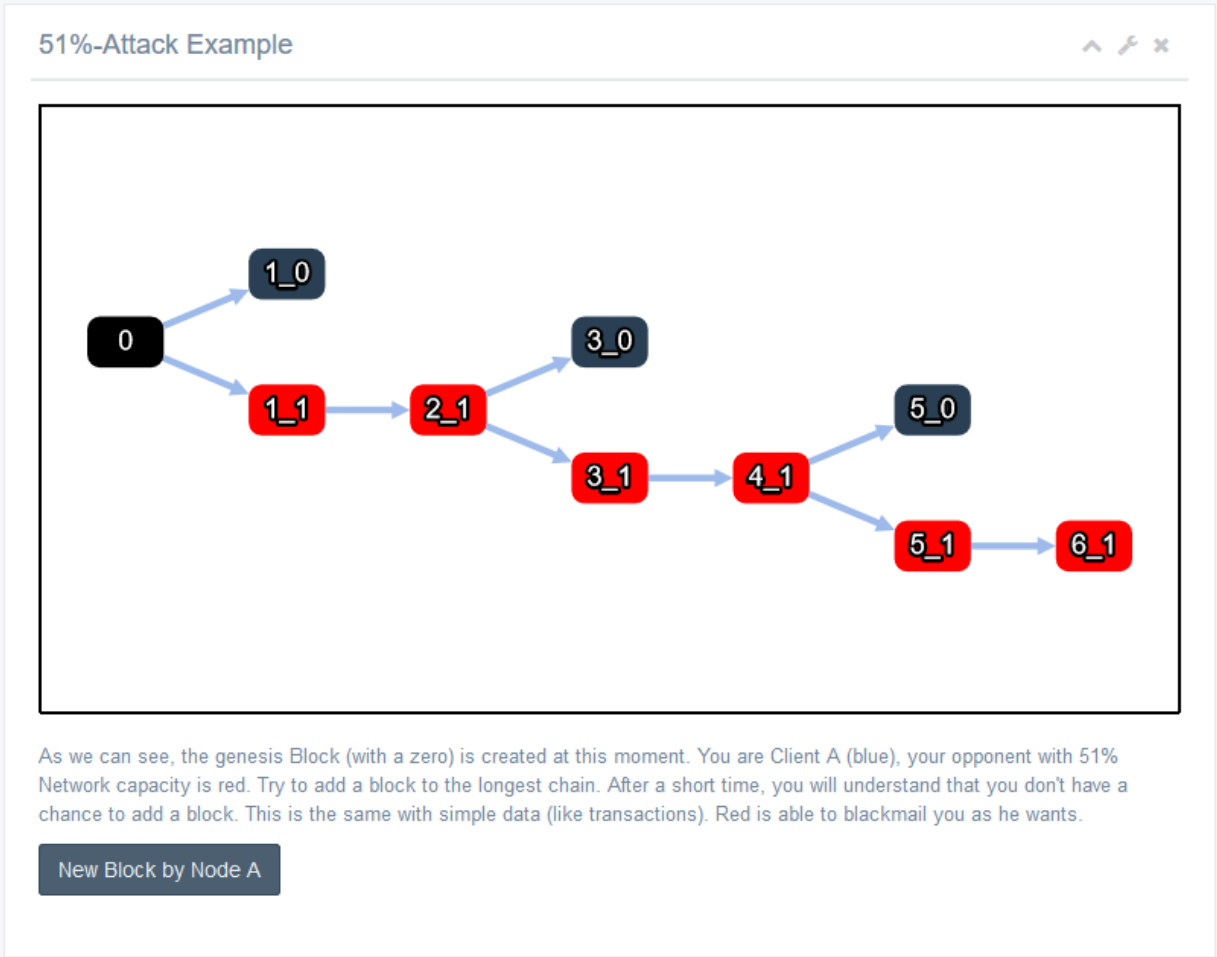


Figure 6.4.: 51%-Attack

Data Manipulation One can perform data manipulation in the Blockchain ecosystem only by a 51%-attack using valid information. The chain cannot contain invalid information. However, it is possible to place invalid information in the client one controls. This example shows that the information does not spread into the network.

The user is able to select a single block which he wants to manipulate. He can change some information, like data, status, or timestamp. The user is not able to change other contents of the block. If he changes information successfully, his own view is updated and contains the manipulated information. Additionally, it the invalid elements of following blocks are shown. When other nodes are connected and own blocks are transferred, these blocks are being rejected due to the invalidity of the information. Thus the node is not able to participate

in the network. The error-message which gets thrown from the other clients is shown in figure 6.5.

```
Node#2: Block contains invalid hash: 12
    c4e26d516a4dc6189c81a3cb01341620fa39122ac4b91950330627dd8f2a91. Should be: 000811487
    d51a0e2bf8ec8399f5631e5fb79d005f79bb29f6e3930c6c6091706
```

Figure 6.5.: Second node rejects block.

Forking The Forking-page allows the user to understand the mechanics behind the forking process in Blockchain technology. However, hard-forks and soft-forks are only described textually. The user is in control of three different nodes: Blue, white, and green. He is able to create new blocks at will that build upon each other. He is also able to fork the current blockchain with any node. If he forks, a second block with the same height occurs and the blockchain is split. Now the nodes can decide on which block they will continue. When the fork is resolved, all new created blocks are placed on top of the longest one. A possible forking example can be viewed in figure 6.6.

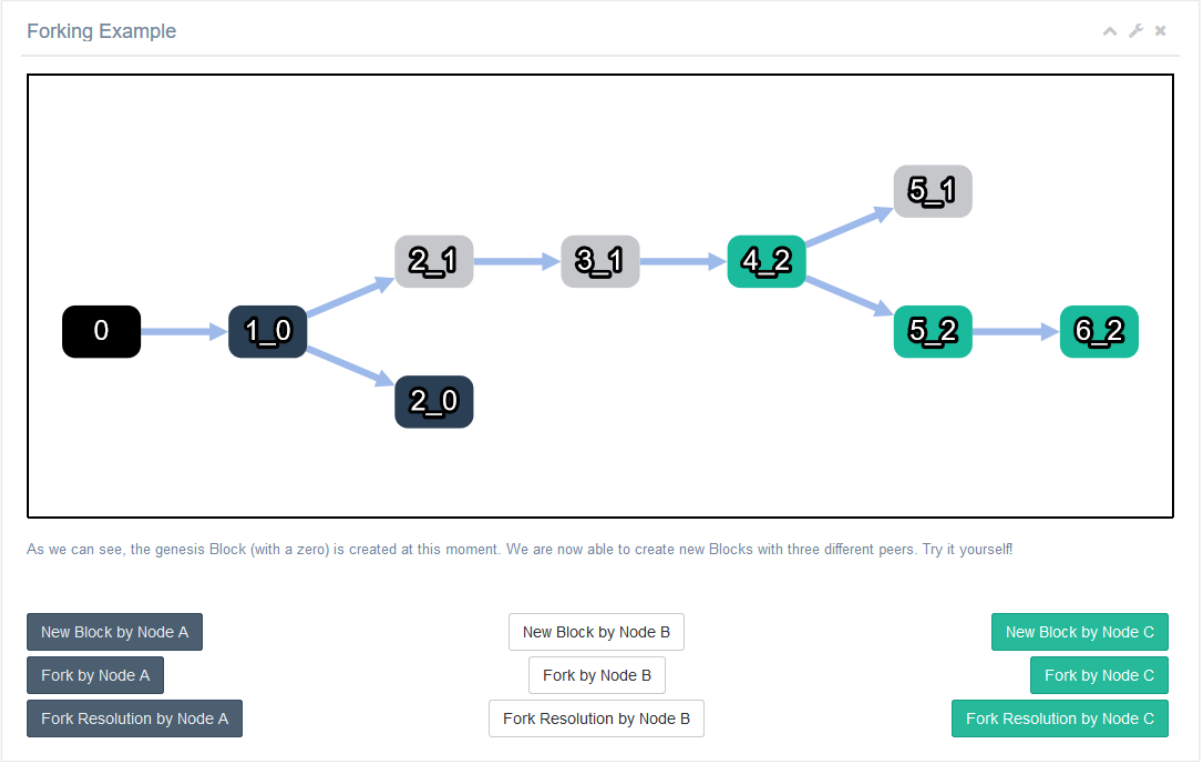


Figure 6.6.: An forking example. Block 2.0 and Block 5.1 are orphan blocks, thus not considered in the blockchain anymore.

7. Discussion and Critical Reflection

In this section, we evaluate this thesis. We compare our aspiration with the results of our work and critically reflect them. We first dig deeper into the proposed theories as the architectural views, the categorisation, and the extracted principles. Afterwards, we take a closer look at the downsides and weaknesses of the prototypical implementation and discuss lessons-learned. At last, we answer the research questions.

7.1. Critical Review of Proposed Theories

The theories extracted in this thesis describe fundamental concepts of Blockchain technology. In this section, we discuss them critically considering their further use.

7.1.1. Architectural Views

We start with the five architectural views.

7.1.1.1. Architecture Overview: Infrastructure Service and Business Service

In discussion with the experts about the architecture overview, we discovered the complexity of differentiation between the tiers infrastructure service and business service. We discuss to which tier the *Mist*-Browser belongs [61], as it provides access to the network (which would account it to the business service tier) but is also part of the network. It is not possible to access information without participating in it. Therefore one cannot precisely draw the lines between those two tiers.

7.1.1.2. Architecture Overview: Structure Discussion

Considering this architecture, underlying layers provide interfaces for layers at a higher level or in other words higher level layers build upon lower layers. The question is: Does the business service layer build upon the application layer? The answer depends very much on the business service itself. Strictly speaking, every business service builds upon the infrastructure service, as it has to have access to the network to provide any service on top of it and it is not possible to access the applications (e.g. smart contracts) without connecting to the infrastructure service first. However, the business services can very much depend on the functionality of the smart contract, as it provides interfaces which cannot be provided by the infrastructure service itself. It is, therefore, worth discussing if this view should be adapted in a way to live up to the actual implementation.

7.1.1.3. Architecture Overview: Placements of Elements

We also discuss on which layer to place different elements. Consider the infrastructure and infrastructure service layer: One can argue that the design of the network belongs to the infrastructure service. The access rights to the blockchain itself highly depend on the implementation of of the blockchain, as we discuss in section 5. Giving access to the blockchain requires access to the same network, but the opposite is also allowed. Using the same network, but not being allowed to connect or use the blockchain is possible. The Internet as we know it is used for public and private blockchains. Therefore there is no need to differentiate between the design on a network level. Nevertheless, the design of the network can be placed in the infrastructure layer, as it is not directly tied to the functionality of the blockchain, as the chain itself works independently from the publicity.

7.1.1.4. Architecture Overview: Application and Underlying Technology

The model does not strictly separate between implementation and environment of the application layer. As an application is pre-defined, all its requirements, rules, and implementation should lie on the infrastructure tier. If it is user-defined, previously named items should be included in the application layer, as it lies above the infrastructure service. For example, in a cryptocurrency, the structure of the ledger containing the coins and the rules of transfer belong to the infrastructure service, but the developer creating a smart contract determines the rules with his code which results in the implementation belonging to the application layer. One can further investigate this disparity.

7.1.1.5. Roles in the Blockchain: Selection of Roles

One can think of additional roles in Blockchain networks. The discussion centers around the question which roles are necessary and which do not belong to a standard blockchain. On the one hand, there are roles required in some implementations, relying on many different factors and parameters: A centralized blockchain requires an instance responsible for the creation of blocks, access restriction (rights management), resulting in functionality superfluous to public blockchains. Fully anonymized Blockchain networks like ZCash or Monero require additional roles [64], such as trustworthy individuals for the bootstrapping process of the coin to function properly. On the other hand, one could argue that some roles in this view are obsolete. As we discuss a standardized Blockchain, it is not clear if one should include the two operator roles light node and full node, as their functionality is just a derivate of the miner. A miner can execute all of their tasks. One therefore has first to discuss the nature of a standardized blockchain to be able to discuss roles. For the deviation of the roles we use the blockchain proposed in chapter 2.

7.1.1.6. Roles in the Blockchain: Active Tiers

Discussing active tiers of the roles, the experts understand the separation into three different developers and deem it as straight forward. However, talking about the operator roles, it is

difficult to draw the line between tiers for single operators. For example, every operator has to operate on a network level, as he cannot take part in the network otherwise. However, we do not want to state the obvious to keep the view clean and simple to understand. When talking about the differentiation between light node, full node, and miner, the differences are small. We therefore want to represent the important roles in the tiers. For example, talking about the infrastructure service, the full node and the miner are the most important operators, as they ensure the integrity of the network. The light node itself uses the infrastructure service (it has to), but it is not its main responsibility[1]. Therefore it is not active in this tier in this view. One can further discuss the allocation of the tiers, as it is not entirely necessary for the ID holder to be active in the business service and business tier. An option to consider would be to just view the three lower tiers (infrastructure, infrastructure service, and application) as the other to tiers do not count to the blockchain itself. One has to make a distinction between the roles of the blockchain as a data-structure or the Blockchain as an ecosystem.

7.1.1.7. Blockchain Process: Simplification

We leave out required activities in the Blockchain network. The process view does not go into detail on how the network works, how new nodes are introduced, or how the information is transmitted in the network. Further research can discuss if an additional view for these processes is required, but it remains questionable if these processes can be represented for a blockchain in a generalized way, as those processes strongly depend on the individual implementation. As already stated in chapter 2, these views already exist in various ways for various implementations.

7.1.1.8. Blockchain Process: Wear-Out

It is questionable if wear-out of blockchain projects has to be considered. The user behaviour is unknown, as there are very few public Blockchains which did go down after some time. Before considering these cases, one has to define how a wear-out or dead blockchain looks like, as there are different symptoms of a non-functional blockchain. It is possible that the mathematical problem becomes too complex for the network. The network might be still up and running, but not be capable of creating new blocks. This might happen as large shares of the computation power leave the network. However, small parts of the network might keep up the network as the blockchain itself has no relevance anymore. Considering the market cap, there are many currencies with very low transaction volume [2]. For example, coins with declining relevance are ReddCoin or Primecoin which have market capitalizations of roughly 2 and 5 million US$ (at may 2017). However, old or unused blockchains could be backup-ed, if their content is relevant to some stakeholders.

[1] In fact, it is not responsible for this tier at all, trusting the miner and the full node to provide a valid blockchain.
[2] `https://coinmarketcap.com/currencies/views/all/`

7.1.2. Use Case Categorization

With the second theory we propose a way to categorize the functionality of Blockchain technology. With this categorization, we are able to generate a unique fingerprint for every of the discussed use cases.

7.1.2.1. Distinction

As already introduced in section 2, it is uncertain if an approach exists in which one can draw a clear line between single categories. We think it can be possible one considers single activities rather than single use cases in the Blockchain. The categorization achieved with this process contains more categories, as there should be no overlaps. Is it a good idea to give a weighted attribution to one category? It has various advantages (as the similarity comparison of use cases), but makes the Blockchain and the process of generating and understanding new use cases more complicated.

7.1.2.2. Manifestation

As we decide to approach the categorization with a weighting, it is not clear if the weightings 1 to 3 are appropriate for the rating of single use cases. One can also consider alternative assessments with weights 1 to 5. The decision depends on the selected classes, as more generalized categories allow more nuances whereas more specific categories often only allow a differentiation between "is utilized" or "is not utilized", resulting in an assignment to categories. As Blockchain is still an emerging technology with limited applications, weights on a scale between 1 and 3 are sufficient. We assume that the use case categorization will be adjusted in the future.

7.1.2.3. Distribution & Depth of Classes

When considering the seven classes, the depth and the scope of the classes vary. We compare two classes: Market creation and Meta-consensus. The market-creation covers many use cases: Cryptocurrencies, many smart contracts, all sorts of ICOs and many more. The meta-consensus class only covers few use cases: Elections which we both saw only in proof-of-concepts [65] and shareholders voting which is executed actively. Obviously, categories do not have the same extent. Market creation could easily been split up into assets with intrinsic value, representative assets and so on. A further split up will give more details about the functionality, but makes it more complicated to talk about the use case.

7.1.3. Principles

Principles represent single components of blockchain technology.

7.1.3.1. Minimum Viable Blockchain

The definition of a Minimum Viable Blockchain is a only of theoretical nature, one can picture it as a hard-fork of all blockchains. As said, we do not know if such theoretical approach can be implemented. An alternative approach to define a basic structure is the definition of an actual blockchain implementation that only provides the basic functionality. With this approach it is possible to actually implement every single component and create a new blockchain selected by components which would increase the value of the proposal. A Minimum Viable Blockchain does not have to be a hard-fork of all blockchains, but only of its derivatives. With little effort, the Naivechain can be redesigned to a basic Blockchain. However, a theoretical approach is more useful if set theory is applied to the field of research.

7.1.3.2. Completeness

The list of proposed principles is not exhaustive. Different components exist in implementations and further research has to be done do identify additional principles. The consideration of actual implementations is, as stated in the introduction, not in the scope of this paper.

7.2. Critical Review of Implementation

After reviewing the proposed theories, we look more closely into the implementation of the selected use case "Collaborative Contract Creation".

7.2.1. Functionality

At first we consider the functionality of the proposed implementation. As stated, it is not possible to actually edit the document or contract due to the simplified data model, therefore we will not go into further details of this and continue with the next functionality.

7.2.1.1. Number of Documents

The proposed implementation has only limited functionality, as it only allows to work on one legal document at a time, and if a new document is required, the server must be restarted. Every node participating in the network works on an identical draft, has the same rights and cannot create additional documents. It is worth discussing whether one should allow multiple documents and participants operating on one Blockchain, as every document gets distributed across all participants[3]. An alternative approach is to provide a basic blockchain which records all unique IDs and this blockchain offers the possibility to create an own blockchain between two participants which is only recorded between them. As they would both be bound to their respective cryptographic keys, it is sufficient if both are part of the blockchain, as both are not

[3]That is not a security issue per se as the documents can be encrypted beforehand, but with later disclosure of the key (Advanced cryptographic attacks or hacks) all documents can be decrypted. This can not happen, if the attacker does not receive the chipertext.

able to create fake messages from the other participant. All security requirements would be fulfilled. The identities would have to be tied somehow to the cryptographic keys. As we only create a working prototype of this use case, and the main goal was to show the functionality of the technology for the use case, we decided to use the most simple approach.

7.2.1.2. Signature

The document cannot be signed by their respective creators. To keep up all security requirements (i.e. no one can tamper with the content) it is required to place the signatures of the contract on the blockchain. This is no problem as both parties can sign a special block which determines the signed version. It is possible that – from a legal perspective – additional requirements have to be fulfilled. Both parties can keep record of the blockchain to later on prove the signing of the contract.

7.2.1.3. Key Exchange

The blockchain itself does not offer any form of key exchange. Both keys (the own private and the other public key) are stored locally, otherwise blocks cannot be exchanged. The problem is that the keys are created at will (and if they are no keys in the directory) and they are not tied to the respective person. If two lawyers want to use such a technology, they had to use a publicly approved method for key exchange. A possible way is to go to a notary and to attest the possession of the keys. This method is time-consuming and costly, thus, one has to introduce other alternatives. We are currently not aware of any service or platform which provides such functionality, but we picture it as a method similar to a Public-Key-Infrastructure for SSL-certificates.

7.2.2. Blockchain-related Functionality

In the implementation section we explain some drawbacks. However, in following we state another problem which can occur when using the platform.

7.2.2.1. Loss of Information in Orphan Blocks

In this implementation it is very unlikely that orphan blocks occur, as blocks are only created when information is inserted. However, it is theoretically possible that both users create new content at the same time. In this case, both users continue with their version of the document, as the chain proposed by the other node is equally long, therefore neither one switches to the other chain. The fork is resolved as soon as one of the two users proposes a new version, thus creating a new block. The other node receives a blockchain which is longer than his own, and replaces it with it. Now there is no functionality which resolves the loss of the information in the orphan block. Other Blockchain implementations check, if the information is included in the chain. If it is not included, it will be contained in a later block. The resolution of this problem is not that easy, because the content of the orphan block is dependent on the precessing blocks. If a new block is introduced, it is possible that the block changes information

that was already changed in the new block. Resolving this issue requires a novel kind of error-handling in which the system displays a dialogue to manually resolve the conflict. This is not implemented in Naivechain and leads to loss of data. Yet, this event is highly unlikely.

7.2.2.2. Malicious Behaviour

As we deal with a private blockchain, maliciously acting users have to be considered in the system. We know all participants, but it is possible to erase new blocks by making them an orphan, the same functionality described as before. This behaviour is randomly triggered, but lawyers can utilize it to delete contents of other users. One has to introduce a mechanism which builds upon the one proposed in the previous paragraph and extend it with additional functionality. As the number of participants is very small, the time it takes to synchronize the nodes is very small. Therefore it could be possible to create a rule that only allows to build on top of the block with the height $n-1$ (n the current height). This would result in $n-1$ fixed blocks, reducing the risk of manipulating the block history. However, this number has to be adapted to cater for many nodes or long synchronization times to prevent the chain from splitting. As both nodes have a unique $n-1$ block, the chains are hard-forked, meaning there is no way to fork them back into one single chain, resulting in a broken functionality.

7.2.3. Lessons Learned

In this chapter we also describe the lessons learned during the implementation phase.

7.2.3.1. Necessity of Private Blockchains

In our opinion, the necessity of private blockchains remains an open question, as one can implement "private" use cases without blockchain paradigms. In the case of lawyer communication, other software is available which achieves the same functionality [4] without an underlying blockchain. The document can be created and signed and transmitted to the other lawyer without putting it in blocks and hashing it together. All requirements are fulfilled if all participants use digital signatures. In our opinion the main advantage (over a normal "non-blockchain" implementation) is the possibility to add redundancy without any problems, as the new nodes just connect to the networks and keep an identical chain. However, this functionality can be implemented without a chain, too.

We think that the most benefits come from making the Blockchain public and let everyone access it. Disruptive use cases have to be publicly available, as they cannot be deployed in a private environment. We are not aware of any high potential use cases with a private design. It is questionable, if use cases are found or if the blockchain can only use its potential in the public area.

[4] `http://clausematch.com/`

7.2.3.2. Application is Key

We think that the underlying technology is only a backbone for more advanced applications. For example, Ethereum is a widely used platform as it is the only one that provides the opportunity to create and execute smart contracts. ZCash and Monero are recognized for their anonymous transaction design. It truly depends on the application a blockchain provides. Of course, the underlying technology has to be adapted to enable more complex applications, but changing these components to more sophisticated ones does not influence the application on top of it. An example is Ethereum switching from Proof of Work to Proof of Stake [23].

7.2.3.3. Continuous Use of Blockchain

An important lesson we learned is that designing the blockchain is a first step, but designing the interfaces to the real world is more important. Considering the "Collaborative Contract Creation" use case: If blockchain is used and every single document is stored and validated, no one is able to manipulate the document. If one lawyer prints out the document, signs it and sends it via letter or mail, the second lawyer cannot trust the document as it was possible to manipulate it between exporting it from the Blockchain and printing it. The security goal "integrity" is not fulfilled anymore and a Blockchain is not needed in the first place, because the lawyer now has to check the document again for manipulation. The integrity of the Blockchain is not violated, but it was circumvented by making the document "offchain". This is a central problem, as one cannot prove the integrity of information which is not completely stored in a Blockchain and retrieved directly from it. For that reason, one has to carefully think about the interfaces between the real world and the Blockchain, as tampering is possible.

7.2.3.4. Understanding of Technology

While designing and developing the application, we learned that the understanding of the underlying technology and its functionality is essential, if one considers Blockchain for some use case. The functionality and the usability depend strongly on how the developer can deal with the unique behaviour of the blockchain: Forking, the speed, the irrevocability, the monolithic approach (and thus not being able to scale), or possible attack vectors. Comprehending these is inevitable to prevent building flawed applications.

7.3. Answers to the Research Questions

At the end of the evaluation chapter we give an overview about the answers to the research questions.

Q1 What is the structure and the architecture of a standardized blockchain and its network?

We propose five different views about the structure and architecture of a blockchain. The architecture of Blockchain can be divided into five distinct tiers, allowing readers to better comprehend the setup of the technology. The four additional views provide a clear insight into

the technical layers, give an overview about involved roles in the network and describe the life cycle of Blockchain networks. The views show a varying and clear picture of the technology. Further information about the answer to this research question can be found in chapter 3.

Q2 What are fundamental parameters used in Blockchain technology?

In public blockchains the most fundamental parameters influence the speed of the network: The chosen consensus algorithm with its parameters (average time between two blocks), the possible applications (pre-defined or possibility for user-defined applications, parameter: chosen language) and the interconnection between consensus algorithm and application (parameters such as coinbase-transaction or coin limitations for cryptocurrencies). In private blockchains the parameters are usually more loosely defined. However, the most important parameters are how the network restricts access and the creation of new content. Additionally, the chosen language determines the applications for the Blockchain. The interested reader can find more information about the parameters in section 3.

Q3 What are risks of applying blockchain technology?

Conducting the interviews, we found out that the risks in Blockchain technology are diverse. We identified different sources of risks and downsides: Technical risks occurring during the life cycle of a Blockchain and risks occurring in in combination with Law & Governments, the Blockchain community, the industry, the technical progress, and the end user. The two most frequently named risks are following: The missing scalability, as it prevents the platform from growing with the higher user demand and leads to possible delays when interacting with the system, and the missing regulation, as there are many legal questions still unanswered and therefore preventing businesses from developing advanced solutions with Blockchain technology. Risks conducted form the interviews can be found in chapter 4.

Q4 What are categories for the usage of Blockchain technology?

Investigating different use cases in the area of Blockchain, we propose seven different categories with weighting from 1 (hardly any or no usage) to 3 (usage focus): Market creation, tracking & provenance, autonomous entities, information storage & retrieval, meta-consensus, communication, and identity management. With these seven categories it is possible to classify every use case which facilitates blockchain technology to reach a goal. Use cases that provide services for or around the Blockchain are not considered in this classification, as their functionality does not rely on the technology itself. Further information about the categorization can be found in chapter 5.

Q5 What requirements emerge from the use case being implemented by blockchain technology?

Requirements for the usage of Blockchain technology regarding use cases are difficult to find. In discussion with interview partners we outlined six different measurements which either favor an implementation or speak against an Blockchain implementation. The six values are number of nodes, trust it the participants, transparency of assets, money, participants and logic, form of data-usage (either read-intensive or write-intensive), usage of digital twins, as well as complexity of operations. These six manifestations either favor or harm a Blockchain, but additional factors as risks have to be considered.

Q6 What are fundamental principles in Blockchain technology?

After conducting interviews, generating use cases and analyzing them, we propose a theory about the fundamental principles of Blockchain technology. The basis is the minimum viable Blockchain, as it provides the underlying functionality for all principles to build upon. The principles are classified into three different tiers: The network-access & participation tier (concerned with all rules and functionalities required for the network and the blockchain), the data tier (concerned with the structure of data and their contents as well as identities) and the operation tier (dealing with the requirements to execute software and allow communication between or with external sources). In these tiers different principles exist. More on the principles can be found in chapter 5.

8. Conclusion and Outlook

First, we state the need for a framework to understand the technology and the implications of it better. The framework provides a clear overview of the functionality and the potential principles, helping to further investigate possible usages of the technology for the society and economy of the future.

We continue with a comprehensive literature review and propose five distinct views about the architecture and structure of a generalized Blockchain, enabling a basic understanding of the technology. Looking at the typical implementations of Blockchains, we identify the relevant parameters of an underlying blockchain setup, exploring possible options in the network.

After providing the technological foundation, we proceed with semi-structured expert interviews. We talk to different companies of varying sizes which deal with Blockchain technology, from start-ups to enterprises. We ask the experts about their opinion on the technology in general, the advantages and disadvantages and their expectations for the future of the Blockchain. We discuss their use cases which facilitate the blockchain. We give an insight into these use cases.

With the extracted use cases we create grounded theories about use case categorization. These categories outline the functionality of the Blockchain itself without the consideration of side products, enabling to understand the potential within the technology entirely. From these categories, we derive blockchain principles one can perceive as fundamental components of the network. With the proposed minimum viable blockchain, we provide a solid foundation for these principles, allowing to implement a blockchain unique to a use case.

After that, we analyze selected use cases according to the architecture tier and use case category they belong to. We provide an explanation for the decisions, giving an insight into the applicability of the proposed theories.

Building upon the knowledge generated in the previous chapters we develop a prototypical implementation of an use case on our own. We describe the concepts, the simplifications and the functionality of the implemented platform serving an educational purpose.

After the combining theory and practice, we critically evaluate the theories and implementation and state possible difficulties and drawbacks. We close with a lessons-learned section and the answers to the research questions.

In this thesis, we provide a blueprint for utilizing Blockchain for individual needs of businesses and society. We also state potential dangers and risks and present the requirements for implementing such projects. Revisiting Don & Alex Tapscott from the introduction: "*Undoubtedly, its best applications are yet to come.*" [1], we are a step closer to finding them.

8.1. Future Work

Further developments in architectural views We propose distinct views which apply to a general blockchain. First, these views can be created for special applications as cryptocurrencies or other applications. Secondly, a deep dive at other tiers can be taken, as we denoted one view purely on the infrastructure service. One can conduct further research about the interconnection between infrastructure service and user-defined applications. Thirdly, one can investigate and consider further blockchain artefacts (such as forking) in the Blockchain life cycle.

Blockchain Parameters and Principles We only consider parameters occurring in standardized Blockchain approaches. As no use case is implemented using a generalized blockchain, one has to review and evaluate individual platforms according to their parameters and principles. A comparison between typical systems such as Bitcoin, Ethereum, Hyperledger, and Ripple and their purposes could yield interesting results. These results are essential for deciding which Blockchain to use, if an own implementation of a blockchain is not considered.

Further investigation of use cases We only talked with a small number of experts as the goal was to generate different theories about the Blockchain technology. However, conducting a questioning with more participants and standardized questions, one can collect more knowledge about the success of the technology, leading to a better understanding of the technology.

Large scale analysis of use cases From literature or further interviews interviews, it is possible to obtain a sufficient number of use cases to compare them according to the use case classification. A statistical comparison backed by an investigation of possible implementations can uncover unknown similarities, leading to an improvement of the proposed categorization.

Advanced Implementation of Naivechain The Naivechain has drawbacks, as stated in the implementation and evaluation chapter. However, the code basis is stable and allows extensions easily. With little effort, it is possible to create a simple cryptocurrency. With it, one can conduct more experiments as the measuring the impact of certain parameters. The programming language node.js allows beginners to get started easily and for expansion of the software.

8.2. Outlook

When we talk about the Blockchain with the experts, the perception of the technology as real distributed computer pleases us. It may not be entirely correct, but it opens up a new perspective at the Blockchain. If one considers a Blockchain as a monolithic computer, community, science, and industry should use the analogy of distributed computers to create new systems. Interconnections between single computers and later on the Internet were created enabling communication, distributed computation and much more. If one can think of

a blockchain as a single computer, can we think of a network of Blockchains to circumvent limitations of the technology, especially the problem of missing scalability? In fact, different approaches exist. Side-chains aim at creating dependent blockchains [66], derived from a central[1] Blockchain which can receive values from the main chain and work with these coins. After participants executed their scheduled transactions, the coins go back to the main-chain. It is fascinating as these side-chains underly their own rules and therefore can be designed in any way, thus speed is not an issue anymore. However, this approach comes at the cost of transparency, as it is not possible to reconstruct the path the coin took in the sidechain. Nonetheless, future developments try to solve the problem of scalability. With the analogy from the beginning, maybe we can look at the achievements of distributed systems and try to apply them to Blockchain technology.

We come up with another perspective at the Blockchain: The Blockchain can be viewed as a model of reality. Why? The Blockchain is the first truly decentralized platform that allows interaction with other participants [2]. In reality, it is too complex (for computers) to get in (business) contact with humans, but the Blockchain provides software standardized means for interaction. Not everything that is possible in the real world is possible on the Blockchain as well, but being able to buy and sell goods, information, and services seems like a reasonable foundation for creating an economy. Using autonomous entities, we allow software and computer systems to participate in the system without complex integration into real-world interaction or payment systems, as they can conduct their business in the blockchain. As IBM and Samsung showed with their autonomous washing machine [67], it is possible to integrate autonomous hardware in our daily life over time.

Given the technical advances, the community has to consider legal implications. Different approaches exist within countries to create legal frameworks to allow some use of the technology, such as conducting business in the network or paying taxes for price gains in cryptocurrencies at the moment [68]. However, the legislation just started, as the technology per se is complex and difficult to understand. It remains an open question how legislation will influence the technology, as it is not possible to delete any information out of the network because of the cryptographic implications. If illegal data is stored in such a chain, it could be punishable to participate in the network, as one would download, store and distribute the illegal data. If the legislator intervenes too frequently and extensively in the technology, it is also possible that all advantages of it become obsolete, if the state can circumvent the original rules to enforce the law.

Considering all factors, the Blockchain remains an important development for the society and economy in the future. We expect community, science, economy, and legislation to find trade-offs between viewpoints to enable valid usage scenarios for the Blockchain.

[1] Central not in the form of centralized, but the main Blockchain

A. Glossary

Term A.1 (Altcoin) *Altcoins are Alternative coins to Bitcoin. They build up on the same technology with small changes to parameters.*

Term A.2 (ASIC) *Short for application-specific integrated circuit. Hardware designed for a specific purpose. One of the most efficient ways to mine cryptocurrencies.*

Term A.3 (Block) *Contains Block Header, Transaction tree and some additional information.*

Term A.4 (Block Header) *Contains Hash of previous block, a Nonce to solve Mining Puzzle, the Hash of Transaction tree and additional information.*

Term A.5 (Blockchain) *Contains all Block which are linked together. Integrity can be checked only with one Hash.*

Term A.6 (Bootstrapping) *The process of starting a new Blockchain or cryptocurrency. Depending on the implementation, the process is more or less complex.*

Term A.7 (BTC) *Abbreviation for Bitcoin.*

Term A.8 (DAO) *Decentralized Autonomous Organization. DAOs are companies enforced through software code within a Blockchain network.*

Term A.9 (Digital Asset) *Is a certain value which is stored either in Transaction outputs or in wallets.*

Term A.10 (DLT) *Distributed Ledger Technologies. Another name for Blockchain technology, as the blockchain represents a ledger.*

Term A.11 (DOS) *Denial of Service. Describes an attack form at any type of system with the goal of making it inaccessible.*

Term A.12 (DSA) *Digital Signature Algorithm. Standard for digital signatures. Required for creation of identities which own some Transactions & Transaction outputs & Addresses.*

Term A.13 (Ether) *Ether is the currency of Ethereum.*

Term A.14 (Ethereum) *Ethereum is a common blockchain platform. Its main focus is on smart contracts.*

Term A.15 (Fork) *A Fork is a split in the Blockchain, resulting in two equally valid chains. Details can be found in 2.3.5.2.*

Term A.16 (FPGA) *FPGA stands for field-programmable gate array. It is an integrated circuit designed for various usages. It is often used for mining. Slower than ASICs, but faster than CPUs or GPUs.*

Term A.17 (Genesis-Block (Block Zero)) *First Block in the Blockchain.*

Term A.18 (Hash) *Is the result of Hash Algorithm executed to some arbitrary information.*

Term A.19 (Hash Algorithm) *Is needed to give Block, Transaction, Transaction tree, and many more unique identifiers. Is also needed to create a mining puzzle. Result of it is a Hash.*

Term A.20 (Hashcash) *Hashcash is an approach to deal with E-Mail spam. Its approach is used for the Proof of Work algorithm in Blockchains [48].*

Term A.21 (ICO) *Initial Coin Offering. A process of raising money (mostly cryptocurrencies) to fund blockchain startups. The offered coins represent a certain voting power in the new venture [69].*

Term A.22 (Identities) *Is a subject which has the private key to a certain public key. Is therefore eligible to spend Transaction outputs linked to their private key.*

Term A.23 (Merkle-Tree) *Also called Hash-Tree. A data-structure for providing integrity of files or data in general. Is used to store transactions in a block.*

Term A.24 (Merkle-Patricia-Tree) *Used in Ethereum for storage of transactions. Its difference is that all elements in the tree are sorted, such information can be retrieved faster.*

Term A.25 (Nonce) *Nonce is an abbreviation for "used only once" in cryptography, often used to secure connections. In mining it is used as a variable to change the content thus the hash of the block until it meets the mining-puzzle requirements.*

Term A.26 (Off-Chain) *Off-chain describes all processes concerning data in and around Blockchains. An example would be off-chain transactions. Instead of transacting the coins in the chain, the key-pair to the coins is sold. The coin changed the owner, but it was not recorded in the ledger.*

Term A.27 (Oracle) *An oracle is a third party in a blockchain providing information, e.g. weather data, randomness, or other external data feeds.*

Term A.28 (P2P) *Stands for peer-to-peer. Describes a network without a central instance.*

Term A.29 (Paxos) *A group of algorithms for providing consensus in unreliable networks. Paxos algorithms are commonly used in public blockchains.*

Term A.30 (Public Key) *The public key is part of a key pair for asymmetric encryption.*

Term A.31 (Satoshi Nakamoto) *Person or group responsible for creating Bitcoin. Real identity remains unknown.*

Term A.32 (Scripting Language) *Enables building of Transaction inputs and Transaction outputs. Enables building of Smart Contracts.*

Term A.33 (SHA-256) *Secure Hash Algorithm 2 with 256 bit output. Is used e.g. in Bitcoin.*

Term A.34 (Sidechains) *Sidechains are blockchains connected to a main chain. Funds can be transferred form one chain to the other. As the sidechain can have different rules, other applications become feasible within the same network [66].*

Term A.35 (Smart Contracts) *Smart Contracts are computable contracts enforcing agreements between two or more parties. They are often used in the Blockchain network Ethereum.*

Term A.36 (Solidity) *Solidity is the programming language of Ethereum.*

Term A.37 (Sybil-Attack) *A Sybil-attack is an attack at P2P-networks by creating false identities. This form of attack is used to manipulate votings, slow down the network, or to manipulate communication.*

Term A.38 (Testnet) *Many public blockchains provide a testnet which are thought for testing software. Certain limitations do not apply, such money can be created at will.*

Term A.39 (Transaction) *Consists of Transaction inputs and Transaction outputs and further data. Are stored in Transaction tree.*

Term A.40 (Transaction tree) *Contains all Transaction of one Block. Transaction Hash are linked together to build up a tree, resulting in a single hash. Single Hash stored into Block Header.*

Term A.41 (Wallet) *A wallet keeps the private keys of assets in a Blockchain. It allows the owner to transfer them to other private keys.*

List of Figures

List of Tables

Bibliography

[1] D. Tapscott and A. Tapscott, *Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World*. Penguin, 2016.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] CoinDesk.com, "Blockchain venture capital," http://www.coindesk.com/bitcoin-venture-capital/, 2017, online; retreived at Mar 21,2017.

[4] coinmarketcap.com, "Cryptocurrency market capitalizations," https://coinmarketcap.com/, 2017, online; retreived at Mar 17,2017.

[5] B. H. Co., "Blockchain for health research," https://blockchainhealth.co/, 2017, online; retreived at Mar 16,2017.

[6] A. Shelkovnikov, "Blockchain applications in the public sector," https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/Innovation/deloitte-uk-blockchain-app-in-public-sector.pdf, 2017, online; retreived at Mar 20,2017.

[7] G. Greenspan, "Avoiding the pointless blockchain project," http://www.multichain.com/blog/2015/11/avoiding-pointless-blockchain-project/, 2015, online; retrieved at Mar 05,2017.

[8] A. R. Hevner, "A three cycle view of design science research," *Scandinavian journal of information systems*, vol. 19, no. 2, p. 4, 2007.

[9] W. D. Fernández *et al.*, "The grounded theory method and case study data in is research: issues and design."

[10] R. W. Gregory, "Design science research and the grounded theory method: Characteristics, differences, and complementary uses," in *Theory-guided modeling and empiricism in information systems research*. Springer, 2011, pp. 111–127.

[11] R. H. Von Alan, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[12] ethercasts.com, "State of the Đapps," http://dapps.ethercasts.com/, 2017, online; retreived at Mar 21,2017.

[13] H. Rehäuser, Jakob; Krcmar, *Wissensmanagement im Unternehmen*. Schreyögg, Georg; Conrad, Peter, Berlin, New York: de Gruyter, 1996, vol. Managementforschung 6, Wissensmanagement, ch. S. 6.

[14] A. Cockburn, "Structuring use cases with goals," *Journal of Object-Oriented Programming*, 1997.

[15] ——, "Use cases, ten years later," *STQE magazine*, 2002.

[16] J. Hage, "What is a legal transaction?" *Law as Institutional Normative Order*, pp. 103–124, 2009.

[17] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press, 2016.

[18] D. Bradbury, "The problem with bitcoin," *Computer Fraud & Security*, vol. 2013, no. 11, pp. 5–8, 2013.

[19] M. Walport, "Distributed ledger technology: Beyond blockchain," *UK Government Office for Science*, 2016.

[20] J. de Kruijff and H. Weigand, "Towards a blockchain ontology."

[21] M. Szydlo, "Merkle tree traversal in log space and time," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 541–554.

[22] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[23] V. Buterin, "Proof of stake," https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ, 2017, online; retrieved at May 02,2017.

[24] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer *et al.*, "On scaling decentralized blockchains," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 106–125.

[25] I. Alqassem and D. Svetinovic, "Towards reference architecture for cryptocurrencies: Bitcoin architectural analysis," in *Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing (CPSCom), IEEE*. IEEE, 2014, pp. 436–443.

[26] B. P. Joshua J. Romero, "How bitcoin works," 2013.

[27] Multichain.com, "Multichain runtime parameters," http://www.multichain.com/developers/runtime-parameters/, 2015, online; retrieved at May 05,2017.

[28] V. Schlatt and A. Schweizer, "Blockchain: Grundlagen, anwendungen und potenziale," Projektgruppe Wirtschaftsinformatik des Fraunhofer-Instituts für Angewandte Informationstechnik FIT, 2016, online; retrieved at May 02,2017.

[29] W. Mougayar, "The crypto-technology and bitcoin landscape," http://startupmanagement.org/2015/03/03/the-crypto-technology-and-bitcoin-landscape/, 2015, online; retrieved at May 02,2017.

[30] AngelList, "Blockchains startups," https://angel.co/blockchains, 2017, online; retrieved at May 02,2017.

[31] R. Williams, "Credit strategy – blockchain technology: Robust, cost-effective applications key to unlocking blockchain's potential credit benefits," https://www.moodys.com/research/Moodys-Blockchain-can-bring-benefits-to-the-financial-industry-and--PR_352414, 2016, online; retrieved at Jan 07,2017.

[32] D. G. Wood, "Ethereum: a secure decentralized generalized transaction ledger," 2017.

[33] D. Siegel, "Understanding the dao attack," http://www.coindesk.com/understanding-dao-hack-journalists/, 2016, online; retrieved at Feb 23,2017.

[34] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 2014, pp. 459–474.

[35] G. Starke, *Effektive Softwarearchitekturen - Ein praktischer Leitfaden*. M: Carl Hanser Verlag GmbH Co KG, 2015.

[36] G. Greenspan, "Four genuine blockchain use cases," http://www.multichain.com/blog/2016/05/four-genuine-blockchain-use-cases/, 2016, online; retrieved at April 12,2017.

[37] M. Stevens, "Fast collision attack on md5." *IACR Cryptology ePrint Archive*, vol. 2006, p. 104, 2006.

[38] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, and Y. Markov, "The first collision for full sha-1," *URL: https://shattered. it/static/shattered. pdf*, 2017.

[39] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[40] V. S. Miller, "Use of elliptic curves in cryptography," in *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1985, pp. 417–426.

[41] National Security Agency, "The case for elliptic curve cryptography," 2009.

[42] D. R. Morrison, "Patricia practical algorithm to retrieve information coded in alphanumeric," *J. ACM*, vol. 15, no. 4, pp. 514–534, Oct. 1968.

[43] A. Biryukov, D. Dinu, and D. Khovratovich, "Fast and tradeoff-resilient memory-hard functions for cryptocurrencies and password hashing." *IACR Cryptology ePrint Archive*, vol. 2015, p. 430, 2015.

[44] Etherscan.io, "Ethereum average blocktime chart," https://etherscan.io/chart/blocktime, 2017, online; retrieved at Jan 5,2017.

[45] J. Wilcke, "The ethereum network is currently undergoing a dos attack," https://blog.ethereum.org/2016/09/22/ethereum-network-currently-undergoing-dos-attack/, 2016, online; retrieved at Jan 6,2017.

[46] U. Gallersdörfer and C. Sprecher, "Challenges and risks of blockchain technology," https://goo.gl/4tQj2W, 2017, online.

[47] S. Micali, "Algorand: The efficient and democratic ledger," *arXiv preprint arXiv:1607.01341*, 2016.

[48] A. Back *et al.*, "Hashcash-a denial of service counter-measure," 2002.

[49] E. Parliament, "Directive 98/26/ec of the european parliament and of the council of 19 may 1998 on settlement finality in payment and securities settlement systems," http://eur-lex.europa.eu/eli/dir/1998/26/oj, 1998.

[50] C. Pierrot and B. Wesolowski, "Malleability of the blockchain's entropy," in *ArcticCrypt 2016*, Longyearbyen, Norway, Jul. 2016. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01364045

[51] Gartner, "Gartner's 2016 hype cycle for emerging technologies identifies three key trends that organizations must track to gain competitive advantage," http://www.gartner.com/newsroom/id/3412017, 2016.

[52] J. Kume, M. Abe, and T. Okamoto, "Lottery protocol for cryptocurrency," *SCIS2015*, 2015.

[53] M. Green and I. Miers, "Bolt: Anonymous payment channels for decentralized currencies," Cryptology ePrint Archive, Report 2016/701, Tech. Rep., 2016.

[54] C. Eckert, *IT-Sicherheit: Konzepte-Verfahren-Protokolle*. Walter de Gruyter, 2013.

[55] Ethereum Foundation, "Create your own crypto-currency with ethereum," https://www.ethereum.org/token, 2017, online; retrieved at April 02,2017.

[56] Merriam-Webster.com, "Consensus." https://www.merriam-webster.com/dictionary/consensus, 2017, online; retrieved at Feb 13,2017.

[57] BuiltWith Pty Ltd, "Ssl by default usage statistics," https://trends.builtwith.com/ssl/SSL-by-Default, 2017, online; retrieved at May 03,2017.

[58] S. Garfinkel, *PGP: pretty good privacy*. " O'Reilly Media, Inc.", 1995.

[59] R. Trinkler, "Whisper," https://github.com/ethereum/wiki/wiki/Whisper, 2016, online; retrieved at May 01,2017.

[60] E. Gamma, *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995.

[61] Ethereum Developer Team, "Mist browser," https://github.com/ethereum/mist, 2017, online; retrieved at May 02,2017.

[62] L. Hartikka, "Naivechain - a blockchain implementation in 200 lines of code," https://github.com/lhartikk/naivechain, 2016, online; retrieved at Dez 03,2016.

[63] The Internet Society, "The base16, base32, and base64 data encodings," 2006.

[64] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *Security and Privacy (SP), 2013 IEEE Symposium on*. IEEE, 2013, pp. 397–411.

[65] D. Yermack, "Corporate governance and blockchains," *Review of Finance*, p. rfw074, 2017.

[66] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille, "Enabling blockchain innovations with pegged sidechains," *URL: http://www. opensciencereview. com/papers/123/enablingblockchain-innovations-with-pegged-sidechains*, 2014.

[67] IBM Institute for Business Value, "Empowering the edge - practical insights on a decentralized internet of things," https://www-935.ibm.com/services/multimedia/GBE03662USEN.pdf.

[68] Estonia, "e-estonia.com - the digital society," https://e-estonia.com/.

[69] Investopedia, "Initial coin offering (ico)," http://www.investopedia.com/terms/i/initial-coin-offering-ico.asp.